Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών

Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή Εργασία

Τίτλος : Μελέτη σύγχρονων JavaScript Frameworks και πως αυτά χρησιμοποιούνται για την ανάπτυξη εφαρμογών. Παράδειγμα εφαρμογής με την χρήση API

Ζερβουδάκη Αικατερίνη (ΑΜ 3232)

e-mail : tp3232@edu.teicrete.gr

Ηράκλειο – Σεπτέμβριος 2016

Επιβλέπων καθηγητής : Παπαδάκης Νικόλαος

Ευχαριστίες

Με την ολοκλήρωση της πτυχιακής μου εργασίας, θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή κ. Παπαδάκη Νικόλαο για τη βοήθεια και τη στήριξη στην υλοποίηση της παρούσας εργασίας. Καθώς και ένα μεγάλο ευχαριστώ στην οικογένεια μου που με την πολύτιμη στήριξή της, ηθική και υλική, που απλόχερα μου πρόσφερε καθ' όλη την διάρκεια των σπουδών μου, συνέβαλε στην υλοποίηση του στόχου μου.

Abstract

The subject of this thesis is the study of modern JavaScript Frameworks, which now monopolize the interest of the developers since their use provide smaller in length, cleaner and more readable code.

This project presents the prevailing technologies, tools and frameworks that developers use for the development of web applications. After studying all these and through the use of the platform for the development of NodeJS software and the appropriate tools, we created an example covering all steps, from the selection and use of suitable technologies till the writing of the code, with detailed presentation of the development process.

The aim of this application is the optimal use of technologies for the development of the web application through the acquired knowledge from the research and study of this project.

Περίληψη

Η παρούσα πτυχιακή εργασία έχει ως θέμα τη μελέτη των σύγχρονων JavaScript Frameworks, τα οποία πλέον μονοπωλούν το ενδιαφέρον των προγραμματιστών μιας και η χρήση τους προσφέρει μικρότερο σε έκταση, καθαρότερο και πιο ευανάγνωστο κώδικα.

Σε αυτή την εργασία παρουσιάζονται οι επικρατέστερες τεχνολογίες, εργαλεία και frameworks που χρησιμοποιούν οι προγραμματιστές για την ανάπτυξη διαδικτυακών εφαρμογών. Έπειτα από την μελέτη όλων αυτών και με την χρήση της πλατφόρμας ανάπτυξης λογισμικού NodeJS και των κατάλληλων εργαλείων δημιουργήσαμε ένα παράδειγμα, καλύπτοντας όλα τα στάδια από την επιλογή και την χρήση των κατάλληλων τεχνολογιών μέχρι και την συγγραφή του κώδικα, με αναλυτική παρουσίαση της διαδικασίας ανάπτυξης.

Στόχος της εφαρμογής είναι η βέλτιστη χρήση των τεχνολογιών για την ανάπτυξη της διαδικτυακής εφαρμογής μέσω των γνώσεων που αποκτήθηκαν από την έρευνα και μελέτη της εργασίας αυτής.

Πίνακας Περιεχομένων

Πίνακας Εικόνων

1. Εισαγωγή

1.1 Κίνητρο για την Διεξαγωγή της Πτυχιακής Εργασίας

Τα τελευταία χρόνια υπάρχει μία ραγδαία εξέλιξη της τεχνολογίας και του διαδικτύου. Όλο και περισσότερες εταιρίες και επιχειρήσεις δημιουργούν τη δική τους ιστοσελίδα με σκοπό να προσελκύσουν πελάτες αλλά και να διαφημίσουν τα προϊόντα τους. Απ' την άλλη πλευρά, προγραμματιστές να ασχολούνται όλο και περισσότερο με τη δημιουργία ιστοσελίδων με τη χρήση διαφορετικών πλατφόρμων και γλωσσών προγραμματισμού. Αυτός ήταν ο κύριος λόγος της επιθυμίας μου να υλοποιήσω αυτή την εργασία , τη μελέτη και τη δημιουργία μίας διαδικτυακής εφαρμογής από το μηδέν.

1.2 Σκοπός και Στόχοι της Εργασίας

Ο σκοπός της εργασίας αυτής ήταν η μελέτη των σύγχρονων Javascript Frameworks και η υλοποίηση μίας βέλτιστης διαδικτυακής εφαρμογής χρησιμοποιώντας το Google Custom Search για την άντληση δεδομένων από το διαδίκτυο. Στόχος της εφαρμογής είναι να δώσει τη δυνατότητα στον αναγνώστη, τη κατανόηση της δημιουργίας μίας τέτοιου είδους εφαρμογής.

1.3 Δομή Εργασίας

Η παρούσα πτυχιακή εργασία είναι χωρισμένη σε τρία μέρη. Σε μία λεπτομερή αναφορά των τεχνολογιών και εργαλείων που συμβάλουν στην δημιουργία διαδικτυακών εφαρμογών , των δημοφιλέστερων Frameworks και τέλος στη δημιουργία της διαδικτυακής εφαρμογής χρησιμοποιώντας τα κατάλληλα εργαλεία έπειτα από την μελέτη και την έρευνα που έγινε για την εργασία αυτή.

