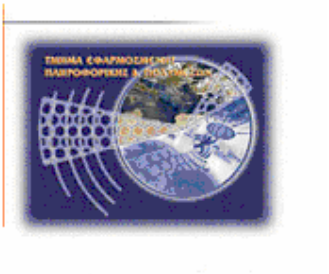
Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανικών Πληροφορικής



Πτυχιακή εργασία

Τίτλος: Ανάπτυξη συστήματος διοίκησης και διαχείρισης έργου βασισμένο στη μέθοδο Prince 2.

Αλεξάνδρα Δεβετζόγλου (ΑΜ: 3557)

Σπύρος Καραφυλλάκης (ΑΜ: 3596)

Επιβλέπων καθηγητής : Γεώργιος Παπαδουράκης

Επιτροπή Αξιολόγησης : Gareth Owens, Σπύρος Παναγιωτάκης

Ημερομηνία παρουσίασης: 11/10/2016

Ευχαριστίες

Θα θέλαμε να ευχαριστήσουμε τον κ. Παπαδουράκη για την ευκαιρία που μας έδωσε να δουλέψουμε μαζί του, καθώς και για την βοήθεια που μας παρείχε κατά τη διάρκεια εκπόνησης της πτυχιακής αυτής καθώς επίσης και την ομάδα του Blended Aim2016 *για την υπέροχη συνεργασία που είχαμε.*

Αλεξάνδρα Δεβετζόγλου: Θα ήθελα να ευχαριστήσω τους γονείς μου που μου πρόσφεραν ότι καλύτερο μπορούσαν και με στήριξαν, όλους τους ανθρώπους που στάθηκαν δίπλα μου και τους δασκάλους που μου δίδαξαν.

Σπύρος Καραφυλλάκης: Θα ήθελα να ευχαριστήσω την οικογένεια μου για την στήριξη τους όλα τα χρόνια των σπουδών μου, τους φίλους μου, καθώς και όλους όσους βοήθησαν για την εκπόνηση αυτής της πτυχιακή εργασίας.

Abstract

The purpose of this thesis is the designing and development of an online version of the printed KICKOFF canvas, within the international academic program Blended Aim. The canvas is a tool based on the principles of PRINCE2 method and it offers to users the opportunity to register to the application and collaborate with other users in real time from different locations.

For the development of the application in the backend, technologies like, apache HTTP server, MySQL, PHP, symfony, let's encrypt, web sockets and others were used. In the frontend, in which we worked during the project, we used technologies like, bootstrap, font awesome, angularJS, jQuery and others. Furthermore, the application was stored on the Microsoft Azure Platform.

The results of this project will be used from UWS Company as a base for a future commercial application.

Σύνοψη

Σκοπός της παρούσας πτυχιακής είναι η σχεδίαση και η ανάπτυξη μίας διαδικτυακής εκδοχής του έντυπου KICKOFF καμβά, στα πλαίσια του διεθνούς ακαδημαϊκού προγράμματος Blended Aim. Ο καμβάς είναι ένα εργαλείο βασισμένο στις αρχές της μεθόδου PRINCE2 και προσφέρει στους χρήστες την δυνατότητα να εγγραφούν στην εφαρμογή και να εργαστούν με άλλους χρήστες σε πραγματικό χρόνο από διαφορετικές τοποθεσίες.

Για την ανάπτυξη της εφαρμογής στο backend χρησιμοποιήθηκαν οι τεχνολογίες, apache HTTP server, MySQL, PHP, symfony, let's encrypt, web sockets και άλλα. Στο frontend, το οποίο εργαστήκαμε εμείς, χρησιμοποιήσαμε τις τεχνολογίες bootstrap, font awesome, angularJS, jQuery και άλλα. Επιπλέον η εφαρμογή φιλοξενήθηκε στην Microsoft Azure πλατφόρμα.

Τα αποτελέσματα της εργασίας θα χρησιμοποιηθούν από την εταιρία UWS ως βάση μιας μελλοντικής εμπορικής εφαρμογής.

Πίνακας περιεχομένων

Πίνακας Εικόνων

Λίστα Πινάκων

1 Εισαγωγή

1.1 Περίληψη

Η παρούσα πτυχιακή αναφέρεται στα πλαίσια ενός διεθνούς ακαδημαϊκού έργου (Blended Aim project), όπου φοιτητές από διάφορες χώρες της Ευρώπης σχεδίασαν και προγραμμάτισαν μία διαδικτυακή εκδοχή του έντυπου KICKOFF καμβά. Ο KICKOFF καμβάς είναι ένα εργαλείο συνεργασίας βασισμένο στις αρχές της μεθόδου PRINCE2 που έχει ως στόχο να βοηθήσει τους πελάτες να ξεκινήσουν ένα έργο. Ο απώτερος στόχος του ψηφιακού καμβά είναι να μειώσει τον χρόνο των συζητήσεων και των διαφωνιών και να θέσει τα άτομα της ομάδας σε ένα κοινό ρεαλιστικό μονοπάτι υλοποίησης. Με τον KICKOFF καμβά οι διαχειριστές έχουν την δυνατότητα να δημιουργήσουν και να εξάγουν ένα δομημένο επιχειρηματικό πλάνο σε τέσσερα εύκολα βήματα. Επιπλέον, βοηθά τις επιχείρησες να εξοικονομήσουν χρήματα, να έχουν επαγγελματικά αποτελέσματα και να είναι κάτι παραπάνω από ένας απλός καμβάς.

Για την ανάπτυξη της διαδικτυακής εφαρμογής χρησιμοποιήθηκαν τεχνολογίες διαδικτυακού προγραμματισμού. Η εφαρμογή χωρίστηκε σε backend και frontend. Όλη η εφαρμογή στηρίχτηκε στην πλατφόρμα Microsoft Azure. Στο backend χρησιμοποιήθηκε ο Apache HTTP server, η βάση δεδομένων MySQL και η γλώσσα προγραμματισμού PHP μαζί με το framework της το Symfony. Για την υλοποίηση της συνεργασίας εξ' αποστάσεως σε πραγματικό χρόνο χρησιμοποιήθηκαν Web Sockets. Στο frontend χρησιμοποιήθηκαν οι βασικές τεχνολογίες HTML, CSS και JavaScript μαζί τη βιβλιοθήκη της jQuery και το framework της AngularJS, ενώ χρησιμοποιήθηκε και το Bootstrap framework.

Ο KICKOFF καμβάς τείνει στο να προσφέρει λύση στις απαιτήσεις των γραφείων διαχείρισης του έργου: εφαρμογή των προτύπων διαχείρισης του έργου, δεξιότητες διαχείρισης της κατάρτισης του σχεδίου, και να διευκολυνθεί η μεταφορά γνώσεων. Ο σκοπός του διαδικτυακού καμβά είναι να γίνεται αντιληπτός ως ένα επαγγελματικό λογισμικό, το οποίο επιτρέπει την online και ασφαλή συνεργασία, και να οδηγεί σε σημαντική εξοικονόμηση χρημάτων από ταξίδια και εκπαίδευση, καθώς και του χρόνου.

1.2 Κίνητρο για τη διεξαγωγή της εργασίας

Στις μέρες μας, ένα κρίσιμο ζήτημα είναι η απόφαση για το αν μία εταιρεία, οργανισμός ή ιδιώτης θα ξεκινήσει ένα έργο ή όχι. Για να παρθεί η απόφαση αυτή πρέπει να εξεταστούν πολλά διαφορετικά ζητήματα όπως είναι το κόστος και το ρίσκο για να διαπιστωθεί τελικά αν συμφέρει ή όχι η έναρξη ενός έργου. Η παρούσα πτυχιακή εργασία αναπτύχθηκε στα πλαίσια του προγράμματος *Blended Aim* και στόχος της είναι να επιλύσει το παραπάνω πρόβλημα με την κατασκευή ενός εργαλείου βασισμένου στο web για την συνεργασία μεταξύ ατόμων ώστε να αποφασίσουν για την έναρξη ενός έργου όπως επίσης και να αναλύσει τους στόχους του *Blended Aim*.

1.3 Σκοπός και στόχοι Εργασίας

Σκοπός της παρούσας πτυχιακής είναι η σχεδίαση και η ανάπτυξη μίας διαδικτυακής εκδοχής του έντυπου KICKOFF καμβά, αναφερόμενη στη δική μας εργασία πάνω στο έργο αυτό, στα πλαίσια του διεθνούς ακαδημαϊκού προγράμματος Blended Aim. Οι στόχοι της εργασίας αυτής είναι η ανάπτυξη ενός αξιόπιστου εργαλείου βασισμένου στις αρχές του PRINCE2 το οποίο θα επιτρέπει στους χρήστες να συνεργάζονται σε πραγματικό χρόνο για την απόφαση της έναρξης ενός έργου καθώς και στην εξοικείωση του σπουδαστή με όλα τα στάδια της ανάπτυξης διαδικτυακών εφαρμογών ώστε να διευρυνθούν οι γνώσεις του. Επίσης στοχεύει στο να αναλύσει και να παρουσιάσει τις θεωρίες, τεχνολογίες και

μεθοδολογίες που χρησιμοποιήθηκαν καθώς επίσης και τα στάδια που ακολουθήθηκαν για την ανάπτυξη της.

1.4 Δομή εργασίας

Η πτυχιακή εργασία αποτελείται από πέντε κύρια κεφάλαια:

- *Κεφάλαιο 1-Εισαγωγή*: Αναλύει το περιεχόμενο της πτυχιακής, τους σκοπούς, τους στόχους της και εξηγεί τι είναι το Blended Aim
- *Κεφάλαιο 2-Μεθοδολογία Υλοποίησης*: Αναλύει τον τρόπο με τον οποίο αναπτύχθηκε η εφαρμογή-τεχνολογίες και μεθοδολογία που χρησιμοποιήθηκαν και επίσης αναλύει θεωρητικά θέματα που είναι σχετικά με την εφαρμογή.
- *Κεφάλαιο 3-Σχέδιο δράσης*: Αναφέρεται στο State of the art της εφαρμογής και αναλύει σε βάθος τις τεχνολογίες και τις θεωρίες πάνω στις οποίες βασίστηκε η εφαρμογή
- *Κεφάλαιο 4-Κύριο μέρος*: Αναφέρεται στην υλοποίηση της εφαρμογής, καταγράφοντας τις απαιτήσεις και αναλύοντας τον τρόπο που εργαστήκαμε για την υλοποίηση της. Επίσης υλοποιείται το εγχειρίδιο χρήσης της εφαρμογής.
- *Κεφάλαιο 5-Αποτελέσματα*: Καταγράφονται τα συμπεράσματα από την πτυχιακή και γίνεται αναφορά σε μελλοντική βελτίωση της εφαρμογής.

1.5 Blended Aim

Το *Blended Aim* ή αλλιώς *BAIM* είναι ένα πρόγραμμα το οποίο χρηματοδοτείται από το *Erasmus+, Συνεργασία για Καινοτομία και την ανταλλαγή Καλών Πρακτικών* της Ευρωπαϊκής Επιτροπής. Το *BAIM* σημαίνει Blended Academic International Mobility και γίνεται εδώ και τρία χρόνια με εταίρους από δέκα διαφορετικά πανεπιστήμια, από χώρες όπως η Πορτογαλία το Ηνωμένο Βασίλειο, το Βέλγιο, η Κύπρος, η Ιταλία, η Γερμανία, η Ελλάδα και η Αυστρία. Σκοπός του είναι να αναδιαμορφώσει τη διεθνή κινητικότητα και να ενδυναμώσει την απασχόληση των φοιτητών μέσω της ανάμεικτης κινητικότητας (blended mobility).

Η επαγγελματική ζωή στις μέρες μας, εξαρτάται σε μεγάλο βαθμό από την κινητικότητα και απαιτεί από τους επαγγελματίες να υπερτερούν σε δεξιότητες επικοινωνίας σε διεθνές, διαπολιτισμικό περιβάλλον. Ωστόσο, οι δεξιότητες όπως επίσης και η διεθνής έκθεση, σπάνια αντιμετωπίζονται από προπτυχιακά μαθήματα. Η ανάμεικτη κινητικότητα ξεπερνά τα τυπικά εμπόδια της, επιτρέποντας έτσι στους μαθητές να επωφεληθούν από τα πλεονεκτήματα που η κινητικότητα και η διεθνής έκθεση προσφέρουν. Ωστόσο, ανεξάρτητα από την προστιθέμενη αξία της, η ανάμεικτη κινητικότητα σπάνια χρησιμοποιείται και μετά βίας αναγνωρίζεται ως σοβαρή εναλλακτική λύση με μεγάλες δυνατότητες για να ξεπεραστούν τα κοινά προβλήματα της διεθνούς κινητικότητας.

Το πρόγραμμα *Blended Aim* θέτει τις βάσεις για να εγκρίνει και να δομήσει την ανάμεικτη κινητικότητα γενικότερα. Πιο συγκεκριμένα, παρέχει τους πόρους, όπως την κατάρτιση, τα εργαλεία υποστήριξης και τις πληροφορίες, για να βοηθήσει τους μαθητές και τις εταιρείες που φιλοξενούν πρακτική άσκηση και εξορθολογεί καινοτόμα παραδείγματα διδασκαλίας με στόχο να αναπτυχθούν οι δεξιότητες των μαθητών. Για να ενισχυθεί αυτό, οι εταίροι ίδρυσαν το δίκτυο *Praxis* το οποίο είναι μία κοινοπραξία των ιδρυμάτων τριτοβάθμιας εκπαίδευσης και των επιχειρήσεων για την ενίσχυση της διεθνούς εμπειρίας φοιτητών σε έργα και πρακτικές ασκήσεις και την προώθηση της καινοτομίας στον τομέα.

Στην πράξη, για να πετύχει το πρόγραμμα τον σκοπό του, υλοποιείται κάθε χρόνο από την έναρξη του, ένα έργο στο οποίο συμμετέχουν φοιτητές από τις χώρες εταίρους και το οποίο συγκαταλέγεται ως μάθημα και δίνει διδακτικές μονάδες στους φοιτητές μετά την

ολοκλήρωση του. Οι φοιτητές είναι από διαφορετικά πεδία μελέτης, όπως την επιστήμη των υπολογιστών, το σχεδιασμό και τη διοίκηση επιχειρήσεων, και προσπαθούν να δώσουν λύση σε ένα πρόβλημα που παρατίθεται από μία εταιρεία όπως για παράδειγμα τη δημιουργία μιας εφαρμογής που εξυπηρετεί τους σκοπούς της. Εργάζονται ως μία πολυεθνική και διεπιστημονική ομάδα, με κοινό στόχο την παράδοση στο τέλος του έργου, ένα προϊόν που ακολουθεί τις απαιτήσεις της εταιρείας.

Κατά τη διάρκεια του προγράμματος γίνονται δύο συναντήσεις πρόσωπο με πρόσωπο μεταξύ των μαθητών, μία στην αρχή του εξαμήνου και μία στο τέλος. Στην πρώτη συνάντηση γνωρίζονται μεταξύ τους, με τους καθηγητές και τους εκπροσώπους της εταιρείας, εξοικειώνονται με το πρόβλημα που πρέπει να λυθεί και συμφωνούν για το ποιος είναι ο καλύτερος τρόπος που πρέπει να εργαστούν κατά τη διάρκεια του εξαμήνου. Μετά δουλεύουν από τα ιδρύματα προέλευσης τους, όπου συνεργάζονται μέσω διαδικτυακών εργαλείων που τους παρέχονται από το πρόγραμμα. Καθ' όλη τη διάρκεια που βρίσκονται στα ιδρύματα τους, οι φοιτητές έχουν διαδικτυακές συναντήσεις όπου συζητούν την πορεία του έργου και τι μπορεί να βελτιωθεί σε αυτό. Στην τελευταία συνάντηση, παρουσιάζουν το προϊόν που οι ίδιοι ανέπτυξαν, αξιολογούνται από καθηγητές και εκπροσώπους της εταιρίας και από την αξιολόγηση αυτή προκύπτει ο βαθμός τους.

2 Μεθοδολογία Υλοποίησης

2.1 Μέθοδος Ανάλυσης & Ανάπτυξης

Η ανάπτυξη συστήματος διοίκησης και διαχείρισης έργου βασισμένου στην μέθοδο PRINCE2 είναι μία web εφαρμογή που επιτρέπει στους χρήστες να επικοινωνούν και να συνεργάζονται μεταξύ τους με σκοπό την υλοποίηση μιας ιδέας. Η εφαρμογή φιλοξενείται και εκτελείται σε έναν web εξυπηρετητή ώστε να είναι διαθέσιμη στο Διαδίκτυο. Μέσω μιας διεπαφής (interface) ο χρήστης θα αλληλοεπιδρά με την εφαρμογή ενώ θα υπάρχει και μία βάση δεδομένων η οποία θα αποθηκεύει τα στοιχεία του χρήστη και λοιπά δεδομένα. Η διεπαφή θα επικοινωνεί με τον web εξυπηρετητή και τη βάση δεδομένων για την άντληση ή την αποστολή περιεχόμενου και δεδομένων, κάνοντας *API κλήσεις*.

Για την υλοποίηση της πτυχιακής εργασίας χρησιμοποιήθηκε σαν υποδομή η πλατφόρμα *Microsoft Azure*, όπου εκεί φιλοξενήθηκε όλη η εφαρμογή. Η ανάπτυξη της εφαρμογής χωρίστηκε σε backend και frontend. Στο backend χρησιμοποιήθηκε ο *Apache HTTP server* σαν web εξυπηρετητής σε περιβάλλον *Linux Debian*. Για βάση δεδομένων χρησιμοποιήθηκε η *MySQL*, ενώ για την αμφίδρομη επικοινωνία σε πραγματικό χρόνο μεταξύ του πελάτη και του web εξυπηρετητή χρησιμοποιήθηκε η *PHP* βιβλιοθήκη *Ratchet* η οποία καθιστά εφικτή αυτή την επικοινωνία χρησιμοποιώντας *Web Sockets*. Όλη η λειτουργία του backend βασίστηκε στην αρχιτεκτονική *MVC* (*Model-View-Controller*) και υλοποιήθηκε με τη γλώσσα προγραμματισμού *PHP* και στο web framework της, το *Symfony*. Τέλος, όλες οι συναλλαγές μεταξύ πελάτη και εξυπηρετητή είναι κρυπτογραφημένες χάρη στη χρήση ψηφιακών πιστοποιητικών από την οντότητα πιστοποιήσεων *Let's Encrypt*.

Στο frontend όπως και στο backend χρησιμοποιήθηκε η αρχιτεκτονική *MVC*. Οι κύριες τεχνολογίες για την ανάπτυξη της διεπαφής ήταν οι *HTML*, *CSS* και *JavaScript*. Για την σχεδίαση της εφαρμογής χρησιμοποιήθηκε το *Bootstrap framework*, το οποίο αντιπροσωπεύει το *View* του *MVC*, ενώ για τη συμπεριφορά της εφαρμογής χρησιμοποιήθηκε η *JavaScript* βιβλιοθήκη *jQuery* και η διεπαφή της *jQuery UI* καθώς και το *JavaScript framework AngularJS*, στην έκδοση 1.5, το οποίο αποτέλεσε τον πυρήνα της εφαρμογής. Χάρη στο *AngularJS* υλοποιήθηκε η αρχιτεκτονική MVC, η οποία διευκόλυνε την ανάπτυξη της εφαρμογής με το χωρισμό της σε τρία μέρη.

Η μεθοδολογία που ακολουθήθηκε για την ανάπτυξη της εφαρμογής βασίστηκε στο *Scrum*. Αναλύθηκαν οι απαιτήσεις της εφαρμογής και με βάση αυτές δημιουργήθηκαν ιστορίες χρήστη (user stories) οι οποίες αξιολογήθηκαν και τους δόθηκε προτεραιότητα ώστε να ολοκληρωθούν πρώτα αυτές που είναι πιο σημαντικές. Κάθε ιστορία χρήστη είχε επιμέρους εργασίες (tasks) όπου και αυτές αξιολογούνταν και γινόταν εκτίμηση του χρόνου που απαιτείται για να ολοκληρωθούν. Αν όλες οι εργασίες μιας ιστορίας ολοκληρώνονταν τότε η ιστορία θεωρούταν τελειωμένη.

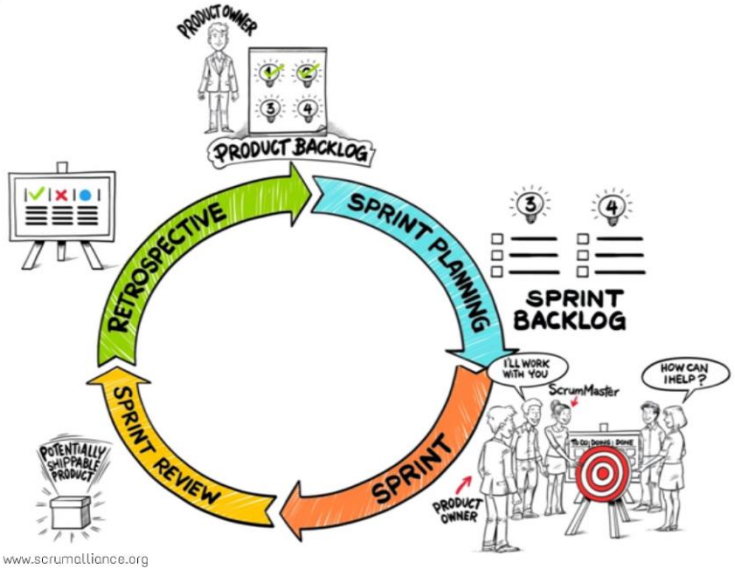
2.2 Θεωρίες

2.2.1 Scrum

Το Scrum είναι ένα ευκίνητο framework ανάπτυξης λογισμικού (*agile software development framework*) το οποίο δείχνει την διαδικασία της διαχείρισης και του ελέγχου που διαπερνούν την πολυπλοκότητα. Ορίζεται ως μία ευέλικτη στρατηγική ανάπτυξης ενός προϊόντος, όπου η ομάδα ανάπτυξης λειτουργεί ως μονάδα για την επίτευξη ενός κοινού στόχου. Το Scrum δίνει έμφαση στην συνεργασία, τη λειτουργία του λογισμικού, τις δυνατότητες διαχείρισης της ομάδας και την ευελιξία να προσαρμοστούν όλα αυτά στην πραγματικότητα των επιχειρήσεων. Επιπλέον, αμφισβητεί τις παραδοσιακές προσεγγίσεις για την ανάπτυξη ενός προϊόντος και επιτρέπει στις ομάδες να αυτο-οργανώνονται μέσω της ενθάρρυνσης από το

φυσικό περιβάλλον, της διαδικτυακής επικοινωνίας μεταξύ των μελών αλλά και της φυσικής επαφής, καθώς και τον τρόπο υλοποίησης του project.

Η πρώτη προσέγγιση του scrum έγινε από τους Hirotaka Takeuchi και Ikujiro Nonaka το 1986 σε μία μελέτη του Harvard Business Review. Οι συγγραφείς περιέγραψαν μια νέα προσέγγιση για την ανάπτυξη εμπορικών προϊόντων που θα αυξήσει την ταχύτητα και την ευελιξία της με βάση μελέτες που έγιναν από επιχειρήσεις. Ονόμασαν αυτή την προσέγγιση ολιστική (*holistic*) ή ράγκμπι (*rugby*), καθώς όλη η διαδικασία εκτελείται με μία διατμηματική ομάδα (*cross-functional team*) σε πολλαπλές επικαλυπτόμενες φάσεις. Το scrum αναφέρεται και ως ράγκμπι (δηλ. μια μορφή ποδόσφαιρου) λόγω της δεμένης ομάδας της οποίας οι παίχτες κρατάνε χαμηλά τα κεφάλια τους με σκοπό να κερδίσουν την κατοχή της μπάλας. Στις αρχές του 1990 ο Jeff Sutherland μαζί με τον John Scumniotales και τον Jeff McKenna ανέπτυξαν μία παρόμοια προσέγγιση και ήταν οι πρώτοι που αναφέρθηκαν σε αυτή μονάχα με την λέξη scrum.



Εικόνα 2.2.1.1 Η λειτουργία του Scrum

Υπάρχουν τρεις βασικοί ρόλοι εντός του scrum framework. Αν και μπορεί σε ένα πραγματικό project να αναφερθούν και άλλου ρόλοι, οι τρεις παρακάτω είναι εκείνοι που αντιπροσωπεύουν ολόκληρη την ομάδα. Ο πρώτος ρόλος είναι αυτός του ιδιοκτήτη του προϊόντος (*product owner*), ο οποίος αντιπροσωπεύει τους ενδιαφερόμενους και είναι η φωνή του πελάτη στην ομάδα και σιγουρεύει ότι θα μπορέσουν να παραδώσουν αξία στην επιχείρηση. Ο product owner είναι εκείνος που γράφει τα κεντρικά ενδιαφέροντα του πελάτη ως ιστορίες χρήστη (*user stories*), τα κατατάσσει και τα ιεραρχεί με βάση την προτεραιότητα που έχουν και τα προσθέτει στο απόθεμα του προϊόντος (*product backlog*). Η επικοινωνία είναι η κύρια λειτουργία του product owner και ο ρόλος του είναι μοναδικός για αυτό και δεν πρέπει να μπλέκεται και να συγχέεται με τους υπόλοιπους ρόλους.

Σκοπός αυτού του ρόλου είναι ο υπεύθυνος να επικεντρωθεί στην επιχειρηματική πλευρά του έργου και να περάσει μεγάλο χρονικό διάστημα με τους ενδιαφερόμενους και δεν πρέπει να επεμβαίνει και να αλληλοεπιδρά με την υπόλοιπη ομάδα σε ότι έχει σχέση με το τεχνικό κομμάτι. Η ικανότητα του να κατηγοριοποιεί τις ανάγκες του project και να δίνει έμφαση στα μέλη της ομάδας και τους ενδιαφερόμενους είναι θέμα ζωτικής σημασίας ώστε το έργο να πάρει την σωστή κατεύθυνση. Με άλλα λόγια ο product owner είναι ο μεσολαβητής μεταξύ της ομάδας και του πελάτη. Το άτομο που είναι υπεύθυνο για αυτό τον ρόλο θα πρέπει να έχει ως χαρακτηριστικά την αποτελεσματικότητα και την κατανόηση, καθώς θα πρέπει αν έρθει σε επαφή με διαφορετικούς ρόλους ανθρώπων από την πλευρά των πελατών, δηλαδή της εταιρίας, και να μεταφέρει όσα αναφέρθηκαν με σαφή τρόπο στα μέλη της ομάδας του.

Πιο συγκεκριμένα, τα καθήκοντα που πρέπει να πραγματοποιεί ο product owner ως ο εκπρόσωπος της ομάδας στους ενδιαφερόμενους πελάτες είναι τα εξής:

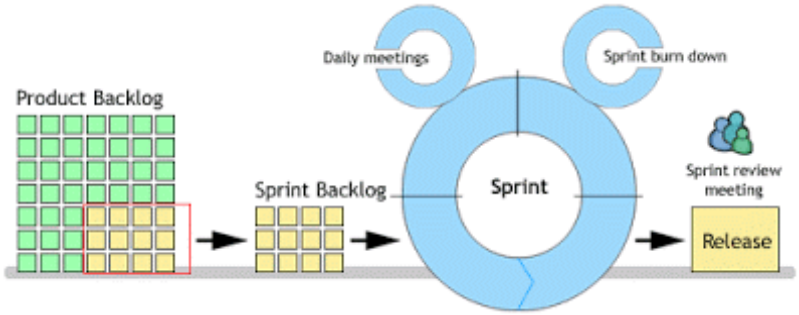
- Επιδεικνύει τα αποτελέσματα στους κύριους ενδιαφερόμενους που δεν μπόρεσαν να είναι παρόν κατά την διάρκεια του *sprint review*.
- Καθορίζει και ανακοινώνει τα *releases*, δηλαδή τα στάδια κατά τα οποία πρέπει να είναι τελειωμένο το προϊόν.
- Επικοινωνεί με την ομάδα για την κατάσταση στην οποία βρίσκονται.
- Διοργανώνει τις ανασκοπήσεις ορόσημο (*milestone reviews*).
- Αναλύει και εξηγεί στους ενδιαφερόμενους την διαδικασία ανάπτυξης και την πρόοδο του προϊόντος τους.
- Διαπραγματεύεται τις προτεραιότητες, τον σκοπό, την χρηματοδότηση και το χρονοδιάγραμμα.
- Διασφαλίζει ότι το product backlog είναι ορατό και ξεκάθαρο για όλους.

Επόμενος ρόλος είναι αυτός του συντονιστή του scrum (*scrum master*), ο οποίος είναι υπεύθυνος να δώσει λύσεις σε προβλήματα και να απομακρύνει τα εμπόδια της ομάδας με σκοπό να παραδοθεί το προϊόν την στιγμή που πρέπει. Ο ρόλος του scrum master δεν είναι ο παραδοσιακός ρόλος του αρχηγού της ομάδας, ούτε του διαχειριστή του έργου, αλλά ενεργεί ως ενδιάμεσος μεταξύ των μελών της ομάδας με σκοπό να λύσει οποιεσδήποτε συγχίσεις. Είναι εκείνος που σιγουρεύει ότι η διαδικασία του scrum framework λειτουργεί όπως θα έπρεπε, βοηθάει τα μέλη να ακολουθήσουν την διαδικασία και συχνά κάνει συναντήσεις ώστε να ενθαρρύνει την ομάδα να βελτιωθεί. Ο ρόλος έχει επίσης αναφερθεί και ως διαμεσολαβητής της ομάδας. Μερικά από τα καθήκοντα του είναι τα εξής:

- Βοηθάει τον product owner να διατηρήσει την πορεία του product backlog ώστε να διασφαλιστεί ότι η δουλειά που πρέπει να γίνει είναι κατανοητή από όλη την ομάδα και η διαδικασία προχωράει μπροστά.
- Βοηθάει την ομάδα να εξασφαλίσει ότι ένα user story έχει τελειώσει με βάση τα δεδομένα τον ενδιαφερόμενων.
- Προπονεί την ομάδα με βάση τις αρχές του scrum, προκειμένου να παραδοθεί ένα υψηλής ποιότητας προϊόν.
- Προωθεί την αυτό-οργάνωση εντός της ομάδας.
- Βοηθά την ομάδα του scrum να αποφύγει και να λύσει προβλήματα εντός και εκτός αυτής.
- Διευκολύνει τα γεγονότα της ομάδας ώστε να εξασφαλιστεί η τακτική πρόοδος.

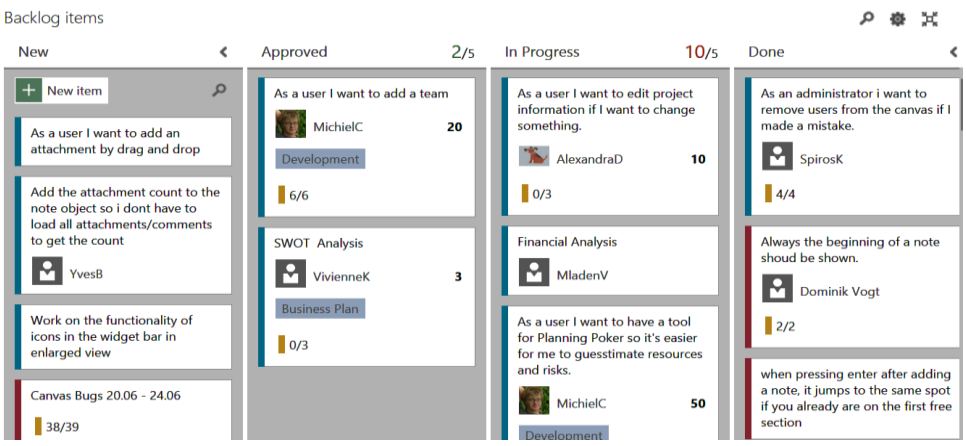
Ο τρίτος ρόλος είναι ολόκληρης της ομάδας ανάπτυξης (*development team*), η οποία είναι υπεύθυνη να υλοποιήσει και να ολοκληρώσει τα user stories που της έχουν ανατεθεί ως το τέλος ενός *sprint*. Μια ομάδα αποτελείται από τρία έως εννέα άτομα που κάνουν την πραγματική εργασία δηλαδή αναλύουν, σχεδιάζουν, προγραμματίζουν, δοκιμάζουν, γράφουν το έργο και επικοινωνούν για τα τεχνικά. Οι ομάδες ανάπτυξης είναι συνήθως *cross-functional* με όλες τις απαραίτητες ικανότητες να υλοποιήσουν τον στόχο, για παράδειγμα, έχει έναν τουλάχιστον σχεδιαστή (*designer*), ένα προγραμματιστή (*developer*) έναν υπεύθυνο για τα οικονομικά (*finance manager*) και έναν υπεύθυνο τον επιχειρήσεων (*business manager*). Η ομάδα ανάπτυξης για το scrum είναι αυτο-οργανωτική (*self-organized*), παρόλα αυτά μπορεί να υπάρξει κάποιου είδους διασύνδεση με τα γραφεία διαχείρισης του γενικού έργου.

Το *sprint*, μια επαναλαμβανόμενη διαδικασία, είναι η βασική μονάδα ή ο πυρήνας της ανάπτυξης του scrum. Είναι μία χρονικά περιορισμένη προσπάθεια (*timeboxed effort*), που κρατάει για συγκεκριμένο σταθερό χρονικό διάστημα. Η διάρκεια του κάθε sprint καθορίζεται εκ των προτέρων και μπορεί να διαρκέσει από μία εβδομάδα το ελάχιστο έως ένα μήνα το μέγιστο, με συχνότερη διάρκεια τις δύο εβδομάδες. Κάθε sprint ξεκινά με ένα γεγονός διοργάνωσης (*sprint planning event*), στο οποίο καθορίζεται ο στόχος του συγκεκριμένου *sprint backlog*, αναγνωρίζεται η εργασία που πρέπει να πραγματοποιηθεί και εκτιμάται ο χρόνος και η προσπάθεια ώστε να πραγματοποιηθεί. Κάθε sprint τελειώνει με το *sprint review* και το *sprint retrospective* τα οποία παρουσιάζουν την πρόοδο και κατά τα οποία αναγνωρίζονται οι βελτιώσεις που πρέπει να γίνουν κατά την διάρκεια του επόμενου sprint.



Εικόνα 2.2.1.2 Τα Product Backlog Items του Sprint

Το scrum δίνει έμφαση σε ένα προϊόν το οποίο μετά την λήξη ενός sprint θα είναι έτοιμο, ως πρότυπο. Στην περίπτωση του λογισμικού, αυτό περιλαμβάνει ότι το λογισμικό θα έχει ενσωματωθεί, θα έχει πλήρως δοκιμαστεί, θα έχει τεκμηριωθεί και θα μπορεί να κυκλοφορήσει. Για αυτό τον λόγο πολύ σημαντικό ρόλο έχει η οργάνωση του κάθε sprint (*sprint planning*) κατά την οποία η ομάδα θα πρέπει να επικοινωνήσει ώστε να κανονιστεί η δουλειά και να εκτιμηθεί ο χρόνος των εργασιων για το νέο sprint. Επίσης, στο sprint planning επιλέγονται τα στοιχεία (*backlog items*) από το product backlog τα οποία είναι πολύ πιθανόν να μπορούν να υλοποιηθούν κατά την διάρκεια του νέου sprint και προετοιμάζονται οι λεπτομέρειες αυτών. Η χρονική διάρκεια ενός sprint planning για ένα sprint δύο εβδομάδων είναι τέσσερις ώρες. Τις πρώτες δύο ώρες, όλη η ομάδα συμφωνεί για τα νέα backlog items και τις επόμενες δύο ώρες, οι προγραμματιστές τα διασπούν σε μικρότερες εργασίες (*tasks*) και τα αναθέτουν μεταξύ τους.



Εικόνα 2.2.1.3 Παράδειγμα υλοποίησης του Scrum

Στο τέλος ενός sprint, όπως προαναφέρθηκε, η ομάδα καλείται σε δύο γεγονότα, το sprint review και το sprint retrospective. Κατά το *sprint review* η ομάδα εξετάζει και αξιολογεί τις εργασίες (tasks) που τελείωσε και αυτή που είναι σε εξέλιξη, παρουσιάζοντας ένα demo

στους ενδιαφερόμενους. Σημαντικό είναι να σημειωθεί ότι τα κομμάτια της εργασίας που δεν δουλεύουν ή είναι λειψά δεν επιδεικνύονται. Η προτεινόμενη διάρκεια του sprint review είναι δύο ώρες ανάλογα και την διάρκεια του κάθε sprint. Το *sprint retrospective* αντανακλά το sprint που πέρασε, καθώς αναγνωρίζεται και γίνεται συμφωνία μιας συνεχούς διαδικασίας για βελτίωση των πράξεων. Οι σημαντικές ερωτήσεις που πρέπει να απαντήσουν τα μέλη της ομάδας είναι, «Τι πήγε καλά κατά την διάρκεια αυτού του sprint;» και «Τι μπορεί να βελτιωθεί για το επόμενο sprint;». Το γεγονός του sprint retrospective πραγματοποιείται από τον scrum master και έχει διάρκεια μία ώρα και 30 λεπτά, για ένα sprint δύο εβδομάδων.

Κατά την διάρκεια ενός sprint όλη η ομάδα πρέπει να ακολουθεί το *daily scrum*, το οποίο είναι ένα καθημερινό *stand-up* με συγκεκριμένους κανόνες. Όλα τα μέλη της ομάδας ανάπτυξης (development team) θα πρέπει να έρχονται προετοιμασμένα, καθώς το daily scrum ξεκινάει στην ώρα του ακόμη κι αν κάποιο μέλος λείπει και περιορίζεται στα 15 λεπτά. Κατά την διάρκεια του daily scrum όλα τα μέλη της ομάδας θα πρέπει να απαντήσουν σε τρεις ερωτήσεις που είναι οι εξής:

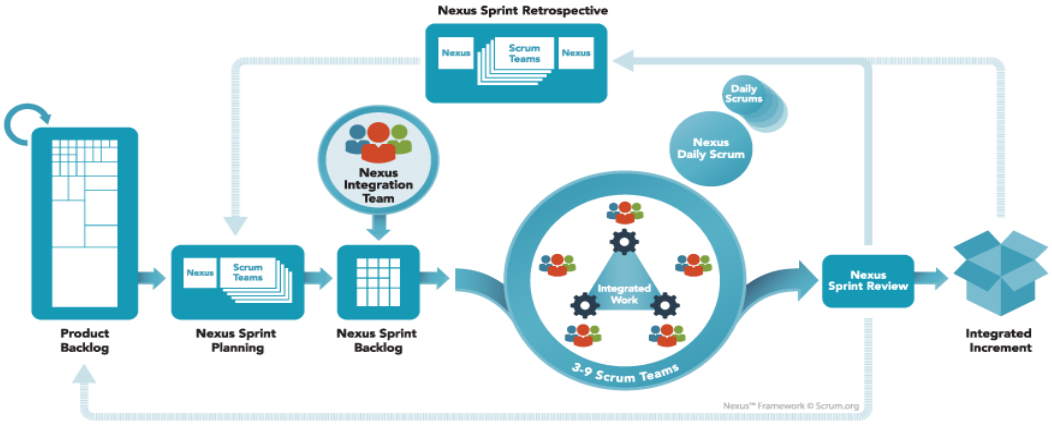
- Τι *έκανες χθες* για να βοηθήσεις την ομάδα σου να φτάσει τον στόχο του τωρινού sprint;
- Τι *θα κάνεις σήμερα* για να βοηθήσεις την ομάδα σου να φτάσει τον στόχο του τωρινού sprint;
- Ποια είναι τα *εμπόδια και οι δυσκολίες* που αντιμετωπίζεις εσύ και η ομάδα σου τα οποία σας αποτρέπουν από το να φτάσετε τον στόχο του τωρινού sprint;

Κάθε εμπόδιο (αδιέξοδα, συγκρούσεις, και άλλα) που αναφέρεται κατά την διάρκεια του daily scrum πρέπει να λαμβάνονται υπόψη από τον scrum master, ο οποίος αφού τελειώσει η συνάντηση θα συζητήσει με το κάθε μέλος ψάχνοντας για λύσεις. Λεπτομέρειες για προβλήματα δεν δίνονται κατά την διάρκεια αυτών των 15 λεπτών.

2.2.2 Nexus Scrum

Η ανάπτυξη λογισμικού είναι μία περίπλοκη και δύσκολη δουλειά. Μεγάλες επιχειρήσεις στην προσπάθεια του να ανταπεξέλθουν στην εξέλιξη και της ανάγκες της τεχνολογίας, αντιμετωπίζουν προβλήματα και δεν μπορούν να συμβαδίσουν με αυτή. Είναι φανερό ότι το Scrum είναι ένα πλεονέκτημα που του βοηθά σε αυτό αλλά το Nexus framework τους οδηγεί ένα βήμα παρακάτω. Το Nexus framework δημιουργήθηκε από τον Ken Schwaber, συν-δημιουργό του Scrum framework και δημοσιεύτηκε από τον οργανισμό του μαζί με τον οδηγό το 2015. Το Nexus είναι βασισμένο στο Scrum framework και χρησιμοποιεί μια επαναληπτική και αυξητική προσέγγιση για την κλιμάκωση του λογισμικού και την ανάπτυξη του προϊόντος. Το Nexus επεκτείνει το Scrum ελάχιστα και διατηρεί τον συντονισμό του. Είναι η βάση για τον σχεδιασμό, την οργάνωση, την έναρξη και την διαχείριση μεγάλων προϊόντων και πρωτότυπης ανάπτυξης λογισμικού. Το Nexus scrum υπάρχει για να χρησιμοποιείται όταν πολλαπλές ομάδες scrum δουλεύουν πάνω στο ίδιο προϊόν, καθώς επιτρέπει στις ομάδες να ενωθούν ως μία ομάδα nexus (*nexus scrum team*), τις προστατεύει και τις ενισχύει με τη δημιουργία συνδέσεων μεταξύ τους.

Ένα Nexus framework αποτελείται από 3-9 scrum teams που εργάζονται σε ένα ενιαίο product backlog για την οικοδόμηση ενός ολοκληρωμένου αποτελέσματος το όποιο θα φτάνει στο επιθυμητό αποτέλεσμα. Το framework αυτό είναι ιδιαίτερα ευεργετικό για τους οργανισμούς οι οποίοι είχαν θετικά αποτελέσματα με τη χρήση του απλού scrum με μία ή δύο ομάδες που εργαζόντουσαν στο ίδιο product backlog και που θέλουν να αναβαθμιστούν ευρύτερα. Από την άλλη πλευρά είναι επίσης εξαιρετικά χρήσιμο για εκείνους που δυσκολεύονται με την κλιμάκωση, καθώς το nexus επικεντρώνεται αυστηρά στα δύο βασικά θέματα κλιμάκωσης (*core scaling matters*), τις εξαρτήσεις μεταξύ των διαφορετικών ομάδων (*cross-team dependencies*) και τα θέματα συνένωσης των αποτελεσμάτων του σε ένα κοινό προϊόν (*integration issues*).