2. Τεχνολογίες και Εργαλεία Υλοποίησης

2.1 Τεχνολογίες Ανάπτυξης Διαδικτυακών Εφαρμογών

2.1.1 Javascript

Εικόνα 1 : Λογότυπο JavaScript

Η JavaScript είναι μία επίπεδη, δυναμική, τυποποιημένη και ερμηνευτική γλώσσα προγραμματισμού. Έχει τυποποιηθεί βάσει της γλωσσικής προδιαγραφής ECMAScript. Μαζί με την HTML και την CSS, είναι μία από τις τρεις βασικές τεχνολογίες παραγωγής του Παγκόσμιου Ιστού· χρησιμοποιείται από την πλειοψηφία των δικτυακών τόπων και υποστηρίζεται από όλα τα σύγχρονα προγράμματα περιήγησης δίχως τη χρήση πρόσθετων εξαρτημάτων λογισμικού. Η JavaScript είναι πρωτότυπη και βασισμένη σε πρώτης τάξης λειτουργίες, κάτι που την καθιστά ως μια παραδειγματική πολύ-γλώσσα, που υποστηρίζει αντικειμενοστραφή, επιτακτικής ανάγκης και λειτουργικά στυλ προγραμματισμού. Διαθέτει μία Διεπαφή Προγραμματισμού Εφαρμογών (API-Application Programming Interface) για την εργασία με κείμενο, πίνακες, ημερομηνίες και κανονικές εκφράσεις, αλλά δεν περιλαμβάνει καμία Είσοδο/Εξοδο (I/O-Input/Output), όπως είναι η δικτύωση, η αποθήκευση ή οι γραφικές εγκαταστάσεις, διότι βασίζεται σε αυτά που διαθέτει το περιβάλλον υποδοχής στο οποίο είναι ενσωματωμένη.

Παρά κάποιες ονοματικές, συντακτικές και προδιαγραφικές ομοιότητες, η JavaScript και η Java δεν σχετίζονται μεταξύ τους και έχουν πολύ διαφορετικές σημασιολογίες. Το συντακτικό της JavaScript προέρχεται στην πραγματικότητα από την C, ενώ η σημασιολογία και ο σχεδιασμός της επηρεάζονται από τις γλώσσες προγραμματισμού Self και Scheme. Η JavaScript χρησιμοποιείται επίσης σε περιβάλλοντα που δεν βασίζονται στο δίκτυο, όπως τα έγγραφα PDF, οι εξειδικευμένοι φυλλομετρητές και οι μικρές εφαρμογές της επιφάνειας εργασίας. Νεότερες και γρηγορότερες εικονικές μηχανές JavaScript και πλατφόρμες βασισμένες πάνω τους έχουν αυξήσει επίσης τη δημοτικότητα της JavaScript για διαδικτυακές εφαρμογές από την πλευρά του διακομιστή. Από την πλευρά του πελάτη, η JavaScript έχει εφαρμοστεί παραδοσιακά ως μία ερμηνευτική γλώσσα, αλλά πιο πρόσφατα προγράμματα περιήγησης μόλις που προλαβαίνουν να εκτελέσουν μεταγλώττιση. Χρησιμοποιείται επίσης στην ανάπτυξη παιχνιδιών, στη δημιουργία εφαρμογών επιφάνειας εργασίας και φορητών εφαρμογών και στον προγραμματισμό δικτύων από πλευράς του διακομιστή(server-side) με νεότερες εικονικές μηχανές όπως το Node.js.

2.1.1.1 Ιστορία

Η JavaScript αναπτύχθηκε αρχικά μέσα σε 10 ημέρες, τον Μάιο του 1995 από τον Brendan Eich, καθώς αυτός εργαζόταν για την Εταιρεία Επικοινωνιών Netscape (Netscape Communications Corporation). Πράγματι, ενώ ανταγωνίζονταν τη Microsoft για την υιοθέτηση της χρήσης διαδικτυακών τεχνολογιών και πλατφόρμων, η Netscape έλαβε υπόψιν της το διακομιστή της από την πλευρά του πελάτη, προσφέροντας ένα κατανεμημένο λειτουργικό σύστημα με μια φορητή έκδοση της Sun Microsystems της Java, παρέχοντας ένα περιβάλλον στο οποίο θα μπορούσαν να «τρέξουν» μικροεφαρμογές. Επειδή η Java ήταν ανταγωνιστής της C++ και στόχο είχε τους επαγγελματίες προγραμματιστές, η Netscape ήθελε επίσης μία ελαφριά ερμηνευτική γλώσσα που θα συμπλήρωνε την Java με την προσφυγή σε μη επαγγελματίες προγραμματιστές, όπως η Visual Basic της Microsoft .

Αν και αναπτύχθηκε με το όνομα Mocha, η γλώσσα επίσημα ονομάστηκε LiveScript όταν διατέθηκε για πρώτη στην αγορά σε δοκιμαστικές εκδόσεις της Netscape Navigator 2.0 τον Σεπτέμβριο του 1995, αλλά μετονομάστηκε σε JavaScript όταν χρησιμοποιήθηκε στην έκδοση 2.0B3 του προγράμματος περιήγησης της Netscape.

Η αλλαγή της επωνυμίας από LiveScript σε JavaScript συνέπεσε στο περίπου με τη Netscape, προσθέτοντας υποστήριξη στη τεχνολογία της Java για το διαδικτυακό πρόγραμμα περιήγησης Navigator της Netscape. Η τελική επιλογή του ονόματος προκάλεσε σύγχυση, δίνοντας την εντύπωση ότι η γλώσσα ήταν ένα παρακλάδι της γλώσσας προγραμματισμού Java και η επιλογή έχει χαρακτηριστεί ως ένα τέχνασμα προώθησης από τη Netscape.