Εικόνα 2.2.2.1 Η διαδικασία του Nexus Framework

Σε αντίθεση με τον απλό scrum, στο nexus scrum έχουμε μία ομάδα συνένωσης (*nexus integration team*). Η Nexus Integration Team είναι υπεύθυνη να διασφαλίσει ότι ο συνδυασμός της δουλειάς όλων των ομάδων είναι ένα ολοκληρωμένο προϊόν στο τέλος του κάθε sprint. Συνήθεις ενέργειες μπορεί να περιλαμβάνουν την αναγνώριση των προβλημάτων εντός των ομάδων (*cross-team issues*), η κατανόηση των εξαρτήσεων μεταξύ τους (*awareness of dependencies*) προτού ξεκινήσει η υλοποίηση και η εξασφάλιση των εργαλείων συνένωσης (*integration tools*). Η Nexus Integration Team είναι μία ομάδα scrum και τα μέλη της μπορεί να εργάζονται εξ' ολοκλήρου ή μερικώς σε αυτήν. Τα μέλη της ομάδας πρέπει να έχουν προτεραιότητες ώστε να δώσουν προτεραιότητες στα υπόλοιπα μέλη και να βοηθήσουν να λυθούν προβλήματα που τυχόν θα επηρεάσουν τις ομάδες.

Ο Product Owner και ένας Scrum Master είναι σε αυτή την ομάδα. Επιπλέον τα μέλη της Integration Team μπορούν να είναι και μέλη των Nexus Scrum Team και να έχουν αποκλειστικά καθήκοντα και μη με βάση τις ανάγκες της εταιρίας. Η συγκρότηση της ομάδας μπορεί να αλλάξει με την πάροδο του χρόνου με εξαίρεση τον Product Owner. Η ομάδα δουλεύει το Product Backlog και αναλαμβάνει οποιοδήποτε πρόβλημα ενσωμάτωσης (*integration issues*). Η ενσωμάτωση περιλαμβάνει την λύση τεχνικών και μη τεχνικών προβλημάτων των ομάδων μεταξύ τους. Τα events στο Nexus είναι τα ίδια με το Scrum με την επισημοποίηση του *Refinement*. Το *Product Backlog Refinement* είναι ένα επίσημο event που συμβαίνει κατά τη διάρκεια ενός Sprint στο Nexus, λόγω της αυξημένης πολυπλοκότητας των πολλαπλών ομάδων που εργάζονται από κοινού. Ο αριθμός και η διάρκεια αυτών των events ποικίλουν με βάση τις εξαρτήσεις που υπάρχουν στο Product Backlog.

Ο στόχος του Refinement είναι να αποσυνθέτει τα στοιχεία Product Backlog αρκετά έτσι ώστε, να είναι φανερό ποιες ομάδες μπορούν να τα υλοποιήσουν και με ποια σειρά και να εντοπίσει, να απεικονίσει και να ελαχιστοποιήσει τις εξαρτήσεις των ομάδων μεταξύ τους. Τα Refinement event συμβαίνουν όσο συχνά χρειάζεται κατά την διάρκεια ενός Sprint. Το Product Backlog πρέπει να είναι καθορισμένο έτσι ώστε να αναγνωρίζονται, να απομακρύνονται και να ελαχιστοποιούνται αρκετά οι εξαρτήσεις πριν το επόμενο Nexus Sprint Planning, διότι οι εξαρτήσεις γίνονται μεγαλύτερες και πιο απαιτητικές κατά την διάρκεια. Για την ελαχιστοποίηση των εξαρτήσεων , θα ήταν σημαντικό να εξετάσουμε τη δομή των ομάδων και την αρχιτεκτονική τους και να απεικονιστούν οι εξαρτήσεις των διαφορετικών ομάδων.

Το Nexus Sprint Planning είναι το γεγονός που γίνεται όταν το Nexus δημιουργεί ένα πλάνο για το επόμενο Sprint και όταν διατυπώνεται ο στόχος αυτού. Ο στόχος του Nexus Sprint είναι ο σκοπός όλων των Scrum ομάδων που ανήκουν στο Nexus και εργάζονται κατά την διάρκεια του Sprint. Το πρώτο μέρος του Nexus Sprint Planning είναι όταν ορισμένα άτομα από κάθε ομάδα κάνουν σύσκεψη μεταξύ τους ώστε να αναγνωριστούν, να

ελαχιστοποιηθούν και να λυθούν οι εξαρτήσεις του συγκεκριμένου Sprint. Κάθε Scrum ομάδα πρέπει να αντιπροσωπεύεται και να μπορεί να αλλάξει την σειρά των εργασιών της αν είναι αναγκαίο. Αφού τελειώσουν την σύσκεψη οι αντιπρόσωποι, τότε καθένας από αυτούς κάνει το Scrum Team Sprint Planning με την ομάδα του. Το Nexus event τελειώνει όταν όλες οι ομάδες έχουν πραγματοποιήσει το Scrum Sprint Planning τους.

Ένα τεχνούργημα που προκύπτει από το Nexus Sprint Planning είναι το Nexus Sprint Backlog. Πρόκειται για μια απεικόνιση των τυχόν εξαρτήσεις που θα υπάρχουν κατά τη διάρκεια του Sprint και ένας τρόπος διαχείρισης της ροής των εργασιών εντός του Nexus. Η Nexus Integration Team δημιουργεί και διαχειρίζεται αυτό το Backlog. Θα πρέπει να αναθεωρείται και να ενημερώνεται τουλάχιστον κάθε μέρα, ειδικά αν το Backlog χρησιμοποιείται για να διαχειριστεί και να απεικονίσει τη ροή της εργασίας για όλες τις ομάδες σε ένα Nexus. Η κατάλληλη στιγμή για να πραγματοποιηθεί η αξιολόγηση του Nexus Sprint Backlog είναι κατά την διάρκεια του Nexus Daily Scrum. Αυτό είναι μία σύντομη συνάντηση που γίνεται πριν από την ξεχωριστή Daily Scrum των ομάδων. Όπως και στον Nexus Sprint Planning, έτσι και στο Nexus Daily Scrum δεν χρειάζεται να παραβρίσκονται όλα τα μέλη αλλά μόνο οι αντιπρόσωποι κάθε ομάδας.

Το Nexus Sprint Review αντικαθιστά τα ξεχωριστά Scrum Sprint Reviews των ομάδων. Οι Scrum Teams του Nexus παραδίδουν την ανασκόπηση μαζί έτσι ώστε να μπορούν να παρουσιάσουν το ολοκληρωμένο ενσωματωμένο αποτέλεσμα ώστε να λάβουν γενικό feedback και όχι ξεχωριστά κομμάτια αυτού. Αυτό το γεγονός διασφαλίζει ότι οι σωστές πληροφορίες θα δοθούν στα σωστά μέλη των ομάδων. Όπως και στο Scrum, υπάρχει το επίσημο γεγονός το Nexus Sprint Retrospective, το οποίο γίνεται αμέσως μετά το Nexus Sprint Review, πριν το Nexus Sprint Planning και είναι χωρισμένο σε τρία μέρη. Στο πρώτο μέρος του οι αντιπρόσωποι κάθε ομάδας θα συναντηθούν ώστε να μοιραστούν μεταξύ του τις προκλήσεις που αντιμετωπίζουν. Το αποτέλεσμα αυτής της συνάντησης θα είναι το θέμα του δεύτερου μέρους που γίνεται εντός κάθε ομάδας ξεχωριστά. Μαζί με τις λύσεις των προβλημάτων από το πρώτο μέρος γίνονται και ξεχωριστές αναδρομές σε κάθε ομάδα. Το τρίτο και τελευταίο μέρος είναι όταν οι αντιπρόσωποι συναντιούνται ξανά ώστε να συζητήσουν τις ενέργειες που θα ληφθούν και τα μέσα για την αντιμετώπιση των προκλήσεων.

Ένα Nexus δουλεύει εντός των ορίων ενός Sprint, δηλαδή 30 ημέρες ή λιγότερες, όπως ισχύει και στο απλό Scrum. Αν είναι απαραίτητο οι διαφορετικές ομάδες μπορούν να δουλεύουν σε διαφορετικής διάρκεια Sprints, όμως πρέπει να υπάρχει ένας κοινός παρονομαστής. Οι ομάδες πρέπει να παραδώσουν τα αποτελέσματα τους στην ομάδα ενσωμάτωσης το αργότερο στο τέλος του δικού τους Sprint. Είναι εύκολο να ξεκινήσει κανείς με το Nexus διότι διατηρεί τις αρχές του Scrum και οι επιχειρήσεις είναι ήδη εξοικειωμένες με αυτό και επωφελούνται από την γνώση του. Όπως όμως το Scrum, έτσι και το Nexus δεν είναι αρκετό ώστε να εγγυηθεί επιτυχία, όμως βοηθάει στην επίλυση προβλημάτων και την συνεχής αναγνώριση και απομάκρυνση των εξαρτήσεων που δημιουργούνται από την πολυπλοκότητα της διαδικασίας. Ως αποτέλεσμα, οι επιχειρήσεις που χρησιμοποιούν το Nexus πρέπει να δημιουργήσουν και να προωθήσουν την τέλεια τεχνική ως βάση για την ανάπτυξη τους.

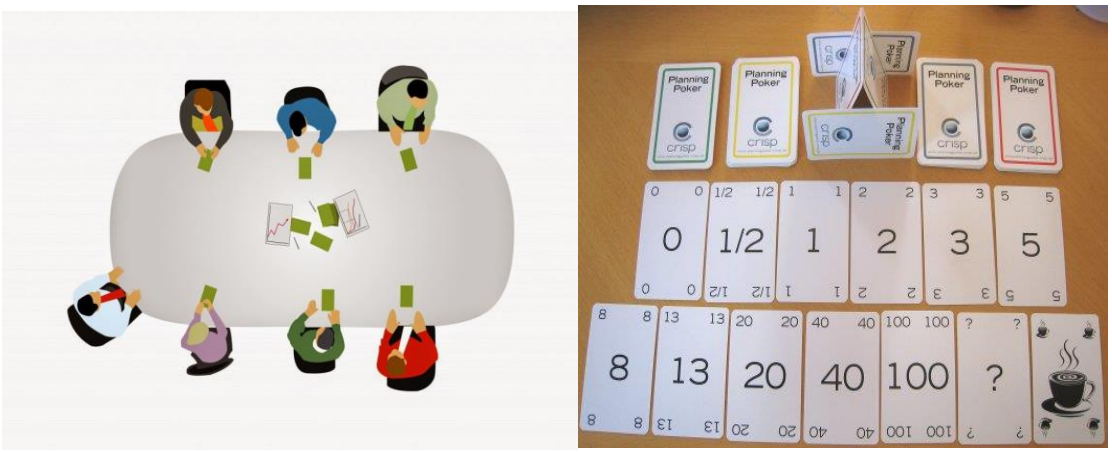
Για να ξεκινήσει το Nexus οι επιχειρήσεις πρέπει πρώτα να:

- Καθορίσουν τις Scrum ομάδες που θα περιέχει το Nexus.
- Να δημιουργήσουν μια αρχική Integration Team.
- Να έχουν ένα και μοναδικό Product Backlog.
- Να έχουν ξεκάθαρη την έννοια της λέξης ‘Ολοκληρωμένο’.
- Να καθορίσουν την διάρκεια και της ημερομηνίες του κάθε Sprint.

2.2.3 Planning Poker

Το *planning poker*, ή αλλιώς scrum poker, είναι μία μέθοδος η οποία χρησιμοποιείται ώστε να εκτιμηθεί το φόρτο ή το κόστος των πόρων και των στόχων. Επίσης μπορεί να χρησιμοποιηθεί και για να εκτιμηθεί η πιθανότητα και η ζημιά των ρίσκων που θα παρθούν. Στην *μη διαδικτυακή εκδοχή* του planning poker, για να ξεκινήσει μια εκτίμηση (*planning poker session*), όλα τα μέλη μαζεύονται σε ένα κοινό χώρο και ο ιδιοκτήτης του προϊόντος (*product owner*) ή ο πελάτης διαβάζει ένα user story ή περιγράφει ένα χαρακτηριστικό στα μέλη που θα κάνουν την εκτίμηση. Κάθε μέλος κρατά στα χέρια του ένα πακέτο από χαρτιά αριθμημένα με τις τιμές 0, 1, 2, 3, 5, 8, 13, 20, 40 και 100. Οι τιμές αντιπροσωπεύουν τους αριθμούς των *story points,* των *ideal days* και *άλλων μονάδων* που θα εκτιμήσουν τα μέλη της ομάδας.

Τα μέλη της ομάδας συζητάνε για τα χαρακτηριστικά ή τα user stories που τους ζητούνται να εκτιμήσουν, ρωτώντας απορίες των product owner. Μόλις λήξη η συζήτηση και έχουν λυθεί οι απορίες, κάθε μέλος κρυφά από τους υπόλοιπους επιλέγει μία κάρτα από το πακέτο η οποία αντιπροσωπεύει την δική του εκτίμηση στο θέμα και την τοποθετεί ανάποδα στο τραπέζι. Όλες οι κάρτες αποκαλύπτονται την ίδια στιγμή. Αν όλα τα μέλη έχουν επιλέξει την ίδια τιμή τότε αυτή η τιμή ορίζεται ως η εκτιμημένη τιμή. Αν όχι, τα μέλη συζητούν τις επιλογές τους. Εκείνος με την υψηλότερη και εκείνος με την χαμηλότερη τιμή πρέπει να αναφέρουν τους λόγους που την επέλεξαν και να την υποστηρίξουν με επιχειρήματα. Αφού τελειώσει η συζήτηση, ξεκινάει καινούργιος γύρος του planning poker. Η διαδικασία αυτή συνεχίζεται μέχρις ότου όλοι να καταλήξουν στην ίδια τιμή.



Εικόνα 2.2.3.1 Διαδικασία του planning poker και χαρτιά τιμών

Στην *διαδικτυακή εκδοχή* την οποία υλοποιήσαμε στην εργασία μας, ένας χρήστης στέλνει προσκλήσεις στα υπόλοιπα μέλη του project, ώστε να εκτιμηθεί κάποια σημείωση του καμβά (canvas note). Δίνεται η δυνατότατα επιλογής κάποιου αριθμού και των σημειώσεων. Οποιαδήποτε στιγμή το άτομο το οποίο ξεκίνησε το planning poker μπορεί να αποκαλύψει τους αριθμούς που επέλεξαν τα υπόλοιπα μέλη και να ξεκινήσει καινούργιο γύρο εκτίμησης ή να προσθέσει την τελική εκτίμηση σε κάποια σημείωση. Αυτός ο τρόπος δεν απαιτεί την φυσική παρουσία των προσώπων τα οποία δηλώνουν την επιλογή τους οποιαδήποτε στιγμή από όπου κι αν βρίσκονται. Επίσης, όπως και την μη διαδικτυακή εκδοχή, όταν οι επιλογές έχουν μεγάλες διαφορές μεταξύ τους απαιτείται από τα μέλη να εξηγήσουν τις επιλογές τους και να υποστηρίξουν τα επιχειρήματα τους. Επιπλέον υπολογίζεται και ο μέσος όρος των επιλογών.

3 Σχέδιο Δράσης

3.1 State of the Art

Ο KICKOFF καμβάς αρχικά σχεδιάστηκε ως αναλογικό εργαλείο. Οι εργαζόμενοι της UWS κάνουν ειδικά σεμινάρια στους πελάτες τους στα οποία χρησιμοποιούν την έντυπη έκδοση του καμβά, ώστε να συγκεντρωθούν σημαντικές πληροφορίες για την έναρξη ενός έργου. Αυτή την στιγμή υπάρχουν αρκετοί έντυποι καμβάδες, όπως οι *openPM Canvas, SmartPM Bluesheet, Project Square, Project Canvas* και *Project Canvas*. Όμως μόνο οι *Project Canvas, TUZZit* και *Canvanizer* δίνουν την δυνατότητα ενός ψηφιακού καμβά. Οι ψηφιακές εκδόσεις είναι πολύ απλές και δεν προσθέτουν καμία πραγματική αξία σε σύγκριση με το αναλογικό εργαλείο, εκτός από την άμεση δυνατότητα τεκμηρίωσης (*direct documentation*) και εξαγωγής (*export*). Ως εκ τούτου η αγορά είναι σχετικά νέα. Οι ψηφιακές εκδόσεις των καμβάδων δεν έχουν εισαχθεί στην αγορά, ακόμη κι αν υπάρχει ήδη μια μικρή ζήτηση αυτών. Ο KICKOFF καμβάς θέλει να είναι ο πρώτος project καμβάς που θα είναι μια πραγματική ευκαιρία για τους έντυπους καμβάδες, λόγω της απλής χρηστικότητας του, των πολλών επιπλέων χαρακτηριστικών και επειδή είναι βασισμένος στις αρχές της μεθόδου PRINCE2.

3.2 Τεχνολογίες πυρήνα της εφαρμογής

3.2.1 Bootstrap

Το Bootstrap είναι μια συλλογή εργαλείων ανοιχτού κώδικα, δηλαδή ελεύθερο λογισμικό, με σκοπό την δημιουργία ιστοσελίδων και διαδικτυακών εφαρμογών. Περιέχει συγκεκριμένα HTML elements και ιδιότητες της CSS για τις μορφές τυπογραφίας, κουμπιά πλοήγησης και άλλα, καθώς και επεκτάσεις της JavaScript. Το Bootstrap δημιουργήθηκε από το Twitter, το οποίο είναι μέσω κοινωνικής δικτύωσης, στα μέσα του 2010 και αρχικά ονομάστηκε Twitter Blueprint. Χρησίμευε ως οδηγός στυλ (style guide) για την ανάπτυξη των εσωτερικών εργαλείων στην εταιρεία του Twitter για πάνω από ένα χρόνο πριν από τη δημόσια κυκλοφορία του. Κυκλοφόρησε για πρώτη φορά το 2011 και από τότε έχει κάνει δύο μεγάλες ανανεώσεις, το Bootstrap 2 και το Bootstrap 3. Στο Bootstrap 3 συμπεριλαμβάνεται μία βιβλιοθήκη ώστε η ιστοσελίδα ή η web εφαρμογή να ανταποκρίνεται από προεπιλογή σε κινητές συσκευές, όπως είναι τα κινητά τηλέφωνα.

Το Bootstrap περιλαμβάνει ένα δυναμικό σύστημα πλέγματος με το οποίο δημιουργούνται τα σχεδιαγράμματα (layouts) για όλα τα σχήματα και τα μεγέθη. Είναι βασισμένο σε 12 στήλες και έχει πολλαπλές βαθμίδες, μία για κάθε είδος συσκευής (μικρές, μεσαίες, μεγάλες και πολύ μεγάλες συσκευές). Υπάρχουν τρία βασικά στοιχεία, το κέλυφος (*container*), τις γραμμές (*rows*) και τις στήλες (*columns*). Οι containers μπορούν να έχουν καθορισμένο μέγεθος (*fixed-width*), μέσω της css κλάσης `.container` ή ολόκληρης της οθόνης (*full-width*) μέσω της css κλάσης `.container-fluid`. Οι γραμμές είναι οριζόντιες ομάδες στηλών, ώστε το περιεχόμενο να εισάγεται σε αυτές και να είναι ευθυγραμμισμένο. Οι στήλες καλούνται μέσω CSS κλάσεων χρησιμοποιώντας τους 12 πιθανούς αριθμούς. Για παράδειγμα αν θέλαμε τρεις στήλες ίσες μεταξύ τους θα τους δίναμε τις κλάσεις `.col-xs-4 .col-sm-4 .col-md-4 .col-lg-4`.

Εικόνα 3.2.1.1 Παράδειγμα των containers του bootstrap

```
<div class="container">
  <div class="row">
    <div class=" col-xs-4 col-sm-4 col-md-4 col-lg-4">
      <p>First of the three equal columns</p>
    </div>
    <div class=" col-xs-4 col-sm-4 col-md-4 col-lg-4">
      <p>Second of the three equal columns</p>
    </div>
    <div class=" col-xs-4 col-sm-4 col-md-4 col-lg-4">
      <p>Third of the three equal columns</p>
    </div>
  </div>
</div>
```

3.2.2 Font Awesome

Το Font Awesome είναι μία γραμματοσειρά που αποτελείται από ένα σύνολο εικονιδίων βασισμένα σε CSS. Δημιουργήθηκε από τον Dave Gandy για την χρήση του στο Twitter Bootstrap και αργότερα ενσωματώθηκε στον ανοιχτό κώδικα του Bootstrap. Στο σύνολο της γραμματοσειράς περιέχονται 634 εικονογραφήσεις οι οποίες δεν απαιτούν JavaScript για να χρησιμοποιηθούν και λειτουργούν σε οποιοδήποτε Web framework. Τα εικονίδια του Font Awesome μπορούν να προσαρμόζουν στιγμιαία το μέγεθος τους, το χρώμα τους, την σκιά και οτιδήποτε αλλάζει από το CSS. Για να χρησιμοποιηθεί σε ένα αρχείο HTML συμπεριλαμβάνονται τα αρχεία αυτού στο `<head>` κάθε σελίδας που θέλουμε να χρησιμοποιηθούν και στην συνέχεια καλούνται μέσα από τον κώδικα οι κλάσεις που περιέχουν τα συγκεκριμένα εικονίδια. Για παράδειγμα,  `<i class="fa fa-camera-retro"></i>`.

3.2.3 jQuery

Η jQuery είναι μία γρήγορη και πλούσια σε χαρακτηριστικά βιβλιοθήκη της JavaScript, που υποστηρίζει πολλαπλούς φυλλομετρητές και είναι σχεδιασμένη για να απλοποιήσει το scripting της HTML στην πλευρά του πελάτη. Είναι ανοιχτού κώδικα και αυτή τη στιγμή είναι η δημοφιλέστερη βιβλιοθήκη της JavaScript. Η σύνταξη της είναι σχεδιασμένη έτσι ώστε να είναι ευκολότερη η πλοήγηση στο έγγραφο, η επιλογή των στοιχείων του DOM, η δημιουργία animations, η διαχείριση των γεγονότων και η ανάπτυξη εφαρμογών Ajax. Επίσης παρέχει τη δυνατότητα στους προγραμματιστές να δημιουργήσουν plug-ins πάνω

από τη βιβλιοθήκη. Αυτό επιτρέπει τη δημιουργία αφαιρέσεων για αλληλοεπιδράσεις και animations χαμηλού επιπέδου καθώς επίσης και τη δημιουργία προηγμένων εφέ. Αυτή η σπονδυλωτή προσέγγιση (modular approach) στην jQuery επιτρέπει τη δημιουργία δυναμικών ιστοσελίδων και web εφαρμογών.

Στον πυρήνα της, η jQuery είναι μία βιβλιοθήκη διαχείρισης και διάσχισης του DOM. Αυτό επιτυγχάνεται μέσα από τη μηχανή επιλογής που έχει, την *Sizzle.* Έτσι δημιουργήθηκε ένα νέο στυλ προγραμματισμού στο οποίο αναμιγνύονται αλγόριθμοι και δομές δεδομένων του DOM. Για παράδειγμα μπορεί να χρησιμοποιηθεί για την εύρεση όλων των *h1* στοιχείων και να αλλάξει μία ή περισσότερες ιδιότητες τους ή να τα κάνει να ανταποκριθούν σε ένα γεγονός όπως το πάτημα ενός κουμπιού. Επίσης βασικά χαρακτηριστικά της, είναι η δυνατότητα που έχει για διαχείριση των γεγονότων, η ανάλυση JSON που προσφέρει και η παροχή πληροφοριών σχετικές με τον φυλλομετρητή.

Τα πλεονεκτήματα που προσφέρει η χρήση της είναι ότι ενθαρρύνει τον διαχωρισμό της JavaScript από την HTML μέσα από τη δυνατότητα προσθήκης χειριστών γεγονότων (event handlers) στο DOM μέσω της JavaScript και όχι χρησιμοποιώντας τις ιδιότητες των στοιχείων της HTML για την κλήση JavaScript συναρτήσεων, προσφέρει σαφήνεια και συντομία καθώς έχει ως χαρακτηριστικά τις αλυσιδωτές συναρτήσεις και τη στενογραφία (shorthand) ονομάτων συναρτήσεων, εξαλείφει τα προβλήματα συμβατότητάς μεταξύ των φυλλομετρητών αφού χειρίζεται τις αντιφάσεις μεταξύ τους και παρέχει μία συνεπή διεπαφή που λειτουργεί σε όλους και προσφέρει επεκτασιμότητα καθώς μπορούν να δημιουργηθούν και να προστεθούν νέα γεγονότα, στοιχεία και μέθοδοι και να επαναχρησιμοποιηθούν σαν πρόσθετο. Για να μπορέσει να χρησιμοποιηθεί, το μόνο που χρειάζεται είναι η δήλωση της στο html αρχείο σε μία *script* ετικέτα (tag):

```
<script src="jquery.js"></script>
```

Για την χρήση της έχουμε δυο διαφορετικά στυλ:

- Τη συνάρτηση *$* που είναι μία factory μέθοδος για το jQuery αντικείμενο
- Τις συναρτήσεις προθέματος (prefixed) *$.* που είναι συναρτήσεις χρησιμότητάς (utility functions) και δεν ενεργούν στο jQuery αντικείμενο άμεσα

Η jQuery διαθέτει και ένα σετ από επιμελημένες (curated) διεπαφές χρήστη, εφέ, widgets και θέματα το οποίο είναι χτισμένο πάνω από εκείνη και ονομάζεται *jQuery UI*. Εκτός από την jQuery η jQuery UI είναι υλοποιημένη και με HTML και CSS. Πιο αναλυτικά μπορεί να προσφέρει αλληλοεπιδράσεις με τα στοιχεία του DOM όπως *draggable, droppable, resizable, selectable* και *sortable*. Για παράδειγμα `$("#draggable").draggable()`.Επίσης μερικά από τα σημαντικότερα widget που διαθέτει είναι το accordion, το autocomplete, η ενίσχυση της εμφάνισης των κουμπιών όπως η μετατροπή των radio κουμπιών σε push κουμπιά, η δημιουργία μενού και η δημιουργία sliders. Τέλος διαθέτει εφέ όπως είναι το animation των χρωμάτων, η ενεργοποίηση και απενεργοποίηση των εφέ και άλλα διάφορα εφέ όπως είναι τα *slide-down* και *fade-in.*

3.2.4 AngularJS

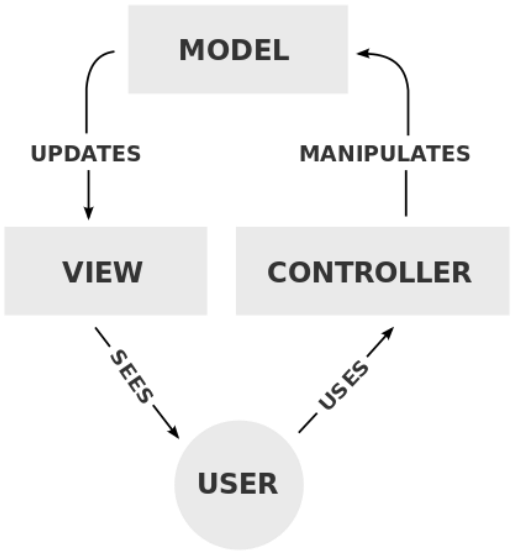
Το AngularJS είναι ένα ολοκληρωμένο, βασισμένο στην JavaScript, ανοιχτού κώδικα framework web εφαρμογής για πελάτες και εξυπηρετητές. Διατηρείται από την *Google* και άλλους ιδιώτες με σκοπό την διευθέτηση των πολλών προκλήσεων που αντιμετωπίζονται στην ανάπτυξη *single-page* εφαρμογών (single-page applications-SPAs). Σκοπός του είναι να απλοποιήσει την ανάπτυξη και τον έλεγχο (testing) τέτοιων εφαρμογών, παρέχοντας ένα framework για τις αρχιτεκτονικές *MVC* (Model-View-Controller) και *MVVM* (Model-View-ViewModel) στην πλευρά του πελάτη, μαζί με συστατικά που χρησιμοποιούνται κυρίως στις *Πλούσιες Διαδικτυακές Εφαρμογές* (Rich Internet Applications).

Είναι χτισμένο πάνω στην πεποίθηση ότι ο δηλωτικός προγραμματισμός (declarative programming) θα πρέπει να χρησιμοποιείται για τη δημιουργία διεπαφών και τη σύνδεση των συστατικών του λογισμικού, ενώ ο προστακτικός προγραμματισμός (imperative programming) ταιριάζει καλύτερα στον ορισμό της επιχειρησιακής λογικής (business logic), δηλαδή τον προσδιορισμό του πως τα δεδομένα μπορούν να δημιουργηθούν, να απεικονιστούν, να αποθηκευτούν και να αλλαχθούν. Το framework υιοθετεί και επεκτείνει την παραδοσιακή HTML για να παρουσιάσει δυναμικό περιεχόμενο μέσα από μία αμφίδρομη δέσμευση δεδομένων (two-way data-binding) που επιτρέπει τον αυτόματο συγχρονισμό των μοντέλων (models) και των όψεων (views). Συγκεκριμένα οι σχεδιαστικοί του στόχοι είναι:

- Η αποσύνδεση του χειρισμού του DOM από την λογική της εφαρμογής. Η δυσκολία αυτού επηρεάζεται δραματικά από τον τρόπο που είναι δομημένος ο κώδικας.
- Η αποσύνδεση της πλευράς του πελάτη της εφαρμογής από την πλευρά του εξυπηρετητή. Αυτό επιτρέπει την ανάπτυξη να προοδεύει παράλληλα και επιτρέπει την επαναχρησιμοποίηση και από τις δύο πλευρές.
- Η παροχή δομής για το χτίσιμο της εφαρμογής· από το σχεδιασμό της διεπαφής, μέχρι τη συγγραφή της επιχειρησιακής λογικής και τον έλεγχο.

Η λειτουργία του AngularJS ξεκινάει αρχικά διαβάζοντας την HTML σελίδα, η οποία έχει ενσωματωμένες μέσα της, επιπλέον ειδικά κατασκευασμένες ιδιότητες ετικετών (tag attributes). Το Angular διερμηνεύει αυτές τις ιδιότητες σαν *directives* για να δεσμεύσει εισαγόμενα και εξαγόμενα μέρη της σελίδας σε ένα μοντέλο το οποίο αναπαρίσταται από τυποποιημένες μεταβλητές της JavaScript. Οι τιμές αυτών των μεταβλητών μπορούν να αρχικοποιηθούν είτε χειροκίνητα μέσα στον κώδικα ή να αποκτηθούν από στατικούς ή δυναμικούς JSON πόρους. Λόγω της MVC αρχιτεκτονικής που υλοποιεί για τον διαχωρισμό της παρουσίασης, των δεδομένων και των λογικών συστατικών, το Angular φέρνει υπηρεσίες που παραδοσιακά γίνονται στον εξυπηρετητή, όπως ελεγκτές εξαρτώμενους από την όψη (view dependent controllers), στις web εφαρμογές στην πλευρά του πελάτη χρησιμοποιώντας *έγχυση εξάρτησης* (*dependency injection*). Ως αποτέλεσμα πολύς από τον φόρτο που υπάρχει στον εξυπηρετητή μπορεί να μειωθεί.

Η MVC αρχιτεκτονική που υλοποιεί το Angular χρησιμοποιείται κυρίως για την υλοποίηση διεπαφών. Παραδοσιακά χρησιμοποιούταν για desktop εφαρμογές όμως τα τελευταία χρόνια έγινε εξαιρετικά δημοφιλής στις web εφαρμογές. Μπορεί οι υλοποιήσεις τις να διαφέρουν αλλά η παραδοσιακή αρχιτεκτονική χωρίζει την εφαρμογή σε τρία διασυνδεδεμένα μέρη: το *μοντέλο* (model), που διαχειρίζεται τα δεδομένα, τη λογική και τους κανόνες της εφαρμογής, την *όψη* (view) η οποία μπορεί να είναι οποιαδήποτε αναπαράσταση πληροφοριών και είναι δυνατό να έχουμε πολλαπλές όψεις για την ίδια πληροφορία και τον *ελεγκτή* (controller) ο οποίος αποδέχεται εισόδους (inputs) και τις μετατρέπει σε εντολές για το μοντέλο ή την όψη. Πέρα από το διαχωρισμό της εφαρμογής σε τρία μέρη, αρχιτεκτονική ορίζει και αλληλοεπιδράσεις μεταξύ των μερών. Το μοντέλο αποθηκεύει δεδομένα τα οποία ανακτώνται σύμφωνα με τις εντολές από τον ελεγκτή και προβάλλονται στην όψη. Μία όψη παράγει νέο αποτέλεσμα στον χρήστη ανάλογα με τις αλλαγές που γίνονται στο μοντέλο. Τέλος, ο ελεγκτής μπορεί να στείλει εντολές είτε στο μοντέλο ώστε να αναβαθμίσει την κατάσταση του είτε στην συσχετισμένη του όψη για να αλλάξει την παρουσίαση του μοντέλου της.



Εικόνα 3.2.4.1 Η τυπική συνεργασία των συστατικών της MVC

Το AngularJS έχει κάποιες βασικές αρχές και λειτουργίες που βοηθούν στην ανάπτυξη των web εφαρμογών. Πέρα από την *αμφίδρομη δέσμευση δεδομένων* μία εφαρμογή που βασίζεται σε αυτό αποτελείται από έννοιες όπως τα *modules* (ενότητες), που είναι κάτι σαν «δοχείο» όπου εκεί μπαίνουν όλα τα διαφορετικά κομμάτια της εφαρμογής, τα *directives*, τα οποία λειτουργούν σαν δείκτες σε ένα στοιχείο του DOM και του προσδίδουν συμπεριφορά, τα *templates* (πρότυπα), που είναι στην ουσία html αρχεία τα οποία περιέχουν στοιχεία και ιδιότητες προσδιορισμένες από το Angular, τους *controllers* (ελεγκτές), οι οποίοι είναι υπεύθυνοι για τη διαχείριση των δεδομένων της εφαρμογής, τα *services*, τα οποία χρησιμοποιούνται για την οργάνωση και τον διαμοιρασμό του κώδικα σε όλη την εφαρμογή, τα *expressions* (εκφράσεις), τα οποία είναι σαν τα snippets (αποσπάσματα) της JavaScript που επιτρέπει στο Angular να διαβάζει και να γράφει μεταβλητές, και το *scope* (έκταση), που είναι ένα αντικείμενο το οποίο αναφέρεται στο μοντέλο της εφαρμογής, τα οποία αποτελούν βασικά συστατικά μιας εφαρμογής.

Όταν το Angular ξεκινάει την εφαρμογή, αναλύει και επεξεργάζεται τα directives χρησιμοποιώντας τον ενσωματωμένο HTML *compiler* που διαθέτει. Το φορτωμένο, μεταμορφωμένο και απεικονισμένο (rendered) DOM ονομάζεται *όψη*. Εκτός από τα directives, ο compiler επεξεργάζεται και τα *expressions* του Angular. Για παράδειγμα, αν συναντήσει {{3*5}} θα το αντικαταστήσει με την τιμή 15. Το Angular παρέχει ένα *scope* για τις μεταβλητές που είναι προσβάσιμες στα expressions. Οι τιμές που αποθηκεύονται στις μεταβλητές του scope αναφέρονται ως το *μοντέλο* στο υπόλοιπο έγγραφο.

Για την υλοποίηση της MVC το Angular χρησιμοποιεί τους controllers και τα services. Με τους controllers προστίθεται λογική στη διεπαφή και σκοπός τους είναι να εκθέσουν μεταβλητές και λειτουργικότητα στα expressions και τα directives. Με τα services μεταφέρουμε από τον controller την λογική που είναι ανεξάρτητη από την όψη ώστε να μπορεί να χρησιμοποιηθεί και από άλλα μέρη της εφαρμογής. Για την επικοινωνία μεταξύ τους χρησιμοποιείται έγχυση εξάρτησης (dependency injection) η οποία ασχολείται με το πώς τα αντικείμενα και οι συναρτήσεις δημιουργούνται και πως διατηρούν τις εξαρτήσεις τους. Τα πάντα στο Angular δημιουργούνται με βάση αυτή τη λογική και το «δοχείο» της dependency injection ονομάζεται *injector*.

Παρακάτω θα δούμε πιο αναλυτικά τα κύρια συστατικά μιας εφαρμογής που χρησιμοποιεί το AngularJS. Πρώτα από όλα θα εξετάσουμε την αμφίδρομη δέσμευση δεδομένων που προσφέρει, η οποία είναι ο αυτόματος συγχρονισμός δεδομένων μεταξύ του μοντέλου και των συστατικών της όψης. Ο τρόπος με τον οποίο υλοποιεί την δέσμευση δεδομένων,

επιτρέπει την αντιμετώπιση του μοντέλου σαν τη «μόνη πηγή αλήθειας στην εφαρμογή. Η όψη είναι μία προβολή του μοντέλου σε όλες τις στιγμές. Όταν το μοντέλο αλλάζει η όψη αντανακλά την αλλαγή και το αντίστροφο.

Στα περισσότερα συστήματα που λειτουργούν με *templates* η δέσμευση των δεδομένων γίνεται σε μία μόνο κατεύθυνση: ενώνουν τα συστατικά του template και του μοντέλου στην όψη. Μετά την ένωση, οι αλλαγές στο μοντέλο ή σε σχετιζόμενους τομείς της όψης δεν αντανακλώνται αυτόματα στην όψη και αντίστροφα ότι αλλαγές κάνει ο χρήστης στην όψη δεν αντανακλώνται στο μοντέλο. Αυτό έχει ως αποτέλεσμα να πρέπει συγχρονίζεται συνεχώς η όψη με το μοντέλο. Από την άλλη, τα templates του Angular λειτουργούν διαφορετικά. Αρχικά το template μεταγλωττίζεται (compiled) στον φυλλομετρητή. Το βήμα της μεταγλώττισης παράγει μία ζωντανή όψη στην οποία ότι αλλαγές γίνουν θα αντανακλαστούν αμέσως στο μοντέλο και το αντίστροφο. Επειδή η όψη είναι απλώς μία προβολή του μοντέλου ο ελεγκτής (controller) είναι τελείως αποκομμένος από αυτή και έχει άγνοια της ύπαρξης της. Αυτό κάνει τον έλεγχο (testing) του κώδικα ευκολότερο επειδή ελέγχεται ο controller απομονωμένα, χωρίς την όψη και την σχετιζόμενη DOM/φυλλομετρητής εξάρτηση.

Τα *templates* του Angular είναι γραμμένα σε HTML η οποία περιέχει συγκεκριμένα στοιχεία και ιδιότητες του. Εκείνο συνδυάζει το *template* με πληροφορίες από το μοντέλο και τον ελεγκτή ώστε να απεικονίσει (render) την δυναμική όψη που βλέπει ο χρήστης στον φυλλομετρητή. Τα στοιχεία και οι ιδιότητες του Angular που μπορούν να χρησιμοποιηθούν είναι τα *directives*, τα *expressions*, τα *φίλτρα* (*filters*) που χρησιμοποιούνται για την μορφοποίηση των δεδομένων για προβολή και *έλεγχοι φόρμας* (*form controls*) που ελέγχουν την ορθότητα αυτών που εισάγει ο χρήστης. Σε μια απλή εφαρμογή, ένα *template* αποτελείται από HTML, CSS, και *directives* του Angular και όλα αυτά περιέχονται συνήθως σε ένα απλό αρχείο *html*. Αντίθετα, σε μία πιο πολύπλοκη εφαρμογή όπου υπάρχουν διαφορετικές όψεις μέσα σε μία σελίδα χρησιμοποιούνται τμήματα (segments) του *template* τα οποία βρίσκονται σε διαφορετικά HTML αρχεία και μπορούν να φορτωθούν χρησιμοποιώντας το *ngView directive*.

Για να ξεκινήσει τη λειτουργία του, το Angular διαθέτει ένα πρόγραμμα εκκίνησης (bootstrapper). Πρώτα από όλα χρειάζεται να φορτωθεί το Angular χρησιμοποιώντας την ετικέτα *script*: `<script src="assets/js/angular.min.js"></script>`. Καλό είναι να είναι κατεβασμένο το Angular και να χρησιμοποιηθεί στην ιδιότητα *src* το κατεβασμένο αρχείο και όχι αυτό που είναι online καθώς θα δημιουργηθεί κενό ασφαλείας. Το Angular αρχικοποιείται αυτόματα είτε επάνω στο γεγονός *DOMContentLoaded* ή όταν το *script angular.js* αξιολογείται και εκείνη τη στιγμή το *document.readyState* είναι ορισμένο σε *complete*. Σε αυτό το σημείο το Angular ψάχνει για το *ngApp directive* το οποίο ορίζει τη ρίζα (root) της εφαρμογής. Αν βρεθεί το *directive* τότε το πρόγραμμα εκκίνησης φορτώνει το *module* που συσχετίζεται με το *directive*, δημιουργεί τον *injector* της εφαρμογής και μεταγλωττίζει το DOM μεταχειρίζοντας το *ngApp directive* σαν τη ρίζα της μεταγλώττισης. Αυτό μας επιτρέπει να του πούμε να μεταχειριστεί μόνο μια μερίδα του DOM σαν μια Angular εφαρμογή.

3.2.4.1 Scope

Μία από τις βασικότερες έννοιες στο AngularJS είναι το *scope* (έκταση). Όπως αναφέρθηκε το *scope* είναι ένα αντικείμενο που αναφέρεται στο μοντέλο της εφαρμογής. Είναι ένα περιεχόμενο εκτέλεσης για τα *expressions*. Τα *scopes* έχουν ιεραρχική δομή το οποίο μιμείται την DOM δομή της εφαρμογής. Μπορούν να παρακολουθούν τα *expressions* και να διαδίδουν γεγονότα είτε αναμεταδίδοντας τα στα παιδιά τους (broadcast) ή εκπέμποντας (emit) τα, τα παιδιά προς τους γονείς. Τα βασικά της χαρακτηριστικά είναι η παρακολούθηση των μεταλλάξεων του μοντέλου με τη χρήση της συνάρτησης $watch και

η διάδοση οποιασδήποτε αλλαγής του μοντέλου μέσω του συστήματος στην όψη έξω από το «βασίλειο του Angular» (ελεγκτές, services και χειριστές γεγονότων). Μπορούν να εμφωλευθούν ώστε να περιοριστεί η πρόσβαση στις ιδιότητες των συστατικών της εφαρμογής ενώ παρέχεται πρόσβαση στις κοινόχρηστες ιδιότητες του μοντέλου. Τα εμφωλευμένα *scopes* είναι είτε *παιδιά* (child) *scopes* ή *απομονωμένα* (isolated) *scopes*. Ένα παιδί *scope* κληρονομεί τις ιδιότητες από τον γονέα του ενώ ένα απομονωμένο *scope* όχι. Τέλος παρέχουν περιεχόμενο κατά το οποίο αξιολογούνται τα *expressions*. Για παράδειγμα, το {{username}} *expression* δεν έχει νόημα αν δεν αξιολογηθεί σε ένα συγκεκριμένο *scope* το οποίο ορίζει την ιδιότητα *username*.

Το *scope* είναι η κόλλα μεταξύ του ελεγκτή της εφαρμογής και της όψης. Κατά τη διάρκεια της φάσης σύνδεσης των *templates* (πρότυπα), τα *directives* εγκαθιδρύουν τα *$watch expressions* στο *scope*. Το *$watch* επιτρέπει στα *directives* να ειδοποιούνται για τις αλλαγές στις ιδιότητες, το οποίο επιτρέπει στο *directive* να απεικονίζει την ενημερωμένη τιμή στο DOM. Και οι ελεγκτές και τα *directives* έχουν αναφορά στο *scope* αλλά όχι μεταξύ τους. Αυτή η συμφωνία απομονώνει τον ελεγκτή από τα *directives* και το DOM και τον κάνει να φαίνεται αγνωστικιστής, αλλά βελτιώνει σε μεγάλο βαθμό την δοκιμή (testing) των εφαρμογών.

Κάθε Angular εφαρμογή έχει ακριβώς ένα *root* (ρίζα) *scope*, αλλά μπορεί να έχει πολλά *παιδιά scopes*. Η εφαρμογή μπορεί να έχει πολλαπλά *scopes* γιατί κάποια *directives* δημιουργούν *παιδιά scopes*. Όταν νέα *scopes* δημιουργούνται, προστίθενται ως παιδιά του γονικού τους *scope*, το οποίο δημιουργεί μία δεντρική δομή η οποία παραλληλίζεται με το DOM στο οποίο είναι προσκολλημένες. Όταν το Angular αξιολογεί ένα *expression* π.χ. {{name}}, πρώτα ψάχνει στο *scope* που συσχετίζεται με δοθέν στοιχείο (element) για την ιδιότητα name. Αν δεν βρεθεί τέτοια ιδιότητα, ψάχνει στο γονικό *scope* και συνεχίζει μέχρι να φτάσει στο *root scope*.

Η φυσιολογική ροή ενός φυλλομετρητή που δέχεται ένα γεγονός, είναι να εκτελέσει μία αντίστοιχη callback (επανάκληση) JavaScript. Με τον όρο γεγονός αναφερόμαστε είτε σε μία αλληλοεπίδραση του χρήστη, είτε σε ένα γεγονός *timer* ή σε ένα γεγονός του δικτύου όπως είναι για παράδειγμα η απάντηση του εξυπηρετητή. Όταν ολοκληρωθεί η callback ο φυλλομετρητής επαναπεικονίζει το DOM και επιστρέφει στο να περιμένει περισσότερα γεγονότα. Όταν ο φυλλομετρητής καλεί την JavaScript, ο κώδικας εκτελείται έξω από το περιεχόμενο εκτέλεσης του Angular το οποίο σημαίνει ότι το Angular δεν γνωρίζει για τις τροποποιήσεις του μοντέλου.

Για τη σωστή κατεργασία των τροποποιήσεων του μοντέλου η εκτέλεση χρησιμοποιεί τη μέθοδο *$apply* για να εισαχθεί στο περιεχόμενο εκτέλεσης του Angular. Μόνο οι τροποποιήσεις μοντέλου που εκτελούνται μέσα στη μέθοδο *$apply* θα υπολογιστούν κατάλληλα από το Angular. Για παράδειγμα αν ένα *directive* ακούει DOM γεγονότα όπως το *ng-click*, πρέπει να αξιολογήσει το *expression* μέσα στην μέθοδο *$apply*. Μετά την αξιολόγηση του *expression*, η μέθοδος *$apply* εκτελεί ένα *$digest*. Στην φάση *$digest* το *scope* εξετάζει όλα τα *expressions* του *$watch* και τα συγκρίνει με την προηγούμενη τιμή τους. Αυτός ο έλεγχος γίνεται ασύγχρονα που σημαίνει ότι το `$scope.username="Mitch"` δεν θα προκαλέσει αμέσως ένα *$watch* να ειδοποιηθεί, αλλά αντίθετα η *$watch* ειδοποίηση καθυστερεί μέχρι την *$digest* φάση. Αυτή η καθυστέρηση είναι επιθυμητή αφού ενσωματώνει πολλαπλές ενημερώσεις του μοντέλου σε μία *$watch* ειδοποίηση όπως επίσης εγγυάται ότι κατά τη διάρκεια της *$watch* ειδοποίησης κανένα άλλο *$watch* δεν τρέχει. Αν ένα *$watch* αλλάξει την τιμή του μοντέλου, θα εξαναγκάσει έναν επιπλέον κύκλο *$digest*.

Συμπεραίνουμε ότι ο κύκλος ζωής ενός *scope* περιλαμβάνει:

1. *Δημιουργία.* Το *root scope* δημιουργείται κατά την εκκίνηση της εφαρμογής από το *$injector*. Κατά τη διάρκεια της σύνδεσης των *templates* κάποια *directives* δημιουργούν νέα *παιδιά scope*.
2. *Εγγραφή παρατηρητή* (*watcher registration*). Κατά τη διάρκεια σύνδεσης των *templates*, τα *directives* εγγράφουν *παρατηρητές* (watches) στο *scope* τους. Αυτοί οι παρατηρητές θα χρησιμοποιηθούν για να διαδώσουν τις τιμές του μοντέλου στο DOM.
3. *Μετάλλαξη του μοντέλου* (*model mutation*). Για να παρατηρηθούν κατάλληλα οι μεταλλάξεις, θα πρέπει να γίνονται μέσα στο *scope.$apply()*, το οποίο το κάνει το Angular σιωπηρά χωρίς την ανάγκη επιπλέον κλήσης του *$apply* για σύγχρονη ή ασύγχρονη δουλειά.
4. *Παρατήρηση μετάλλαξης* (*mutation observation*). Στο τέλος του *$apply*, το Angular πραγματοποιεί έναν *$digest* κύκλο στο *root scope*, ο οποίος στη συνέχεια διαδίδεται και σε όλα τα *παιδιά scope*. Κατά τη διάρκεια του κύκλου όλα τα *expressions* ή οι συναρτήσεις που έχουν γίνει *$watch*, ελέγχονται για μετάλλαξη του μοντέλου και αν ανιχνευθεί μετάλλαξη ο *$watch* listener καλείται.
5. *Καταστροφή του Scope* (*scope destruction*). Όταν τα *παιδιά scopes* δεν χρειάζονται πλέον, είναι ευθύνη του δημιουργού του *παιδιού scope* να το καταστρέψει μέσω του *scope.$destroy()*. Αυτό θα σταματήσει τη διάδοση των *$digest* κλήσεων στα *παιδιά scope* και θα επιτρέψει τη μνήμη που χρησιμοποιείται από αυτά αξιοποιηθεί από τον *garbage collector*.

3.2.4.2 Expressions

Όπως αναφέρθηκε τα *expressions* του Angular είναι σαν τα snippets (αποσπάσματα) της JavaScript που τοποθετούνται ανάμεσα σε {{}} όπως για παράδειγμα {{*member.firstName*}}. Είναι σαν τα JavaScript expressions με τις εξής διαφορές:

- *Περιεχόμενο*. Τα JavaScript *expressions* αξιολογούνται κατά την καθολική (global) μεταβλητή *window*, ενώ του Angular κατά το αντικείμενο *scope* χρησιμοποιώντας το *$parse* service για την κατεργασία των *expressions* και το *$window* service για την πρόσβαση σε καθολικές μεταβλητές.
- *Συγχώρεση* (*Forgiving*). Στην JavaScript, η προσπάθεια αξιολόγησης απροσδιόριστων (undefined) ιδιοτήτων παράγει τα σφάλματα *ReferenceError* και *TypeError*. Στο Angular η αξιολόγηση των *expressions* συγχωρεί σε *undefined* και *null*.
- *Φίλτρα*. Μπορούν να χρησιμοποιηθούν φίλτρα μέσα σε *expressions* ώστε να μορφοποιηθούν τα δεδομένα προτού προβληθούν.
- *Δεν υπάρχουν δηλώσεις ελέγχου ροής* (*No Control Flow Statements*). Σε ένα Angular *expression* δεν μπορούν να χρησιμοποιηθούν if συνθήκες, βρόγχοι επανάληψης και εξαιρέσεις (exceptions).
- *Δεν υπάρχουν δηλώσεις συναρτήσεων*. Δεν μπορούν να δηλωθούν συναρτήσεις μέσα σε ένα *expression* ακόμα και μέσα στο *ng-init* directive.
- *Δεν δημιουργούνται RegExp με κυριολεκτική σημειογραφία* (*No RegExp With Literal Notation*). Δεν μπορούν να δημιουργηθούν *regular expressions* μέσα σε ένα Angular *expression*.
- *Δεν μπορεί να δημιουργηθεί νέο αντικείμενο χρησιμοποιώντας τον τελεστή new*. Δεν μπορεί να χρησιμοποιηθεί ο τελεστής *new* σε ένα Angular *expression*.
- *Δεν υπάρχουν Bitwise, κόμμα και κενοί (void) τελεστές*. Δεν μπορούν να χρησιμοποιηθούν *Bitwise*, κόμμα (,) και κενοί τελεστές μέσα σε ένα Angular *expression*.

3.2.4.3 Ελεγκτές (Controllers)

Στο AngularJS οι *ελεγκτές* αποτελούν σημαντικό στοιχείο και είναι υπεύθυνοι για τη διαχείριση των δεδομένων της εφαρμογής. Ένας *ελεγκτής*, ορίζεται από μία JavaScript συνάρτηση κατασκευαστή (constructor function) η οποία χρησιμοποιείται για να αυξήσει το Angular *scope*. Όταν ένας *ελεγκτής* συνδέεται στο DOM μέσω του *ng-controller directive*, το Angular θα δημιουργήσει ένα στιγμιότυπο ενός *Controller* αντικειμένου χρησιμοποιώντας την καθορισμένη συνάρτηση κατασκευαστή του *ελεγκτή*. Ένα *παιδί scope* θα δημιουργηθεί και θα είναι διαθέσιμο σαν μία ενέσιμη (injectable) παράμετρος στην συνάρτηση κατασκευαστή του *ελεγκτή* ως *$scope*.

Οι ελεγκτές χρησιμοποιούνται για την εγκαθίδρυση της αρχικής κατάστασης του *$scope* αντικειμένου και την προσθήκη συμπεριφοράς στο *$scope* αντικείμενο. Επίσης αλληλοεπιδρούν με τα *scopes* χρησιμοποιώντας τα για να εκθέσουν τις μεθόδους τους στα *templates*, ορίζουν μεθόδους (συμπεριφορά) που μπορούν να μεταλλάξουν το μοντέλο και μπορούν να εγγράψουν *παρατηρητές* (*watches*) στο μοντέλο, οι οποίοι εκτελούνται αμέσως μετά την εκτέλεση της συμπεριφοράς του *ελεγκτή*. Αντίθετα οι *ελεγκτές* δεν θα πρέπει να χρησιμοποιούνται για τον χειρισμό του DOM (πρέπει να περιέχουν μόνο τη *business logic*), τη μορφοποίηση της εισόδου (δεδομένα που εισάγει ο χρήστης), το φιλτράρισμα της εξόδου, τον διαμοιρασμό κώδικα ή κατάστασης ανάμεσα σε *ελεγκτές* (χρησιμοποιούνται τα *services* για αυτό) και τη διαχείριση του κύκλου ζωής άλλων συστατικών όπως για παράδειγμα το να δημιουργούν στιγμιότυπα των *services*.

Τυπικά, κατά τη δημιουργία μιας εφαρμογής, χρειάζεται η εγκαθίδρυση της αρχικής κατάστασης για το Angular *$scope*. Αυτό γίνεται με την επισύναψη ιδιοτήτων (properties) στο *$scope* αντικείμενο. Οι ιδιότητες περιέχουν το *μοντέλο όψη* (*view model*) το οποίο είναι το μοντέλο που θα παρουσιαστεί από την όψη. Όλες οι ιδιότητες του *$scope* θα είναι διαθέσιμες στο *template* στο σημείο του DOM όπου ο ελεγκτής είναι γραμμένος. Για παράδειγμα:

```
<nav class="navbar navbar-inverse navbar-fixed-top" ng-
controller="NavbarController">

...

</nav>

<div id="canvas" ng-controller="CanvasController" ng-
class="{'hide-all-icons': hideIcons == true}">

...

</div>

.controller('NavbarController',function($scope, $rootScope,
CanvasService, $translate, $interval, $uibModal,$timeout){

$scope.userToBeNotified;

)}
```

Στο παράδειγμα βλέπουμε ότι έχουμε δύο διαφορετικούς *ελεγκτές*: τον *NavbarController* και τον *CanvasController*. Κάθε ένας έχει το δικό του *$scope* με τις δικές του ιδιότητες και ο ένας είναι γραμμένος στο στοιχείο *nav* και ο άλλος στο *div*. Επίσης βλέπουμε την εγκαθίδρυση της αρχικής κατάστασης του *$scope* του *NavbarController* χρησιμοποιώντας την μέθοδο *controller()* καθώς και την επισύναψη της ιδιότητας *userToBeNotified* στο *$scope*.

Πέρα από την αρχικοποίηση της κατάστασης του *$scope*, προκειμένου να γίνεται αντίδραση σε γεγονότα ή να γίνονται υπολογισμοί στην όψη πρέπει να προστεθεί συμπεριφορά στο *scope*. Αυτό επιτυγχάνεται με την επισύναψη μεθόδων στο αντικείμενο *$scope*, οι οποίες στη συνέχεια είναι διαθέσιμες για να κληθούν στην όψη. Για παράδειγμα:

Η προσθήκη μίας μεθόδου σε έναν *ελεγκτή* με όνομα *AddUserToCanvasController*:

```
$scope.addNewMemberToCanvas = function(members){…}
```

Η κλήση της μεθόδου στην όψη:

```
<div class="row">

  <button type="submit" id="sendInvite"  class="button-canvas white
light rounded-5"                                                ng-
click=" addNewMemberToCanvas(members)">Add</button>

 </div>
```

Τέλος, είναι συνηθισμένο να συνδέονται *ελεγκτές* σε διαφορετικά επίπεδα της ιεραρχίας του DOM. Αφού το *ng-controller directive* δημιουργεί ένα νέο *παιδί scope* έχουμε μία ιεραρχία από *scopes* που κληρονομούν το ένα από το άλλο. Το *$scope* που λαμβάνει κάθε *ελεγκτής* θα έχει πρόσβαση στις ιδιότητες και τις μεθόδους που είναι ορισμένες από *ελεγκτές* οι οποίοι είναι ψηλότερα στην ιεραρχία. Στο παράδειγμα που παρατίθεται παραπάνω με τους δύο *ελεγκτές*, δεν έχουμε κληρονομικότητα καθώς κανένας *ελεγκτής* δεν βρίσκεται μέσα στον άλλο στο DOM (το στοιχείο *div* δεν βρίσκεται μέσα στο *nav* και αντίστροφα), οπότε και κανείς τους δεν έχει πρόσβαση στις ιδιότητες και τις μεθόδους του άλλου.

3.2.4.4 Directives

Τα *directives* αποτελούν επίσης σημαντικό συστατικό στο AngularJS. Πριν γίνει αναφορά σε αυτά, θα εξεταστεί ο *HTML μεταγλωττιστής* του Angular ο οποίος συνδέεται με τα *directives*. Ο *HTML μεταγλωττιστής* επιτρέπει να διδάξουμε στον φυλλομετρητή νέα HTML σύνταξη. Επιτρέπει την επισύναψη συμπεριφοράς σε οποιοδήποτε στοιχείο ή ιδιότητα ή ακόμη και τη δημιουργία νέων HTML στοιχείων ή ιδιοτήτων με προσαρμοσμένη (custom) συμπεριφορά. Αυτές οι επεκτάσεις συμπεριφοράς ονομάζονται *directives*. Επειδή η HTML είναι δηλωτική γλώσσα μας επιτρέπει να κάνουμε συγκεκριμένα πράγματα όπως το να κεντράρουμε ένα στοιχείο στο 1/2 του παραθύρου του φυλλομετρητή. Με τα *directives* την επεκτείνουμε και μπορούμε να ευθυγραμμίσουμε για παράδειγμα ένα στοιχείο στο 1/3 της θέσης.

Τα περισσότερα συστήματα που λειτουργούν με *templates*, καταναλώνουν ένα στατικό *string template* και το συνδυάζουν με δεδομένα, καταλήγοντας σε ένα νέο *string*. Το κείμενο που προκύπτει γίνεται *innerHTML* σε ένα στοιχείο. Αυτό σημαίνει ότι οποιεσδήποτε αλλαγές γίνουν στα δεδομένα θα πρέπει να επανενωθούν με το *template* και να γίνουν *innerHTML* στο DOM. Αυτή η προσέγγιση έχει κάποια θέματα όπως το διάβασμα εισόδου του χρήστη και την ένωση της με δεδομένα, την αντικατάσταση των δεδομένων του χρήστη, τη διαχείριση της όλης διαδικασίας αναβάθμισης και την έλλειψη συμπεριφοράς εκφραστικότητας (behavior expressiveness).

Αντίθετα, το Angular είναι διαφορετικό. Ο *μεταγλωττιστής* του καταναλώνει το DOM, όχι *string templates*. Το αποτέλεσμα είναι μία συνάρτηση σύνδεσης η οποία όταν συνδυάζεται με ένα *scope* μοντέλου παράγεται η ζωντανή όψη. Οι δεσμοί της όψης και του *scope* του μοντέλου είναι διάφανοι (transparent). Δεν χρειάζεται κάποια ιδιαίτερη κλήση για την αναβάθμιση της όψης. Επειδή το *innerHTML* δεν χρησιμοποιείται, δεν θα «χτυπηθούν

(clobber)» τα δεδομένα του χρήστη ενώ τα *directives* του Angular μπορούν να περιέχουν εκτός από «δέστρες (bindings) κειμένου» και κατασκευές συμπεριφοράς. Αυτή η προσέγγιση του Angular παράγει ένα σταθερό DOM. Η δέσμευση του στιγμιότυπου του DOM στοιχείου σε ένα αντικείμενο μοντέλου δεν αλλάζει για όλη τη διάρκεια της δέσμευσης. Αυτό σημαίνει ότι ο κώδικας μπορεί να πάρει στα χέρια του τα στοιχεία και να εγγράψει χειριστές γεγονότων και να ξέρει ότι η αναφορά (reference) δεν θα καταστραφεί από τη συγχώνευση δεδομένων του *template*.

Όπως αναφέρθηκε το Angular λειτουργεί σε DOM κόμβους και όχι σε strings. Πιο αναλυτικά, η μεταγλώττιση συμβαίνει σε τρεις φάσεις:

1. Το *$compile service* διασχίζει το DOM και ταιριάζει τα *directives*. Αν ο μεταγλωττιστής βρει ότι ένα στοιχείο ταιριάζει με ένα *directive* τότε το *directive* προστίθεται στη λίστα με τα *directives* που ταιριάζουν με το DOM στοιχείο. Ένα μονό στοιχείο μπορεί να έχει πολλαπλά *directives*.
2. Όταν όλα τα *directives* που ταιριάζουν ένα DOM στοιχείο αναγνωριστούν, ο μεταγλωττιστής τα ταξινομεί με βάση την προτεραιότητα τους. Η συνάρτηση *compile* του κάθε *directive* εκτελείται. Κάθε συνάρτηση *compile* έχει την ευκαιρία να τροποποιήσει το DOM και παράγει και επιστρέφει μία συνάρτηση σύνδεσης (link function). Αυτές οι συναρτήσεις σύνδεσης συντίθενται σε μία «συνδυασμένη» συνάρτηση σύνδεσης, η οποία επικαλείται την επιστρεφόμενη συνάρτηση σύνδεσης του κάθε *directive*.
3. Το *$compile* συνδέει το *template* με το *scope* καλώντας τη συνδυασμένη συνάρτηση σύνδεσης από το προηγούμενο βήμα. Αυτή με τη σειρά της θα καλέσει τη συνάρτηση σύνδεσης των ατομικών *directives*, εγγράφοντας listeners στα στοιχεία και εγκαθιδρύοντας *$watch*'s με το *scope* όπως κάθε *directive* είναι ρυθμισμένο να κάνει.

Το αποτέλεσμα είναι ένας ζωντανός δεσμός (binding) μεταξύ του *scope* και του DOM. Έτσι σε αυτό το σημείο, μία αλλαγή στο μοντέλο στο μεταγλωττισμένο *scope* θα αντανακλαστεί στο DOM.

Ο λόγος που χρειαζόμαστε μία φάση μεταγλώττισης και μία σύνδεσης είναι γιατί ο χωρισμός της μεταγλώττισης και της σύνδεσης χρειάζεται κάθε φορά που μία αλλαγή στο μοντέλο προκαλεί αλλαγή στην δομή του DOM. Είναι σπάνιο για τα *directives* να έχουν μία συνάρτηση μεταγλώττισης καθώς τα περισσότερα ενδιαφέρονται να δουλεύουν με ένα συγκεκριμένο στιγμιότυπο DOM στοιχείου παρά να αλλάζουν την όλη του δομή. Πιο συχνά, τα *directives* έχουν μία συνάρτηση σύνδεσης η οποία επιτρέπει στο *directive* να εγγράφει listeners σε συγκεκριμένα κλωνοποιημένα στιγμιότυπα DOM στοιχείου, όπως επίσης και να αντιγράφει περιεχόμενο στο DOM από το *scope*.

Αφού αναλύθηκε ο τρόπος λειτουργίας του μεταγλωττιστή και η σχέση του με τα *directives*, θα εξεταστούν με περισσότερη λεπτομέρεια τα *directives*. Όπως ειπώθηκε, τα *directives* είναι δείκτες (markers) σε ένα DOM στοιχείο όπως για παράδειγμα μία ιδιότητα, ένα όνομα στοιχείου, ένα σχόλιο ή μία CSS κλάση, που λένε στον HTML μεταγλωττιστή να επισυνάψει μία συγκεκριμένη συμπεριφορά σε αυτό το DOM στοιχείο ή ακόμη και να μεταμορφώσει αυτό και τα παιδιά του. Το Angular έχει ήδη έτοιμα *directives* όπως *ng-app, ng-controller,ng-bind,ng-model* και άλλα, όμως δίνει τη δυνατότητα να δημιουργηθούν και προσαρμοσμένα (custom).

Ένα στοιχείο ταιριάζει με ένα *directive* όταν το *directive* είναι μέρος της δήλωσης του. Για παράδειγμα: στο `<input ng-model="foo">` το *input* ταιριάζει με το *ng-model directive*. Το Angular κανονικοποιεί το όνομα της ετικέτας και της ιδιότητας ενός στοιχείου για να καθορίσει ποια *directives* ταιριάζουν με ποια στοιχεία. Συνήθως αναφερόμαστε σε αυτά

γράφοντας το κανονικοποιημένο όνομα τους ενωμένο με πεζά κεφαλαία (camelCase) (π.χ. ngModel). Επειδή όμως η HTML είναι case-insensitive αναφερόμαστε στα *directives* στο DOM με παύλες (-), κάτω παύλες (_) και άνω κάτω τελείες (:) όπως για παράδειγμα *ng-model* ή *ng:model* ή *ng_model*. Κατά τη διάρκεια της κανονικοποίησης αφαιρούνται τα *x-* και *data-* μπροστά από τα στοιχεία και τις ιδιότητες και στη συνέχεια μετατρέπονται τα (:), (-) και (_) οριοθετώντας το όνομα σε camelCase. Συνοψίζοντας, υπάρχουν τέσσερις διαφορετικοί τύποι *directives* που μπορεί να ταιριάξει το *$compile*: βασισμένο στα ονόματα στοιχείων (π.χ. <my-dir></my-dir>), βασισμένο στις ιδιότητες (π.χ.), βασισμένο στα ονόματα κλάσεων (π.χ.) και βασισμένο στα σχόλια (π.χ. <!-- directive: my-dir exp -->).

Πολύ χρήσιμο στοιχείο είναι η δυνατότητα δημιουργίας δικών μας *directives*. Όπως και οι *ελεγκτές* έτσι και τα *directives* εγγράφονται σε *modules*. Για την εγγραφή ενός *directive* χρησιμοποιείται το *module.directive* API το οποίο παίρνει το κανονικοποιημένο όνομα του *directive* ακολουθούμενο από μία *factory* συνάρτηση. Αυτή η συνάρτηση θα πρέπει να επιστρέψει ένα αντικείμενο με τις διαφορετικές επιλογές ώστε να πει στο *$compile* πως πρέπει να συμπεριφέρεται το *directive* όταν ταιριαστεί. Η *factory* συνάρτηση επικαλείται μόνο όταν ο μεταγλωττιστής ταιριάζει το *directive* για πρώτη φορά. Για την επίκληση της χρησιμοποιείται το *$injector.invoke* το οποίο την κάνει injectable όπως ακριβώς έναν ελεγκτή.

Εξετάζουμε το παρακάτω παράδειγμα:

```
angular.module('canvas-grid',
['services','ui.bootstrap','ui.bootstrap.contextMenu', 'filters'])

    .directive('canvasGrid', function() {

        return {

            require: '^canvasSection',

            restrict: 'E',

            transclude: true,

            scope: { grid: '='},

            controller: function($scope, $element){

                var grid = $scope.grid;

            },

            link:       function(scope,       element,       attrs,
canvasSectionController, CanvasService) {

            },

            templateUrl: 'app/templates/canvas/canvas-grid.html',

            replace: true

        }

    });
```

Στο παράδειγμα αυτό έχουμε δημιουργήσει ένα custom *directive* με όνομα *canvasGrid*. Βλέπουμε ότι το *directive* είναι εγγεγραμμένο σε ένα *module* με όνομα *canvas-grid*. Για τη δημιουργία του *directive* χρησιμοποιείται το *module.directive* API. Μέσα στο *directive()*

ορίζεται το όνομα και μία συνάρτηση που επιστρέφει ορισμένα πεδία. Το πρώτο πεδίο, το *require* δηλώνει πως απαιτείται ένας *ελεγκτής* αλλιώς το *$compile* θα πετάξει ένα σφάλμα. Στη συγκεκριμένη περίπτωση χρειάζεται ο *ελεγκτής* με όνομα *canvasSection* ο οποίος είναι ελεγκτής ενός *directive*. Το σύμβολο ^ μπροστά από το όνομα του *ελεγκτή* δηλώνει στο *directive* να ψάξει για τον *ελεγκτή* στο δικό του στοιχείο ή στων γονέων του ενώ τα ^^ δηλώνουν να ψάξει μόνο στους γονείς του. Τέλος αν δεν υπάρχει σύμβολο πριν το όνομα του *ελεγκτή* το *directive* θα ψάξει μόνο στο δικό του στοιχείο.

Το πεδίο *restrict* δηλώνει έναν περιορισμό που βάζουμε στο *directive* και μπορεί να πάρει τις τιμές 'A', 'E', 'C'και 'M'. Το *A* (attribute) ταιριάζει μόνο με όνομα ιδιότητας (π.χ.), το *E* (element) με όνομα στοιχείου (π.χ. <my-dir></my-dir>), το *C* (class) με όνομα κλάσης (π.χ.) και το *M* (comment) με σχόλιο (π.χ. <!-- directive: my-dir exp -->). Μπορεί όμως να υπάρξει και συνδυασμός των περιορισμών όπως για παράδειγμα *AEC* το οποίο ταιριάζει είτε με ιδιότητα είτε με στοιχείο ή με όνομα κλάσης.

Το πεδίο *transclude* έχει τεθεί στην τιμή *true*. Αυτό σημαίνει ότι τα περιεχόμενα του *directive* θα έχουν πρόσβαση στο *scope* που είναι έξω από αυτό παρά μέσα. Η επιλογή αυτή αλλάζει τον τρόπο με τον οποίο είναι εμφωλευμένα τα *scopes*. Με λίγα λόγια δίνει στα περιεχόμενα του πρόσβαση στο εξωτερικό *scope*. Αυτό συνήθως γίνεται όταν ένα *directive* τυλίγει κάποιο περιεχόμενο γιατί διαφορετικά θα έπρεπε να περνιέται στο *directive* κάθε *μοντέλο* που θέλουμε να χρησιμοποιήσουμε ξεχωριστά.

Το πεδίο *scope* χρησιμοποιείται για να φτιάξουμε ένα ξεχωριστό *scope* μέσα στο *directive*, το οποίο ονομάζεται *απομονωμένο* (*isolate*) *scope*. Αυτό που θέλουμε να κάνουμε ουσιαστικά είναι να απομονώσουμε το *scope* μέσα σε ένα *directive* από το εξωτερικό *scope* και στη συνέχεια να χαρτογραφήσουμε το εξωτερικό *scope* σε ένα εσωτερικό του directive. Ο λόγος που θέλουμε να φτιάξουμε ένα *απομονωμένο scope* είναι για να μπορούμε να φτιάξουμε *directives* τα οποία θα μπορούν να επαναχρησιμοποιηθούν καθώς με αυτόν τον τρόπο εμποδίζεται ένα συστατικό (component) από το να αλλάξει την κατάσταση του μοντέλου εκτός από τα μοντέλα που έχουν περαστεί μέσα ρητώς. Όπως προτείνει και το όνομα του, το *απομονωμένο scope* δεν κληρονομεί σε επίπεδο πρωτοτύπου από τον γονέα του όπως κάνει ένα κανονικό *scope*. Αντίθετα απομονώνει τα πάντα εκτός από τα *μοντέλα* που έχουν προστεθεί ρητώς στο *scope:{}* αντικείμενο.

Το αντικείμενο *scope{}* περιέχει μία ιδιότητα για κάθε δεσμό (binding) *απομονωμένου scope*. Στο παράδειγμα μας έχει μόνο μία ιδιότητα με όνομα *grid*. Το όνομα ανταποκρίνεται στην ιδιότητα του *απομονωμένου scope* του *directive*, *grid* και η τιμή του «=» λέει στο *$compile* να δεσμευτεί στην ιδιότητα *grid*. Στο παράδειγμα μας έχουμε ως τιμή το «=» *scope:{grid:'='}* το οποίο χρησιμοποιείται με αυτόν τον τρόπο στη περίπτωση που θέλουμε το όνομα της ιδιότητας να είναι ίδιο με την τιμή που θέλουμε να δεσμεύσουμε μέσα στο *scope* του *directive*. Το *scope:{grid:'='}* είναι το ίδιο με το να πούμε *scope:{grid:'=grid'}*. Άλλες επιλογές πέρα από το «=» είναι το «@» και το «&». Με το «@» τα περιεχόμενα του *directive* αποκτούν πρόσβαση σε *string* τιμές έξω από αυτό ενώ το «&» χρησιμοποιείται όπως το «=» με τη διαφορά ότι χρησιμοποιείται για τη δέσμευση συναρτήσεων.

Το πεδίο *controller* αναφέρεται στον *ελεγκτή* του *directive*. Αυτή η επιλογή επισυνάπτει έναν *ελεγκτή* στο *template* του *directive*. Αμέσως μετά τον *ελεγκτή* υπάρχει η επιλογή *link* στη συνάρτηση της οποίας βρίσκεται σαν τέταρτο όρισμα το όνομα του *ελεγκτή* που απαιτεί το *directive* (canvasSectionController). Γενικά όταν ένα *directive* απαιτεί κάποιον *ελεγκτή*, τότε δέχεται αυτόν τον *ελεγκτή* σαν τέταρτο όρισμα της *link* συνάρτησης του.

Το πεδίο *link* χρησιμοποιείται για να δώσει στο *directive* τη δυνατότητα να τροποποιήσει το DOM εγγράφοντας DOM listeners όπως επίσης και να αναβαθμίσει (update) το DOM. Εκτελείται αφού κλωνοποιηθεί το *template* και είναι το μέρος όπου τοποθετείται όλη η λογική του *directive*. Το *link* παίρνει μία συνάρτηση με την εξής μορφή: *function link(scope, element, attrs, controller, transcludeFn) { ... },* όπου το *scope* είναι το Angular *scope* αντικείμενο, το *element* είναι ένα στοιχείο τυλιγμένο με jqLite το οποίο ταιριάζει το *directive*, το *attrs* είναι ένα hash αντικείμενο με ζευγάρια κλειδί-τιμή από κανονικοποιημένα ονόματα ιδιοτήτων και τις αντίστοιχες τιμές ιδιοτήτων, το *controller* είναι είτε το στιγμιότυπο(α) του απαιτούμενου *ελεγκτή* του *directive* ή ο *ελεγκτής* του ίδιου του *directive*. Αυτό εξαρτάται από το πεδίο *require* του *directive*. Τέλος, το *transcludeFn* είναι μία *transclude* συνάρτηση σύνδεσης προ-δεσμευμένη στο σωστό *transclusion scope*.

Το πεδίο *templateUrl* χρησιμοποιείται για να φορτωθεί ένα εξωτερικό αρχείο HTML το οποίο είναι *template*. Ο λόγος που χρησιμοποιείται αυτή η επιλογή είναι γιατί το *template* του *directive* είναι μεγάλο και είναι καλό να βρίσκεται σε ξεχωριστό αρχείο. Εκτός από τη φόρτωση αρχείων HTML, το *templateUrl* μπορεί επίσης να είναι και μία συνάρτηση η οποία επιστρέφει το URL ενός HTML *template* το οποίο θα φορτωθεί και θα χρησιμοποιηθεί για το *directive*. Η συνάρτηση έχει δύο παραμέτρους: ένα στοιχείο πάνω στο οποίο κλήθηκε το *directive* και ένα *attr* αντικείμενο το οποίο συσχετίζεται με το στοιχείο. Δηλαδή: *templateUrl: function(elem, attr) {}*. Στη περίπτωση που το *template* είναι μικρό αντί για το *templateUrl* χρησιμοποιείται η επιλογή *template* η οποία παίρνει ως τιμή ένα στατικό *template*. Για παράδειγμα: *template: 'Name: {{customer.name}} Address: {{customer.address}}'*.

Τέλος, το πεδίο *replace* αναφέρεται στο αν το στοιχείο στο οποίο εφαρμόζεται το *directive* θα παραμείνει ή όχι. Στη περίπτωση που η τιμή του πεδίου είναι *true* το στοιχείο στο οποίο εφαρμόζεται το *directive* θα αντικατασταθεί από το *template* του *directive* ενώ στην περίπτωση που η τιμή είναι *false* το στοιχείο θα παραμείνει και το *template* του *directive* θα προστεθεί σαν παιδί του. Και στις δύο περιπτώσεις όμως τα παιδιά του στοιχείου θα χαθούν. Για να διατηρηθούν θα πρέπει να χρησιμοποιηθεί η επιλογή *trnasclude* και να τεθεί η τιμή της σε *true*.

3.2.4.5 Dependency Injection

Το *dependency injection* είναι ένα σχεδιαστικό πρότυπο λογισμικού το οποίο ασχολείται με το πώς τα συστατικά παίρνουν στα χέρια τους τις εξαρτήσεις τους. Το injector υποσύστημα του Angular είναι υπεύθυνο για τη δημιουργία συστατικών, για την επίλυση των εξαρτήσεων τους και την παροχή τους σε άλλα συστατικά όπως ζητήθηκε. Το *dependency injection* είναι διαβρωτικό σε όλο το Angular. Μπορεί να χρησιμοποιηθεί όταν ορίζονται τα συστατικά ή όταν παρέχονται *run* και *config* blocks για ένα *module*.

Συστατικά όπως τα *services*, τα *directives*, τα *φίλτρα* και τα *animations* ορίζονται από μία injectable *factory μέθοδο* ή μία συνάρτηση constructor. Αυτά τα συστατικά μπορούν να εγχυθούν (injected) με “service” και “value” συστατικά ως εξαρτήσεις. Για παράδειγμα μπορούμε να δημιουργήσουμε ένα *service* με όνομα *depService* και να το εγχύσουμε στη συνέχεια σε ένα *directive* ως *dependency*: ang*ular.module('myModule', []).directive('directiveName', ['depService', function(depService) {}])*. Οι *factory* μέθοδοι εγγράφονται σε *modules*. Μπορούμε επίσης να ορίσουμε συναρτήσεις που θα τρέχουν κατά τη διαμόρφωση και το χρόνο εκτέλεσης για ένα *module* καλώντας τις μεθόδους *config* και *run*. Αυτές οι συναρτήσεις είναι injectable με εξαρτήσεις όπως ακριβώς και οι *factory* συναρτήσεις. Για παράδειγμα σε ένα *module:* τρέχουμε τη συνάρτηση *config*:

angular.module('myModule', []).config(['depProvider', function(depProvider) {}]). Ακριβώς με τον ίδιο τρόπο τρέχει και η συνάρτηση *run*.

Το Angular επικαλείται ορισμένες συναρτήσεις (όπως τα *service factories* και τους *ελεγκτές*) μέσω του injector. Χρειάζεται να σχολιαστούν (annotate) αυτές οι συναρτήσεις έτσι ώστε να γνωρίζει ο injector τι *services* να εγχύσει στη συνάρτηση. Υπάρχουν τρεις τρόποι για να σχολιαστεί ο κώδικας με πληροφορίες ονόματος *service*: είτε χρησιμοποιώντας τον σχολιασμό inline πίνακα, είτε την ιδιότητα σχολιασμού *$inject* ή έμμεσα από τα ονόματα παραμέτρων της συνάρτησης.

Με τη χρήση inline πίνακα περνάμε έναν πίνακα του οποίου τα στοιχεία αποτελούνται από μία λίστα από στοιχεία (ονόματα των εξαρτήσεων) ακολουθούμενη από τη συνάρτηση. Δηλαδή: *someModule.controller('MyController', ['$scope', 'greeter', function($scope, greeter) {}])*. Στο παράδειγμα βλέπουμε την εγγραφή ενός ελεγκτή σε ένα *module* που έχει ήδη φτιαχτεί καθώς επίσης και τις εξαρτήσεις *$scope* και *greeter* μέσα στον πίνακα. Ο τρόπος αυτός είναι ο πιο επιθυμητός για το σχολιασμό (annotation) των συστατικών της εφαρμογής. Τέλος, θα πρέπει να σημειωθεί ότι χρησιμοποιώντας αυτόν τον τρόπο σχολιασμού πρέπει ο σχολιασμός μέσα στον πίνακα να είναι συγχρονισμένος με τις παραμέτρους στη δήλωση της συνάρτησης.

Χρησιμοποιώντας την ιδιότητα σχολιασμού *$inject* επιτρέπουμε στους minifiers να αλλάξουν το όνομα των παραμέτρων της συνάρτησης και να μπορούν να εγχύσουν τα σωστά *services*. Η ιδιότητα είναι ένας πίνακας από ονόματα *services* τα οποία θα εγχυθούν. Παραδείγματος χάρη:

```
var MyController = function($scope, greeter) {
  ...
}
MyController.$inject = ['$scope', 'greeter'];
someModule.controller('MyController', MyController);
```

Η σειρά των τιμών στον πίνακα *$inject* πρέπει να ταιριάζει τη σειρά των παραμέτρων στο *MyController*. Όπως και στον προηγούμενο τρόπο, το *$inject* πρέπει να είναι συγχρονισμένο με τις παραμέτρους στη δήλωση της συνάρτησης.

Με τον τελευταίο τρόπο υποθέτουμε ότι τα ονόματα παραμέτρων της συνάρτησης είναι τα ονόματα των εξαρτήσεων. Δηλαδή: *someModule.controller('MyController', function($scope, greeter) {})*. Δεδομένης μίας συνάρτησης, ο injector μπορεί να συμπεράνει τα ονόματα των *services* που θα εγχύσει, εξετάζοντας τη δήλωση της συνάρτησης και εξάγοντας τα ονόματα των παραμέτρων. Στο παράδειγμα τα *$scope* και *greeter* είναι δύο *services* τα οποία χρειάζεται να εγχυθούν στη συνάρτηση. Το πλεονέκτημα αυτής της προσέγγισης είναι ότι δεν υπάρχει πίνακας ονομάτων ο οποίος πρέπει να είναι συγχρονισμένος με τις παραμέτρους της συνάρτησης και επιπλέον μπορούμε να αλλάξουμε την σειρά των εξαρτήσεων. Αντίθετα η προσέγγιση αυτή δεν θα δουλέψει με minifiers/obfuscators της JavaScript επειδή αλλάζουν το όνομα των παραμέτρων.

Υπάρχουν μόνο τρεις τρόποι ώστε ένα συστατικό (αντικείμενο ή συνάρτηση) να πάρει στα χέρια του τις εξαρτήσεις του: είτε το συστατικό να δημιουργήσει την εξάρτηση, τυπικά χρησιμοποιώντας τον τελεστή *new*, είτε να ψάξει την εξάρτηση αναφερόμενο σε μία καθολική (global) μεταβλητή ή μπορεί να περαστεί σε αυτό η εξάρτηση όποτε χρειάζεται. Οι δύο πρώτοι τρόποι δεν είναι άριστοι καθώς κάνει *hard code* την εξάρτηση με το

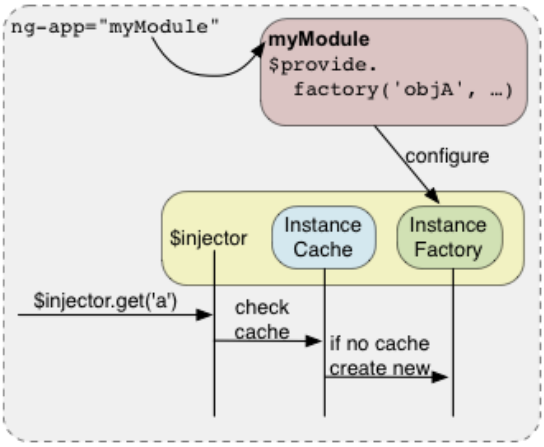
συστατικό, κάνοντας έτσι δύσκολη, μπορεί και ακατόρθωτη, την τροποποίηση των εξαρτήσεων.

Ο τρίτος τρόπος είναι πιο βιώσιμος, αφού αφαιρεί την ευθύνη εύρεσης της εξάρτησης από το στοιχείο. Η εξάρτηση απλώς παραδίδεται στο στοιχείο.

```
function SomeClass(greeter) {
  this.greeter = greeter;
}
SomeClass.prototype.doSomething = function(name) {
  this.greeter.greet(name);
}
```

Στο παραπάνω παράδειγμα, η *SomeClass* δεν ενδιαφέρεται στο να δημιουργήσει ή να εντοπίσει την εξάρτηση από το στοιχείο, απλώς παραδίδεται στο *greeter* μόλις αρχικοποιηθεί. Αυτό είναι επιθυμητό αλλά βάζει την ευθύνη, του να πάρει στα χέρια του την εξάρτηση, στον κώδικα που κατασκευάζει τη *SomeClass*.