- **Τυποποίηση**

Το Νοέμβριο του 1996, η Netscape ανακοίνωσε ότι είχε υποβάλλει την JavaScript στο Διεθνή Οργανισμό Ecma (Ecma International) προκειμένου να εξεταστεί ως ένα βιομηχανικό πρότυπο, και στη συνέχεια το έργο είχε ως αποτέλεσμα την τυποποιημένη έκδοση με το όνομα ECMAScript. Τον Ιούνιο του 1997, ο Διεθνής Οργανισμός Ecma δημοσίευσε την πρώτη έκδοση της προδιαγραφής ECMA-262. Τον Ιούνιο του 1998, έγιναν ορισμένες τροποποιήσεις για να προσαρμοστεί στο πρότυπο ISO/IEC-16262, και δημοσιεύτηκε η δεύτερη έκδοση. Η τρίτη έκδοση της ECMA-262 δημοσιεύτηκε το Δεκέμβριο του 1999.

Η ανάπτυξη της τέταρτης έκδοσης του προτύπου ECMAScript δεν ολοκληρώθηκε ποτέ. Η πέμπτη έκδοση δημοσιεύτηκε το Δεκέμβριο του 2009. Η τρέχουσα έκδοση του προτύπου ECMAScript είναι η έκτη, η οποία κυκλοφόρησε τον Ιούνιο του 2015.

- **Μετέπειτα εξελίξεις**

Η JavaScript έχει γίνει μία από τις δημοφιλέστερες γλώσσες προγραμματισμού στο Παγκόσμιο Ιστό. Η έλευση της τεχνολογίας Ajax επέστρεψε την JavaScript στο προσκήνιο και τράβηξε περισσότερο την προσοχή των επαγγελματιών προγραμματιστών. Το αποτέλεσμα ήταν μία γρήγορη εξάπλωση των

συνολικών πλαισίων και βιβλιοθηκών, βελτιωμένες πρακτικές προγραμματισμού της JavaScript και αυξημένη χρήση της JavaScript έξω από τα προγράμματα περιήγησης στο Παγκόσμιο Ιστό.

Τον Ιανουάριο του 2009, ιδρύθηκε το CommonJS με στόχο τον καθορισμό ενός κοινού προτύπου βιβλιοθήκης, κυρίως για την ανάπτυξη της JavaScript έξω από το πρόγραμμα περιήγησης.

2.1.1.2 Χαρακτηριστικά

Τα παρακάτω χαρακτηριστικά είναι κοινά σε όλες τις προσαρμοσμένες εφαρμογές της ECMAScript, εκτός εάν ορίζεται ρητά κάτι διαφορετικό.

- Επιτακτική και δομημένη

  Η JavaScript υποστηρίζει ένα μεγάλο μέρος του δομημένου προγραμματισμού σύνταξης της C (π.χ. εντολές if, while loops, εντολές switch, do while loops κ.τ.λ.). Μία μερική εξαίρεση είναι η οριοθέτηση πεδίου: η JavaScript αρχικά είχε μόνο τη λειτουργική οριοθέτηση πεδίου με var. Η ECMAScript 2015 προσθέτει μία λέξη κλειδί let για block scoping, που σημαίνει ότι τώρα η JavaScript έχει τόσο τη λειτουργική οριοθέτηση πεδίου όσο και την οριοθέτηση πεδίου block. Όπως και η C, η JavaScript κάνει διάκριση μεταξύ των εκφράσεων και των εντολών. Μία συντακτική διάφορα σε σχέση με τη C είναι η αυτόματη εισαγωγή ερωτηματικού, η οποία επιτρέπει στα ερωτηματικά, που κανονικά θα τερμάτιζαν οι εντολές, να παραλειφθούν.

- Δυναμική

  Πληκτρολόγηση

  Όπως στις περισσότερες γλώσσες προγραμματισμού, οι τύποι σχετίζονται με τιμές, όχι με μεταβλητές. Για παράδειγμα, μία μεταβλητή x θα μπορούσε να συνδεθεί με έναν αριθμό και αργότερα να επανασυνδεθεί με μια συμβολοσειρά. Η JavaScript υποστηρίζει διάφορους τρόπους για τον έλεγχο του τύπου ενός αντικειμένου, συμπεριλαμβανομένης και της πληκτρολόγησης duck (duck typing).

  Αντικειμενοστραφής

  Η JavaScript είναι σχεδόν εξ' ολοκλήρου αντικειμενοστραφής Τα αντικείμενα της JavaScript είναι συνειρμικοί πίνακες, εμπλουτισμένοι με πρωτότυπα. Οι ονομασίες των αντικειμένων είναι λέξεις κλειδιά. Αυτές υποστηρίζουν δύο ισοδύναμες συντάξεις: τη σημείωση κουκίδας (dot notation) (obj.x = 10) και τη σημείωση παρένθεσης (bracket notation) (obj ['x'] = 10). Οι ιδιότητες και οι τιμές τους μπορούν να προστεθούν, να αλλαχθούν ή να διαγραφούν κατά τον χρόνο εκτέλεσης. Οι περισσότερες ιδιότητες ενός αντικειμένου μπορούν να απαριθμηθούν χρησιμοποιώντας ένα βρόγχο for …in. Η JavaScript έχει ένα μικρό αριθμό ενσωματωμένων αντικειμένων όπως είναι η Function και η Date.

Αξιολόγηση χρόνου εκτέλεσης

Η JavaScript περιλαμβάνει μία λειτουργία eval η οποία μπορεί να εκτελέσει εντολές που δίνονται ως συμβολοσειρές κατά τον χρόνο εκτέλεσης.

- Αντικειμενοστραφής προγραμματισμός με βάση τα πρωτότυπα

Πρωτότυπα

Η JavaScript χρησιμοποιεί πρωτότυπα καθώς πολλές άλλες αντικειμενοστραφείς γλώσσες χρησιμοποιούν κλάσεις για κληρονομικότητα. Είναι δυνατό να προσομοιώσουμε πολλά χαρακτηριστικά βασισμένα στις κλάσεις με πρωτότυπα στη JavaScript.

Οι λειτουργίες ως κατασκευαστές αντικειμένων

Οι λειτουργίες διπλασιάζονται ως κατασκευαστές αντικειμένων μαζί με τον τυπικό τους ρόλο. Η πρόσθεση μίας νέας λειτουργίας σε μία λειτουργία call, θα δημιουργήσει ένα υπόδειγμα ενός πρωτοτύπου, το οποίο θα κληρονομήσει ιδιότητες και μεθόδους από τον κατασκευαστή (συμπεριλαμβανομένων και ιδιοτήτων από το πρωτότυπο Object (Αντικείμενο)). Η ECMAScript 5 προσφέρει τη μέθοδο δημιουργίας Object, επιτρέποντας τη ρητή δημιουργία ενός υποδείγματος χωρίς να κληρονομήσει αυτόματα από το πρωτότυπο Object. Η ιδιότητα του κατασκευαστή του πρωτοτύπου ορίζει το αντικείμενο που θα χρησιμοποιηθεί για το καινούριο εσωτερικό πρωτότυπο του αντικειμένου. Νέες μέθοδοι μπορούν να προστεθούν με την τροποποίηση της λειτουργίας του πρωτοτύπου ως ένας κατασκευαστής. Οι ενσωματωμένοι κατασκευαστές της JavaScript, όπως είναι ο Array(Πίνακας) και το Object, έχουν επίσης πρωτότυπα που μπορούν να τροποποιηθούν. Αν και είναι εφικτό να τροποποιηθεί το πρωτότυπο Object, θεωρείται γενικά μια κακή πρακτική επειδή τα περισσότερα αντικείμενα στη JavaScript θα κληρονομήσουν μεθόδους και ιδιότητες από το πρωτότυπο Object και ίσως να μην αναμένουν την τροποποίηση του πρωτοτύπου.

Οι συναρτήσεις ως μέθοδοι

Σε αντίθεση με πολλές αντικειμενοστραφείς γλώσσες, δεν υπάρχει διάκριση μεταξύ του ορισμού της συνάρτησης και του ορισμού της μεθόδου. Αντίθετα, η διάκριση γίνεται κατά τη διάρκεια που καλείται μία συνάρτηση· όταν κληθεί μία συνάρτηση ως μία μέθοδος ενός αντικειμένου, η λέξη κλειδί της συνάρτησης συνδέεται με εκείνο το αντικείμενο για αυτή την επίκληση.

- Έμμεση και άμεση ανάθεση

Η JavaScript είναι μία γλώσσα ανάθεσης.

Οι συναρτήσεις ως Ρόλοι (Traits και Mixins)

Η JavaScript υποστηρίζει εγγενώς διάφορες εφαρμογές με βάση λειτουργίες ως πρότυπα Ρόλων (Role patterns) όπως είναι τα Traits και Mixins. Μία τέτοια συνάρτηση καθορίζει επιπλέον συμπεριφορά με τουλάχιστον μία λειτουργία συνδεδεμένη με τη λέξη κλειδί this μέσα στο σώμα της συνάρτησης. Ένας Role μετά πρέπει να ανατεθεί άμεσα μέσω του call ή να εφαρμοστεί σε

αντικείμενα που χρειάζονται να παρουσιάσουν επιπλέον συμπεριφορά η οποία δεν διανέμεται μέσω της αλυσίδας του πρωτοτύπου.

Σύνθεση Αντικειμένου και Κληρονομικότητα

Ενώ η άμεση ανάθεση με βάση λειτουργίες δεν καλύπτει τη σύνθεση στη JavaScript, η έμμεση ανάθεση ήδη συμβαίνει κάθε φορά που η πρωτότυπη αλυσίδα προχωράει προκειμένου, π.χ., να βρει μία μέθοδο με την οποία ίσως να σχετίζεται αλλά και η οποία μέθοδος δεν θα ανήκει άμεσα σε κάποιο αντικείμενο. Μόλις η μέθοδος βρεθεί, λαμβάνει το όνομά της μέσα στο πλαίσιο του αντικειμένου. Έτσι, η κληρονομικότητα στη JavaScript καλύπτεται από μία αυτόματη ανάθεση η οποία συνδέεται με την ιδιότητα του πρωτοτύπου των κατασκευαστικών λειτουργιών.