Για τη διαχείριση της ευθύνης δημιουργίας εξάρτησης, κάθε Angular εφαρμογή έχει τον injector. Όπως αναφέρθηκε, ο injector είναι ένας *εντοπιστής service* που είναι υπεύθυνος για την κατασκευή και την αναζήτηση εξαρτήσεων. Κατά την εκτέλεση του προγράμματος εκκίνησης του Angular (bootstrapper) δημιουργείται ο injector και μέσα από αυτόν ζητείται το συστατικό. Η ζήτηση των εξαρτήσεων λύνει το πρόβλημα του hard coding αλλά σημαίνει επίσης ότι ο injector χρειάζεται να περνιέται σε όλη την εφαρμογή. Περνώντας τον injector, παραβιάζεται ο νόμος του *Demeter*. Για να λυθεί αυτό χρησιμοποιείται δηλωτική σημειογραφία στα HTML *templates*, ώστε να παραδοθεί η ευθύνη δημιουργίας συστατικών στον injector.



Εικόνα 3.2.4.2 Η λειτουργία του Angular injector

3.2.4.6 Services

Τα *services* του Angular είναι υποκατάστατα αντικείμενα, τα οποία είναι ενωμένα μαζί χρησιμοποιώντας *dependency injection* και χρησιμοποιούνται για την οργάνωση και τον διαμοιρασμό του κώδικα στην εφαρμογή. Τα Angular *services* αρχικοποιούνται νωχελικά, δηλαδή το Angular αρχικοποιεί ένα *service* μόνο όταν ένα συστατικό της εφαρμογής εξαρτάται από αυτό, και είναι Singletons, δηλαδή κάθε στοιχείο εξαρτώμενο από ένα *service* παίρνει μία αναφορά στο μονό στιγμιότυπο που παράγεται από το *factory* του *service*.

Υπάρχουν πολλά χρήσιμα *services* που παρέχονται από το Angular (όπως το *$http*) αλλά είναι δυνατό να δημιουργήσει κάποιος τα δικά του.

Για να χρησιμοποιηθεί ένα *service* προστίθεται σαν εξάρτηση για το στοιχείο (*ελεγκτής, service, φίλτρο* ή *directive*) που εξαρτάται από το *service*. Το υποσύστημα *dependency injection* του Angular χειρίζεται τα υπόλοιπα. Όπως αναφέρθηκε μπορεί ο καθένας να δημιουργήσει το δικό του *service*. Για να γίνει αυτό αρκεί να εγγράψει το όνομα του *service* και την *factory συνάρτηση* του σε ένα *module*. Για παράδειγμα:

var myModule = angular.module('myModule', []);

myModule.factory('serviceId', ['$interval', '$log', function() {}]);

Δημιουργούμε το *module* και εγγράφουμε σε αυτό το *service* με όνομα *serviceId* το οποίο εξαρτάται από τα *$interval* και *$*log *services*. Να σημειωθεί ότι δεν εγγράφεται ένα στιγμιότυπο του *service* αλλά μία *factory συνάρτηση* η οποία θα δημιουργήσει αυτό το στιγμιότυπο όταν κληθεί. Η *factory συνάρτηση* παράγει το μονό αντικείμενο ή τη μονή συνάρτηση που αντιπροσωπεύει το *service* στην υπόλοιπη εφαρμογή. Το αντικείμενο ή η συνάρτηση που επιστρέφεται, μπορεί να εγχυθεί σε οποιοδήποτε συστατικό που προσδιορίζει εξάρτηση στο *service*.

3.2.4.7 Modules

Ένα *module* μπορεί να θεωρηθεί σαν ένα «δοχείο» για τα διαφορετικά μέρη της εφαρμογής (*ελεγκτές, services, φίλτρα* ή *directives*). Οι περισσότερες εφαρμογές έχουν μία κύρια (main) μέθοδο η οποία αρχικοποιεί και ενώνει μαζί τα διαφορετικά μέρη της εφαρμογής. Οι Angular εφαρμογές δεν έχουν κύρια μέθοδο. Αντί αυτής, τα *modules* προσδιορίζουν δηλωτικά πώς πρέπει να εκκινηθεί μία εφαρμογή. Τα πλεονεκτήματα αυτής της προσέγγισης είναι ότι η δηλωτική διαδικασία κατανοείται πιο εύκολα, ο κώδικας μπορεί να πακεταρηστεί σαν επαναχρησιμοποιήσιμα *modules*, τα *modules* μπορούν να φορτωθούν με οποιαδήποτε σειρά (ή ακόμα και παράλληλα) γιατί καθυστερούν την εκτέλεση, τα unit tests πρέπει να φορτώσουν μόνο τα σχετικά *modules*, το οποίο τα κρατάει γρήγορα και τα end-to-end tests μπορούν να χρησιμοποιήσουν τα *modules* για να παραμερίσουν τη διαμόρφωση (configuration).

Για να ορίσουμε ένα *module* χρησιμοποιούμε το *module* API. Αφού έχουμε ορίσει το *module*: ***var*** *app = angular.module('app', ['canvas', 'dashboard',...])*, όπου ο πίνακας είναι η λίστα από *modules* από τα οποία εξαρτάται το *app*, το καλούμε στην όψη: *<html lang="en" ng-app="app">* ώστε να εκκινήσει την εφαρμογή χρησιμοποιώντας το *module*. Το *module* είναι μία συλλογή από configuration και run μπλοκ τα οποία εφαρμόζονται στην εφαρμογή κατά τη διαδικασία εκκίνησης (bootstrap process). Στην πιο απλή του μορφή αποτελείται από μία συλλογή δύο τύπων μπλοκ: τα *configuration* και τα *run*.

Τα *configuration* μπλοκ εκτελούνται κατά τις εγγραφές του παρόχου (provider) και τη φάση διαμόρφωσης (configuration phase). Μόνο πάροχοι και σταθερές (constants) μπορούν να εγχυθούν στα *configuration* μπλοκ. Αυτό συμβαίνει για να αποφευχθεί η κατά λάθος αρχικοποίηση *services* προτού έχουν ρυθμιστεί πλήρως. Για παράδειγμα: *app.config(function($locationProvider){})* χρησιμοποιούμε το *app module* του προηγούμενου παραδείγματος και εγγράφουμε σε αυτό ένα *config* μπλοκ στου οποίου τη συνάρτηση βάζουμε τους providers.

Τα *run* μπλοκ είναι το πιο κοντινό πράμα στο Angular με τη κύρια μέθοδο. Είναι ο κώδικας που χρειάζεται να τρέξει για να εκκινηθεί η εφαρμογή.Εκτελούνται μετά την διαμόρφωση όλων των *services* και τη δημιουργία του injector. Μόνο στιγμιότυπα (instances) και

σταθερές μπορούν να εγχυθούν σε *run* μπλοκ. Αυτό συμβαίνει για να εμποδιστεί η επιπλέον διαμόρφωση του συστήματος κατά τη διάρκεια του χρόνου εκτέλεσης της εφαρμογής. Τα *run* μπλοκ τυπικά περιέχουν κώδικα που είναι δύσκολο να γίνει unit testing για αυτό δηλώνονται σε απομονωμένα *module*, ώστε να αγνοούνται στα unit tests. Για παράδειγμα: *app.run(function(Socket, CanvasService, $rootScope){})* χρησιμοποιούμε ξανά το *app module* του προηγούμενου παραδείγματος και εγγράφουμε σε αυτό ένα *run* μπλοκ στου οποίου τη συνάρτηση βάζουμε τα στιγμιότυπα (instances).

Τέλος όπως αναφέρθηκε, τα *modules* μπορούν να καταγράψουν άλλα *modules* σαν εξαρτήσεις τους. Η εξάρτηση από ένα *module* υποδηλώνει ότι το απαιτούμενο *module* χρειάζεται να φορτωθεί πριν το *module* που απαιτείται. Με άλλα λόγια, τα *configuration* μπλοκ των απαιτούμενων *modules* εκτελούνται πριν από τα *configuration* μπλοκ των *module* που απαιτούνται. Το ίδιο ισχύει και για τα *run* μπλοκ. Κάθε *module* μπορεί να φορτωθεί μία φορά ακόμα και αν άλλα πολλαπλά *modules* το απαιτούν.

3.2.5 PRINCE 2

Ο δημόσιος τομέας είχε έλλειψη της ικανότητας να παραδώσει εργασίες (projects) στην ώρα τους, εντός του προϋπολογισμού, του πεδίου δράσης και με την σωστή ποιότητα. Τα PROMPTII, PRINCE και PRINCE2 είχαν εισαχθεί για την αντιμετώπιση των κοινών αιτιών της αποτυχίας ενός project. Το PRINCE2 είναι μια μία de facto μεθοδολογία που βασίζεται στην διαδικασία της αποτελεσματικής διαχείρισης ενός έργου. Προέρχεται από το ακρώνυμο του ***PR***ojects ***IN*** ***C***ontrolled ***E***nvironments (Εργασίες σε Ελεγχόμενο Περιβάλλον). Αναπτύχθηκε από την κυβέρνηση του Ηνωμένου Βασιλείου και χρησιμοποιούνται διεθνώς, τόσο στον ιδιωτικό τομέα όσο και τον δημόσιο τομέα. Το PRINCE2 χρησιμοποιεί τις βέλτιστες αποδεδειγμένα πρακτικές (best practices) από διάφορους κλάδους και υπόβαθρα. Τα σαφή σημεία λήψης αποφάσεων είναι τα χαρακτηριστικά αυτής της μεθοδολογίας. Υπάρχουν τρεις εξετάσεις με βάση τα επίπεδα της πιστοποίησης για τους ιδιώτες, τα θεμέλια (Foundation), ο ειδικευόμενος (Practitioner) και ο επαγγελματίας (Professional).

Στις πρώτες μέρες της Διαχείρισης Εργασιών (project management), τα projects τα διαχειρίζονταν με ad-hoc basis, δηλαδή κάτι που γίνεται για ένα συγκεκριμένο σκοπό και δεν προβλέπεται πριν αυτό συμβεί, χρησιμοποιώντας διαγράμματα Gantt και άτυπα εργαλεία και τεχνικές. Το 1989, η Κεντρική Υπηρεσία Υπολογιστών και Τηλεπικοινωνιών (central computer and telecommunications agency) της τότε κυβέρνησης του Ηνωμένου Βασιλείου, υιοθέτησε ένα Project Management μοντέλο το οποίο ονόμασε PRINCE. Το 1996 κυκλοφόρησε η τελειωμένη έκδοσή του ως PRINCE2 και έγινε πρότυπο σε πολλά τμήματα της Κυβέρνησης του Ηνωμένου Βασιλείου. Μια σημαντική αναθεώρηση από τις συχνές αιτήσεις των χρηστών έκανε την μέθοδο πιο απλή και εύκολα προσαρμόσιμη. Το PRINCE2 στοχεύει στο να δώσει στους διαχειριστές του project (project managers) ένα καλύτερο πακέτο εργαλείων ώστε το έργο να παραδοθεί στην ώρα του, εντός του προϋπολογισμού και στην σωστή ποιότητα. Έχει ρίζες από την δεκαετία του 70, αλλά είναι το πιο δημοφιλές framework για την διαχείριση έργων όλων των ειδών.

Η μέθοδος περιγράφει πώς ένα έργο χωρίζεται σε διαχειρίσιμα στάδια που επιτρέπουν τον αποτελεσματικό έλεγχο των πόρων και την τακτική παρακολούθηση της προόδου σε όλη την διάρκεια του έργου. Επίσης, περιγράφει πλήρως τα διάφορα καθήκοντα και τις ευθύνες για τη διαχείριση ενός έργου και τα ορίζει έτσι ώστε να ταιριάζουν με το μέγεθος και την πολυπλοκότητα του έργου, καθώς και τις ικανότητες του οργανισμού. Ο σχεδιασμός του έργου (project planning), με την χρήση του PRINCE2 ορίζει ότι τα σχέδια του έργου επικεντρώνονται στην επίτευξη αποτελεσμάτων και όχι μόνο στον σχεδιασμό, όταν οι διάφορες δραστηριότητες θα πραγματοποιηθούν. Το business case αναθεωρείται συχνά κατά την διάρκεια του έργου ώστε να εξασφαλιστούν οι στόχοι, οι οποίοι αλλάζουν καθώς το έργο εξελίσσεται.

Το PRINCE2 παρέχει οφέλη στους διαχειριστές και τους διευθυντές του έργου και σε μια οργάνωση γενικότερα, μέσω της ελέγξιμης χρήσης των πόρων και την ικανότητα να διαχειρίζονται οι επιχειρήσεις τους κινδύνους του έργου πιο αποτελεσματικά. Επιπλέον, ενσωματώνει καθιερωμένες και δοκιμασμένες βέλτιστες πρακτικές στη διαχείριση έργων. Είναι ευρέως αναγνωρισμένο και κατανοητό, παρέχοντας μια κοινή γλώσσα για όλους τους συμμετέχοντες του έργου. Το PRINCE2 ενθαρρύνει την επίσημη αναγνώριση των ευθυνών μέσα σε ένα έργο και εστιάζει στο αποτέλεσμα που πρέπει να παραδοθεί, γιατί πρέπει να δημιουργηθεί, πότε και για ποιον. Το PRINCE2 παρέχει στα projects: Μια ελεγχόμενη και οργανωμένη αρχή, μέση και τέλος, τακτικές αξιολογήσεις της προόδου με βάση το Business Case καθώς και ευέλικτα σημεία λήψης αποφάσεων, αυτόματο έλεγχο της διαχείρισης των τυχόν αποκλίσεις, την συμμετοχή της διοίκησης και των ενδιαφερόμενων στο σωστό χρόνο και τόπο κατά τη διάρκεια του έργου και καλή επικοινωνίας μεταξύ όλων.

Οι διευθυντές που χρησιμοποιούν το PRINCE είναι σε θέση να καθιερώσουν όρους αναφοράς ως προαπαιτούμενο για την έναρξη του έργου, να χρησιμοποιήσουν μια καθορισμένη δομή για τις αρμοδιότητες και την επικοινωνία μεταξύ των μελών της ομάδας, να χωρίσουν το έργο σε διαχειρίσιμα στάδια για πιο ακριβή σχεδιασμό, να παρέχουν τακτικές αλλά σύντομες αναφορές (reports) και άλλα. Για να οργανωθεί και να ελέγχεται ένα έργο πρέπει να υπάρχει ένα υπεύθυνος, αυτό ο υπεύθυνος ονομάζεται διευθυντής του έργου (Project Manager). Η δουλειά του Project Manager είναι να επιλέγει τους ανθρώπους που είναι ικανοί να φέρουν εις πέρας στο στόχο του project και βεβαιώνεται ότι η δουλειά γίνεται σωστά και στον χρόνο που είναι καθορισμένη. Ο Project Manager συντάσσει τα σχέδια του έργου (Project Plans) που περιγράφουν το τι πράγματι θα κάνει η ομάδα του έργου και πότε αναμένεται να ολοκληρωθεί η διαδικασία. Το άτομο ή η εταιρία που πληρώνει για αυτό το έργο ονομάζεται πελάτης (customer) ή διευθυντής (executive). Ο άνθρωπος που θα χρησιμοποιήσει τα αποτελέσματα αυτού του έργου ή που θα επηρεαστεί από αυτά, ονομάζεται χρήστης (user). Σε μερικά projects ο πελάτης και ο χρήστης μπορεί να είναι το ίδιο άτομο.

Εκείνος που παρέχει την τεχνογνωσία για να κάνει την πραγματική δουλειά του έργο, όπως τον σχεδιασμό και προγραμματισμό, ονομάζεται προμηθευτής (supplier) ή ειδικός (specialist). Όλα αυτά τα πρόσωπα χρειάζεται να οργανωθούν και να συντονιστούν έτσι ώστε να πραγματοποιηθεί το απαιτούμενο αποτέλεσμα εντός του προϋπολογισμού (budget), στην ώρα του και στην κατάλληλη ποιότητα. Κάθε έργο που υλοποιείται με το PRINCE2 πρέπει να έχει μία Επιτροπή του Έργου (Project Board) φτιαγμένη από τον πελάτη, κάποιον ο οποίος θα αντιπροσωπεύει την μεριά των χρηστών και κάποιον που θα αντιπροσωπεύει την μεριά των προμηθευτών. Αυτά τα άτομα ονομάζονται Customer, Senior User και Senior Supplier. Ο Project Manager δίνει τακτικές αναφορές στην Επιτροπή του Έργου κρατώντας τους ενημερωμένους για την πρόοδο και τονίζοντας τους κάποια προβλήματα που ίσως είναι ικανοί να επιλύσουν. Η Επιτροπή του Έργου είναι υπεύθυνη να παρέχει στον Project Manager με τις απαραίτητες αποφάσεις για την διαδικασία του έργου και να ξεπερνά οποιοδήποτε πρόβλημα.

Εκτός από την περιγραφή των διάφορων ανθρώπων που εμπλέκονται στα project βασισμένα στο PRINCE2 και ποιες είναι οι αρμοδιότητες και οι ευθύνες τους για αυτό, η μέθοδος αυτή εξηγεί και δίνει πληροφορίες για το πώς να διαχειριστούν τους κινδύνους, την ποιότητα και πως θα ελέγξουν τις σταδιακές αλλαγές του έργου. Η διαχείριση του κινδύνου (Risk Management) έχει να κάνει με την αντιμετώπιση εκείνων που μπορεί να πάνε στραβά και την εύρεση λύσεων. Η διαχείριση της ποιότητας (Quality Management) έχει να κάνει με τον έλεγχο της ποιότητα της εργασίας που ολοκληρώθηκε, είτε από δοκιμή είτε από επανεξέταση αυτής. Υπάρχουν πάντοτε αλλαγές που γίνονται κατά την διάρκεια της εκτέλεσης του έργου, καθώς οι γνώμες αλλάζουν και άλλοι εξωτερικού παράγοντες επηρεάζουν την εξέλιξη του. Το PRINCE2 έχει μία τεχνική να ελέγχει τον τρόπο που οι αλλαγές αυτές θα επηρεάσουν το έργο προκειμένου να το αποτρέψει από λάθος κατεύθυνση.

Το PRINCE2 είναι μια διαδικασία προσέγγισης για τη διαχείριση του έργου που παρέχει μία εύκολα προσαρμοσμένη και κλιμακούμενη μέθοδο για τη διαχείριση όλων των τύπων των έργων. Κάθε διαδικασία ορίζεται με τα κλειδιά της εισόδου και της εξόδου της μαζί με τους συγκεκριμένους στόχους που πρέπει να επιτευχθούν και τις δραστηριότητες που θα πραγματοποιηθούν. Η ανανεωμένη έκδοση PRINCE2 βασίζεται σε επτά αρχές, επτά θέματα και επτά διεργασίες που συμβάλλουν στην επιτυχία ενός project. Οι επτά αρχές είναι: συνεχής επιχειρηματική αιτιολόγηση, μάθηση από την εμπειρία, καθορισμένοι ρόλοι και ευθύνες, διαχείριση μέσω σταδίων, διαχείριση μέσω εξαιρέσεων, εστίαση στο προϊόν και προσαρμογή στο περιβάλλον του project. Τα επτά θέματα είναι: οι επιχειρηματικές περιπτώσεις (business case), η οργάνωση, η ποιότητα, τα σχέδια, ο κίνδυνος (risks), η αλλαγή και η πρόοδος. Οι αρχές και τα θέματα έρχονται στο προσκήνιο στις επτά διαδικασίες, οι οποίες είναι οι εξής:

1. *Καθοδήγηση ενός project* (directing a project – DP). Η καθοδήγηση ενός έργου ξεκινάει από την έναρξη αυτό έως το τέλος του. Αυτή η διαδικασία στοχεύει την Επιτροπή του Έργου (Project Board). Η Επιτροπή του έργου διαχειρίζεται κατ' εξαίρεση, παρακολουθεί μέσω αναφορών και ελέγχει μια σειρά αποφάσεων. Οι βασικές διαδικασίες για τη Επιτροπή του Έργου χωρίζεται σε τέσσερις κύριους τομείς: *Την σωστή έναρξη του έργου, τα όρια του σχεδίου, την παρακολούθηση της προόδου γίνοντας συμβουλές και καθοδήγηση και το κλείσιμο του έργου.* Αυτές οι διαδικασίες δεν αναιρούν ή καλύπτουν τις καθημερινές δραστηριότητες του Project Manager.

2. *Έναρξη ενός project* (starting up a project – SU). Είναι πρώτη διαδικασία του PRINCE. Είναι προετοιμασία του έργου, ο σχεδιασμός του και η εξασφάλιση ότι τα προ απαιτούμενα για το αρχικό προϊόν είναι έτοιμα. Η διαδικασία αναμένει την ύπαρξη μια εντολής έργου που ορίζεται από τους λόγους της δημιουργίας του έργου καθώς και τα επιθυμητά αποτελέσματα μετά την ολοκλήρωση αυτού. Η έναρξη του έργου είναι μια πολύ σύντομη διαδικασία. Το έργο της διαδικασία είναι χτισμένο γύρω από την παραγωγή τριών στοιχείων: *Την διασφάλιση ότι οι απαιτούμενες πληροφορίες για την ομάδα του έργου είναι διαθέσιμες, τον σχεδιασμό και προσδιορισμό της ομάδας της διαχείρισης του έργου και την δημιουργία ενός αρχικού πλάνου*.

3. *Αρχικοποίηση ενός project* (initiating a project – IP). Ο σκοπός και οι στόχοι της αρχικοποίησης ενός project είναι οι εξής:

    - Η συμφωνία για το αν υπάρχει επαρκής αιτιολόγηση για την υλοποίηση του project.
    - Να δημιουργηθεί μια σταθερή βάση διαχείρισης με τη οποία θα πορευτεί το project.
    - Να καταγράφεται και να επιβεβαιώνεται οποιοδήποτε υπάρχον Business Case για το project.
    - Να εξασφαλιστεί για σταθερή και αποδεκτή βάση για την έναρξη εργασιών.
    - Να γίνει συμφωνία για την δέσμευση των πόρων στο αρχικό στάδιο του project.
    - Να ενεργοποιηθεί και να ενθαρρυνθεί η Επιτροπή του Έργου στο να αναλάβει την κυριότητα του project.
    - Να παρέχονται οι βάσεις για της διαδικασίες λήψεων των αποφάσεων καθ' όλη την διάρκεια του project.
    - Να διασφαλιστεί ότι η επένδυση του χρόνου και η προσπάθεια που απαιτούνται από το project γίνονται με σύνεση λαμβάνοντας υπόψη τα ρίσκα και τους κινδύνους.

4. *Διαχείριση των ορίων του σταδίου* (managing stage boundaries – SB). Αυτή η διαδικασία παρέχει στο Project Board τις αποφάσεις κλειδί για το αν θα συνεχίσουν με την υλοποίηση του project ή όχι. Οι στόχοι αυτής της διαδικασίας είναι οι εξής:

    - Να διαβεβαιώνει το Project Board ότι όλα τα παραδοτέα που ήταν στο πλάνο για εκείνη την χρονική στιγμή είναι ολοκληρωμένα όπως θα έπρεπε.
    - Να παρέχει τις πληροφορίες που χρειάζεται το Project Board ώστε να έχει πρόσβαση στην συνεχή βιωσιμότητα του project.
    - Να παρέχει στο Project Board τις πληροφορίες που χρειάζεται ώστε να εγκρίνει το τρέχων στάδιο ολοκλήρωσης και να επιτρέψει την έναρξη της επόμενης φάσης.
    - Να καταγράψει μετρήσεις ή μαθήματα τα οποία μπορεί να βοηθήσουν σε μεταγενέστερα στάδια του project ή και σε άλλα projects.

5. *Έλεγχος ενός σταδίου* (controlling a stage – CS). Αυτή η διαδικασία περιγράφει τις δραστηριότητες παρακολούθησης και ελέγχου του Project Manager που εμπλέκονται στην διασφάλιση ότι ένα στάδιο παραμένει στην πορεία του και μπορεί να ανταπεξέλθει σε αναπάντεχα γεγονότα. Η διαδικασία σχηματίζει τον πυρήνα της προσπάθειας του Project Manager για το project, δηλαδή είναι η διαδικασία η οποία διαχειρίζεται το project κάθε μέρα. Καθ' όλη την διάρκεια ενός σταδίου θα υπάρχει ένας κύκλος που θα αποτελείται από: *Εξουσιοδοτημένες δουλειές που έχουν ολοκληρωθεί, συγκεντρωμένες πληροφορίες για την εξέλιξη της εργασίας, έλεγχος των αλλαγών, επανεξέταση της κατάστασης, αναφορές, λήψη απαραίτητων διορθωτικών πράξεων*. Η διαδικασία αυτή καλύπτει αυτές τις δραστηριότητες σε συνδυασμό με *την εξέλιξη του έργου, τους κινδύνους και την διαχείριση των αλλαγών.*

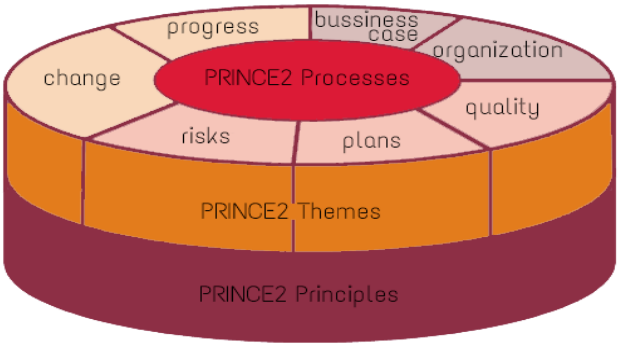
6. *Διαχείριση της παράδοσης του προϊόντος* (managing product delivery – MP). Ο στόχος αυτής της διαδικασίας είναι να διασφαλίσει ότι τα project που έχουν σχεδιαστεί δημιουργούνται και παραδίδονται, δηλαδή:

    - Εξασφαλίζει ότι οι εργασίες που διατίθενται για την ομάδα σχετικά με τα προϊόντα είναι αποτελεσματικές και έχουν εγκριθεί σύμφωνα με τα πακέτα εργασίας (work packages).
    - Εξασφαλίζει ότι οι εργασίες προσδιορίζονται σύμφωνα με τις απαιτήσεις του πακέτου εργασιών.
    - Εξασφαλίζει ότι μια εργασία είναι ολοκληρωμένη.
    - Αξιολογεί την πρόοδο της εργασίας και προβλέπει την εξέλιξη τακτικά.
    - Εξασφαλίζει ότι ολοκληρωμένα προϊόντα συναντούν την κριτήρια ποιότητα.
    - Εγκρίνει τα ολοκληρωμένα προϊόντα.

7. *Τέλος του project* (closing a project – CP). Σκοπός αυτής της διαδικασίας είναι να εκτελέσει ένα ελεγχόμενο κλείσιμο για το project. Η διαδικασία καλύπτει την δουλειά του Project Manager και τυλίγει το project είτε αυτό έχει ολοκληρωθεί, είτε έχει διακοπεί πρόωρα. Το μεγαλύτερο μέρος της εργασίας είναι να προετοιμαστούν όλα εκείνα που πρέπει να δοθούν στο Project Board ώστε να ληφθεί επιβεβαίωση ότι το project ίσως κλείσει. Στόχοι του κλεισίματος του έργου είναι να:

    - Ελέγχει τον βαθμό στον οποίο τελικοί οι στόχοι και οι σκοποί του project έχουν συναντήσει τους αρχικούς.
    - Επιβεβαιώνει την έκταση της εκπλήρωσης του αρχικού project και την ικανοποίηση του πελάτη αφού του παραδοθεί το προϊόν.
    - Αποκτά επίσημη αποδοχή των παραδοτέων.
    - Εξασφαλίζει σε ποιο βαθμό έχουν παραδοθεί όλα τα αναμενόμενα προϊόντα και αν έγιναν αποδεκτά από τον πελάτη.

- Επιβεβαιώνει ότι οι κανονισμοί της λειτουργίας και συντήρησης είναι στην θέση τους.
- Κάνει οποιεσδήποτε συστάσεις για μελλοντικές πράξεις.
- Καταγράφει τα μαθήματα που προέκυψαν από τα αποτελέσματα του project όταν αυτό ολοκληρώθηκε (Lessons Learned Report).
- Προετοιμάζει την αναφορά για την ολοκλήρωση του έργου.



Εικόνα 3.2.5.1 Δομή του PRINCE2

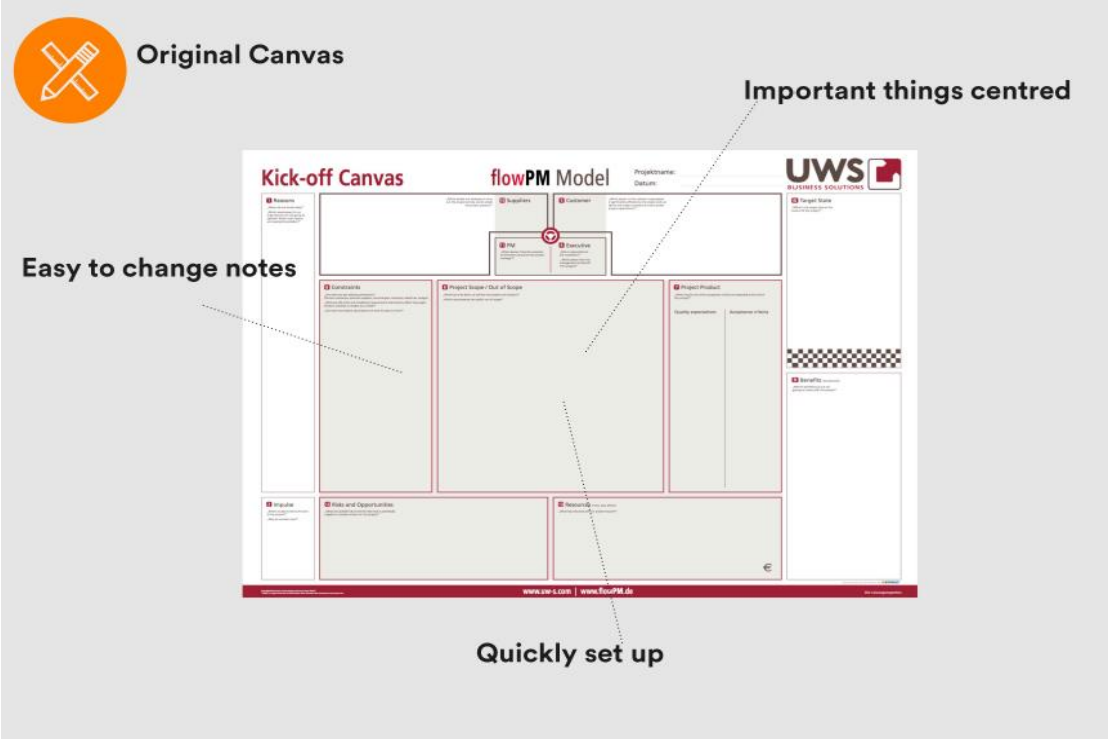
4 Κύριο Μέρος

4.1 Ανάλυση προβλήματος

4.1.1 Περιγραφή προβλήματος

Η *UWS Business Solutions GmbH* είναι η εταιρία που επένδυσε και δημιούργησε τον έντυπο KICKOFF καμβά (*KICKOFF canvas printed version*). Η UWS είναι εταιρία που παρέχει βοήθεια και συμβουλές σε ότι έχει σχέση με το management και βρίσκεται στο Paderborn της Γερμανίας. Δύο κεντρικά πρόσωπα της είναι ο Klaus-Oliver Welsow και ο André Unger οι οποίοι εφαρμόζουν την κατευθυντήρια αρχή της άριστης αλληλεπίδρασης των επιχειρηματικών διαδικασιών, παρέχουν προσαρμοσμένες λύσεις πληροφορικής και μεταφέρουν τις γνώσεις τους. Για περισσότερα από 20 χρόνια έχουν αποκτήσει πολύτιμες γνώσεις και μάθηση του management, IT λύσεις, καθώς και εμπειρία στην οργανωτική βελτιστοποίηση. Η παραγωγικότητα, η ποιότητα, η καινοτομία και η προσαρμοστικότητα είναι οι σημαντικοί παράγοντες που συμβάλλουν στην επιτυχία τους. Ο γενικός στόχος του έργου είναι η ανάπτυξη ενός λογισμικού που υποστηρίζει την εκκίνηση και την έναρξη της φάσης PRINCE2, την κορυφαία μέθοδο διαχείρισης ενός έργου στον κόσμο.

Με την εμπειρία που έχει αποκτήσει με τα πάνω από 1000 έργα η UWS διαθέτει την απαραίτητη τεχνογνωσία και επενδυτική ικανότητα να δημιουργήσει ένα πολύτιμο εργαλείο για τους διαχειριστές ενός έργου. Έχουν ήδη ένα ισχυρό δίκτυο πελατών και συνεργατών για την εισαγωγή του KICKOFF καμβά με επιτυχία στην αγορά. Η επαγγελματική ζωή στις μέρες μας βασίζεται σε μεγάλο βαθμό στην κινητικότητα και τις απαιτήσεις των επαγγελματιών να επεκταθούν σε διεθνές και διαπολιτισμικό περιβάλλον. Για αυτό τον λόγο στόχος της εταιρίας είναι να παρέχει στους πελάτες και στους χρήστες μία ενιαία διαδικτυακή λύση ώστε να βελτιωθεί η έναρξη μιας ιδέας. Ο έξυπνος σχεδιασμός, η εύκολη πλοήγηση, η προσβασιμότητα, οι ενσωματωμένες καθιερωμένες αρχές του PRINCE2 και η ώθηση για παραγωγικότητα είναι τα χαρακτηριστικά κλειδιά.



Εικόνα 4.1.1.1 Ο έντυπος KICKOFF καμβάς

Ο KICKOFF καμβάς είναι ένα *online εργαλείο* συνεργασίας για την έναρξη ενός έργου. Βοηθάει τον χρήστη να επικεντρωθεί σε σημαντικές πληροφορίες και λεπτομέρειες έτσι ώστε να είναι κατανοητός ο στόχος αυτού. Η έλλειψη της σαφήνειας και των πληροφοριών κατά την έναρξη του έργου μπορεί να οδηγήσει σε απογοήτευση και παρεξηγήσεις. Οι άνθρωποι αναπτύσσουν διαφορετικές και μη ρεαλιστικές προσδοκίες, αρχίζουν να εργάζονται σε αντίθετη κατεύθυνση από αυτή του στόχου του έργου και αυτό οδηγεί όλη την ομάδα σε σύγχυση. Το *KICKOFF canvas template* που σχεδιάσαμε και υλοποιήσαμε μπορεί να βοηθήσει τους παλιούς και τους καινούργιους διαχειριστές έργων να οργανώσουν όλα τα άτομα της ομάδας από την αρχή ώστε όλοι να βρίσκονται προς την ίδια κατεύθυνση από την αρχή δημιουργώντας ένα δομημένο και ολοκληρωμένο *business case* σε τέσσερα βασικά βήματα.

Το *business case* είναι ένα έγγραφο υψηλού επιπέδου για την οργάνωση των projects, το οποίο ενώνει όλες τις πληροφορίες που χρειάζονται ώστε να ξεκινήσει το έργο. Με ένα καλά ορισμένο business case, οι πελάτες μπορούν να γνωστοποιήσουν σε όλους και να τους κάνουν να καταλάβουν που κατευθύνεται το project από την αρχή και να παρέχουν τις πληροφορίες που χρειάζονται ώστε να είναι παραγωγικοί και αποτελεσματικοί. Με τη χρήση του KICKOFF καμβά οι ερωτήσεις ***γιατί***, ***ποιος***, και ***πως*** θα απαντηθούν και ένα ξεκάθαρος στόχος θα φανερωθεί. Ο καμβάς δημιουργείται από 14 κομμάτια, τα οποία είναι: ώθηση (*impulse*), λόγοι (*reasons*), οφέλη (*benefits*), περιορισμοί (*constrains*), στόχος (*target*), πελάτες (*customers*), προμηθευτές (*suppliers*), στέλεχος (*executive*), διευθυντής του έργου (*product manager*), ομάδα (*team*), πεδίο εφαρμογής του έργου (*project scope*), πεδίο εφαρμογής του προϊόντος (*product scope*), κινδύνους και ευκαιρίες (*risks and opportunities*) και πόρους (*resources*).

Η μέθοδος του να δουλεύει κανείς με καμβάδες αποδεδειγμένα φτάνει στον πυρήνα μιας ιδέας γιατί έχει τα εξής *πλεονεκτήματα*:

- Είναι σαφές και ελκυστικό οπτικά. Τα πάντα βρίσκονται σε μία σελίδα.
- Είναι εύκολο στη χρήση του.
- Δομεί τις διαδικασίες.
- Εξασφαλίζει ότι όλα έχουν ληφθεί υπόψη.
- Μπορεί να χρησιμοποιηθεί από τις ομάδες ώστε να συζητηθούν διαφορετικές απόψεις και προσανατολισμοί.
- Αυξάνει την επικοινωνία μεταξύ της ομάδας.

Μετά την εγγραφή του ο χρήστης έχει τη *δυνατότητα να προσκαλέσει άλλους χρήστες* για να δουλέψουν μαζί σε ένα έργο αλλά και να εργάζονται ασύγχρονα σε αυτό. Οι χρήστες μπορούν να δημιουργήσουν ομάδες και να ορίσουν άμεσα σχέδια για αυτούς.

Εικόνα 4.1.1.2 Ο καμβάς και το dashboard της εφαρμογής

Όπως αναφέραμε και παραπάνω, ο KICKOFF καμβάς είναι βασισμένος στην μέθοδο του PRINCE2. Άλλες αποδεδειγμένες μέθοδοι για την διαχείριση ενός έργου, όπως το *planning poker*, θα βοηθήσουν τους πελάτες να αποκτήσουν περισσότερες χρήσιμες πληροφορίες για την δουλειά τους. Οι πληροφορίες που συλλέγονται θα προστίθενται απευθείας στον καμβά και θα εμφανίζονται στο τελικό αρχείο εξαγωγής (*export file*). Με αυτό τον τρόπο ο KICKOFF καμβάς αποτρέπει την περίπτωση οι πελάτες να ξεχάσουν σημαντικά θέματα και αυξάνει τις γνώσεις όλων σε ότι αναφορά το έργο τους. Ένας επιπλέον στόχος του προϊόντος μας είναι να ενισχύσει τους πελάτες να μάθουν τη διαχείριση του έργου καθώς δουλεύουν. Με μία μικρή βοήθεια, η οποία θα εξηγεί βήμα-βήμα πως πρέπει να κινηθούν μέσα στον καμβά και ποια κατεύθυνση πρέπει να ακολουθήσουν ώστε να δημιουργηθεί το τέλειο business case.

Επεξηγήσεις και παραδείγματα δίνουν την δυνατότητα καινούργιων ιδεών στα στελέχη και τις νέες επιχειρήσεις, ώστε να έχουν το αίσθημα ότι μπορούν να πείσουν τους υπεύθυνους τους και τους πελάτες για αυτές. Επιπλέον, συμπεριλαμβάνονται στο dashboard του καμβά, tutorial videos τα οποία βοηθούν και αυτά τους χρήστες να χειριστούν την εφαρμογή μας. Συνοπτικά, την KICKOFF καμβάς εφαρμογή μας μπορούν οι χρήστες να την χρησιμοποιήσουν για:

- Να περιγράψουν και να αποδείξουν μία ιδέα ενός έργου.
- Να καθορίσουν το έργο τους στην πληρότητα του και αποκτήσουν κοινή κατανόηση.
- Ως οδηγός καθ' όλη την διάρκεια, ο οποίος θα διατηρεί την ροή των πραγμάτων.
- Την ανασκόπηση του έργου αφότου τελειώσει.

Χαρακτηριστικά	
Διαδικτυακός καμβάς	*Δίνει στην ομάδα το κεντρικό μέρος ώστε να συλλέξει όλες τις απαραίτητες πληροφορίες για την γρήγορη έναρξη του έργου.*
Εύκολος σχεδιασμός που καθοδηγεί τον χρήστη	*Μένει στην κορυφή των πραγμάτων μέσω του διαισθητικού και φιλικού σχεδιασμού του. Περιγραφές και tutorials θα καθοδηγούν τον χρήστη σε όλο το καμβά και θα τον βοηθήσουν να γίνει ειδικός στα projects.*
Χρήσιμα widgets	*Ενισχύει την αξία των πληροφοριών με εργαλεία όπως, το planning poker, την SWOT-ανάλυση και την σχεδίαση των προϊόντων. Μια έξυπνη έναρξη είναι η βάση ενός επιτυχημένου έργου.*
Αρχές του PRINCE2	*Οι χρήστες ωφελούνται από την πιο διάσημη μέθοδο στο project management. Το σχέδιο δοκιμάστηκε και ελέγχθηκε από αμέτρητα workshops.*
Εξαγωγή του business case	*Το τελικό business case έχει επαγγελματική μορφή και είναι καλά δομημένο.*
Διαμόρφωση	*Ο KICKOFF καμβάς είναι η μόνη λύση για την έναρξη των έργων, καθώς ο χρήστης μπορεί να διαμορφώσει τον κάθε καμβά ξεχωριστά ως προς τον σχεδιασμό και τα χαρακτηριστικά.*
SSL-Ασφάλεια	*Η δουλειά του χρήστη και τα δεδομένα μεταφέρονται με ασφάλεια μέσω υψηλής ποιότητας SSL/TLS και κρυπτογράφηση 256-bit.*

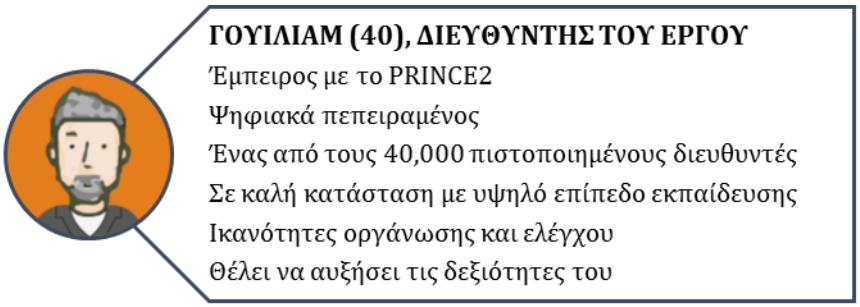
Πίνακας 4.1.1 Τα χαρακτηριστικά της εφαρμογής

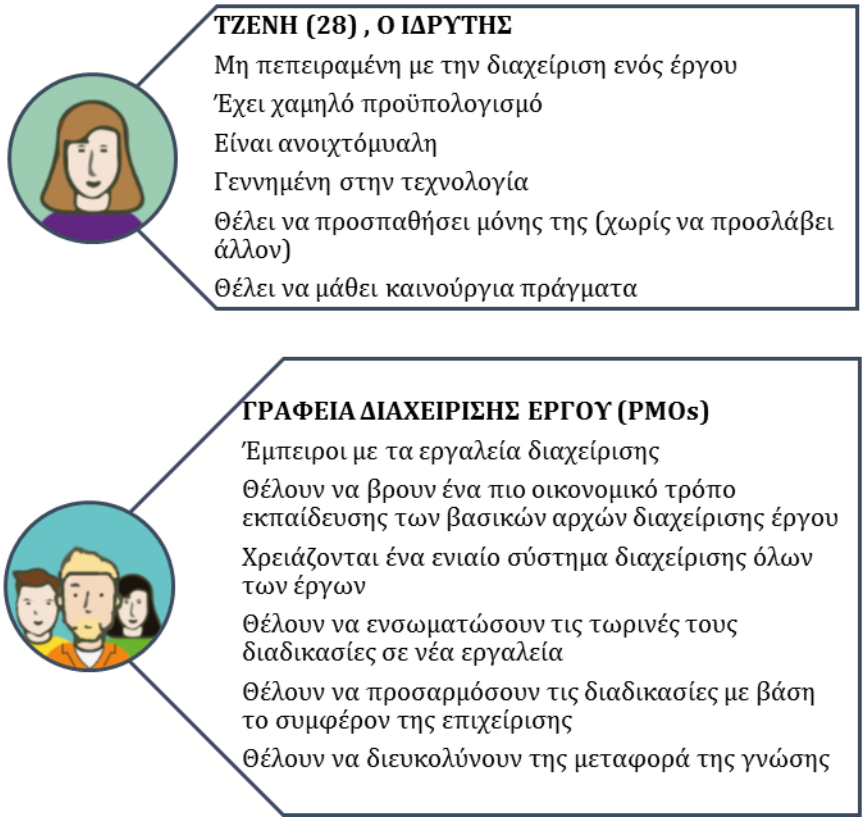
Εικόνα 4.1.1.3 Το ανανεωμένο logo του καμβά

4.1.2 Απαιτήσεις Συστήματος

Το *Scrum* σε αντίθεση με τα παραδοσιακά μοντέλα ανάπτυξης λογισμικού όπως είναι το *Waterfall* μπορεί να ανταπεξέλθει σε αλλαγή των απαιτήσεων ενός πελάτη. Οι φάσεις ανάπτυξης λογισμικού στο *Waterfall* είναι οι εξής: Ανάλυση και προσδιορισμός των απαιτήσεων, σχεδιασμός συστήματος και λογισμικού, υλοποίηση και έλεγχος υποσυστήματος, ενοποίηση και έλεγχος συστήματος και λειτουργία και συντήρηση. Το μοντέλο αυτό απαιτεί ότι για να προχωρήσει η επόμενη φάση, θα πρέπει η προηγούμενη να έχει αναθεωρηθεί και επικυρωθεί. Αντίθετα το *Scrum* υλοποιεί όλες τις φάσεις αυτές μαζί σε κάθε *Sprint* και δεν απαιτεί να ολοκληρωθεί η μία για να αρχίσει η άλλη, πράγμα που το κάνει ευέλικτο.

Ορίσαμε δύο κύριες ομάδες ως στόχο του KICKOFF καμβά. Από τη μία πλευρά, θέλουμε να συμπεριλάβουμε επαγγελματίες στην διαχείριση του έργου (project management), χωρισμένους στους ενεργείς διευθυντές του έργου (project managers) και στα γραφεία υποστήριξης αυτού. Από την άλλη πλευρά, ένας άλλος σημαντικός στόχος είναι οι νέοι ιδιώτες επιχειρηματίες και οι μικρές ομάδες, οι οποίοι θέλουν να έχουν τις πρώτες εμπειρίες τους στο project management και μπορεί να γίνουν οι επόμενοι επαγγελματίες project managers. Οι δύο διαφορετικές περιπτώσεις των πελατών εκτιμούν και διαφορετικές πτυχές του KICKOFF καμβά. Παρακάτω καθορίσαμε τα χαρακτηριστικά των διαφορετικών ομάδων των πελατών, ώστε να προσδιοριστούν οι απαιτήσεις και να προσαρμοστεί το marketing πάνω σε αυτές. Με βάση αυτές τις ομάδες παράχθηκαν τα user stories των απαιτήσεων του project μας.





Εικόνα 4.1.2.1 Συνοπτική περιγραφή των χρηστών του καμβά

Στο *Scrum* λοιπόν η καταγραφή των απαιτήσεων του πελάτη γίνεται μέσω των *user stories*. Η ανάπτυξη της εφαρμογής μας, είχε πέντε *releases*, όπου κάθε *release* αποτελούταν από κάποια *user stories* τα οποία ανήκαν σε *Sprints*. Οι πίνακες παρακάτω παρουσιάζουν τα *user stories* που υπήρχαν σε κάθε *release* και αναφέρονται στη δουλειά που έγινε από εμάς.

1ο Release, Sprint 1
Ως χρήστης θέλω ένα πρότυπο (template) καμβά, βασισμένο στο web ώστε να μπορώ να το ανοίξω σε φυλλομετρητή και να δω τα διαφορετικά τμήματα.
Ως χρήστης θέλω να διαβάζω επιπλέον πληροφορίες σε κάθε τμήμα του καμβά έτσι ώστε να χρησιμοποιήσω τον καμβά όπως πρέπει.

Πίνακας 4.1.2 Τα user stories του πρώτου release

2ο Release, Sprints 2,3,4,5,6
Ως χρήστης θέλω να είμαι σε θέση να εγγράφομαι, ώστε να μπορώ να ξεκινήσω το δικό μου νέο καμβά
Ως χρήστης θέλω να συνδέομαι, ώστε να μπορώ να έχω πρόσβαση στο δικό μου καμβά
Ως χρήστης θέλω έναν εύκολο σχεδιασμό που θα δείχνει μια απλή σειρά έτσι ώστε να ξέρω με ποια περιοχή να αρχίσω και ποια να ακολουθήσω μετά.
Ως χρήστης θέλω να βλέπω άμεσα το λογότυπο της ιστοσελίδας, έτσι ώστε να ξέρω πού είμαι
Ως χρήστης θέλω να φθάνω την σελίδα προορισμού του καμβά KICK-OFF αν έχω το URL έτσι ώστε να μπορώ να πάρω περισσότερες πληροφορίες σχετικά με το προϊόν και την εταιρεία.

Ως χρήστης θέλω να πηγαίνω εύκολα στον καμβά μου, έτσι ώστε να αρχίσω να εργάζομαι
Ως test-χρήστης θέλω μία καθαρή από σφάλματα landing page
Ως χρήστης θέλω μια γερμανική και μια αγγλική έκδοση της landing page.
Ως test-χρήστης θέλω ένα ελκυστικό σχεδιασμό της landing page που να είναι σύγχρονο, ομοιογενές και σύμφωνο με τον οδηγό στιλ της UWS.
Ανατροφοδότηση για το landing page από το Sprint Review

Πίνακας 4.1.3 Τα user stories του δεύτερου release

3ο Release, Sprints 7,8,9,10
Ως χρήστης θέλω να κατανοώ άμεσα τη δομή πίσω από τον καμβά έτσι καταλαβαίνω τις αρχές του
Ως χρήστης θέλω να είμαι σε θέση να εγγράφομαι, ώστε να μπορώ να ξεκινήσω το δικό μου νέο καμβά
Ως διαχειριστής θέλω να καταργώ χρήστες από τον καμβά, αν έκανα λάθος.

Πίνακας 4.1.4 Τα user stories του τρίτου release

4ο Release, Sprints 11,12,13
Ως χρήστης θέλω ένα ωραίο και καθαρό από σφάλματα ταμπλό (dashboard)
Ως χρήστης θέλω να προειδοποιούμαι αν έβαλα λάθος στοιχεία σύνδεσης
Ως χρήστης θέλω να έχω ένα popup παράθυρο για να προσθέτω ένα νέο καμβά, όπου θα μπορώ να καλώ άμεσα τους ανθρώπους.
Ως χρήστης θέλω μια επισκόπηση των ανθρώπων που συνδέονται στον καμβά.
Ως χρήστης θέλω να προσθέτω άμεσα πολλά μέλη, έτσι ώστε να μην χρειάζεται να ανοίγω το popup κάθε φορά.
Ως χρήστης θέλω μια πλήρη επισκόπηση όλων των έργων μου
Ως χρήστης θέλω να αλλάζω το προφίλ, το όνομα χρήστη και διεύθυνση ηλεκτρονικού ταχυδρομείου μου.
Ως εγγεγραμμένος χρήστης θέλω να παίρνω μια πρόσκληση σε ένα νέο καμβά, αν κάποιος με κάλεσε, είτε με email ή με ειδοποίηση στον καμβά
Ως μη εγγεγραμμένος χρήστης θέλω να παίρνω μια πρόσκληση στον KICKOFF καμβά, αν κάποιος με προσκάλεσε σε ένα καμβά
Ως χρήστης θέλω να λαμβάνω ένα email επιβεβαίωσης μετά την εγγραφή

Πίνακας 4.1.5 Τα user stories του τέταρτου release

5ο Release, Sprints 14,15,16,17,18
Ως χρήστης θέλω μια επισκόπηση όλων των ομάδων μου
Ως χρήστης θέλω μια πλήρη επισκόπηση όλων των έργων μου
Ως διαχειριστής θέλω να καταργώ χρήστες από τον καμβά, αν έκανα λάθος.
Ως εγγεγραμμένος χρήστης θέλω να παίρνω μια πρόσκληση σε ένα νέο καμβά, αν κάποιος με κάλεσε, είτε με email ή με ειδοποίηση στον καμβά
Ως μη εγγεγραμμένος χρήστης θέλω να παίρνω μια πρόσκληση στον KICKOFF καμβά, αν κάποιος με προσκάλεσε σε ένα καμβά
Ως χρήστης θέλω να λαμβάνω μια προειδοποίηση αν δεν μπορούσα να προσθέσω ένα χρήστη σε ένα έργο, ώστε να ξέρω τι είναι λάθος.
Ως χρήστης θέλω να έχω μια επιβεβαίωση ότι προστέθηκε με επιτυχία ένα μέλος στο έργο, έτσι ώστε να ξέρω ότι όλα είναι μια χαρά.
Ως χρήστης θέλω προτάσεις μετά την πληκτρολόγηση των πρώτων γράμματών για να προσθέσω ένα μέλος.
Ως χρήστης θέλω να εισάγω χρήστες από τα ονόματά προφίλ τους
Ως χρήστης θέλω να λαμβάνω ένα email επιβεβαίωσης μετά την εγγραφή
Ως χρήστης θέλω να αλλάζω το προφίλ, το όνομα χρήστη και διεύθυνση ηλεκτρονικού ταχυδρομείου μου.
Ως χρήστης θέλω να επεξεργάζομαι τις πληροφορίες του έργου, αν θέλω να αλλάξω κάτι.

Ως χρήστης θέλω να αλλάζω τις ρυθμίσεις ειδοποίησης μου έτσι ώστε να μην ενοχλούμαι από email

Πίνακας 4.1.6 Τα user stories του πέμπτου release

Κάθε ένα από τα παραπάνω *user stories* αποτελείται από επιμέρους *καθήκοντα* (*tasks*) και *κριτήρια αποδοχής* (*acceptance criteria*). Τα *καθήκοντα* είναι μικρότερες εργασίες οι οποίες πρέπει να ολοκληρωθούν ώστε να ολοκληρωθεί και το *user story*. Τα κριτήρια αποδοχής ελέγχονται μετά την ολοκλήρωση των *καθηκόντων* και είναι ορισμένες προϋποθέσεις οι οποίες πρέπει να πληρούνται ώστε το *user story* να θεωρηθεί ολοκληρωμένο και να αφαιρεθεί από το *product backlog*. Οι παρακάτω πίνακες δείχνουν αναλυτικά τα *καθήκοντα* και τα *κριτήρια αποδοχής* για κάθε ένα από τα *user stories* μας. Με μπλε είναι το κάθε *user story*,με κίτρινο τα *καθήκοντα* (*tasks*) και με άσπρο τα *κριτήρια αποδοχής* (*acceptance criteria*).

1ο Release, Sprint 1

Ως χρήστης θέλω ένα πρότυπο (template) καμβά, βασισμένο στο web ώστε να μπορώ να το ανοίξω σε φυλλομετρητή και να δω τα διαφορετικά τμήματα.	
1. Επικοινώνησε με τους σχεδιαστές για το πώς πρέπει να δείχνει 2. Σχεδίασε μια mobile έκδοση του καμβά	Χρησιμοποιείται ένας οδηγός στυλ από την ομάδα σχεδιασμού; Υλοποιείται μία πλοήγησης; Έχει δημιουργηθεί μία κεφαλίδα (header); Έχει οριστεί μια θέση για το λογότυπο; Υπάρχει η δυνατότητα προσθήκης νέας ανάρτησης; Είναι σχεδιασμένος ο kickoff καμβάς; Χρησιμοποιείται footer; Έχει σχεδιαστεί μία σημείωση-παράδειγμα;

Πίνακας 4.1.7 1ο release-1ο user story

Ως χρήστης θέλω να διαβάζω επιπλέον πληροφορίες σε κάθε τμήμα του καμβά έτσι ώστε να χρησιμοποιήσω τον καμβά όπως πρέπει.	
1. Επικοινώνησε με τους σχεδιαστές 2. Αντέγραψε τις πληροφορίες από τον τρέχων καμβά	Υπάρχει ένα popup παράθυρο; Έχει αποφασιστεί αν το κουμπί πληροφοριών θα ενεργοποιείται με hover ή κάνοντας κλικ πάνω του; Έχει περιεχόμενο Έχει σχεδιαστεί ένα popup παράθυρο

Πίνακας 4.1.8 1ο release-2ο user story

2ο Release, Sprints 2,3,4,5,6

Ως χρήστης θέλω να είμαι σε θέση να εγγράφομαι, ώστε να μπορώ να ξεκινήσω το δικό μου νέο καμβά	
1. Δημιούργησε μία φόρμα εγγραφής με τα κατάλληλα πεδία και ελέγχους στο frontend 2. Υπέβαλε τα δεδομένα από τη φόρμα εγγραφής στο API	Υπάρχει μία φόρμα για να γραφτείς (όνομα, email, κωδικός,…); Τα δεδομένα της εγγραφής στέλνονται στο API; Αποθηκεύονται τα δεδομένα στη βάση δεδομένων; Δημιουργείται ένας νέος καμβάς στη βάση δεδομένων με τον εγγεγραμμένο χρήστη συνδεδεμένο σε αυτόν;

	Στέλνεται ο χρήστης στον δικό του καμβά αμέσως μετά την εγγραφή; Υπάρχει προειδοποίηση αν κάποιο πεδίο είναι άδειο; Υπάρχει προειδοποίηση εάν το πεδίο με το email δεν περιέχει το @; Υπάρχει προειδοποίηση εάν το πεδίο με το email δεν περιέχει .de, .com κ.τ.λ.; Υπάρχει προειδοποίηση εάν ο κωδικός είναι πολύ μικρός; Υπάρχει προειδοποίηση εάν υπάρχει λογαριασμός με τα ίδια στοιχεία; Συνδέεται αυτόματα ο χρήστης αμέσως μετά την εγγραφή; Παίρνει ο χρήστης ένα email επιβεβαίωσης μετά την εγγραφή;

Πίνακας 4.1.9 2ο release-1ο user story

Ως χρήστης θέλω να συνδέομαι, ώστε να μπορώ να έχω πρόσβαση στο δικό μου καμβά	
1. Δημιούργησε μία φόρμα σύνδεσης στην εφαρμογή 2. Υπέβαλε τα δεδομένα σύνδεσης στο API	Είναι συνδεδεμένος ο χρήστης μετά την υποβολή των σωστών διαπιστευτηρίων; Ελέγχονται τα δεδομένα του χρήστη μέσω μίας API κλήσης; Στέλνεται ο χρήστης σε έναν από τους δικούς του καμβάδες μετά από επιτυχή σύνδεση; Υπάρχει προειδοποίηση εάν τα δεδομένα είναι λάθος;

Πίνακας 4.1.10 2ο release-2ο user story

Ως χρήστης θέλω έναν εύκολο σχεδιασμό που θα δείχνει μια απλή σειρά έτσι ώστε να ξέρω με ποια περιοχή να αρχίσω και ποια να ακολουθήσω μετά.	
1. Υλοποίησε τον επανασχεδιασμένο καμβά	Έχει σταλθεί η επανασχεδιασμένη έκδοση στους προγραμματιστές;

Πίνακας 4.1.11 2ο release-3ο user story

Ως χρήστης θέλω να βλέπω άμεσα το λογότυπο της ιστοσελίδας, έτσι ώστε να ξέρω πού είμαι	
1. Υλοποίησε το λογότυπο	Έχει σταλθεί το λογότυπο στους προγραμματιστές;

Πίνακας 4.1.12 2ο release-4ο user story

Ως χρήστης θέλω να φθάνω την σελίδα προορισμού του καμβά KICK-OFF αν έχω το URL έτσι ώστε να μπορώ να πάρω περισσότερες πληροφορίες σχετικά με το προϊόν και την εταιρεία.

1. Υλοποίησε τον σχεδιασμό 2. Βρες ένα πρότυπο (template) για τη σελίδα προορισμού και να μίλα με τους σχεδιαστές	Βλέπει ο χρήστης την ιστοσελίδα εάν πληκτρολογήσει το URL; Είναι ο σχεδιασμός της αποκριτικός (responsive);

Πίνακας 4.1.13 2ο release-5ο user story

Ως χρήστης θέλω να πηγαίνω εύκολα στον καμβά μου, έτσι ώστε να αρχίσω να εργάζομαι	
1. Υλοποίησε ένα κουμπί εκκίνησης που θα στέλνει τον χρήστη στον καμβά	Ο χρήστης μπορεί να κάνει κλικ στο κουμπί και ο καμβάς θα ανοίξει

Πίνακας 4.1.14 2ο release-6ο user story

Ως test-χρήστης θέλω μία καθαρή από σφάλματα landing page	
1. Έλεγξε όλους τους συνδέσμους και τα κουμπιά 2. Φτιάξε το πρόβλημα με το μενού 3. Κάνε το λογότυπο να εμφανίζεται 4. Φτιάξε το σφάλμα με το κουμπί για τις μικρότερες οθόνες στο τμήμα «our mission» 5. Φτιάξε το πρόβλημα για τις μικρότερες οθόνες, στον πίνακα τιμών 6. Φτιάξε το σφάλμα της εικόνας στο τμήμα benefits	Είναι όλα τα σφάλματα διορθωμένα;

Πίνακας 4.1.15 2ο release-7ο user story

Ως χρήστης θέλω μια γερμανική και μια αγγλική έκδοση της landing page.	
1. Υλοποίησε ένα κουμπί για να αλλάζει μεταξύ των γλωσσών 2. Αντέγραψε την ιστοσελίδα και είσαγε το Γερμανικό περιεχόμενο	Είναι δυνατό να αλλάξεις μεταξύ των γλωσσών; Είναι η επιλεγμένη γλώσσα επισημασμένη; Συμβαίνει τίποτα εάν κάνω κλικ στη γλώσσα που είναι ήδη επιλεγμένη; Υπάρχει προειδοποίηση εάν τα δεδομένα είναι λάθος;

Πίνακας 4.1.16 2ο release-8ο user story

Ως test-χρήστης θέλω ένα ελκυστικό σχεδιασμό της landing page που να είναι σύγχρονο, ομοιογενές και σύμφωνο με τον οδηγό στιλ της UWS.

1. Ενσωμάτωσε τον ελκυστικό σχεδιασμό που έφτιαξαν οι σχεδιαστές 2. Άλλαξε το κουμπί σύνδεσης σε ένα «Try Kick-off canvas» κουμπί και επισήμανε το 3. Κάνε το τμήμα επικοινωνίας πιο φωτεινό 4. Άλλαξε το χρώμα στο υπόβαθρο (background) από γκρι σε άσπρο ή ακόμα καλύτερα χρησιμοποίησε όλο το πλάτος 5. Γράψε «10€/month» και «80€/month» (ίδιο ύψος για όλα τα γράμματα) 6. Κάνε πιο ορατή την εικόνα δίπλα στο τμήμα benefits	Έχει υλοποιηθεί σωστά ο σχεδιασμός; Είναι σύγχρονος; Είναι ελκυστικός; Είναι ομοιογενής και σύμφωνος με τον οδηγό στυλ της UWS;

Πίνακας 4.1.17 2ο release-9ο user story

Ανατροφοδότηση για το landing page από το Sprint Review	
1. Υλοποίησε το σχέδιο της Heidi για το footer 2. Ίδιο μέγεθος για όλες τις επικεφαλίδες, ίδια χρώματα κ.τ.λ. 3. Υλοποίησε τα εικονίδια του Thomas 4. Πρόσθεσε κουκίδες όπως του tumblr 5. Φτιάξε ένα κουμπί για τη σύνδεση (login) 6. Πρόσθεσε ένα κουμπί benefits στο μενού 7. Άλλαξε τα χρώματα των πινάκων τιμών και πρόσθεσε περισσότερες πληροφορίες για την ακαδημαϊκή έκδοση 8. Φτιάξε το λογότυπο	Είναι όλες οι επικεφαλίδες ίδιες; Έχει υλοποιηθεί το σχέδιο για το footer; Έχουν υλοποιηθεί τα εικονίδια; Προστέθηκαν οι κουκίδες; Υπάρχει κουμπί για τη σύνδεση; Υπάρχει κουμπί benefits στο μενού; Έχουν αλλαχθεί οι πίνακες τιμών; Διορθώθηκε το λογότυπο;

Πίνακας 4.1.18 2ο release-10ο user story

3ο Release, Sprints 7,8,9,10

Ως χρήστης θέλω να κατανοώ άμεσα τη δομή πίσω από τον καμβά έτσι καταλαβαίνω τις αρχές του	
1. Υλοποίησε το σχέδιο για το overlay 2. Πρόσθεσε animation κατά την εξαφάνιση του	Είναι το overlay του καμβά υλοποιημένο σύμφωνα με το σχέδιο; Εξαφανίζεται όταν γίνεται κλικ πάνω του; Υπάρχει κάποιο animation κατά την εξαφάνιση του;

Πίνακας 4.1.19 3ο release-1ο user story

Ως χρήστης θέλω να είμαι σε θέση να εγγράφομαι, ώστε να μπορώ να ξεκινήσω το δικό μου νέο καμβά	
1. Δημιούργησε μία φόρμα εγγραφής με τα κατάλληλα πεδία και ελέγχους στο frontend 2. Υπέβαλε τα δεδομένα από τη φόρμα εγγραφής στο API	Υπάρχει μία φόρμα για να γραφτείς (όνομα, email, κωδικός,…); Τα δεδομένα της εγγραφής στέλνονται στο API; Αποθηκεύονται τα δεδομένα στη βάση δεδομένων; Δημιουργείται ένας νέος καμβάς στη βάση δεδομένων με τον εγγεγραμμένο χρήστη συνδεδεμένο σε αυτόν; Στέλνεται ο χρήστης στον δικό του καμβά αμέσως μετά την εγγραφή; Υπάρχει προειδοποίηση αν κάποιο πεδίο είναι άδειο; Υπάρχει προειδοποίηση εάν το πεδίο με το email δεν περιέχει το @; Υπάρχει προειδοποίηση εάν το πεδίο με το email δεν περιέχει .de, .com κ.τ.λ.; Υπάρχει προειδοποίηση εάν ο κωδικός είναι πολύ μικρός; Υπάρχει προειδοποίηση εάν υπάρχει λογαριασμός με τα ίδια στοιχεία; Συνδέεται αυτόματα ο χρήστης αμέσως μετά την εγγραφή; Παίρνει ο χρήστης ένα email επιβεβαίωσης μετά την εγγραφή;

Πίνακας 4.1.20 3ο release-2ο user story

Ως διαχειριστής θέλω να καταργώ χρήστες από τον καμβά, αν έκανα λάθος.	
1. Εκχώρησε δικαίωμα διαγραφής χρηστών μόνο στον διαχειριστή 2. Δημιούργησε ένα κουμπί που θα διαγράφει τους χρήστες 3. Κάνε την API κλήση για τη διαγραφή των χρηστών από τη βάση	Υπάρχει κάποιο κουμπί που διαγράφει τους χρήστες; Εμφανίζεται αυτό το κουμπί μόνο στον διαχειριστή; Διαγράφονται χρήστες από τον καμβά, στη βάση;

Πίνακας 4.1.21 3ο release-3ο user story

4ο Release, Sprints 11,12,13

Ως χρήστης θέλω ένα ωραίο και καθαρό από σφάλματα ταμπλό (dashboard)	
1. Υλοποίησε το σχεδιασμό του πρώτου συνδέσμου 2. Ενεργοποίησε όλους τους συνδέσμους, έτσι ώστε ο χρήστης να μπορεί να κάνει κλικ πάνω τους	Είναι υλοποιημένος ο σχεδιασμός του πρώτου συνδέσμου; Λειτουργούν όλοι οι σύνδεσμοι; Έχουν διορθωθεί όλα τα σφάλματα;

Πίνακας 4.1.22 4ο release-1ο user story

Ως χρήστης θέλω να προειδοποιούμαι αν έβαλα λάθος στοιχεία σύνδεσης

1. Προειδοποίησε το χρήστη εάν ο κωδικός δεν ταιριάζει με το email	Εμφανίζεται προειδοποιητικό μήνυμα αν ο κωδικός δεν ταιριάζει με το email; Ο χρήστης δεν εισέρχεται στο λογαριασμό του αν ο κωδικός δεν ταιριάζει με το email

Πίνακας 4.1.23 4ο release-2ο user story

Ως χρήστης θέλω να έχω ένα popup παράθυρο για να προσθέτω ένα νέο καμβά, όπου θα μπορώ να καλώ άμεσα τους ανθρώπους.	
1. Φτιάξε ένα popup παράθυρο 2. Πρόσθεσε επιλογές όπως: πρόσκληση για μέλη ομάδας, όνομα καμβά, περιγραφή καμβά κ.τ.λ. 3. Συμπεριέλαβε το νέο popup παράθυρο στον καμβά 4. Συμπεριέλαβε το νέο popup παράθυρο στον ταμπλό (dashboard)	Υπάρχει το popup στον καμβά; Υπάρχει το popup στο ταμπλό (dashboard); Αναγράφεται στο popup τα στοιχεία του καμβά στον οποίο ανήκει; Μπορώ να προσκαλέσω νέα μέλη σε ένα καμβά μέσω του popup;

Πίνακας 4.1.24 4ο release-3ο user story

Ως χρήστης θέλω μια επισκόπηση των ανθρώπων που συνδέονται στον καμβά.	
1. Κάνε την API κλήση για να δεις τα τρέχοντα μέλη 2. Δείξε τα τρέχοντα δεδομένα από την API κλήση στο popup 3. Πρόσθεσε στυλ στο popup 4. Δημιούργησε ένα popup όταν κάνεις κλικ στα μέλη στον καμβά 5. Δώσε στυλ στο popup των μελών	Γίνεται με επιτυχία η API κλήση; Φαίνονται τα μέλη στον καμβά; Έχει το popup ωραίο σχεδιασμό; Δημιουργείται ένα popup παράθυρο όταν γίνεται κλικ στο παράθυρο;

Πίνακας 4.1.25 4ο release-4ο user story

Ως χρήστης θέλω να προσθέτω άμεσα πολλά μέλη, έτσι ώστε να μην χρειάζεται να ανοίγω το popup κάθε φορά.	
1. Δημιούργησε ένα κουμπί ώστε να προστεθούν περισσότερες είσοδοι (selection lists) στο «add members to canvas» popup 2. Κάνε την API κλήση για την προσθήκη πολλών μελών	Υπάρχει το κουμπί δημιουργίας περισσότερων selection lists; Δημιουργούνται περισσότερα selection lists κατά το πάτημα του κουμπιού; Υπάρχει τρόπος να αφαιρεθεί ένα selection list αν γίνει λάθος; Προστίθενται πολλά μέλη ταυτόχρονα;

Πίνακας 4.1.26 4ο release-5ο user story

Ως χρήστης θέλω μια πλήρη επισκόπηση όλων των έργων μου

1. Δημιούργησε ένα κουμπί στο dashboard όπου θα φαίνονται όλα τα έργα ενός χρήστη 2. Δώσε στυλ στην επισκόπηση των έργων	Οδηγεί το κουμπί στην επισκόπηση όλων των έργων; Έχει η επισκόπηση ωραία εμφάνιση Αν κάνω κλικ σε ένα έργο οδηγούμαι στον αντίστοιχο καμβά;

Πίνακας 4.1.27 4ο release-6ο user story

Ως χρήστης θέλω να αλλάζω το προφίλ, το όνομα χρήστη και διεύθυνση ηλεκτρονικού ταχυδρομείου μου.	
1. Υλοποίησε τον σχεδιασμό των ρυθμίσεων (settings)	Αν κάνω κλικ στο κουμπί «ρυθμίσεις» οδηγούμαι σε μία φόρμα όπου μπορώ να επεξεργαστώ τα στοιχεία του προφίλ μου; Μπορώ αλλάξω το όνομα χρήστη; Μπορώ αλλάξω τη διεύθυνση ηλεκτρονικού ταχυδρομείου; Αποθηκεύονται οι αλλαγές επιτυχώς;

Πίνακας 4.1.28 4ο release-7ο user story

Ως εγγεγραμμένος χρήστης θέλω να παίρνω μια πρόσκληση σε ένα νέο καμβά, αν κάποιος με κάλεσε, είτε με email ή με ειδοποίηση στον καμβά	
1. Υλοποίησε το πρότυπο σχέδιο του email από τους σχεδιαστές σε HTML πρωτότυπο 2. Πρόσθεσε κείμενο στο πρότυπο 3. Κάνε την API κλήση για την αποστολή του email 4. Δημιούργησε ένα εικονίδιο για τις ειδοποιήσεις 5. Πρόσθεσε στο εικονίδιο τον αριθμό των ειδοποιήσεων που έχει ένας χρήστης 6. Κάνε την API κλήση για την λήψη των ειδοποιήσεων	Είναι ο σχεδιασμός του προτύπου σύμφωνος με το σχέδιο των σχεδιαστών; Υπάρχει κείμενο στο πρότυπο; Δουλεύει η υλοποίηση του προτύπου σε email μηνύματα; Στέλνεται email με πρόσκληση σε καμβά αν ο χρήστης είναι εγγεγραμμένος; Υπάρχει στον καμβά ένα εικονίδιο για τις ειδοποιήσεις; Εμφανίζεται ο ακριβής αριθμός των ειδοποιήσεων; Αν κάνω κλικ πάνω στο εικονίδιο εμφανίζεται το όνομα του χρήστη και το όνομα του καμβά στον οποίο με κάλεσαν; Αν κάνω κλικ στο όνομα του καμβά που με κάλεσαν οδηγούμαι σε αυτόν; Αν διαβάσω τις ειδοποιήσεις μου, εξαφανίζεται ο αριθμός των ειδοποιήσεων από το εικονίδιο;

Πίνακας 4.1.29 4ο release-8ο user story

Ως μη εγγεγραμμένος χρήστης θέλω να παίρνω μια πρόσκληση στον KICKOFF καμβά, αν κάποιος με προσκάλεσε σε ένα καμβά

1. Υλοποίησε το πρότυπο σχέδιο του email από τους σχεδιαστές σε HTML πρότυπο 2. Πρόσθεσε κείμενο στο πρότυπο 3. Κάνε την API κλήση για την αποστολή του email	Είναι ο σχεδιασμός του προτύπου σύμφωνος με το σχέδιο των σχεδιαστών; Υπάρχει κείμενο στο πρότυπο; Δουλεύει η υλοποίηση του προτύπου σε email μηνύματα; Στέλνεται email με πρόσκληση σε καμβά αν ο χρήστης είναι μη εγγεγραμμένος; Αν κάνω κλικ στο όνομα του καμβά που με κάλεσαν οδηγούμαι σε αυτόν;

Πίνακας 4.1.30 4ο release-9ο user story

Ως χρήστης θέλω να λαμβάνω ένα email επιβεβαίωσης μετά την εγγραφή	
1. Υλοποίησε το πρότυπο σχέδιο του email από τους σχεδιαστές σε HTML πρότυπο 2. Πρόσθεσε κείμενο στο πρωτότυπο 3. Κάνε την API κλήση για την αποστολή του email	Είναι ο σχεδιασμός του προτύπου σύμφωνος με το σχέδιο των σχεδιαστών; Υπάρχει κείμενο στο πρότυπο; Δουλεύει η υλοποίηση του προτύπου σε email μηνύματα; Στέλνεται email επιβεβαίωσης μετά την εγγραφή του χρήστη

Πίνακας 4.1.31 4ο release-10ο user story

5ο Release, Sprints 14,15,16,17,18

Ως χρήστης θέλω μια επισκόπηση όλων των ομάδων μου	
1. Υλοποίησε ένα πρωτότυπο σχέδιο για τις ομάδες 2. Υλοποίησε το πρωτότυπο σχέδιο στην εφαρμογή	Υπάρχει ένα κουμπί στο dashboard που οδηγεί στην επισκόπηση των ομάδων του χρήστη; Είναι ο σχεδιασμός σύμφωνος με το σχέδιο από τους σχεδιαστές;

Πίνακας 4.1.32 5ο release-1ο user story

Ως χρήστης θέλω μια πλήρη επισκόπηση όλων των έργων μου	
1. Δημιούργησε ένα κουμπί στο dashboard όπου θα φαίνονται όλα τα έργα ενός χρήστη 2. Δώσε στυλ στην επισκόπηση των έργων	Οδηγεί το κουμπί στην επισκόπηση όλων των έργων; Έχει η επισκόπηση ωραία εμφάνιση Αν κάνω κλικ σε ένα έργο οδηγούμαι στον αντίστοιχο καμβά;

Πίνακας 4.1.33 5ο release-2ο user story

Ως διαχειριστής θέλω να καταργώ χρήστες από τον καμβά, αν έκανα λάθος.

1. Υλοποίησε το σχεδιασμό 2. Εκχώρησε δικαίωμα διαγραφής χρηστών μόνο στον διαχειριστή 3. Δημιούργησε ένα κουμπί που θα διαγράφει τους χρήστες 4. Κάνε την API κλήση για τη διαγραφή των χρηστών από τη βάση 5. Αφαίρεσε τον χρήστη από την όψη	Υπάρχει κάποιο κουμπί που διαγράφει τους χρήστες; Εμφανίζεται αυτό το κουμπί μόνο στον διαχειριστή; Διαγράφονται χρήστες από τον καμβά, στη βάση; Διαγράφεται ο χρήστης από την όψη;

Πίνακας 4.1.34 5ο release-3ο user story

Ως εγγεγραμμένος χρήστης θέλω να παίρνω μια πρόσκληση σε ένα νέο καμβά, αν κάποιος με κάλεσε, είτε με email ή με ειδοποίηση στον καμβά	
1. Επανασχεδίασε το email πρότυπο 2. Κάνε την API κλήση για την λήψη των ειδοποιήσεων 3. Ειδοποίησε τον προσκεκλημένο χρήστη στην όψη	Είναι ο σχεδιασμός του προτύπου σύμφωνος με το σχέδιο των σχεδιαστών; Υπάρχει κείμενο στο πρότυπο; Δουλεύει η υλοποίηση του προτύπου σε email μηνύματα; Στέλνεται email με πρόσκληση σε καμβά αν ο χρήστης είναι εγγεγραμμένος; Ειδοποιείται ο χρήστης στην όψη; Υπάρχει στον καμβά ένα εικονίδιο για τις ειδοποιήσεις; Εμφανίζεται ο ακριβής αριθμός των ειδοποιήσεων; Αν κάνω κλικ πάνω στο εικονίδιο εμφανίζεται το όνομα του χρήστη και το όνομα του καμβά στον οποίο με κάλεσαν; Αν κάνω κλικ στο όνομα του καμβά που με κάλεσαν οδηγούμαι σε αυτόν; Αν διαβάσω τις ειδοποιήσεις μου, εξαφανίζεται ο αριθμός των ειδοποιήσεων από το εικονίδιο;

Πίνακας 4.1.35 5ο release-4ο user story

Ως μη εγγεγραμμένος χρήστης θέλω να παίρνω μια πρόσκληση στον KICKOFF καμβά, αν κάποιος με προσκάλεσε σε ένα καμβά	
1. Επανασχεδίασε το email πρότυπο	Είναι ο σχεδιασμός του προτύπου σύμφωνος με το σχέδιο των σχεδιαστών; Υπάρχει κείμενο στο πρότυπο; Δουλεύει η υλοποίηση του προτύπου σε email μηνύματα; Στέλνεται email με πρόσκληση σε καμβά αν ο χρήστης είναι μη εγγεγραμμένος; Αν κάνω κλικ στο όνομα του καμβά που με κάλεσαν οδηγούμαι σε αυτόν;

Πίνακας 4.1.36 5ο release-5ο user story

Ως χρήστης θέλω να λαμβάνω μια προειδοποίηση αν δεν μπορούσα να προσθέσω ένα χρήστη σε ένα έργο, ώστε να ξέρω τι είναι λάθος.

1. Πρόσθεσε τα κατάλληλα μηνύματα στο popup	Εμφανίζεται μήνυμα όταν ο χρήστης είναι ήδη μέλος του καμβά; Εμφανίζεται μήνυμα αν ο χρήστης δεν υπάρχει;

Πίνακας 4.1.37 5ο release-6ο user story

Ως χρήστης θέλω να έχω μια επιβεβαίωση ότι προστέθηκε με επιτυχία ένα μέλος στο έργο, έτσι ώστε να ξέρω ότι όλα είναι μια χαρά.	
1. Πρόσθεσε ένα μήνυμα για την επιτυχή προσθήκη	Εμφανίζεται το μήνυμα αν προστεθεί επιτυχώς ένα μέλος; Εξαφανίζεται μετά από ένα χρονικό διάστημα;

Πίνακας 4.1.38 5ο release-7ο user story

Ως χρήστης θέλω προτάσεις μετά την πληκτρολόγηση των πρώτων γράμματών για να προσθέσω ένα μέλος.	
1. Κάνε την API κλήση για τις προτάσεις 2. Εφάρμοσε την αυτόματη συμπλήρωση στην όψη	Εμφανίζονται προτάσεις από email χρηστών αν αρχίσω να πληκτρολογώ τα πρώτα γράμματα από ένα email; Υπάρχει ο χώρος ώστε να εμφανίζονται τα προτεινόμενα email; Εμφανίζεται το ονοματεπώνυμο του χρήστη μαζί με το email του; Αν σβήσω τελείως τα γράμματα που έχω πληκτρολογήσει, εξαφανίζεται ο χώρος για τα ονόματα; Εμφανίζεται ξανά όταν πληκτρολογήσω πάλι;

Πίνακας 4.1.39 5ο release-8ο user story

Ως χρήστης θέλω να εισάγω χρήστες από τα ονόματά προφίλ τους	
1. Κάνε την API κλήση για τις προτάσεις 2. Εφάρμοσε την αυτόματη συμπλήρωση στην όψη	Εμφανίζονται προτάσεις από ονόματα χρηστών αν αρχίσω να πληκτρολογώ τα πρώτα γράμματα από ένα όνομα; Υπάρχει ο χώρος ώστε να εμφανίζονται τα προτεινόμενα ονόματα; Εμφανίζεται το email του χρήστη μαζί με το ονοματεπώνυμο του; Αν σβήσω τελείως τα γράμματα που έχω πληκτρολογήσει, εξαφανίζεται ο χώρος για τα ονόματα; Εμφανίζεται ξανά όταν πληκτρολογήσω πάλι;

Πίνακας 4.1.40 5ο release-9ο user story

Ως χρήστης θέλω να λαμβάνω ένα email επιβεβαίωσης μετά την εγγραφή

1. Επανασχεδίασε το πρότυπο email	Είναι ο σχεδιασμός του προτύπου σύμφωνος με το σχέδιο των σχεδιαστών; Υπάρχει κείμενο στο πρότυπο; Δουλεύει η υλοποίηση του προτύπου σε email μηνύματα; Στέλνεται email επιβεβαίωσης μετά την εγγραφή του χρήστη

Πίνακας 4.1.41 5ο release-10ο user story

Ως χρήστης θέλω να αλλάζω το προφίλ, το όνομα χρήστη και διεύθυνση ηλεκτρονικού ταχυδρομείου μου.	
1. Κάνε την API κλήση για να πάρεις τα δεδομένα προφίλ του χρήστη 2. Εγκατέστησε την initial.js για να δείχνει τα αρχικά του ονόματος των χρηστών αν δεν έχουν ανεβάσει εικόνα προφίλ ακόμα 3. Κάνε την API κλήση για να αλλάζεις τα δεδομένα προφίλ του χρήστη 4. Δείξε τις πληροφορίες του χρήστη στην όψη 5. Δημιούργησε ένα popup για τον χρήστη για να αλλάζει την εικόνα προφίλ του	Αν κάνω κλικ στο κουμπί «ρυθμίσεις» οδηγούμαι σε μία φόρμα όπου μπορώ να επεξεργαστώ τα στοιχεία του προφίλ μου; Μπορώ αλλάξω το όνομα χρήστη; Μπορώ αλλάξω τη διεύθυνση ηλεκτρονικού ταχυδρομείου; Αποθηκεύονται οι αλλαγές επιτυχώς; Φαίνεται ως εικόνα προφίλ του χρήστη τα αρχικά του ονόματος και του επιθέτου του αν δεν έχει ανεβάσει φωτογραφία; Μπορεί ο χρήστης να αλλάξει την εικόνα προφίλ του; Φαίνονται τα στοιχεία του χρήστη στην όψη;

Πίνακας 4.1.42 5ο release-11ο user story

Ως χρήστης θέλω να επεξεργάζομαι τις πληροφορίες του έργου, αν θέλω να αλλάξω κάτι.	
1. Φτιάξε ένα popup και δείξε τις πληροφορίες του κάθε έργου 2. Δείξε τα μέλη του έργου 3. Φτιάξε ένα κουμπί αποθήκευσης για να αλλάζει δεδομένα	Δημιουργείται ένα popup παράθυρο; Δείχνει τις πληροφορίες του έργου; Δείχνει τα μέλη του έργου; Μπορώ να αποθηκεύσω τις αλλαγές που έχω κάνει; Μπορώ να αλλάξω το όνομα του έργου; Μπορώ να προσθέσω μέλη; Μπορώ να αφαιρέσω μέλη;

Πίνακας 4.1.43 5ο release-12ο user story

Ως χρήστης θέλω να αλλάζω τις ρυθμίσεις ειδοποίησης μου έτσι ώστε να μην ενοχλούμαι από email	
1. Δημιούργησε ένα κουτί (checkbox) όπου ο χρήστης θα ενεργοποιεί και απενεργοποιεί τις ειδοποιήσεις από email 2. Κάνε την API κλήση για την ενεργοποίηση/απενεργοποίηση των ειδοποιήσεων	Υπάρχει ένα checkbox στις ρυθμίσεις για να επιλέγει ο χρήστης; Αν το επιλέξει τότε θα λαμβάνει ειδοποιήσεις μέσω email; Αν δεν το επιλέξει θα είναι απενεργοποιημένες οι ειδοποιήσεις;

Πίνακας 4.1.44 5ο release-13ο user story

Αυτό που παρατηρείται είναι ότι κάποια *user stories* υπάρχουν σε διαφορετικά releases με διαφορετικά όμως καθήκοντα. Αυτό συμβαίνει διότι κάποια από τα *user stories* δεν μπορούσαν να θεωρηθούν τελειωμένα σε εκείνο το release είτε δεν ολοκληρώθηκαν λόγω χρόνου ή γιατί μετά τις συσκέψεις που είχαμε οι φοιτητές μεταξύ μας αποφασίσαμε ότι πρέπει να προστεθούν και άλλα καθήκοντα ή να αλλάξει λίγο η πορεία του *user story* λόγω της αλλαγής των απαιτήσεων. Επίσης ένα release περιλαμβάνει πολλά *Sprints*. Αυτό σημαίνει ότι κατά τη διάρκεια ενός *Sprint* ένα *user story* μπορεί να υπήρχε πολλές φορές γιατί δεν ολοκληρωνόταν και περνούσε στο επόμενο *Sprint*. Στην περίπτωση αυτή όλα τα καθήκοντα ενός *user story* που υπήρχε σε πολλά *Sprints* ενός release συγκεντρώθηκαν σε έναν πίνακα στο αντίστοιχο release.