- Ποικίλης

Περιβάλλον χρόνου εκτέλεσης

Η JavaScript βασίζεται συνήθως σε ένα περιβάλλον χρόνου εκτέλεσης (π.χ., σε ένα πρόγραμμα περιήγησης του Ιστού) προκειμένου να παρέχει αντικείμενα και μεθόδους βάσει των οποίων σενάρια μπορούν να αλληλοεπιδρούν με το περιβάλλον (π.χ., μία ιστοσελίδα DOM). Βασίζεται επίσης στο περιβάλλον χρόνου εκτέλεσης προκειμένου να παρέχει την ι κανότητα να συμπεριληφθούν εισαγωγικά σενάρια (import scripts) (πχ., HTML <script> στοιχεία). Αυτό δεν είναι ένα καθαυτού χαρακτηριστικό της γλώσσας, αλλά είναι συνηθισμένο στις περισσότερες εφαρμογές της JavaScript.

Η JavaScript επεξεργάζεται ένα-ένα τα μηνύματα από μία σειρά. Κατά τη φόρτωση ενός μηνύματος, η JavaScript καλεί μία λειτουργία που είναι συνδεδεμένη με αυτό το μήνυμα, η οποία δημιουργεί ένα πλαίσιο στοίβας κλήσεων (call stack frame) (τα arguments των συναρτήσεων και τις τοπικές μεταβλητές). Η στοίβα των κλήσεων μειώνεται και αυξάνεται βάσει των αναγκών των συναρτήσεων. Μετά την ολοκλήρωση της λειτουργίας, όταν η στοίβα είναι άδεια, η JavaScript προχωράει στο επόμενο μήνυμα της σειράς. Αυτό ονομάζεται event loop, και περιγράφεται ως «run to completion» επειδή κάθε μήνυμα δέχεται πλήρη επεξεργασία πριν το επόμενο μήνυμα ληφθεί υπόψη. Ωστόσο, το μοντέλο συγχρονισμού της γλώσσας περιγράφει το event loop ως non-blocking: η είσοδος/έξοδος (input/output) του προγράμματος γίνεται με τη χρήση των συναρτήσεων event και callback. Αυτό σημαίνει, για παράδειγμα, ότι η JavaScript μπορεί να επεξεργαστεί ένα κλικ του ποντικιού καθώς περιμένει να της επιστρέψει πληροφορίες ένα ερώτημα βάσης δεδομένων.

Variadic συναρτήσεις

Ένας απροσδιόριστος αριθμός παραμέτρων μπορεί να μεταβιβαστεί σε μία λειτουργία. Η λειτουργία μπορεί να αποκτήσει πρόσβαση σε αυτές μέσω επίσημων παραμέτρων και μέσω, επίσης, local arguments αντικειμένων. Οι Variadic συναρτήσεις μπορούν ακόμα να δημιουργηθούν μέσω της χρήσης της μεθόδου δέσμευσης (bind method).

Πίνακες και αντικείμενα

Όπως σε πολλές γλώσσες υλοποίησης σεναρίων, οι πίνακες και τα αντικείμενα (συνειρμικοί πίνακες σε άλλες γλώσσες) μπορούν να δημιουργηθούν με μία σαφή σύντομη σύνταξη. Στην πραγματικότητα, αυτοί οι τύποι αποτελούν τη βάση των δεδομένων διαμόρφωσης της JSON.

Κανονικές εκφράσεις (Regular expressions)

Η JavaScript υποστηρίζει επίσης κανονικές εκφράσεις με τρόπο παρόμοιο όπως η Perl, ο οποίος παρέχει ένα συνοπτικό και ισχυρό συντακτικό για χειραγώγηση κειμένου, το οποίο είναι πιο εκλεπτυσμένο από τις λειτουργίες των ενσωματωμένων συμβολοσειρών.

2.1.1.3 Σύνταξη

- Παράδειγμα 1ο :

Οι μεταβλητές στη JavaScript μπορούν να οριστούν χρησιμοποιώντας την λέξη κλειδί var :

```
1
2   var x; // ορίζει την μεταβλητή x
3   var y = 3; // ορίζει την μεταβλητή y και της αναθέτει την τιμή 3
4
```

Εικόνα 2 : Παράδειγμα ορισμού μεταβλητών

Μία απλή αναδρομική συνάρτηση :

```
5
6   function f(n) {
7       if (n == 0){
8           return 1;
9       }
10      return n*f(n-1);
11  }
12
```

Εικόνα 3 : Παράδειγμα αναδρομικής συνάρτησης

Επίδειξη λειτουργίας Variadic :

```
13
14  function sum() {
15      var x = 0;
16
17      for (var i = 0; i < arguments.length; ++;){
18          x += arguments[i];
19      }
20      return x;
21  }
22
23  sum(1, 2); // επιστρέφει 3
24  sum(1, 2, 3); // επιστρέφει 6
25
```

Εικόνα 4 : Παράδειγμα Variadic

- Παράδειγμα 2ο :

Ο κώδικας Javascript μιας σελίδας περικλείεται από τις ετικέτες της HTML

<script type="text/javascript"> και </script>.