4.2 Υλοποίηση

4.2.1 Διαδικασία υλοποίησης

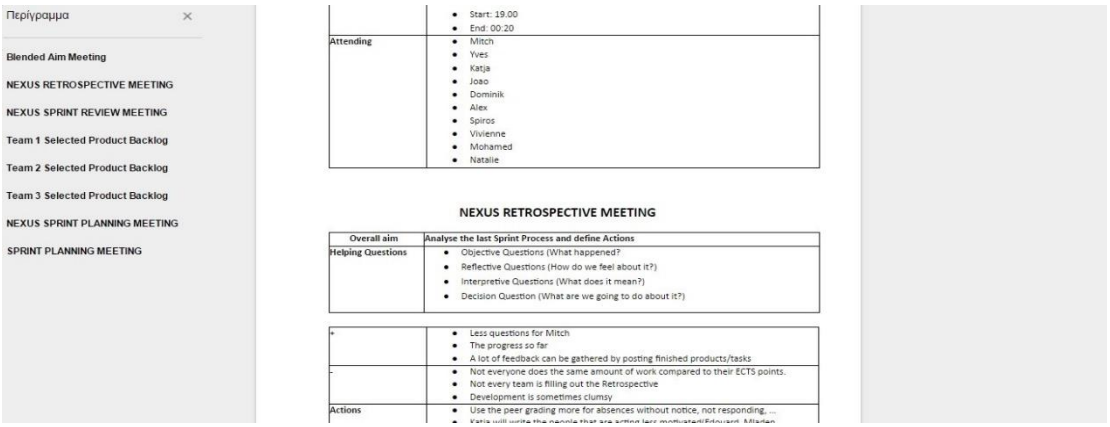
Στο Blended Aim 2016 πήραν μέρος 15 φοιτητές από πανεπιστήμια της Γερμανίας, της Αυστρίας, της Ελλάδας, της Πορτογαλίας, της Σκωτίας και του Βελγίου. Για την συνεργασία τους στην υλοποίηση του project έπρεπε να μελετήσουν και να ακολουθήσουν τους κανόνες και τις οδηγίες του Scrum Framework. Παραβαίνοντας το κανόνα που θέλει την scrum ομάδα να είναι από τρία έως εννέα άτομα, οι φοιτητές σχημάτισαν μία μοναδική ομάδα των 15 ατόμων με έναν product owner και έναν scrum master. Το πρώτο sprint έλαβε χώρα στο πανεπιστήμιο του Paderborn της Γερμανίας με διάρκεια της μίας εβδομάδας. Η διαδικασία του scrum κύλησε ομαλά με μερικές διαφωνίες μεταξύ των φοιτητών, καθώς όλοι βρίσκονταν στο ίδιο χώρο και τις ίδιες ώρες. Παρόλα αυτά, η διαδικασία δυσκόλεψε κατά την διάρκεια του δεύτερου sprint στο οποίοι οι φοιτητές είχαν επιστρέψει στη χώρα τους. Κάθε Τρίτη βράδι τα μέλη της ομάδας έπρεπε να πραγματοποιήσουν τα προκαθορισμένα, *sprint review meeting, sprint retrospective meeting* και *sprint planning meeting*.

Εικόνα 4.2.1.1 Συνεργασία των business, των σχεδιαστών και των προγραμματιστών

Όλη η εργασία ήταν χωρισμένη σε 18 sprints με διάρκεια της μίας εβδομάδας το καθένα. Η επικοινωνία ήταν καθημερινή και γραπτή μέσω του *Slack*. Ο τρόπος επικοινωνίας για τα meetings θα ήταν το Skype, όμως λόγω του ότι δεν δέχεται ομαδικές βιντεοκλήσεις των 15 ατόμων, οι συναντήσεις γινόντουσαν στο *Team Speak* μέσω ενός server. Τα

προβλήματα που αντιμετωπίσαμε με την ενιαία scrum ομάδα και τον team speak ήταν τα εξής:

- Τα meeting διαρκούσαν πολλές ώρες.
- Οι εργασίες (tasks) που είχε να υλοποιήσει κάθε μέλος δεν ήταν ξεκάθαρα.
- Υπήρχαν πολλές εξαρτήσεις ανάμεσα στα user stories.
- Δεν μπορούσαν όλοι να μιλούν ελεύθερα στα meeting λόγω του χρόνου και του θορύβου.
- Δεν γινόταν να υπάρχει οπτική επικοινωνία, ούτε κοινή χρήση οθόνης.
- Η υλοποίηση του project κυλούσε αργά.
- Ο Scrum master δεν μπορούσε να ελέγξει όλα τα άτομα.

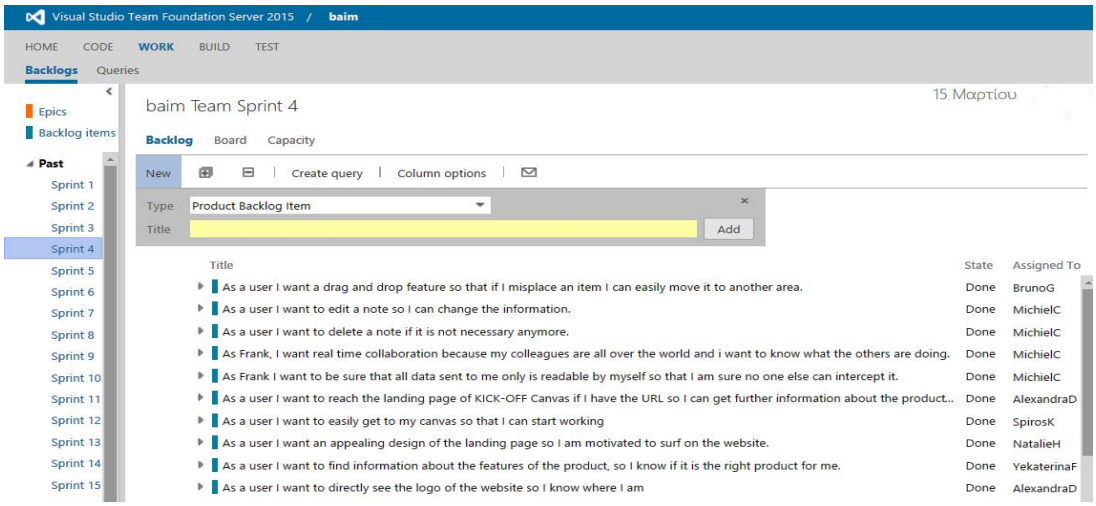


Εικόνα 4.2.1.2 Εβδομαδιαία Scrum Meeting

Για αυτό το λόγο αποφασίσαμε να χωριστούμε σε τρεις nexus scrum ομάδες. Κάθε ομάδα διαμορφωνόταν από πέντε φοιτητές κάθε είδους, σχεδιαστές, προγραμματιστές και business. Υπήρχε μόνο ένας product owner, όμως κάθε ομάδα είχε δικό της scrum master, ο οποίος ήταν υπεύθυνος και βρίσκει λύσεις σε προβλήματα και να ελέγχει αν όλα τα μέλη δουλεύουν και είναι στον σωστό δρόμο της υλοποίησης. Τα παραπάνω προβλήματα λύθηκαν, οι εξαρτήσεις μειώθηκαν και όλοι οι φοιτητές γνώριζαν ακριβώς τι πρέπει να κάνουν. Όσον αφορά τα meetings, αποφασίσαμε ότι κάθε εβδομάδα θα συμμετέχουν σε αυτά ο product owner και ένας αντιπρόσωπος από κάθε ομάδα, ο οποίος ήταν υπεύθυνος να γνωρίζει τι έχει κάνει ο καθένας, τι θα κάνει και που βρίσκει δυσκολίες. Όλα τα meetings λάμβαναν χώρα στο *Google Hangouts*, ώστε να υπάρχει οπτική επαφή και κοινή χρήση οθόνης. Με αυτό τον τρόπο ο χρόνος και η ποιότητα των συναντήσεων βελτιώθηκε σε μεγάλο βαθμό και η πρόοδος της υλοποίησης αυξήθηκε σημαντικά.

Όλα τα user stories βρισκόντουσαν στο *Visual Studio Team Foundation Server 2015*. Κάθε ομάδα είχε δικά της user stories, κάθε user story είχε δικά του tasks τα οποία ήταν ανατεθειμένα σε μέλη της ομάδας. Στο τέλος του κάθε sprint ο product owner έβαζε καινούργια user stories από το product backlog, τα οποία έπρεπε να υλοποιηθούν σε εκείνο το χρονικό διάστημα. Όταν ένα user story δεν είχα τελειώσει στο τέλος του τρέχων sprint, ο product owner το μετέφερε στο επόμενο. Αν και πάλι δεν υπήρχε κάποια πρόοδος στην υλοποίηση του τότε κάποιο άλλο μέλος της ομάδας αναλάμβανε να το φέρει εις πέρας αφαιρώντας το από εκείνον που το είχε αναλάβει αρχικά. Επιπλέον, υπήρχαν πέντε releases κατά τα οποία γινόταν ένα overview και μία αξιολόγηση του υπάρχον προϊόντος από τον υπεύθυνο της εταιρίας αλλά και από ομάδες φοιτητών του γερμανικού πανεπιστημίου, καταγράφοντας τυχόν προβλήματα και λεπτομέρειες που πρέπει να βελτιωθούν. Αυτά τα σχόλια ονομάζονταν *Bugs* και το άτομο στο οποίο ήταν ανατεθειμένα έπρεπε να ανατρέξει στον κώδικα και να προσθέσει, να αφαιρέσει ή να βελτιώσει αυτά που ζητήθηκαν από τους χρήστες. Με αυτό τον τρόπο το προϊόν βελτιώνονταν διαρκώς κατά την υλοποίηση του. Τέλος, τα κομμάτια κώδικα της εφαρμογής αποθηκεύονταν στο αποθετήριο (repository) που είχε δημιουργηθεί στον *Team Foundation Server* μέσω του *Git*. Το *Git* είναι ένα σύστημα

ελέγχου εκδόσεων (λέγεται και σύστημα ελέγχου αναθεωρήσεων ή σύστημα ελέγχου πηγαίου κώδικα) με έμφαση στην ταχύτητα, στην ακεραιότητα των δεδομένων και στην υποστήριξη για κατανεμημένες μη γραμμικές ροές εργασίας. Χάρη στο *Git* μπορούσαμε να ελέγχουμε τις εκδόσεις της εφαρμογής και να αναπτύσσουμε το λογισμικό μας με ταχύτητα και εξοικονόμηση χρόνου.



Εικόνα 4.2.1.3 User stories στο TFS

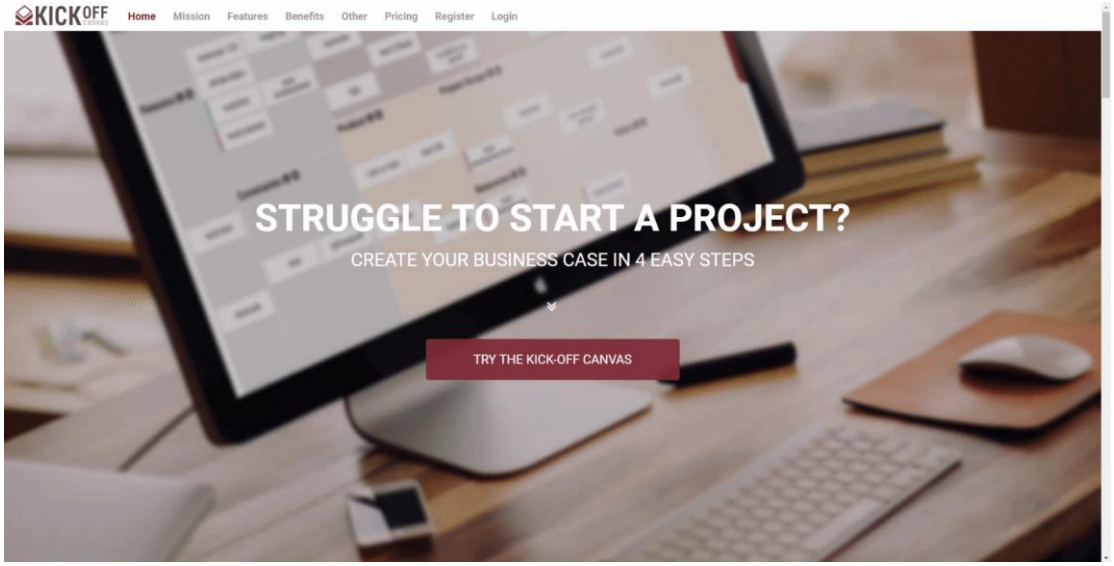
Οι εργασίες τις οποίες αναλάβαμε εμείς ήταν για το frontend της εφαρμογής. Για την υλοποίηση των εργασιών μας χρησιμοποιήσαμε το *bootstrap framework* στο landing page και το dashboard, ώστε να είναι responsive σε οποιαδήποτε συσκευή. Το *font awesome* για ποικιλία έτοιμων και ελεύθερων για χρήση εικονιδίων στο διαδίκτυο. Το angularJS και το jQuery χρησιμοποιήθηκαν για την προσθήκη λειτουργικότητας στην εφαρμογή. Το jQuery περισσότερο για εφέ, animation και άλλων λειτουργιών που πρόσφεραν πρακτικότητα στην εμφάνιση, ενώ το angularJS για την ανάπτυξη όλης της εφαρμογής. Η ολοκλήρωση κάθε user story μας είχε ως αποτέλεσμα μία ολοκληρωμένη ιστοσελίδα στην οποία ο χρήστης μπορεί να βρει πληροφορίες και να διαλέξει το πακέτο που επιθυμεί και ένα βασικό dashboard, στο οποίο ο χρήστης διαχειρίζεται τα project του, τα προσωπικά του στοιχεία και άλλα. Η ολοκλήρωση tasks ενός user story είχε ως αποτέλεσμα βασικές λειτουργίες όπως η είσοδος του χρήστη στο dashboard, η αποστολή email επιβεβαίωσης και πρόσκλησης, η διαχείριση χρηστών στον καμβά, καθώς και ο αυτόματος έλεγχος εγκυρότητας των στοιχείων κατά την είσοδο και την εγγραφή, για παράδειγμα έλεγχος αν το email που πληκτρολογεί ο χρήστης είναι πράγματι email.

Το τελευταίο sprint ήταν η προετοιμασία του τελευταίου release κατά το οποίο θα παρουσιάζαμε το τελικό προϊόν στους καθηγητές και τον υπεύθυνο της εταιρίας. Το sprint έλαβε χώρα στο πανεπιστήμιο της Γλασκόβης, όπου για ακόμη μία φορά όλοι οι φοιτητές βρισκόμασταν στον ίδιο χώρο την ίδια ώρα για πέντε ημέρες. Κατά την διάρκεια αυτού, ο στόχος μας ήταν να τελειοποιήσουμε την εφαρμογή μας βελτιώνοντας τις λειτουργίες και λύνοντας τα προβλήματα. Σε αυτό βοήθησαν οι καθηγητές οι οποίο δοκίμασαν και σχολίασαν ολόκληρη την εφαρμογή. Στην παρουσίαση συμμετείχαν αρκετά άτομα της ομάδας, καθώς και ο ένας από εμάς. Μετά την λήξη της παρουσίασης οι καθηγητές βαθμολόγησαν τους φοιτητές ως ομάδα και οι φοιτητές κάθε μέλος ξεχωριστά. Έτσι βγήκε ένας τελικό βαθμός με άριστα το πέντε.

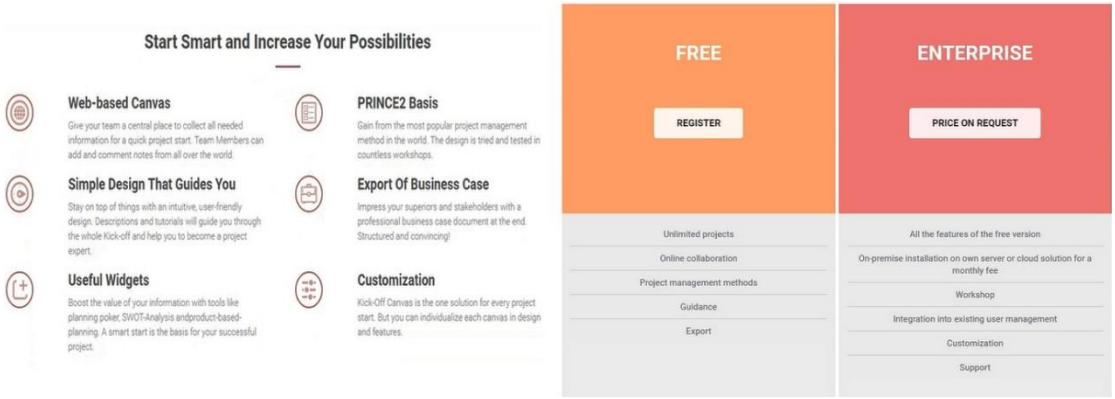
4.2.2 Εγχειρίδιο χρήσης

Ο παρακάτω οδηγός απευθύνεται στο ολοκληρωμένο τελικό προϊόν που αναπτύχθηκε από την ένωση των user stories όλων των φοιτητών και από τις τρεις nexus scrum ομάδες. Αρχικά, ένας

χρήστης επισκέπτεται την *ιστοσελίδα* του KICKOFF καμβά. Αν είναι καινούργιος χρήστης τότε θα περιηγηθεί στη σελίδα ώστε να μάθει περισσότερες πληροφορίες για τον καμβά, τα οφέλη και τα χαρακτηριστικά, καθώς και τα πακέτα που μπορεί να αγοράσει.

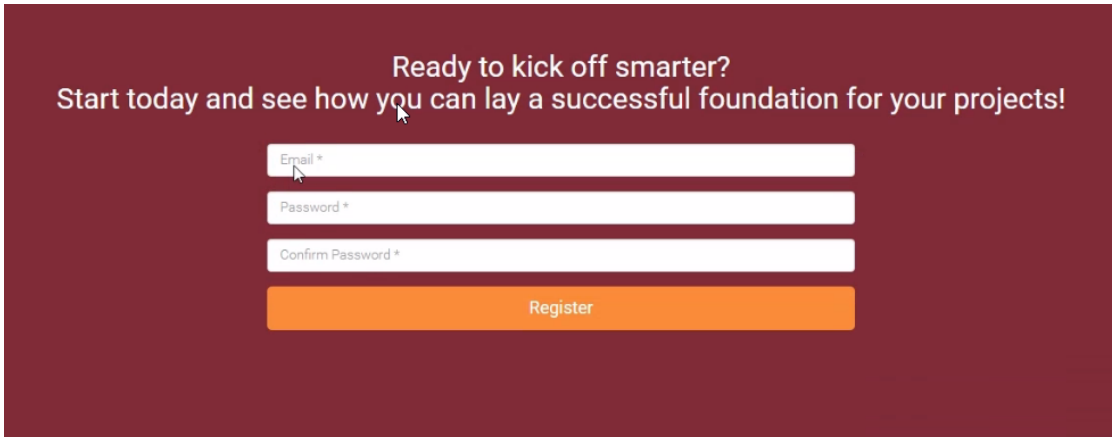


Εικόνα 4.2.2.1 Αρχική σελίδα του landing page



Εικόνα 4.2.2.2 Τα χαρακτηριστικά της εφαρμογής και τα πακέτα εγγραφής

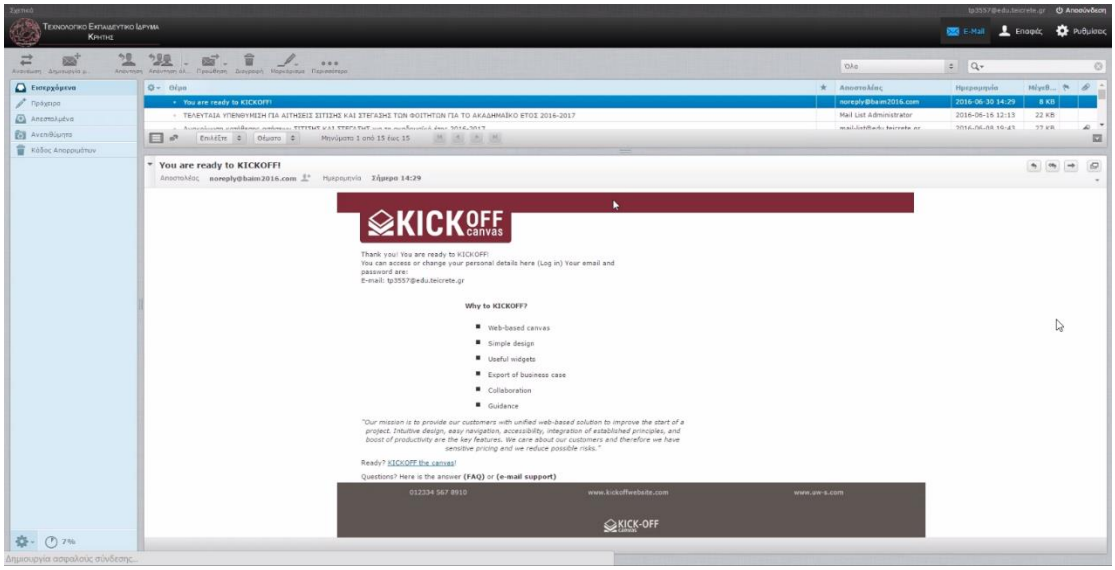
Στην συνέχεια μπορεί να εγγραφεί ως νέος χρήστης πατώντας το *TRY KICKOFF CANVAS* κουμπί ή *Register* από το menu. Θα εμφανιστεί η παρακάτω φόρμα συμπλήρωσης. Κατά την εγγραφή γίνεται αυτόματος έλεγχος στο email για την εγκυρότητα του, δηλαδή αν πράγματι είναι διεύθυνση email, καθώς και επιβεβαίωση του κωδικού πρόσβασης, δηλαδή ο χρήστης συμπληρώνει δύο φορές τον κωδικό του και η εφαρμογή ελέγχει αν ταιριάζουν οι δύο κωδικοί.



Εικόνα 4.2.2.3 Φόρμα εγγραφής νέου χρήστη στην εφαρμογή

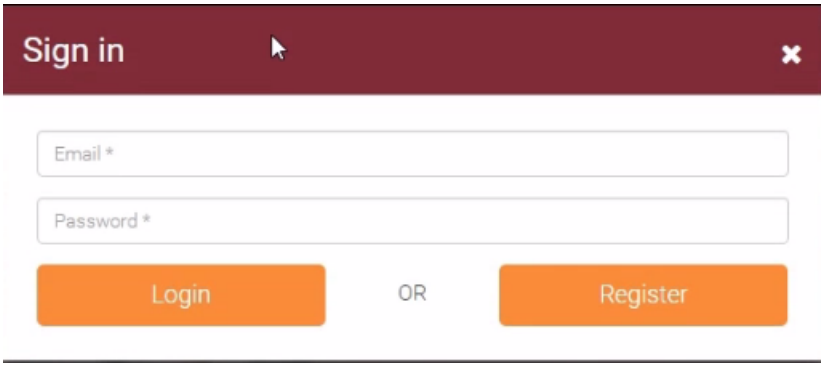
Αφού συμπληρώσει τα στοιχεία του και πατήσει register θα είναι έτοιμος να περάσει στο dashboard της εφαρμογής μας.

Μετά την εγγραφή του, ο χρήστης δέχεται ένα email καλωσορίσματος στην εφαρμογή το οποίο του εξηγεί τα βασικά της χαρακτηριστικά και την αποστολή της εταιρείας.



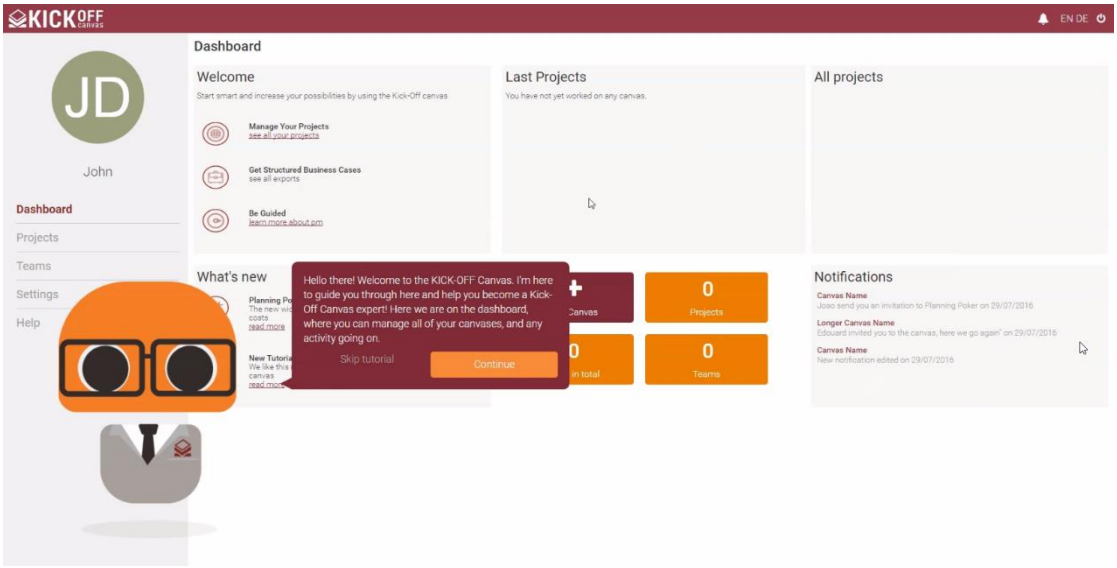
Εικόνα 4.2.2.4 Το email καλωσορίσματος μετά την εγγραφή

Αν ο χρήστης είναι ήδη εγγεγραμμένος τότε θα πατήσει στο *login* κουμπί του menu. Και θα του εμφανιστεί σε popup παράθυρο η παρακάτω φόρμα συμπλήρωσης. Στην περίπτωση που έκανε λάθος και δεν είναι εγγεγραμμένος μπορεί να πατήσει το register κουμπί και το παράθυρο θα κλείσει αυτόματα οδηγώντας τον χρήστη στην σωστή φόρμα. Πρώτα γίνεται έλεγχος αν το email αντιστοιχεί σε κάποιον χρήστη και αν ο κωδικός πρόσβασης ανήκει στο email. Αν τα στοιχεία του είναι σωστά και πράγματι υπάρχει στην βάση δεδομένων μας τότε θα οδηγηθεί στο dashboard.



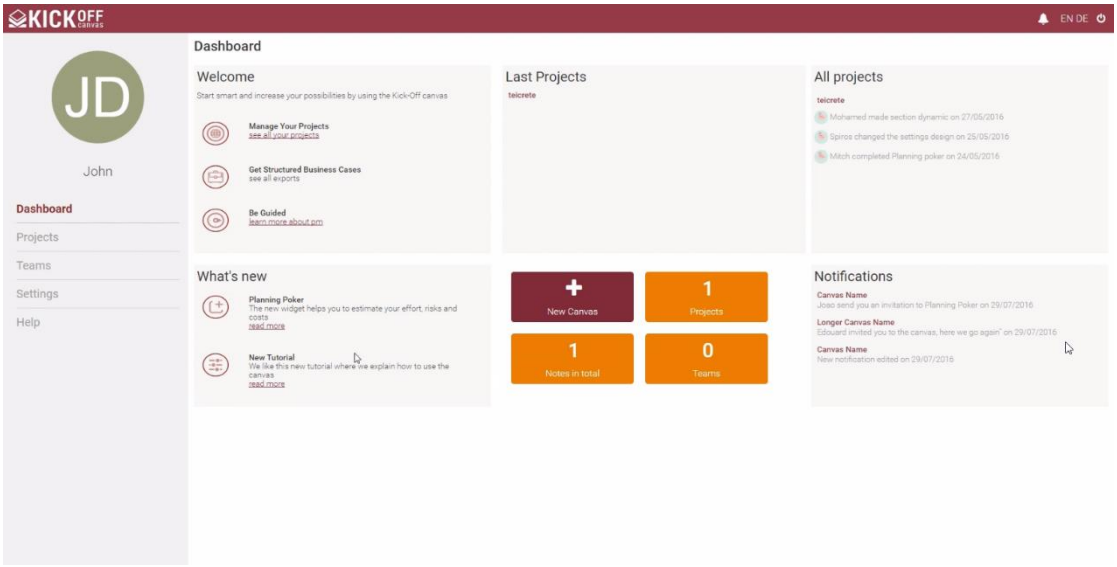
Εικόνα 4.2.2.5 Popup παράθυρο για την είσοδο εγγεγραμμένου χρήστη στην εφαρμογή

Το *dashboard* είναι ο προσωπικός χώρος κάθε χρήστη, στον οποίο έχει την δυνατότητα να διαχειρίζεται τα project του, τα στοιχεία του και άλλες πληροφορίες. Μετά την είσοδο του χρήστη στο dashboard για πρώτη φορά, θα εμφανιστεί στην αρχική του σελίδα ένας βοηθός (*bot*), το οποίο θα το καθοδηγήσει βήμα-βήμα, εξηγώντας του τι πρέπει αν κάνει. Αν ο χρήστης δεν επιθυμεί αυτή την βοήθεια μπορεί να πατήσει skip tutorial και το bot θα εξαφανιστεί.



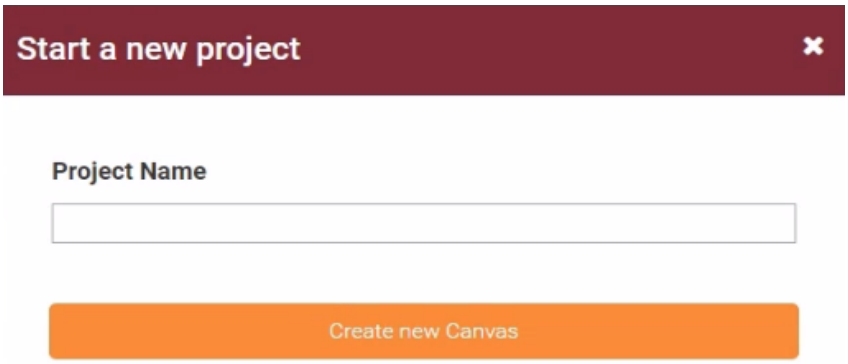
Εικόνα 4.2.2.6 Το bot ως tutorial βοηθός στο dashboard

Το bot βοηθάει στην καλύτερη κατανόηση της εφαρμογής και διατρέχει όλο το dashboard δίνοντας πληροφορίες και παραδείγματα. Αφού το bot τελειώσει ο χρήστης είναι έτοιμος και σίγουρος να προχωρήσει.



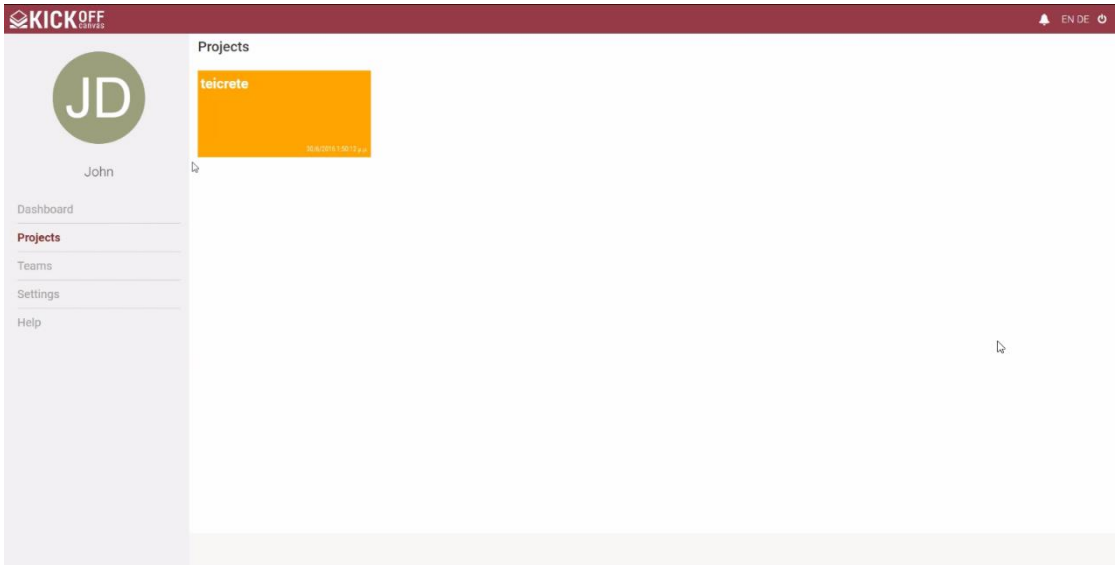
Εικόνα 4.2.2.7 Το dashboard του χρήστη

Όλες οι σημαντικές πληροφορίες, όπως ο αριθμός των project, τα τελευταία project, κάποιες πληροφορίες για αυτά, ειδοποιήσεις από άλλους χρήστες και το κουμπί δημιουργίας νέου καμβά βρίσκονται στη κεντρική σελίδα. Πάνω δεξιά υπάρχουν τρία εικονίδια. Στο πρώτο εικονίδιο εμφανίζεται ένας κόκκινος κύκλος με τον αριθμό των νέων ειδοποιήσεων και πατώντας το μπορεί ο χρήστης να τις δεις και να τις διαβάσει. Το δεύτερο είναι για τις αλλαγές των γλώσσες από Αγγλικά σε Γερμανικά και το αντίστροφο. Και το τρίτο και τελευταίο είναι το εικονίδιο της εξόδους, το όποιο αν πατηθεί θα εμφανίσει μία ειδοποίηση στον χρήστη για το αν πράγματι θέλει να βγει από το dashboard. Μόλις πατήσει «Ναι» ο χρήστης θα μεταφερθεί πίσω στο landing page. Πατώντας το μωβ κουμπί ”+ *new canvas*” εμφανίζεται ένα popup παράθυρο για την δημιουργία νέου καμβά.



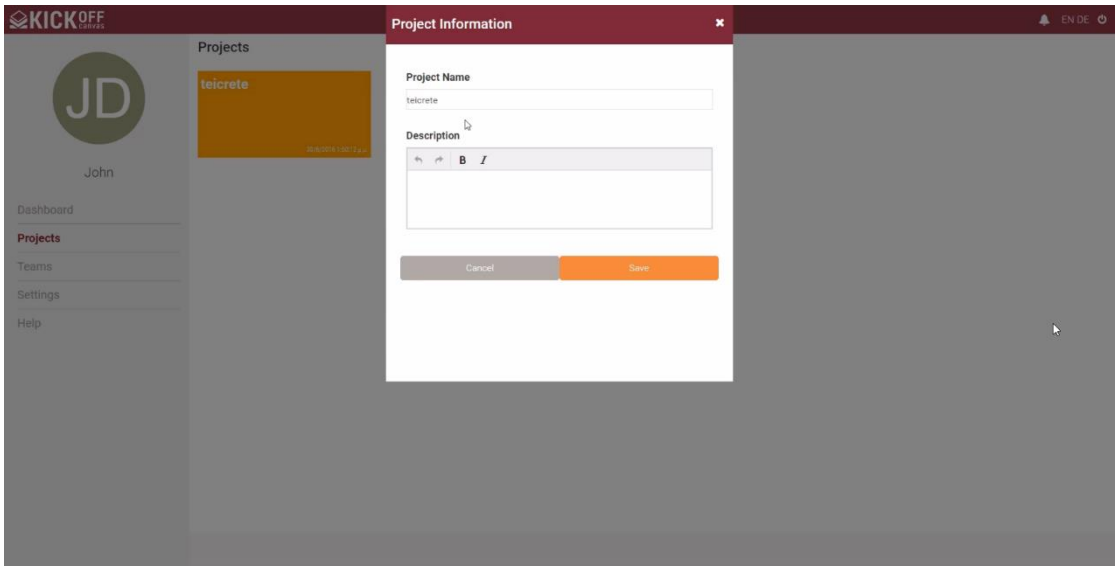
Εικόνα 4.2.2.8 Popup παράθυρο δημιουργίας νέου καμβά

Ο χρήστης έχει την δυνατότητα να δει και να διαχειριστεί όλα του τα project πατώντας το *Projects* από το menu.



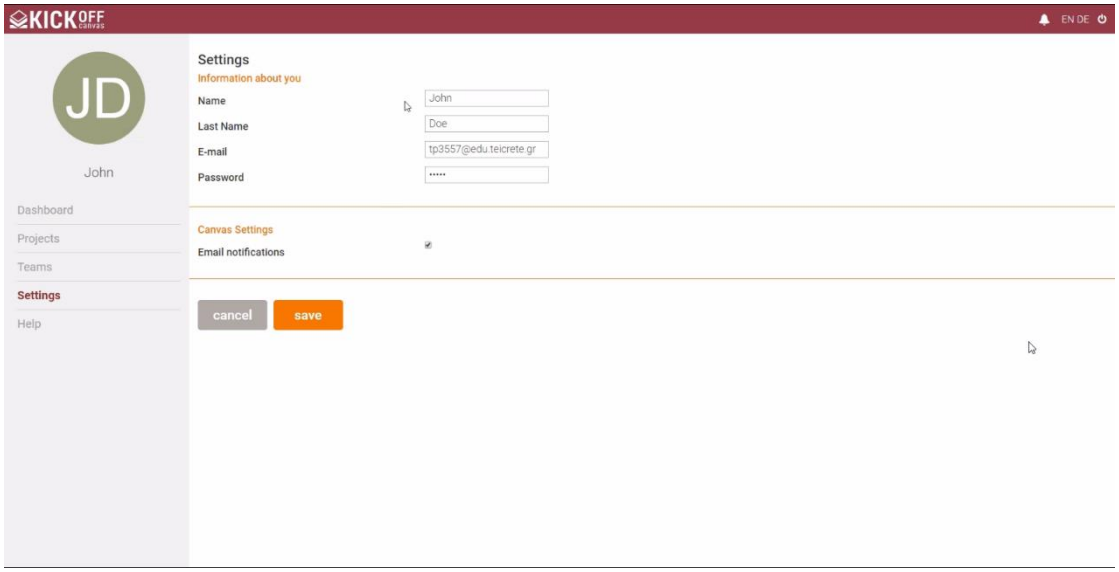
Εικόνα 4.2.2.9 Επισκόπηση όλων των έργων του χρήστη

Για να ανοίξει το project και να μεταφερθεί στον καμβά αρκεί να κάνει ένα αριστερό click. Για να επεξεργαστεί το όνομα του project πρέπει να κάνει click στο εικονίδιο που θα εμφανιστεί δεξιά όταν κάνει mouse over πάνω από το καθένα πορτοκαλί κουτάκι. Μόλις πατήσει το εικονίδιο θα του εμφανιστεί στην οθόνη ένα popup παράθυρο.



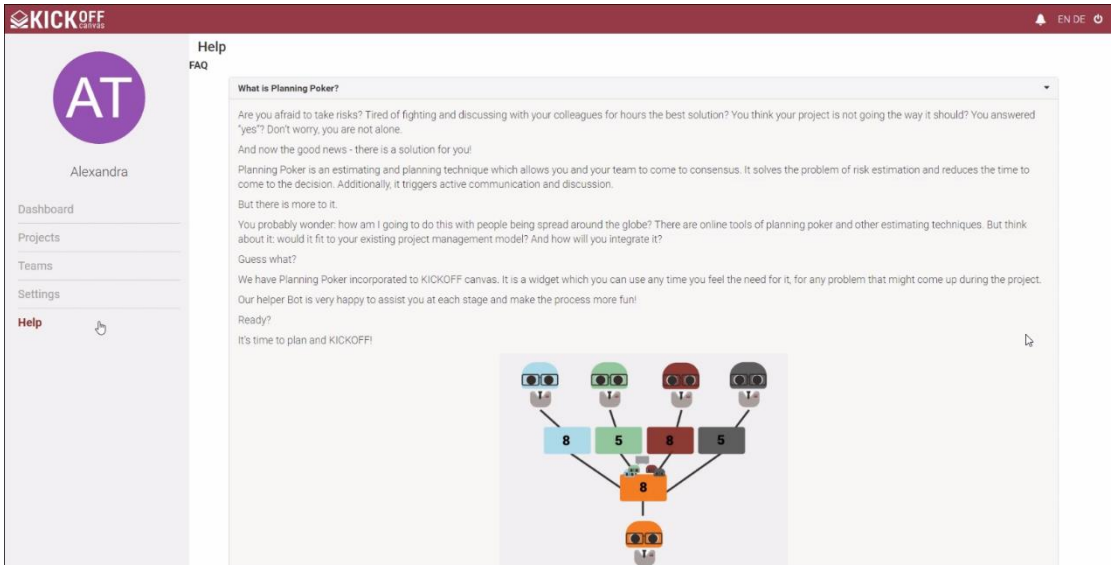
Εικόνα 4.2.2.10 Επεξεργασία πληροφοριών του έργου

Από τα *Setting* του menu έχει την δυνατότητα να αλλάξει τα προσωπικά του στοιχεία όπως, το όνομα, τον κωδικό πρόσβασης, το email και μπορεί να ενεργοποιήσει/απενεργοποιήσει τις ειδοποιήσεις που του έρχονται στο email από άλλους χρήστες. Για να σωθούν οι αλλαγές θα πρέπει να πατήσει save. Το όνομα και η εικόνα με τα αρχικά του θα αλλάξουν αμέσως.



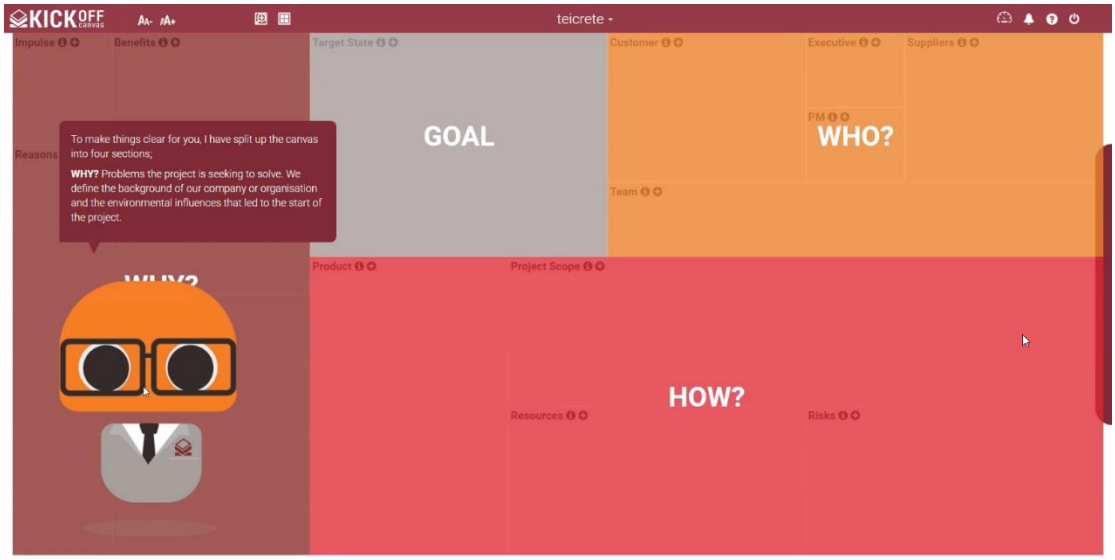
Εικόνα 4.2.2.11 Προσωπικά στοιχεία του χρήστη

Εκτός από το bot που προαναφέραμε, υπάρχει το *Help* το οποίο είναι μια επιπλέον βοήθεια με συχνές ερωτήσεις και απορίες που μπορεί να του δημιουργηθούν, Q&A, όπως το ”*Τι είναι το Planning Poker;*”



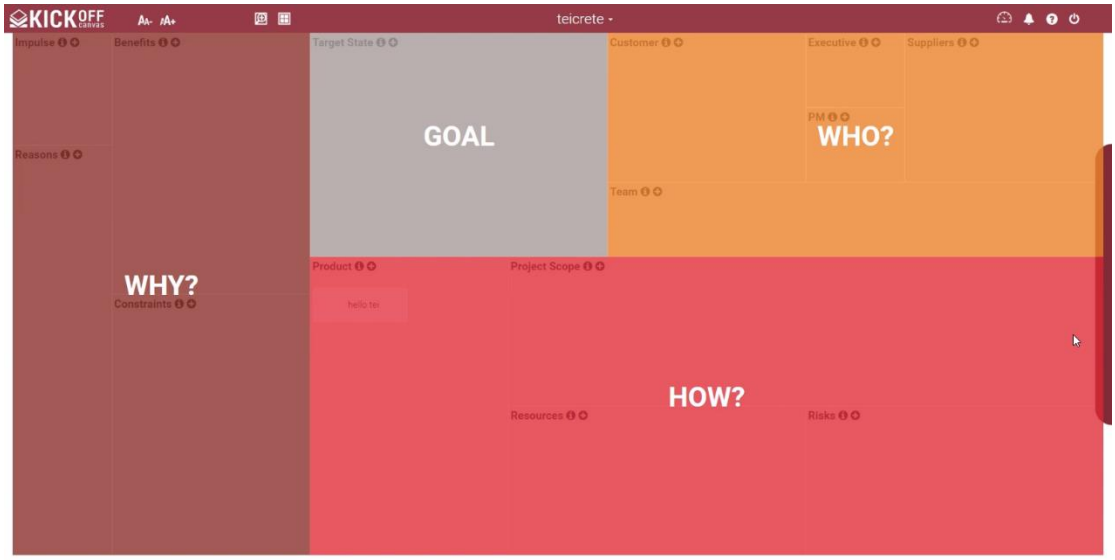
Εικόνα 4.2.2.12 Dashboard help

Όταν συνδεθεί ο χρήστης σε ένα συγκεκριμένο καμβά, το πρώτο που βλέπει είναι ο βοηθός της εφαρμογής (bot) ο οποίος υπάρχει για να εξηγεί σε νέους χρήστες τον τρόπο λειτουργίας της εφαρμογής.



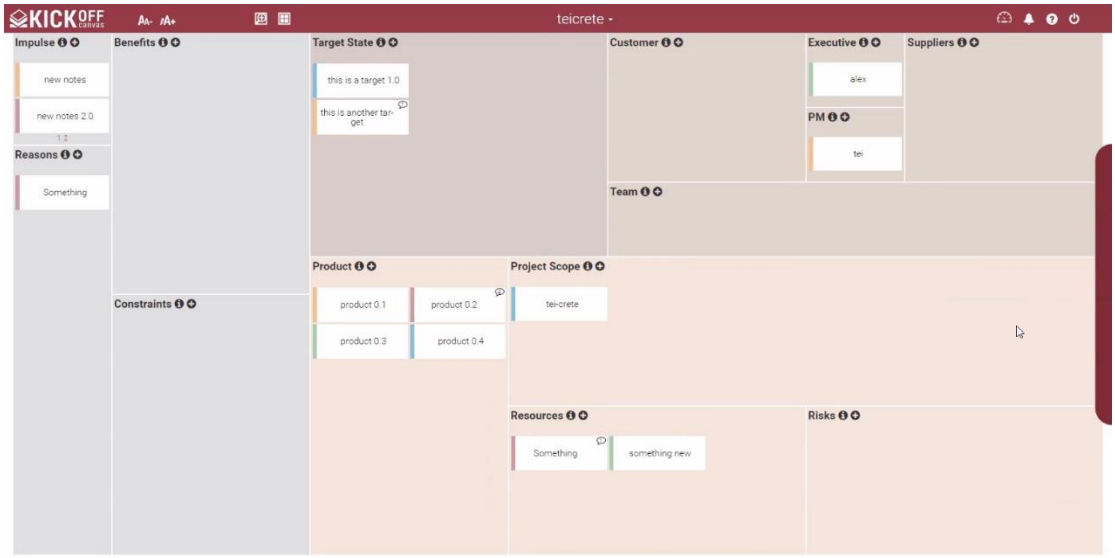
Εικόνα 4.2.2.13 Η πρώτη εικόνα που έχει ο χρήστης όταν συνδέεται στον καμβά

Αν έχει επιλέξει να κλείσει τον βοηθό τότε του εμφανίζεται το επικάλυμμα (overlay) του καμβά. Αυτό βοηθάει τον χρήστη να καταλάβει εύκολα τη δομή του καμβά.



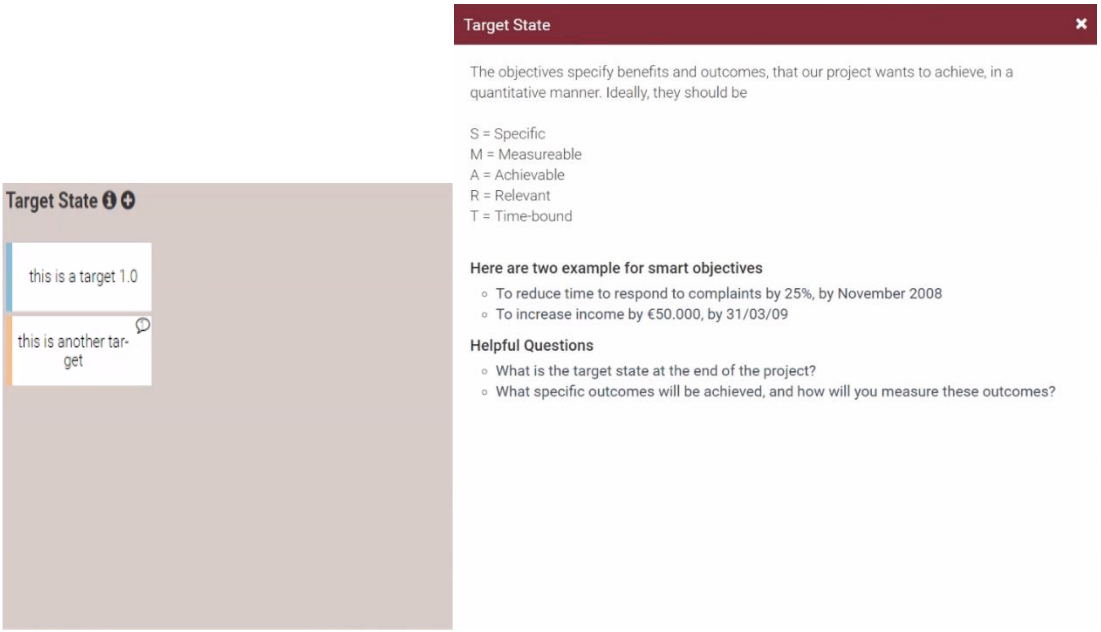
Εικόνα 4.2.2.14 Το επικάλυμμα (overlay) του καμβά

Αφού δει ο χρήστης την δομή του καμβά και είναι έτοιμος να ξεκινήσει να τον χρησιμοποιεί, μπορεί να κάνει κλικ σε οποιοδήποτε σημείο του επικαλύμματος και αυτό θα εξαφανιστεί και θα εμφανιστεί ο καμβάς.



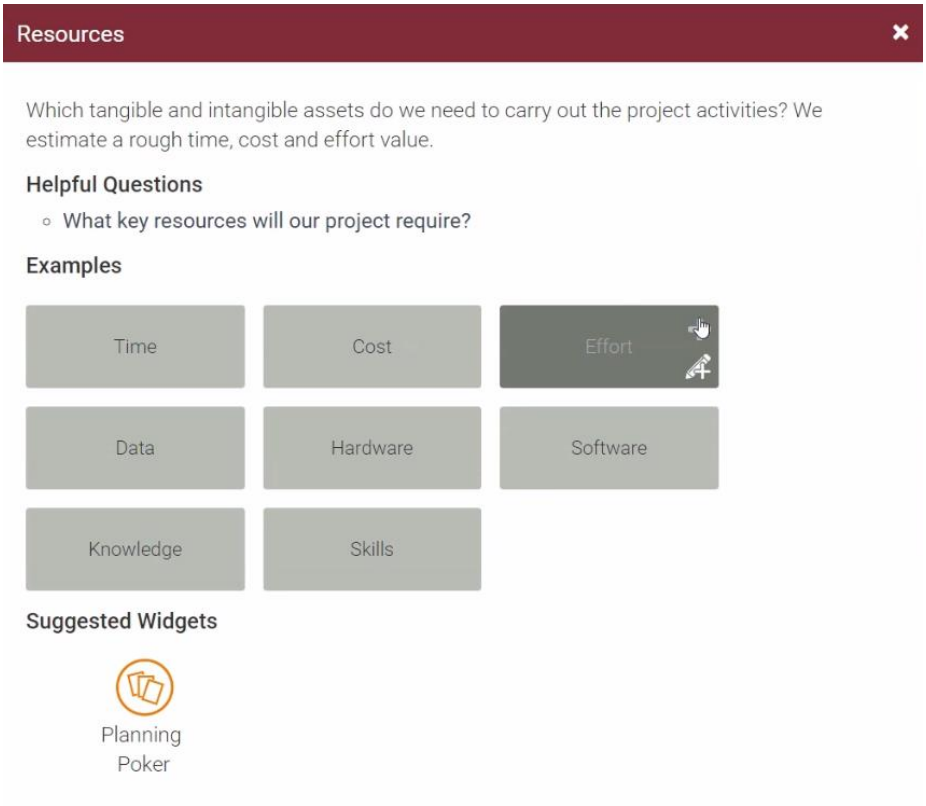
Εικόνα 4.2.2.15 Η εικόνα του καμβά μετά την απομάκρυνση του επικαλύμματος (overlay)

Αυτό που φαίνεται στον καμβά μετά την απομάκρυνση του επικαλύμματος είναι οι σημειώσεις που υπάρχουν στα διαφορετικά τμήματα τα οποία είναι αντίστοιχα με αυτά που υπάρχουν στην έντυπη έκδοση του. Πάνω στη μέση υπάρχει το όνομα του (*teicrete*), όπου μπορεί ο χρήστης να κάνει κλικ πάνω του και να δει τις πληροφορίες του έργου, να διαχειριστεί τα μέλη, να δει τις πρόσφατες δραστηριότητες, και να εξάγει τον καμβά σε word αρχείο. Αριστερά και δεξιά του ονόματος υπάρχουν κουμπιά που επιτελούν διάφορες λειτουργίες όπως τη μεγέθυνση και τη σμίκρυνση της γραμματοσειράς των σημειώσεων, την αλλαγή του τρόπου προβολής του καμβά, την επαναφορά του επικαλύμματος, τη μετάβαση στο dashboard, την εμφάνιση των ειδοποιήσεων, την παροχή βοήθειας και την έξοδο από την εφαρμογή. Δεξιά της οθόνης υπάρχει μία μωβ μπάρα η οποία περιέχει *widgets* όπως των κάδο με τις διαγραμμένες σημειώσεις, τον χώρο όπου τοποθετούνται οι σημειώσεις όπου δεν έχει αποφασιστεί ακόμα σε ποιο τμήμα ανήκουν, το *parking lot* και το *planning poker* και ενεργοποιείται με mouseover.



Εικόνα 4.2.2.16 Το τμήμα target state και οι επιπλέον πληροφορίες για αυτό

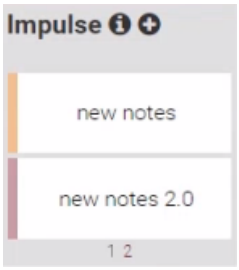
Αν κοιτάξουμε πιο προσεκτικά τα τμήματα του καμβά, θα παρατηρήσουμε ότι υπάρχουν δύο κουμπιά σε κάθε ένα από αυτά· ένα «*i*» το οποίο δίνει πληροφορίες για το συγκεκριμένο τμήμα και ένα «+» το οποίο δημιουργεί μία σημείωση στο τμήμα αυτό. Στις εικόνες παραπάνω φαίνεται το τμήμα *target state* και δίπλα οι επιπλέον πληροφορίες του τμήματος αν πατηθεί το κουμπί «*i*». Επιπλέον σε κάποια τμήματα δίνεται η δυνατότητα προσθήκης (π.χ. *Resources*) πρότυπων (template) σημειώσεων.



Εικόνα 4.2.2.17 Τα templates σημειώσεων στο τμήμα Recourses

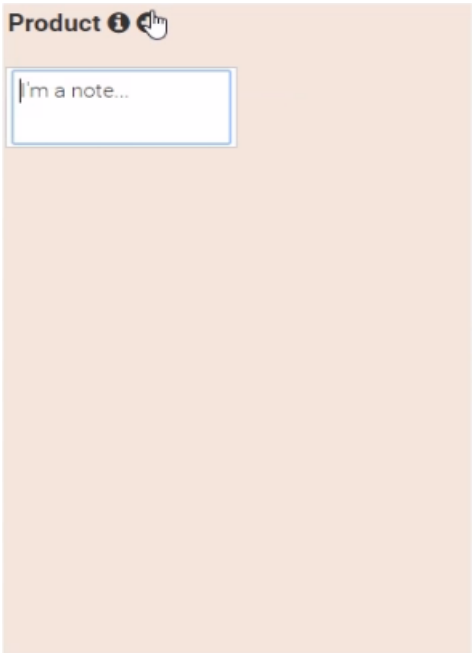
Στην εικόνα παραπάνω βλέπουμε έτοιμες σημειώσεις, που παρέχονται από την εφαρμογή, οι οποίες βρίσκονται στις πληροφορίες που παίρνουμε για το τμήμα πατώντας το κουμπί «*i*». Όπως φαίνεται και στην εικόνα οι σημειώσεις αυτές μπορούν είτε να προστεθούν κατευθείαν στον καμβά πατώντας το κουμπί «+» ή να επεξεργαστούν πρώτα πατώντας στο σύμβολο με το μολύβι.

Παρατηρούμε ότι δεν έχουν όλα τα τμήματα το ίδιο μέγεθος, οπότε σε περίπτωση που οι σημειώσεις δεν χωράνε σε ένα τμήμα, δημιουργούνται σελίδες σε αυτό ώστε να χωρέσουν και οι υπόλοιπες σημειώσεις. Οι σελίδες δημιουργούνται και διαγράφονται μετά από ερώτηση προς τον χρήστη.



Εικόνα 4.2.2.18 Η δημιουργία σελίδων όταν ένα τμήμα γεμίζει από σημειώσεις

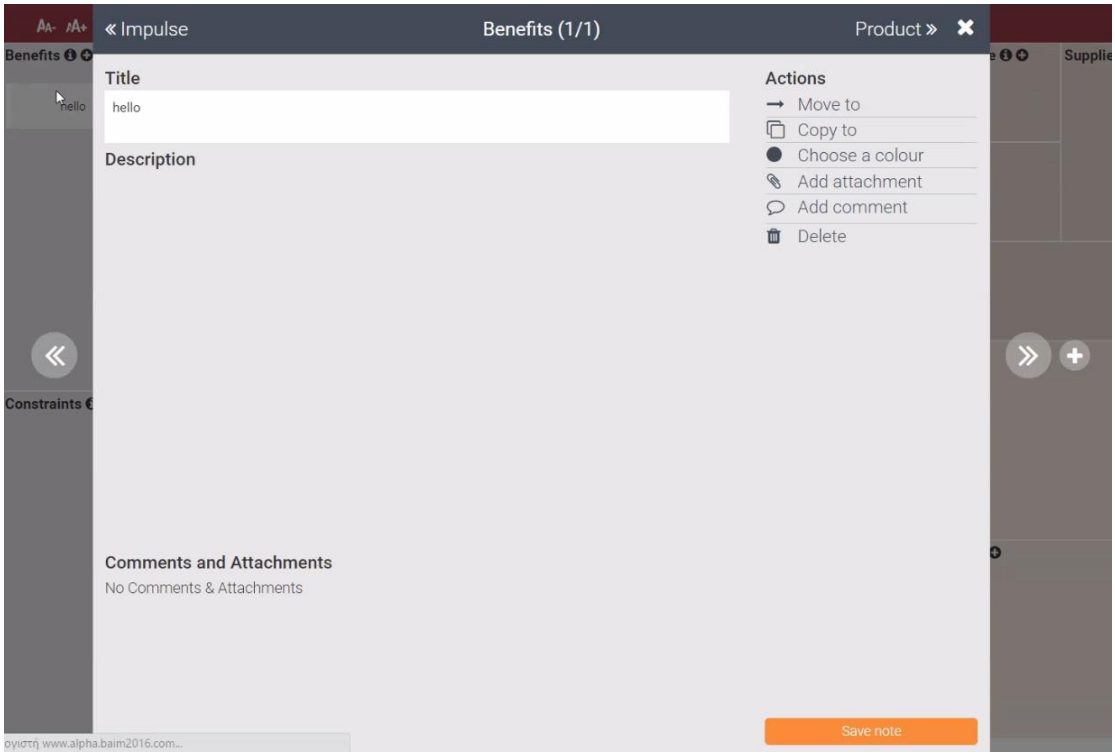
Το άλλο κουμπί που υπάρχει σε κάθε τμήμα, το «+» χρησιμοποιείται για τη δημιουργία μίας σημείωσης. Εναλλακτικός τρόπος για τη δημιουργία σημείωσης είναι να γίνει διπλό κλικ σε οποιοδήποτε σημείο του τμήματος.



Εικόνα 4.2.2.19 Το πάτημα του κουμπιού «+» για την δημιουργία σημείωσης

Όταν ο χρήστης φτιάξει τη σημείωση του, μπορεί απλά να πατήσει *enter* ή να κάνει κλικ κάπου έξω από τη σημείωση και αυτή θα δημιουργηθεί. Τη σημείωση αυτή ο χρήστης μπορεί να την σύρει και να την μετακινήσει οπουδήποτε μέσα στο τμήμα ή σε έναν άλλο, να την αντιγράψει, να της αλλάξει χρώμα, να την διαγράψει, να της προσθέσει σχόλιο ή κάποια επισύναψη.

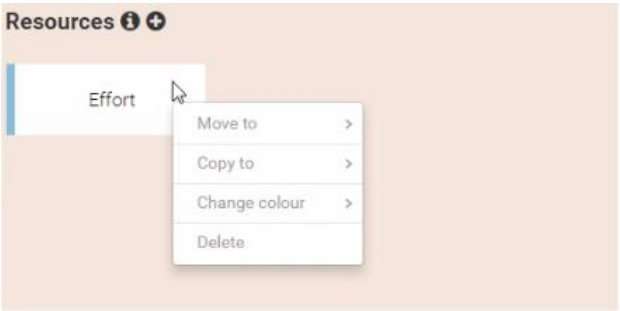
Κάνοντας διπλό κλικ πάνω στην σημείωση, ανοίγει ένα παράθυρο όπου μπορούμε να κάνουμε όλες τις ενέργειες που αναφέρθηκαν πιο πάνω.



Εικόνα 4.2.2.20 Το popup παράθυρο για την επεξεργασία μιας σημείωσης

Στο πάνω μέρος του παραθύρου βλέπουμε το όνομα του τμήματος στον οποίο ανήκει η σημείωση (*Benefits*) καθώς επίσης και τον αριθμό των σημειώσεων που έχει εκείνη τη στιγμή το τμήμα (*1/1*), όπου το αριστερό *1* είναι η σειρά της σημείωσης που επεξεργαζόμαστε και το δεξιό το σύνολο των σημειώσεων του τμήματος. Αριστερά από το όνομα μπορούμε να πάμε στο προηγούμενο τμήμα (*Impulse*) και δεξιά στο επόμενο (*Product*) διατηρώντας ταυτόχρονα ανοιχτό το παράθυρο. Στο κύριο σώμα του παραθύρου βλέπουμε τον τίτλο της σημείωσης, την περιγραφή της και τα σχόλια ή τις επισυνάψεις που μπορεί να έχει. Μπορούμε να αλλάξουμε τον τίτλο της σημείωσης όπως και την περιγραφή της. Δεξιά στο κύριο σώμα υπάρχουν οι ενέργειες που μπορούν να γίνουν σε μία σημείωση, οι οποίες αναφέρθηκαν πιο πάνω. Επίσης, αριστερά και δεξιά στο κύριο σώμα της σημείωσης υπάρχουν βέλη ενώ από τη δεξιά μεριά υπάρχει και ένα «+». Τα βέλη αυτά χρησιμοποιούνται για την πλοήγηση στις σημειώσεις του τμήματος (αν έχει παραπάνω από μία) χωρίς να κλείσει το παράθυρο ενώ το «+» χρησιμοποιείται για τη δημιουργία σημείωσης χωρίς πάλι να κλείσει το παράθυρο και η οποία σημείωση δημιουργείται αμέσως μετά την τρέχουσα. Όποιες αλλαγές έχουν γίνει στη σημείωση μπορούν να αποθηκευτούν πατώντας το κουμπί *save note* το οποίο βρίσκεται κάτω δεξιά.

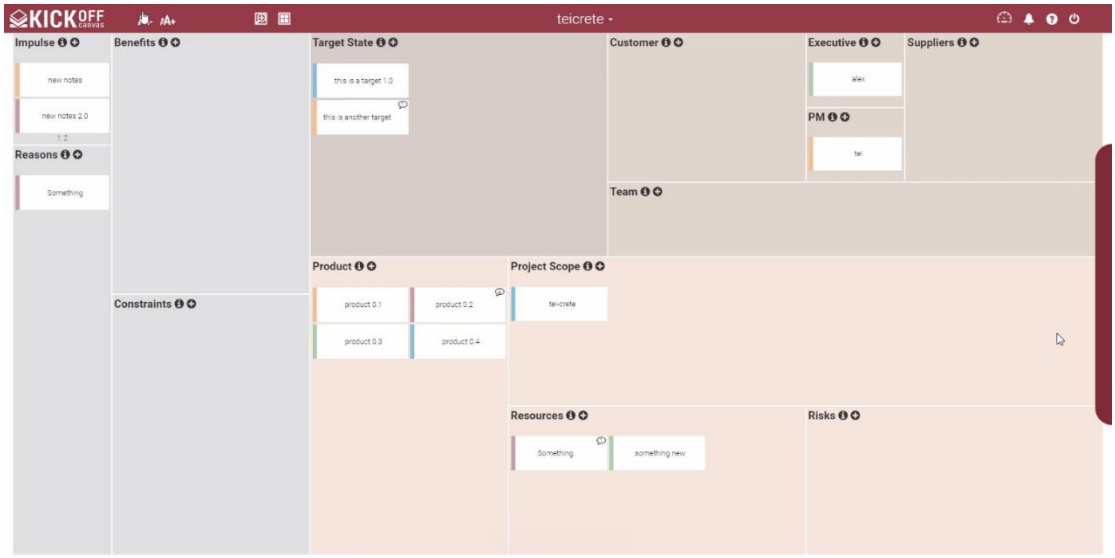
Εναλλακτικός τρόπος για την επεξεργασία μίας σημείωσης είναι να γίνει δεξί κλικ πάνω της.



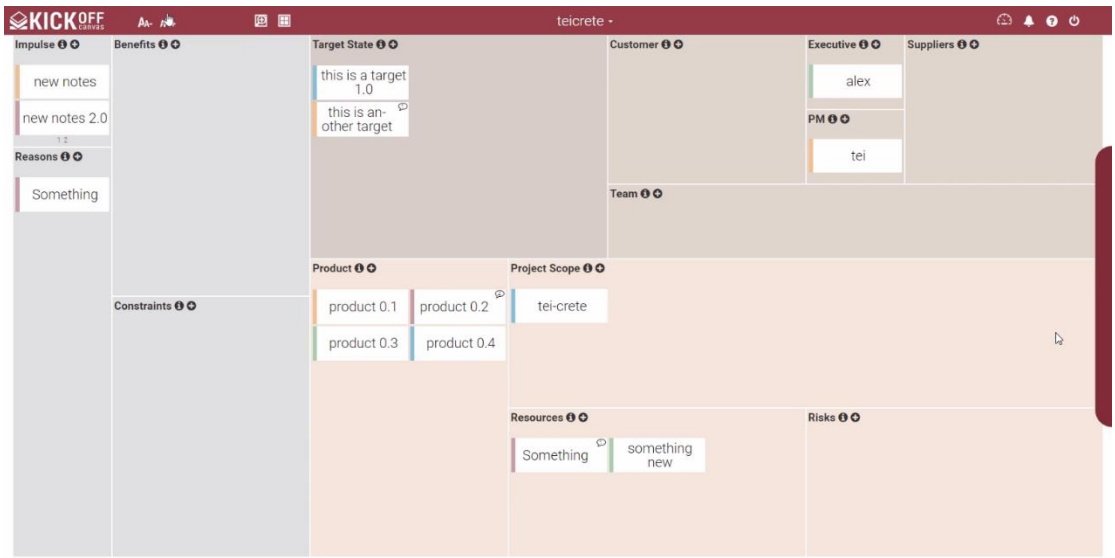
Εικόνα 4.2.2.21 Κάνοντας δεξί κλικ σε σημείωση

Το δεξί κλικ αποτελεί έναν πιο παραδοσιακό τρόπο για την επεξεργασία της σημείωσης. Η διαφορά εδώ είναι ότι έχουμε τη δυνατότητα να μετακινήσουμε, να αντιγράψουμε ή να διαγράψουμε τη σημείωση και να της αλλάξουμε χρώμα αλλά δεν μπορούμε να προσθέσουμε κάποιο σχόλιο ή κάποια επισύναψη.

Όπως είδαμε στη γενική επισκόπηση του καμβά, αριστερά και δεξιά από το όνομα του υπάρχουν κάποια εικονίδια. Στα αριστερά βλέπουμε δύο εικονίδια με σύμβολα *AA-* και *AA+*, τα οποία χρησιμοποιούνται για την σμίκρυνση και τη μεγέθυνση της γραμματοσειράς των σημειώσεων αντίστοιχα.

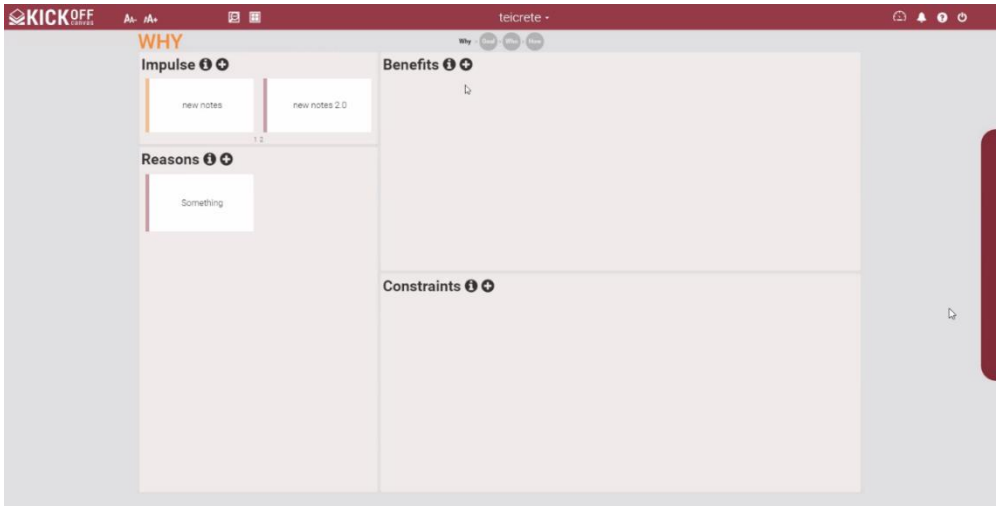


Εικόνα 4.2.2.22 Η σμίκρυνση της γραμματοσειράς των σημειώσεων



Εικόνα 4.2.2.23 Η μεγέθυνση της γραμματοσειράς των σημειώσεων

Δεξιά από τα κουμπιά σμίκρυνσης και μεγέθυνσης της γραμματοσειράς, υπάρχει το κουμπί με τον μεγεθυντικό φακό. Το κουμπί αυτό αλλάζει την προβολή του καμβά, ώστε να μπορεί να λειτουργεί στην περίπτωση που κάποιος θέλει να χρησιμοποιήσει την εφαρμογή με προβολέα.



Εικόνα 4.2.2.24 Η αλλαγή της προβολής του καμβά για να λειτουργεί με προβολείς

Στην παραπάνω εικόνα παρατηρούμε ότι το μόνο που έχει αλλάξει είναι ο καμβάς. Τα υπόλοιπα κουμπιά είναι όλα ίδια και στη θέση τους. Η νέα αυτή προβολή του καμβά χωρίστηκε σε τέσσερα τμήματα βασισμένα στο επικάλυμμα του: *Why*, *Goal*, *Who* και *How*. Κάθε ένα από αυτά τα τμήματα περιέχει τα τμήματα που του αντιστοιχούν, σύμφωνα με τη δομή του καμβά. Για παράδειγμα το *Why* περιέχει τα τμήματα *Impulse*, *Benefits*, *Reasons* και *Constraints*. Η λειτουργία του καμβά παραμένει ίδια καθώς μπορούν κανονικά να δημιουργηθούν σημειώσεις, να μετακινηθούν, να αλλάξουν χρώμα, να αντιγραφούν και να τους προστεθούν σχόλια και επισυνάψεις. Το μόνο διαφορετικό είναι ότι δεν φαίνονται όλα τα τμήματα τα οποία είναι χωρισμένα.

Εικόνα 4.2.2.25 Η περιήγηση στα τμήματα του καμβά στην μεγεθυμένη προβολή του

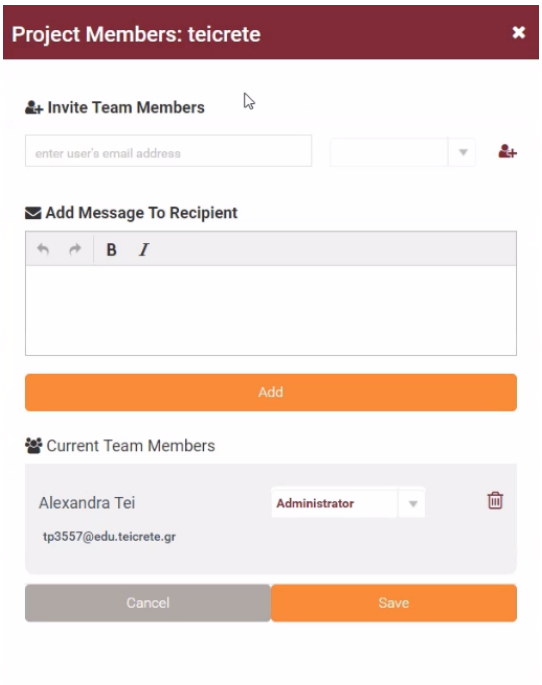
Στην παραπάνω εικόνα φαίνεται ο τρόπος πλοήγησης στην μεγεθυμένη προβολή του καμβά. Για τη μετακίνηση από το ένα τμήμα στο άλλο υπάρχουν σφαιρίδια με ονόματα *Why*, *Goal*, *Who* και *How* στα οποία αν γίνει κλικ μετακινείται ο χρήστης στο αντίστοιχο τμήμα. Για να εξέλθει ο χρήστης από την μεγεθυμένη προβολή πατά απλώς ξανά το κουμπί με τον μεγεθυντικό φακό.

Δίπλα από το κουμπί αλλαγής προβολής του καμβά υπάρχει το κουμπί που επαναφέρει το επικάλυμμα του καμβά. Ο λόγος που υπάρχει είναι σε περίπτωση που ο χρήστης θέλει να θυμηθεί την δομή του καμβά. Αν υπάρχει το επικάλυμμα και πατηθεί το κουμπί τότε αυτό εξαφανίζεται.

Αν πατήσουμε πάνω στο όνομα του καμβά τότε μας εμφανίζονται επιλογές οι οποίες αναφέρθηκαν παραπάνω.

Εικόνα 4.2.2.26 Οι επιλογές που εμφανίζονται στο πάτημα του ονόματος ενός καμβά

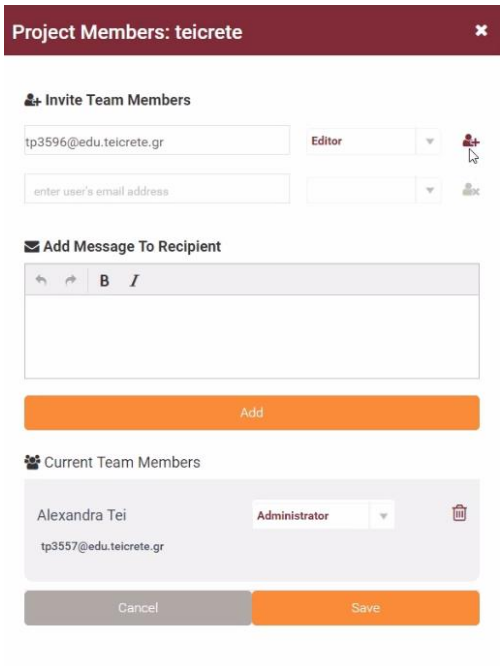
Με την επιλογή *project members* διαχειριζόμαστε τα μέλη ενός καμβά. Δηλαδή, μπορούμε να προσθέσουμε μέλη σε αυτόν, να δούμε τα τρέχοντα ή να διαγράψουμε μέλη. Για τη διαγραφή των μελών χρειαζόμαστε δικαιώματα διαχειριστή.



Εικόνα 4.2.2.27 Το popup παράθυρο για την διαχείριση των μελών

Η παραπάνω εικόνα απεικονίζει το popup παράθυρο που εμφανίζεται στην επιλογή για τη διαχείριση των μελών. Στην ενότητα *Invite Team Members* έχουμε τη δυνατότητα να πληκτρολογήσουμε το όνομα ή το email του χρήστη που θέλουμε να προσκαλέσουμε στον καμβά. Αν ο χρήστης είναι εγγεγραμμένος στην εφαρμογή τότε θα ειδοποιηθεί με email καθώς και με ειδοποίηση μέσα στην εφαρμογή. Σε περίπτωση που δεν είναι λαμβάνει ένα email πρόσκληση για εγγραφή στην εφαρμογή και στη συνέχεια μπορεί να είναι μέλος στον καμβά που προσκλήθηκε. Στην ενότητα *Current Team Members* μπορεί να δει τα τρέχοντα μέλη και τον ρόλο του καθενός, ενώ αν έχει δικαιώματα διαχειριστή τότε μπορεί να διαγράψει κάποιον αν το επιθυμεί. Στην εικόνα μας είμαστε συνδεδεμένοι ως διαχειριστές γι' αυτό και βλέπουμε το εικονίδιο του κάδου δίπλα από το όνομα του μέλους. Αν δεν ήμασταν διαχειριστές τότε θα βλέπαμε απλά το όνομα του μέλους και τον ρόλο που έχει.

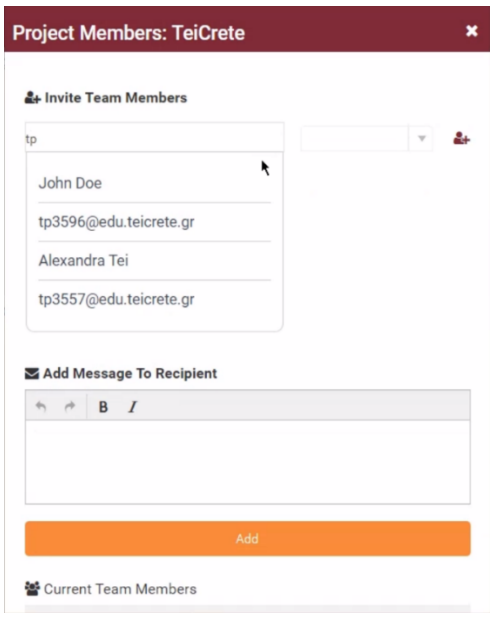
Έχουμε το δικαίωμα να προσθέσουμε ταυτόχρονα περισσότερους από έναν χρήστες πατώντας το μωβ κουμπί που βρίσκεται δίπλα στη selection λίστα που έχει ως σύμβολο έναν άνθρωπο με το «+» πάνω του. Έτσι εμφανίζονται περισσότερες είσοδοι.



Εικόνα 4.2.2.28 Η δυνατότητα προσθήκης περισσότερων εισόδων για την προσθήκη πολλών μελών ταυτόχρονα

Κάθε ένα από τα πεδία που δημιουργούνται έχουν δίπλα τους ένα κουμπί το οποίο τα αφαιρεί σε περίπτωση που ο χρήστης τα δημιούργησε κατά λάθος. Οι χρήστες που υπάρχουν σε ένα καμβά μπορούν να συνεργαστούν online και ταυτόχρονα. Για παράδειγμα όταν ένας χρήστης δημιουργήσει μία σημείωση σε κάποιο τμήμα του καμβά τότε οι υπόλοιποι χρήστες του καμβά θα τη δουν να δημιουργείται ζωντανά στην δικιά τους οθόνη. Αυτό επιτυγχάνεται χάρη στο web socket που είναι υλοποιημένο στην εφαρμογή.

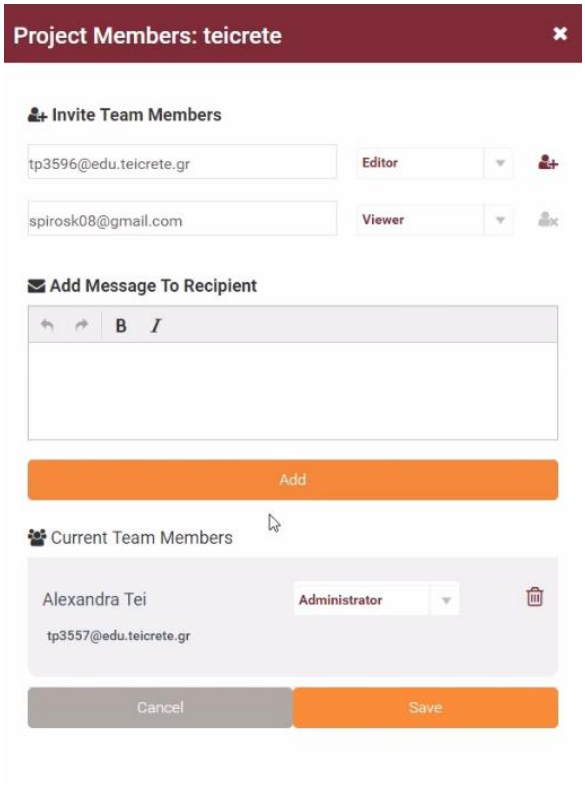
Κατά την πληκτρολόγηση του ονόματος ή του email ενός χρήστη για την προσθήκη του στον καμβά μας δίνεται η δυνατότητα της αυτόματης συμπλήρωσης.



Εικόνα 4.2.2.29 Η λειτουργία αυτόματης συμπλήρωσης (autocomplete)

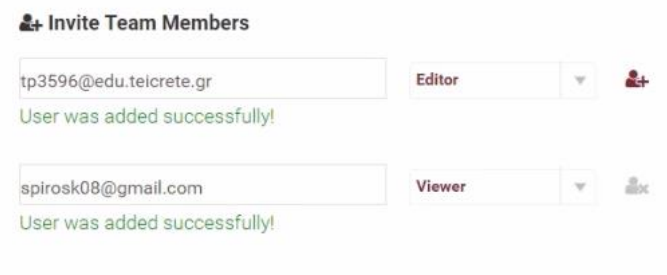
Όπως φαίνεται στην εικόνα παραπάνω με την πληκτρολόγηση είτε του email ή του ονόματος του χρήστη μας εμφανίζεται και το όνομα και το email του. Φυσικά στην αυτόματη συμπλήρωση δεν εμφανίζονται τα ονόματα και τα email όλων των εγγεγραμμένων χρηστών της εφαρμογής. Για λόγους ασφαλείας εμφανίζονται μόνο ονόματα χρηστών με τους οποίους ο χρήστης ο οποίος τους προσκαλεί συσχετίζεται μαζί τους σε άλλους καμβάδες.

Αφού έχουμε πληκτρολογήσει τα ονόματα ή τα email των χρηστών που θέλουμε να προσθέσουμε, πρέπει να επιλέξουμε και τον ρόλο που θα έχουν στον καμβά. Οι επιλογές είναι *Administrator*, *Viewer* και *Editor*.