Για παράδειγμα , ο ακόλουθος κώδικας Javascript εμφανίζει ένα πλαίσιο διαλόγου με το κείμενο «Hello World!».

```
27
28  <script type="text/javascript">
29  alert('Hello World!');
30  </script>
```

Εικόνα 5 : Παράδειγμα με πλαίσιο διαλόγου

2.1.2 SQL

Εικόνα 6 : Λογότυπο SQL

Η SQL(αρχικά από το Structured Query Language) είναι μία γλώσσα προγραμματισμού ειδικού σκοπού σχεδιασμένη για την διαχείριση δεδομένων, σε ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (Relational Database Management System, RDBMS) ή για την επεξεργασία της ροής σε ένα σύστημα διαχείρισης ροής δεδομένων (Relational Data Stream Management System, RDSMS).

Βασίστηκε στη σχεσιακή άλγεβρα και πλειάδα σχεσιακού λογισμού, η SQL αποτελείται από μία γλώσσα ορισμού δεδομένων, τη γλώσσα χειρισμού δεδομένων και τη Data Control Language. Η SQL ήταν μία από τις πρώτες εμπορικές γλώσσες για το σχεσιακό μοντέλο του Edgar F. Codd, όπως περιγράφετε στο σημαντικό άρθρο του το 1970, και έγινε η πιο διαδεδομένη γλώσσα για τις σχεσιακές βάσεις δεδομένων.

2.1.2.1 Ιστορία

Η SQL αναπτύχθηκε αρχικά στην IBM από τους Andrew Richardson, Donald D. Chamberlin και Raymond F. Boyce στις αρχές της δεκαετίας του 1970. Αυτή η έκδοση, αρχικά ονομαζόταν SEQUEL (Structured Query Language English), σχεδιάστηκε για να χειριστεί και να ανακτήσει τα δεδομένα που είναι αποθηκευμένα στο πρώτο RDBMS της IBM, το System R.

Στα τέλη της δεκαετίας του 1970, η Relational Software, Inc. (τώρα Oracle Corporation) είδε τη δυνατότητα των εννοιών που περιγράφονται από Codd, Chamberlin, και Boyce, και ανάπτυξε την SQL βασισμένο στο RDBMS με τις φιλοδοξίες πώλησης στο Πολεμικό Ναυτικό των ΗΠΑ, την Κεντρική Υπηρεσία Πληροφοριών, και άλλες κυβερνητικές υπηρεσίες των ΗΠΑ. Τον Ιούνιο του 1979,η Relational Software, Inc. παρουσίασε τη πρώτο εμπορικά διαθέσιμη εφαρμογή της SQL, την Oracle V2 (VERSION2) για υπολογιστές VAX.

2.1.2.2 Σύνταξη

Γλωσσικά στοιχεία

Η γλώσσα SQL υποδιαιρείται σε διάφορα γλωσσικά στοιχεία, που περιλαμβάνουν :

- Clauses, οι οποίες είναι συστατικά στοιχεία των δηλώσεων και ερωτήσεων(Σε ορισμένες περιπτώσεις, αυτές είναι προαιρετικές),

- Expressions, οι οποίες μπορούν να παράγουν είτε τις κλιμακωτές τιμές είτε πίνακες που αποτελούνται από τις στήλες και τις σειρές των δεδομένων
- Predicates, τα οποία καθορίζουν τους όρους που μπορούν να αξιολογηθούν σαν σωστό ή λάθος.
- Queries που ανακτούν τα στοιχεία βασισμένες σε συγκεκριμένα κριτήρια. Αυτό είναι ένα σημαντικό στοιχείο της SQL.
- Statements που μπορούν να έχουν μια επίμονη επίδραση στα σχήματα και τα στοιχεία, ή που μπορούν να ελέγχουν τις συναλλαγές, τη ροή του προγράμματος, τις συνδέσεις, τις συνεδρίες ή τα διαγνωστικά. Τα SQL Statement περιλαμβάνουν επίσης το ερωτηματικό « ; » δήλωση τερματισμού. Αν και δεν απαιτείται σε κάθε πλατφόρμα, ορίζεται ως ένα τυπικό μέρος της SQL γραμματικής.
- Το κενό συνήθως αγνοείται στις Statementsκαι τις Queries SQL, όμως καθιστά ευκολότερη την διαμόρφωση κώδικα SQL για την ανάγνωσή του.

- Παράδειγμα 1ο :

Ένα απλό παράδειγμα είναι το ακόλουθο :

```
2
3   SELECT *
4   FROM foitites
5   where onoma = 'Maria'
6
```

Εικόνα 7 : Απλό παράδειγμα SQL

- Παράδειγμα 2ο :

```
8
9   CASE WHEN n > 0
10          THEN 'positive'
11      WHEN n < 0
12          THEN 'negative'
13          ELSE 'zero'
14  END
15
```

Εικόνα 8 : Παράδειγμα SQL

2.2 Πλατφόρμες Ανάπτυξης Διαδικτυακών Εφαρμογών

2.2.1 NodeJS

Εικόνα 9 : Λογότυπο NodeJS

Το Node.js είναι ένα ανοιχτού κώδικα, διασταύρωσης πλατφόρμων, περιβάλλον χρόνου εκτέλεσης για τον προγραμματισμό εφαρμογών από την πλευρά του διακομιστή και είναι χτισμένη σε περιβάλλον Javascript.

Το Node.js παρέχει μία αρχιτεκτονική βάσει συμβάντων και μία non-blocking I/O API σχεδιασμένη να βελτιστοποιεί την απόδοση και κλιμάκωση μιας εφαρμογής. Σε αντίθεση με τα περισσότερα σύγχρονα περιβάλλοντα ανάπτυξης εφαρμογών δικτύων μία διεργασία node στηρίζεται σε ένα μοντέλο ασύγχρονης επικοινωνίας είσόδου/εξόδου και περιέχει μία ενσωματωμένη βιβλιοθήκη για να επιτρέπει στις εφαρμογές να λειτουργούν ως ένας αυτόνομος διαδικτυακός διακομιστής.

Κάποιοι από τους χρήστες του λογισμικού NodeJS είναι το Linkedln , Microsoft, Netflix, Paypal και Yahoo!.

2.2.1.1 Ιστορία

Το Node.js εφευρέθηκε το 2009 από τον Ryan Dahl και άλλους προγραμματιστές που εργάζονταν στη Joyent. Το Node.js δημιουργήθηκε και εκδόθηκε πρώτη φορά για χρήση των Linux το 2009. Η ανάπτυξη και διατήρησή του καθοδηγήθηκαν από τον Dahl και χρηματοδοτήθηκαν από την Joyent, την εταιρεία που εργαζόταν ο Dahl. Ο Dahl εμπνεύστηκε τη δημιουργία του node από την ανάγκη του να ενημερώσει τον χρήστη σε πραγματικό χρόνο για την κατάσταση ενός αρχείου που ανεβάζει στο διαδίκτυο. Μετά από αρκετές αποτυχημένες προσπάθειες σε C, Lua και Haskell , η κυκλοφορία της V8 Javascript Engine της Google τον παρέτρεψε να ασχοληθεί με την Javascript.

Το 2011, ένας διαχειριστής πακέτων εισήχθη στη βιβλιοθήκη Node.js, με το όνομα npm.Ο διαχειριστής πακέτων επιτρέπει τη δημοσίευση και τον διαμοιρασμό των ανοιχτού κώδικα βιβλιοθηκών Node.js από την κοινότητα, και είναι σχεδιασμένος να απλοποιεί την εγκατάσταση, αναβάθμιση και απεγκατάσταση των βιβλιοθηκών. Το πρώτο Node.js που κατασκευάστηκε για να υποστηρίζει τα Windows κυκλοφόρησε τον Ιούλιο του 2011.

2.2.1.2 Επισκόπηση

Το Node.js επιτρέπει τη δημιουργία διαδικτυακών διακομιστών και εργαλείων δικτύωσης που χρησιμοποιούν την JavaScript και μία συλλογή από modules που χειρίζονται διάφορες βασικές λειτουργίες. Τα modules χειρίζονται το σύστημα αρχείων εισόδου/εξόδου, τη δικτύωση (HTTP, TCP, UDP, DNS, ή TLS/SSL), δυαδικά δεδομένα (ενδιάμεσες μνήμες), λειτουργίες κρυπτογράφησης, ροές δεδομένων και άλλες βασικές λειτουργίες. Τα modules του Node χρησιμοποιούν ένα API που είναι σχεδιασμένο να μειώνει την πολυπλοκότητα της συγγραφής εφαρμογών διακομιστών. Τα πλαίσια μπορούν να χρησιμοποιηθούν για να επιταχύνουν την ανάπτυξη εφαρμογών και κοινών πλαισίων όπως τα Express.js, Socket.IO και Connect.

Το Node.js αρχικά χρησιμοποιήθηκε για την κατασκευή προγραμμάτων δικτύου, όπως διαδικτυακών διακομιστών, κάτι που το καθιστούσε παρόμοιο με την PHP και την Python. Η μεγαλύτερη διαφορά μεταξύ της PHP και του Node.js είναι ότι η PHP είναι μια blocking γλώσσα, όπου οι εντολές εκτελούνται αφού πρώτα έχει ολοκληρωθεί η προηγούμενη εντολή, ενώ το Node.js είναι μια non-blocking γλώσσα όπου οι εντολές εκτελούνται παράλληλα, και χρησιμοποιεί ανακλήσεις για να σηματοδοτήσει κάποια ολοκλήρωση.

2.2.1.3 Τεχνικές Λεπτομέρειες

Το NodeJS είναι ένα ένα Javascript περιβάλλον χρόνου εκτέλεσης που επεξεργάζεται τις εισερχόμενες αιτήσεις σε έναν βρόγχο, ο οποίος ονομάζεται βρόγχος συμβάντων(event loop).

- Threading

Το Node.js λειτουργεί σε ένα μονό νήμα (thread), χρησιμοποιώντας non-blocking κλήσεις εισόδου/εξόδου, κάτι που του επιτρέπει να υποστηρίζει δεκάδες χιλιάδες ταυτόχρονες συνδέσεις χωρίς να αναλαμβάνει το βάρος μεταγωγής του κόστους του περιεχομένου του νήματος. Ο σχεδιασμός του διαμοιρασμού ενός μονού νήματος μεταξύ όλων των αιτημάτων προορίζεται για την κατασκευή εξαιρετικά ταυτόχρονων εφαρμογών, όπου οποιαδήποτε λειτουργία που εκτελεί είσοδο/έξοδο πρέπει να χρησιμοποιεί μία ανάκληση. Για να φιλοξενηθεί ο βρόγχος συμβάντων μονού σήματος, το Node.js χρησιμοποιεί την βιβλιοθήκη libuv η οποία με την σειρά της χρησιμοποιεί μία σταθερών διαστάσεων πισίνα (pool) νήματος που είναι υπεύθυνη για όλες τις non-blocking ασύγχρονες εισόδου/εξόδου λειτουργίες.