Εικόνα 4.2.2.30 Η προσθήκη πολλών μελών σε ένα καμβά

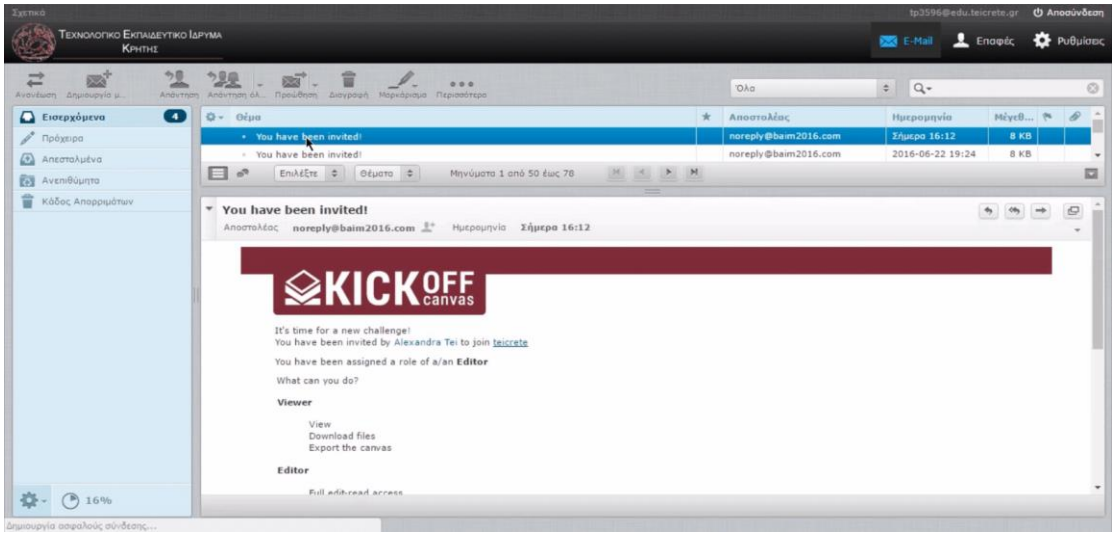
Στην εικόνα βλέπουμε πως η *Alexandra Tei* θέλει να προσθέσει τους χρήστες *tp3596@edu.teicrete.gr* και *spirosk08@gmail.com*. Για να προστεθούν οι χρήστες πρέπει να πατήσουμε το κουμπί *Add*. Αφού ενημερωθεί με το κατάλληλο μήνυμα ότι οι χρήστες προστέθηκαν το popup παράθυρο κλείνει αυτόματα.



Εικόνα 4.2.2.31 Τα μηνύματα επιτυχούς προσθήκης μελών στον καμβά

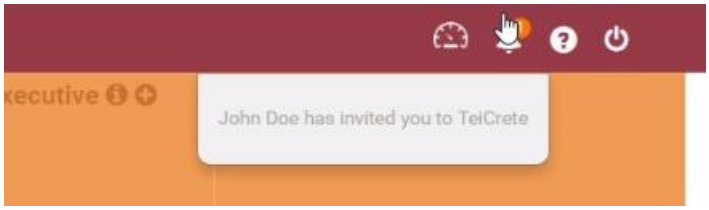
Εκτός από το μήνυμα της εικόνας υπάρχουν και άλλοι έλεγχοι στα πεδία όπως για το αν αυτό που πληκτρολογείται είναι διεύθυνση email, αν ο χρήστης που προσπαθεί να προστεθεί είναι ήδη μέλος στον καμβά, αν ο χρήστης που προσπαθεί να προστεθεί είναι εγγεγραμμένος, αν το πεδίο που εισάγεται το όνομα και το email είναι κενό ή το πεδίο με τη λίστα επιλογών δεν έχει κάποια τιμή. Σε κάθε μία από τις περιπτώσεις, εμφανίζονται τα κατάλληλα μηνύματα.

Όπως αναφέραμε, οι χρήστες που προστίθενται σε έναν καμβά ειδοποιούνται μέσω email πρόσκλησης ή με ειδοποίηση στην εφαρμογή.



Εικόνα 4.2.2.32 Το email πρόσκλησης ενός χρήστη σε καμβά

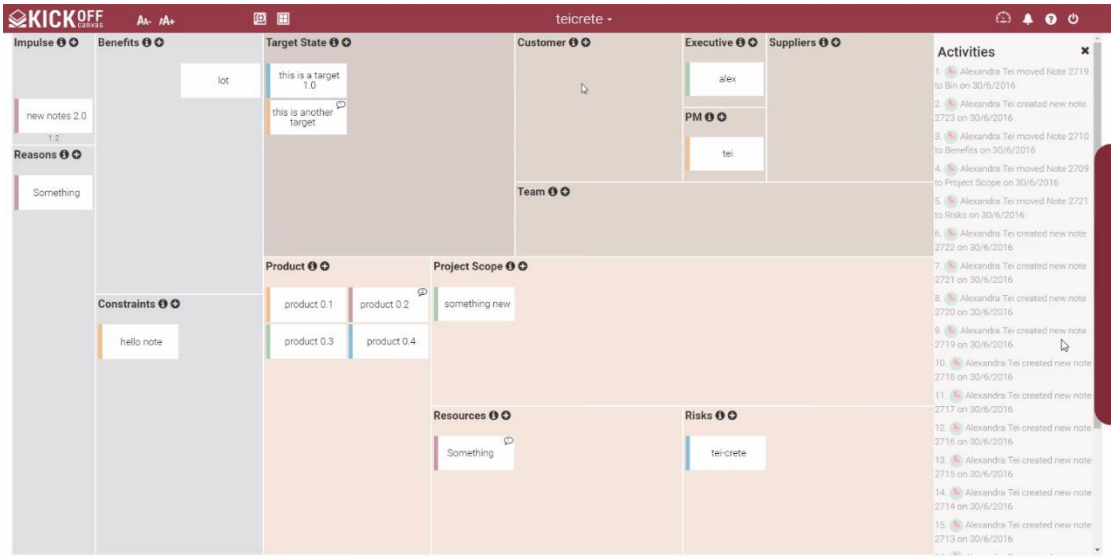
Στην παραπάνω εικόνα βλέπουμε ότι στο email tp3596@edu.teicrete.gr έχει έρθει ένα μήνυμα από την εφαρμογή που ειδοποιεί τον χρήστη ότι έχει προσκληθεί από την *Alexandra Tei* στον καμβά *teicrete*. Το email τον ενημερώνει επίσης για το ρόλο που έχει στον καμβά και τι μπορεί να κάνει σε αυτόν.



Εικόνα 4.2.2.33 Η ειδοποίηση στον καμβά για πρόσκληση ενός χρήστη σε καμβά

Βλέπουμε επίσης στην παραπάνω εικόνα την ειδοποίηση που παίρνει ο χρήστης στον καμβά του. Η ειδοποίηση βρίσκεται στο δεξιά του ονόματος του καμβά στο εικονίδιο που μοιάζει με κουδούνι. Αν ο χρήστης πατήσει πάνω στο μήνυμα της ειδοποίησης τότε πηγαίνει αυτόματα στον καμβά που προσκλήθηκε.

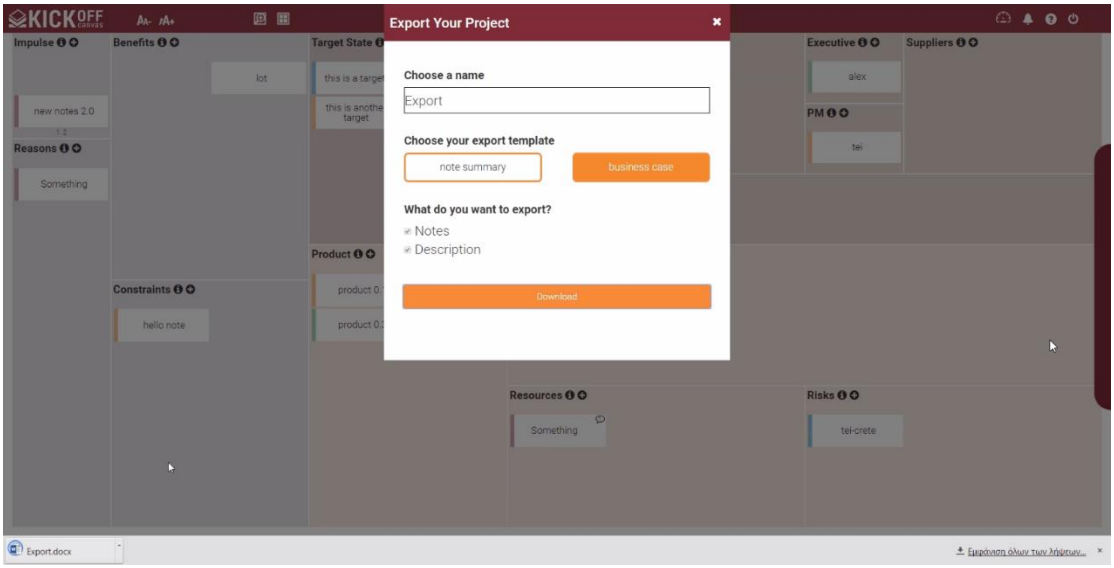
Με την επιλογή *latest activities*, εμφανίζεται στον καμβά μία στήλη όπου αναγράφονται όλες οι πρόσφατες δραστηριότητες που έγιναν στον καμβά όπως είναι η δημιουργία, η μετακίνηση ή η διαγραφή σημειώσεων.



Εικόνα 4.2.2.34 Η στήλη με τις πρόσφατες δραστηριότητες

Υπάρχει δυνατότητα κλεισίματος αυτής της στήλης πατώντας το κουμπί «*X*» που βρίσκεται πάνω δεξιά σε αυτή.

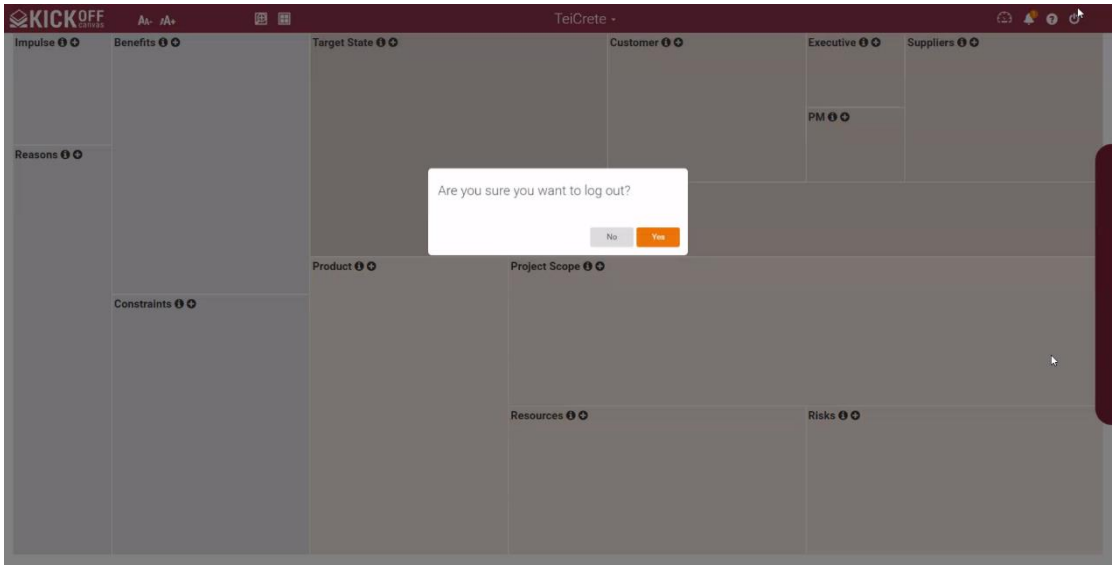
Η τελευταία επιλογή, η *export* δίνει τη δυνατότητα να εξάγουμε τον καμβά και τις σημειώσεις που υπάρχουν στα τμήματα σε *word* αρχείο.



Εικόνα 4.2.2.35 Η εξαγωγή των σημειώσεων του καμβά σε word

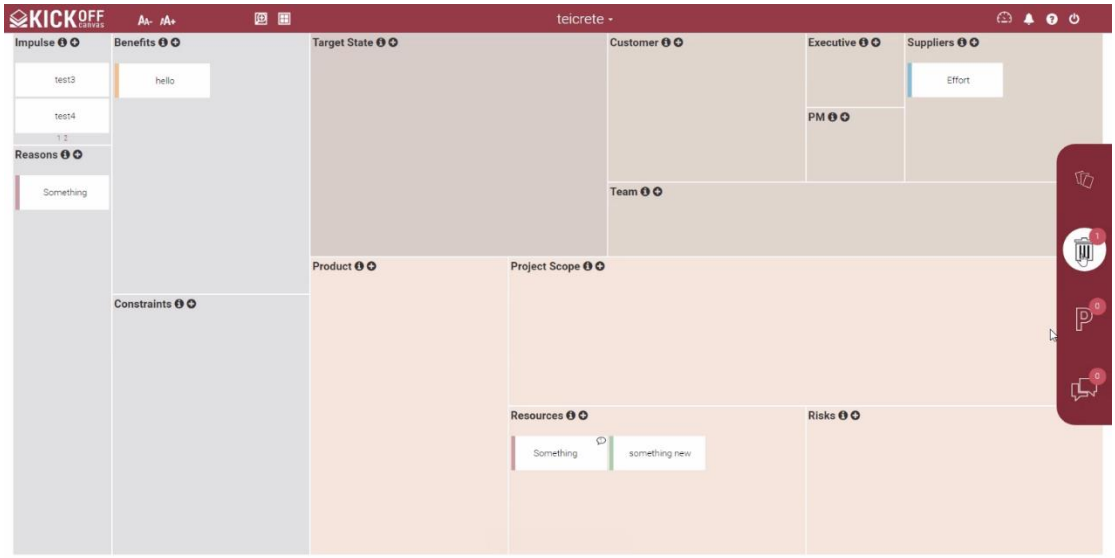
Με την επιλογή αυτή εμφανίζεται ένα popup παράθυρο το οποίο μας δίνει τη δυνατότητα να δώσουμε όνομα στο word αρχείο που θα δημιουργηθεί, να επιλέξουμε το πρότυπο (template) του αρχείου, δηλαδή την εμφάνιση που θα έχει το αρχείο και τέλος να επιλέξουμε αν το αρχείο θα περιέχει μόνο τις σημειώσεις ή μόνο την περιγραφή των τμημάτων του καμβά ή και τα δύο μαζί. Αφού ολοκληρώσουμε τις επιλογές μας πατάμε το κουμπί *download* και όπως βλέπουμε στο κάτω αριστερό άκρο της παραπάνω εικόνας το word αρχείο έχει κατέβει.

Τα υπόλοιπα κουμπιά που υπάρχουν δεξιά από το όνομα του καμβά είναι το κουμπί που μοιάζει με ταχύμετρο και οδηγεί πίσω στο dashboard, δίπλα του είναι το κουμπί για τις ειδοποιήσεις για το οποίο μιλήσαμε πιο πάνω, δίπλα από αυτό είναι το κουμπί για τη βοήθεια και το τελευταίο κουμπί, είναι το κουμπί της αποσύνδεσης.



Εικόνα 4.2.2.36 Η αποσύνδεση του χρήστη από την εφαρμογή

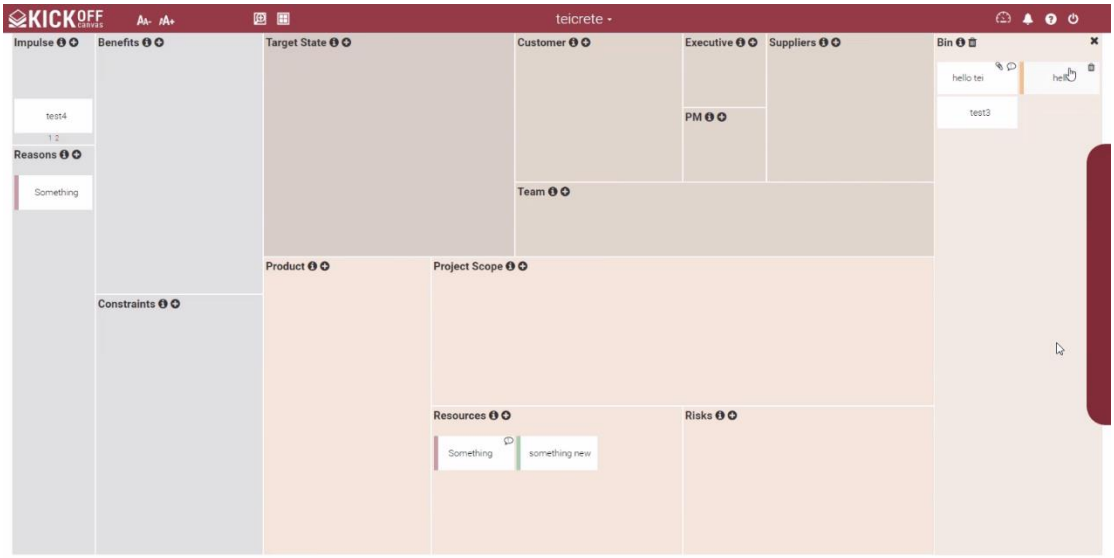
Όπως αναφέρθηκε στην αρχή, στα δεξιά του καμβά υπάρχει η μπάρα με τα widgets η οποία ενεργοποιείται με *mouseover*.



Εικόνα 4.2.2.37 Η μπάρα με τα widgets

Η μπάρα περιέχει τέσσερα widgets: το *planning poker*, τον *κάδο απορριμμάτων*, το *parking lot* και το *open issues* (ανοιχτά θέματα).

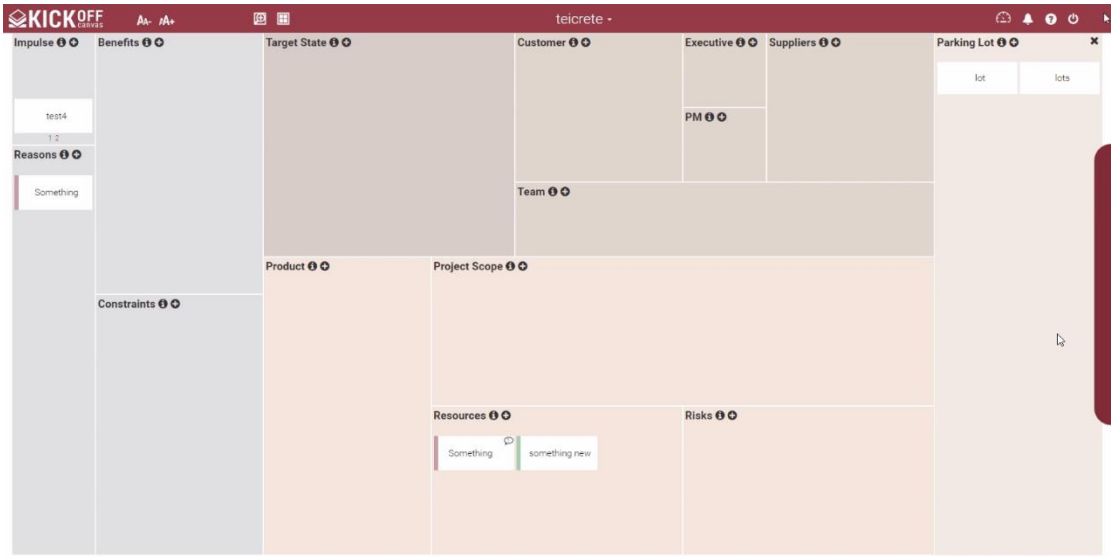
Το δεύτερο εικονίδιο είναι αυτό του κάδου απορριμμάτων. Μόλις πατηθεί εμφανίζεται δεξιά του καμβά μία στήλη που δείχνει τις διαγραμμένες σημειώσεις.



Εικόνα 4.2.2.38 Η στήλη του κάδου απορριμμάτων

Από εκεί ο χρήστης έχει τη δυνατότητα να επαναφέρει τις σημειώσεις στον καμβά μέσω *drag and drop* ή να τις διαγράψει τελείως πατώντας πάνω στο εικονίδιο του κάδου που βρίσκεται στη σημείωση ή να τις διαγράψει όλες μαζί πατώντας στο εικονίδιο του κάδου που βρίσκεται στη στήλη. Για να κλείσει η στήλη πατάμε στο «*X*» που βρίσκεται σε αυτή.

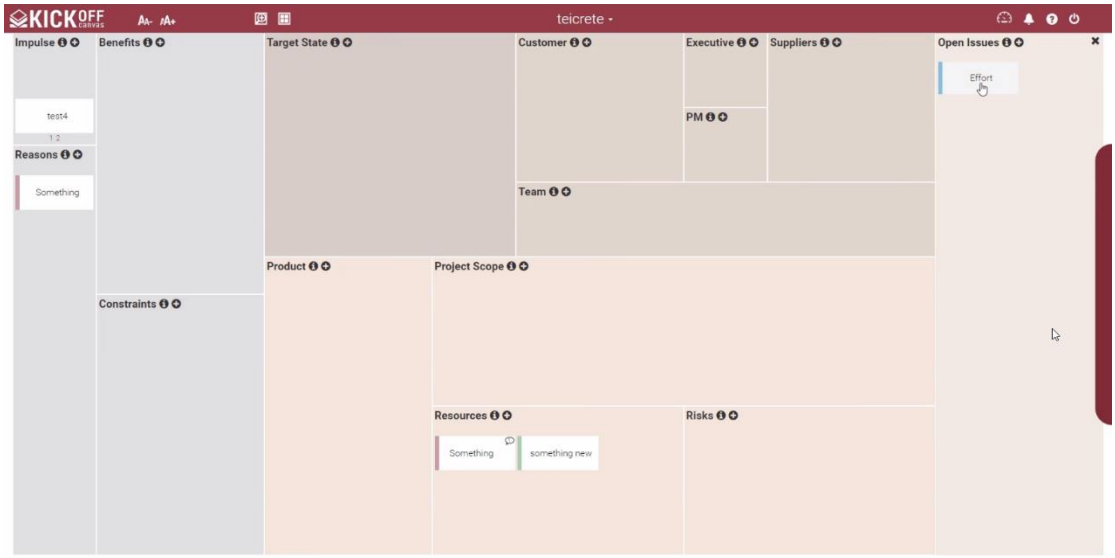
Το εικονίδιο που βρίσκεται κάτω από τον κάδο απορριμμάτων είναι αυτό του *parking lot*. Και εδώ, όταν πατηθεί εμφανίζεται στα δεξιά του καμβά μία στήλη.



Εικόνα 4.2.2.39 Η στήλη του parking lot

Στο parking lot μπαίνουν οι σημειώσεις για τις οποίες δεν υπάρχει κάποιο τμήμα στον καμβά στον οποίο να ταιριάζουν. Και εδώ μπορούν να μεταφερθούν στον καμβά οι σημειώσεις μέσω *drag and drop*. Επίσης η στήλη διαθέτει κουμπί δημιουργίας σημειώσεων για να μπορεί ο χρήστης να δημιουργεί απευθείας εκεί τις σημειώσεις ενώ υπάρχει και το κουμπί «*i*» που δίνει περισσότερες πληροφορίες για την στήλη. Για να κλείσει η στήλη πατάμε στο «*X*» που βρίσκεται σε αυτή.

Το τελευταίο εικονίδιο είναι αυτό των *ανοιχτών θεμάτων*. Στα ανοιχτά θέματα τοποθετούνται οι σημειώσεις που περιέχουν θέματα τα οποία δεν μπορούν να λυθούν κατά τη διάρκεια της Kickoff συνάντησης και πρέπει να λυθούν από τον επικεφαλής του έργου.

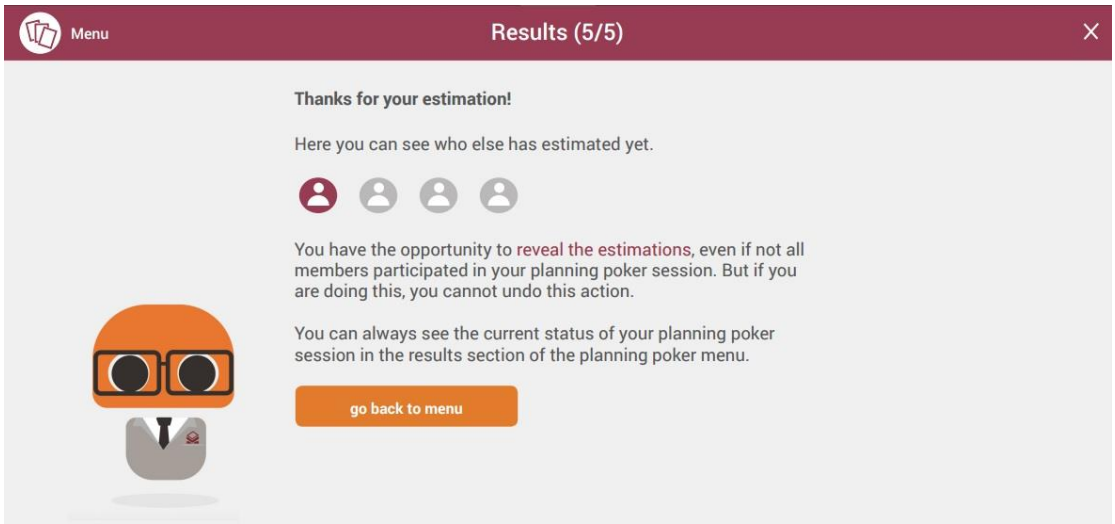


Εικόνα 4.2.2.40 Η στήλη με τα ανοιχτά θέματα

Όπως και πριν έτσι και εδώ με το πάτημα στο εικονίδιο εμφανίζεται μία στήλη δεξιά του καμβά. Οι σημειώσεις μπορούν να μεταφερθούν με *drag and drop* και επίσης υπάρχει δυνατότητα απευθείας δημιουργίας σημειώσεων στην στήλη ενώ υπάρχει και το κουμπί «*i*» που δίνει περισσότερες πληροφορίες για αυτή. Για να κλείσει η στήλη πατάμε στο «*X*» που βρίσκεται σε αυτή.

Και τα τρία widgets δουλεύουν με τον ίδιο ακριβώς τρόπο και στην μεγεθυμένη προβολή του καμβά.

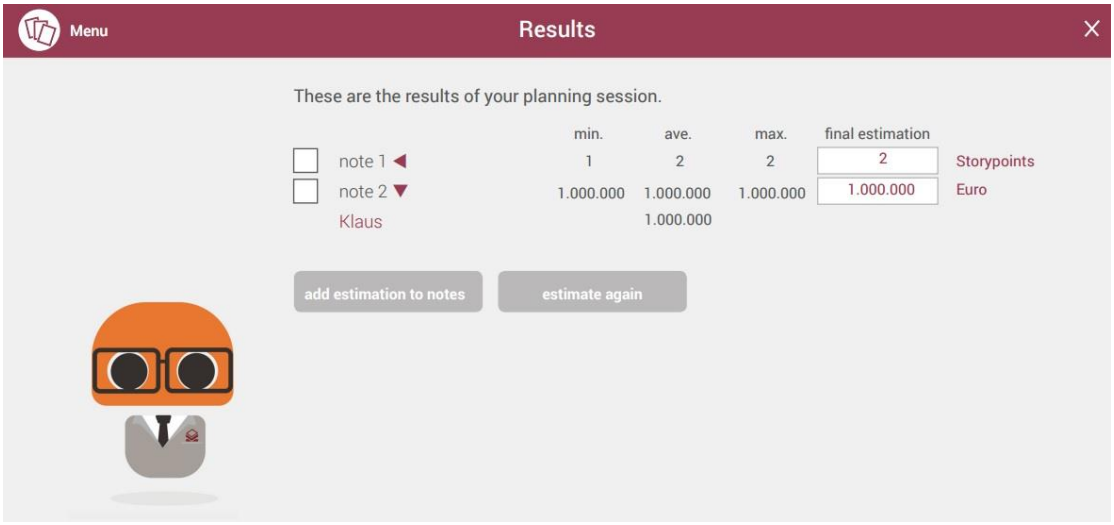
Το πρώτο εικονίδιο είναι αυτό του *planning poker*. Όταν πατηθεί εμφανίζεται ένα popup παράθυρο ώστε να αρχίσει το planning poker μεταξύ των μελών του καμβά. Στα πρώτα βήματα υπάρχει μία επεξήγηση σχετικά με το τι είναι το planning poker και πως μπορεί να χρησιμοποιηθεί το widget της εφαρμογής. Σημαντικό ρόλο εδώ παίζει ο βοηθός (bot) της εφαρμογής ο οποίος εξηγεί σε κάθε βήμα τι πρέπει να γίνει.



Εικόνα 4.2.2.41 Το πέμπτο βήμα του planning poker της εφαρμογής

Η διαδικασία ολοκληρώνεται σε πέντε βήματα. Κάθε μέλος του καμβά πληκτρολογεί την εκτίμηση του για το κόστος στο θέμα που υπάρχει στη σημείωση για την οποία γίνεται το planning poker.

Αφού ολοκληρωθούν τα βήματα εμφανίζονται τα αποτελέσματα για την εκτίμηση κόστους.



Εικόνα 4.2.2.42 Τα αποτελέσματα του planning poker

5 Αποτελέσματα

5.1 Συμπεράσματα

Η εν λόγω πτυχιακή αποτέλεσε μέρος ενός μεγαλύτερου έργου και αναφέρθηκε στη δική μας συνεισφορά στο έργο αυτό. Μέσω της συμμετοχής μας στο πρόγραμμα Blended Aim προσπαθήσαμε να αναπτύξουμε σε συνεργασία με άλλους φοιτητές μία διαδικτυακή εφαρμογή που βοηθά τον χρήστη της να αποφασίσει για την έναρξη ή όχι ενός έργου. Η εφαρμογή αναπτύχθηκε για την εταιρεία UWS και βασίστηκε πάνω σε δικά της ήδη υπάρχοντα εργαλεία. Το αποτέλεσμα ήταν ένα πρωτότυπο το οποίο δέχεται εξελίξεις. Ήταν μία λειτουργική εφαρμογή η οποία εξυπηρετούσε σε μεγάλο βαθμό τις αρχικές απαιτήσεις της εταιρείας.

Η εφαρμογή που αναπτύχθηκε μπορεί να εξυπηρετήσει τους σκοπούς εταιριών, διευθυντών αλλά και μικρών ομάδων ανθρώπων και επιχειρήσεων βγάζοντας τους από τη δύσκολη θέση του ρίσκου, εξοικονομώντας τους παράλληλα χρήματα και χρόνο. Πέραν της εφαρμογής η ίδια η πτυχιακή μπορεί να ωφελήσει φοιτητές της επιστήμης υπολογιστών που ενδιαφέρονται για τον διαδικτυακό προγραμματισμό αλλά και φοιτητές που ενδιαφέρονται να μάθουν για τις μεθοδολογίες ανάπτυξης λογισμικού ή οποιουδήποτε άλλου έργου. Τέλος μπορεί να ωφελήσει οποιονδήποτε θέλει να μάθει περισσότερα για τον τρόπο οργάνωσης και υλοποίησης του προγράμματος Blended Aim αλλά και τι είναι το ίδιο το πρόγραμμα.

Όσον αφορά τα δικά μας οφέλη από την πτυχιακή αυτή, ήταν πολλά. Μας δόθηκε η ευκαιρία να δουλέψουμε πάνω στον διαδικτυακό προγραμματισμό και να επεκτείνουμε τις γνώσεις που είχαμε ήδη από την σχολή αλλά και να αποκτήσουμε νέες. Εργαστήκαμε στο frontend της εφαρμογής και επεκτείναμε τις γνώσεις μας πάνω στις HTML, CSS και JavaScript αλλά ταυτόχρονα μάθαμε το AngularJS, το Bootstrap, το jQuery. Επίσης μας δόθηκε η ευκαιρία να αναπτύξουμε λογισμικό με επαγγελματικό τρόπο δουλεύοντας με μεθοδολογίες που χρησιμοποιούνται από εταιρείες λογισμικού (Scrum) αλλά και χρησιμοποιώντας σύστημα ελέγχου εκδόσεων για το λογισμικό όπως είναι το Git, κάνοντας μας έτσι πιο έτοιμους για την αγορά εργασίας. Παράλληλα χάρη στο Blended Aim νιώσαμε για μικρό χρονικό διάστημα την εμπειρία του Erasmus καθώς μετακινηθήκαμε στο εξωτερικό, γνωρίσαμε φοιτητές από άλλες χώρες με τους οποίους δουλέψαμε μαζί και με τους οποίους αναπτύξαμε φιλίες.

Επιπλέον, η συγκεκριμένη εργασία πήρε μέρος στο *International Symposium on Ambient Intelligence and Embedded Systems* ως δημοσίευση (*paper*). Ο στόχος αυτού του διεθνούς συμποσίου είναι να μοιραστούν οι γνώσεις και η εμπειρία στους τομείς της Διάχυτης Νοημοσύνης (Ambient Intelligence), των Ενσωματωμένων Συστημάτων (Embedded Systems), του Διαδικτυακού Προγραμματισμού (Internet Programming) και της Τεχνολογίας Πληροφοριών (Information Technology). Απευθύνεται σε ερευνητές και προγραμματιστές από την ακαδημαϊκή κοινότητα και τη βιομηχανία, καθώς και σε έμπειρους φοιτητές που εργάζονται σε αυτούς τους τομείς ειδικότητας. Φέτος η εκδήλωση διοργανώθηκε σε συνεργασία με το Τεχνολογικό Εκπαιδευτικό Ίδρυμα (TEI) Κρήτης, στο Ηράκλειο, στην Ελλάδα και με το Kiel University of Applied Sciences, της Γερμανίας. Η σειρά των εργαστηρίων ξεκίνησε από κοινού το 2002 από το Vaasa UAS και το Kiel UAS. Από τότε έχει διοργανωθεί με μεγάλη επιτυχία στο Vaasa, το Kiel, το Aveiro, το Geel, το Ηράκλειο, τη Madeira, τα Χανιά, το Espoo, το Βερολίνο, και την Oostende.

5.2 Μελλοντική Εργασία και Επεκτάσεις

Το τελικό προϊόν που σχεδίασε και υλοποίησε η ομάδα του blended aim 2016 είναι ένα πρότυπο της εφαρμογής με βασικές λειτουργίες, η οποία μπορεί να επεκταθεί περαιτέρω.

Στην εφαρμογή του KICKOFF canvas θα μπορούσαν να προστεθούν τα εξής χαρακτηριστικά.

Χαρακτηριστικά	
Άλλα templates καμβάδων	*Λόγω των ανταγωνιστών οι όποια αυξάνονται, συνετό θα ήταν να συμπεριληφθούν και διαφορετικοί καμβάδες, όπως είναι ο business model canvas, ο lean canvas, καθώς και ο value proposition canvas. Από την μία πλευρά, υπάρχουν ήδη πολλοί πελάτες που προτιμούν αυτούς τους καμβάδες και το KICKOFF canvas θα μπορούσε να επεκταθεί και σε αυτούς. Από την άλλη πλευρά, είναι σχετικά εύκολο να ενσωματωθούν κι άλλα templates διότι οι τεχνολογίες που θα χρησιμοποιηθούν θα είναι ίδιες.*
Τα δικά τους templates	*Πολλές εταιρίες έχουν ήδη σταθεροποιήσει την διαδικασία της διαχείρισης έργου περιλαμβάνοντας δικές τους απαιτήσεις. Για να είναι χρήσιμο το εργαλείο για αυτές τις εταιρίες θα πρέπει να τους δίνεται η δυνατότητα να δημιουργούν τα δικά τους templates και να ρυθμίζουν τα υπάρχοντα με τις δικές τους προτιμήσεις.*
Επιπλέον μεθόδους για την διαχείριση του έργου	*Είναι σημαντικό να μπορεί η εφαρμογή να ανταπεξέλθει στις απαιτήσεις κάθε εποχής ώστε να καλείται ειδικός στη διαχείριση του έργου. Προσθέτοντας νέες μεθόδους και τεχνικές στον καμβά, η UWS θα μπορεί να εγγυηθεί ότι είναι η καλύτερη λύσει για τους πελάτες της. Τα widgets είναι εκείνα τα οποία θα υποστηρίξουν αυτή την καινοτόμα διαδικασία και θα επικεντρωθούν στα εσωτερικά έργα.* *Πιθανά επιπλέον widgets:* • *SWOT-Analysis* • *Product Breakdown Structure* • *Empathy Map* • *Creativity Tools* • *Experiment Tools* • *Task Manager* • *Chat function* • *Calculation tools*
Επεξήγηση όλης της διαδικασίας του έργου	*Ώστε να ενισχυθεί το "Learning on the go", η UWS θα πρέπει να παρουσιάσει μια προοπτική της όλης διαδικασίας του έργου και πώς ενσωματώνεται το kick-off εντός της διαδικασίας. Με το flowPM μοντέλο τους έχουν ήδη μία πειστική διαδικασία, οπότε θα είναι εύκολο να παρέχουν περισσότερες πληροφορίες και υποστήριξη.*
Ενσωμάτωση σε άλλες πλατφόρμες	*Για το λόγο ότι το kick-off είναι ενσωματωμένο σε ολόκληρη την διαδικασία, θα ήταν συνετό θα ληφθούν υπόψη και άλλες πλατφόρμες οι οποίες χρησιμοποιούνται κατά την διάρκεια, ώστε να εγγυηθούν ότι το KICKOFF canvas προσφέρει μια πολύτιμη βάση για τις μελλοντικές φάσεις και αποφεύγονται οι περιττές ρυθμίσεις. Επιπλέον, η δυνατότητα να μοιράζονται τα αρχεία εύκολα θα διασφάλιζε ότι το KICKOFF canvas ταιριάζει στην ρουτίνα των εταιριών.* *Πιθανές πλατφόρμες είναι:*

- *E-Mail Integration*
- *Dropbox*
- *Google Drive*
- *Sharepoint*
- *Slack*
- *Outlook*

Πίνακας 5.2.1 Τα μελλοντικά χαρακτηριστικά της εφαρμογής

6 Βιβλιογραφία

[1] Kurose|Ross, 'Δικτύωση υπολογιστών, προσέγγιση από πάνω προς τα κάτω', 6η έκδοση

[2] Frank Turley, (2010) 'An introduction to PRINCE2'

[3] Simon Bourk & Patricia Kong, (Ιούνιος 2016) 'An Introduction to the Nexus™ Framework'

[4] Tutorials Point, (2016) 'Bootstrap tutorial' <http://www.tutorialspoint.com/bootstrap/index.htm>

[5] Wikipedia, (Οκτώβριος 2016) 'Font Awesome' <https://en.wikipedia.org/wiki/Font_Awesome>

[6] Font Awesome < http://fontawesome.io/>

[7] WhatIs.com, (1999-2016) 'PRINCE2' <http://whatis.techtarget.com/definition/PRINCE2>

[8] Wikipedia, (Σεπτέμβριος 2016) 'PRINCE2' <https://en.wikipedia.org/wiki/PRINCE2>

[9] PRINCE2.com, (2016) 'What is PRINCE2' <https://www.prince2.com/eur/what-is-prince2>

[10] Richard Moore, (Ιανουάριος 2016) 'A Brief History of PRINCE2 Project Management' <http://blog.goodelearning.com/prince2/a-brief-history-of-prince2-project-management/>

[11] Axelos GLOBAL BEST PRACTICE, (2016) 'Agile PRINCE2' <https://www.axelos.com/>

[12] JISC infoNET 'An introduction to PRINCE2' <http://www.prince2trainingen.nl/PRINCE2Websitefiles/Prince2%20introduction.pdf>

[13] Wikipedia, (Αύγουστος 2016) 'jQuery UI' <https://en.wikipedia.org/wiki/JQuery_UI>

[14] Wikipedia, (Οκτώβριος 2016) 'Hypertext' <https://en.wikipedia.org/wiki/Hypertext>

[15] Wikipedia, (Οκτώβριος 2016) 'Scrum (software development)' <https://en.wikipedia.org/wiki/Scrum_(software_development)>

[16] Scrum Alliance, (2016) 'Learn About Scrum' <https://www.scrumalliance.org/why-scrum>

[17] Scrum.org, (2016) 'Nexus Framework' <https://www.scrum.org/Resources/The-Nexus-Guide>

[18] Wikipedia, (Αύγουστος 2016) 'Cross-functional team' <https://en.wikipedia.org/wiki/Cross-functional_team>

[19] Wikipedia, (Αύγουστος 2016) 'Planning Poker' <https://en.wikipedia.org/wiki/Planning_poker>

[20] Mountain Goat, (1998-2016) 'Planning Poker' <https://www.mountaingoatsoftware.com/agile/planning-poker>

[21] Wikipedia, (Μάιος 2016) 'JavaScript engine' <https://en.wikipedia.org/wiki/JavaScript_engine>

[22] Wikipedia, (Οκτώβριος 2016) 'JavaScript' <https://en.wikipedia.org/wiki/JavaScript>

[23] Wikipedia, (Μάιος 2016) 'JavaScript' <https://el.wikipedia.org/wiki/JavaScript>

[24] Wikipedia, (Σεπτέμβριος 2016) 'JQuery' <https://el.wikipedia.org/wiki/JQuery>

[25] Wikipedia, (Οκτώβριος 2016) 'JQuery' <https://en.wikipedia.org/wiki/JQuery>

[26] jQuery, ‘What is jQuery?’ <https://jquery.com/>

[27] Wikipedia, (Οκτώβριος 2016) ‘AngularJS’ <https://en.wikipedia.org/wiki/AngularJS>

[28] AngularJS, (2010-2016) ‘Conceptual Overview’ <https://docs.angularjs.org/guide/concepts>

[29] AngularJS, (2010-2016) ‘Data Binding’ <https://docs.angularjs.org/guide/databinding>

[30] AngularJS, (2010-2016) ‘Understanding Controllers’ <https://docs.angularjs.org/guide/controller>

[31] AngularJS, (2010-2016) ‘Services’ <https://docs.angularjs.org/guide/services>

[32] AngularJS, (2010-2016) ‘What are Scopes?’ <https://docs.angularjs.org/guide/scope>

[33] AngularJS, (2010-2016) ‘Templates’ <https://docs.angularjs.org/guide/templates>

[34] AngularJS, (2010-2016) ‘Angular Expressions’ <https://docs.angularjs.org/guide/expression>

[35] AngularJS, (2010-2016) ‘Creating Custom Directives’ <https://docs.angularjs.org/guide/directive>

[36] AngularJS, (2010-2016) ‘What is a Module?’ <https://docs.angularjs.org/guide/module>

[37] AngularJS, (2010-2016) ‘HTML Compiler’ <https://docs.angularjs.org/guide/compiler>

[38] AngularJS, (2010-2016) ‘Bootstrap’ <https://docs.angularjs.org/guide/bootstrap>

[39] PRAXIS, (Δεκέμβριος 2015) ‘Praxis and bAIM project’ <http://www.praxisnetwork.eu/news/12/>

[40] GCU Newsroom, (Οκτώβριος 2015) ‘GCU supports international student employability with blended mobility project’ <http://www.gcu.ac.uk/newsroom/news/article/?id=125980>