- V8

Η V8 είναι η μηχανή εκτέλεσης της JavaScript που είναι κατασκευασμένη για το Google Chrome και ανοιχτού κώδικα από την Google το 2008. Γραμμένη σε C++, η V8 μεταγλωττίζει τον πηγαίο κώδικα JavaScript σε εγγενή κώδικα μηχανής αντί να τον ερμηνεύσει σε πραγματικό χρόνο.

Το Node.js περιέχει libuv για να χειρίζεται ασύγχρονα συμβάντα. Το libuv είναι επίπεδο αφαίρεσης για δικτυακή και συστημική λειτουργικότητα αρχείων, τόσο στα Windows όσο και στα συστήματα βασισμένα σε POSIX όπως τα Linux, Mac OS X, OSS στο NonStop και Unix.

Η κύρια λειτουργικότητα του Node.js βρίσκεται σε μία JavaScript βιβλιοθήκη. Οι συνδέσεις Node.js γραμμένες σε C++, συνδέουν αυτές τις τεχνολογίες μεταξύ τους και με το λειτουργικό σύστημα.

- Διαχείριση πακέτων

Το npm είναι ο προ-εγκατεστημένος διαχειριστής πακέτων για την πλατφόρμα διακομιστή Node.js. Χρησιμοποιείται για να εγκαθιστά Node.js προγράμματα από το μητρώο npm, οργανώνοντας την εγκατάσταση και τη διαχείριση των τρίτων προγραμμάτων Node.js. Το npm δεν πρέπει να συγχέεται με την δήλωση CommonJS require(). Δεν χρησιμοποιείται για την φόρτωση κώδικα: αντ' αυτού, χρησιμοποιείται για την εγκατάσταση κώδικα και τον χειρισμό εξαρτήσεων του κώδικα από τη γραμμή εντολών. Τα πακέτα που βρίσκονται στο μητρώο του nmp μπορούν να κυμαίνονται από απλές βοηθητικές βιβλιοθήκες, όπως η Underscore.js, μέχρι λειτουργείς έργων όπως το Grant.

- Unified API

Το Node.js μπορεί να συνδυαστεί με ένα πρόγραμμα περιήγησης, μία βάση δεδομένων (όπως η MongoDB ή η CouchDB) και μία JSON για μία ενοποιημένη προγραμματιστική JavaScript στοίβα (stack). Με την προσαρμογή του για το τι ήταν ουσιαστικά τα προγραμματιστικά μοτίβα από την πλευρά του διακομιστή όπως τα MVC, MVP, MVVM κ.τ.λ., το Node.js επιτρέπει την επαναχρησιμοποίηση του ίδιου μοντέλου και της ίδιας υπηρεσιακής διεπαφής μεταξύ της πλευράς του πελάτη και της πλευράς του διακομιστή.

- Βρόγχος Συμβάντων

Το Node.js καταγράφει τον εαυτό του στο λειτουργικό σύστημα έτσι ώστε να ειδοποιείται όταν γίνεται μία σύνδεση, και το λειτουργικό σύστημα θα εκδώσει ένα callback, το οποίο είναι ένα από τα κύρια χαρακτηριστικά του node. Μέσα στον χρόνο εκτέλεσης του Node.js κάθε σύνδεση είναι μία μικρή κατανομή στοίβας, χρησιμοποιώντας έναν βρόγχο συμβάντων για επεκτασιμότητα Σε αντίθεση με άλλους διακομιστές βασισμένους σε συμβάντα, ο βρόγχος συμβάντων του Node.js δεν χρειάζεται να καλείται ρητά. Αντ' αυτού ορίζονται οι ανακλήσεις, και ο διακομιστής εισέρχεται αυτόματα στον βρόγχο στο τέλος ορισμού της ανάκλησης. Το Node.js εξέρχεται του βρόγχου συμβάντων όταν δεν υπάρχουν περαιτέρω ανακλήσεις που πρέπει να εκτελεστούν.

2.2.1.4 Σύνταξη

- Παράδειγμα 1ο : HTTP Server

Ένα χαρακτηριστικό παράδειγμα Node.js για έναν απλό HTTP Server είναι το ακόλουθο:

```
3
4   var http = require('http');
5
6   http.createServer(function (req, res) {
7       res.writeHead(300, {'Content-Type': 'text/plain'});
8       res.end('Hello World!\n');
9   }).listen(3000, '127.0.0.1');
10
11  console.log('Server running at http://127.0.0.1:3000/');
12
```

Εικόνα 10 : Παράδειγμα HTTP Server

Στο παράδειγμα, ο κώδικας δημιουργεί έναν server ο οποίος τρέχει τοπικά στην θύρα «3000». Έτσι ο χρήστης μπορεί να γράψει στην μπάρα του browser του την διεύθυνση URL «http://127.0.0.1:3000/» και να του εμφανίσει μία σελίδα η οποία να γράφει "Hello World!". Για την εκτέλεση του παραπάνω προγράμματος απαιτείται η αποθήκευση του παραπάνω κώδικα σε ένα αρχείο με όνομα my_first_http_server.js στην κονσόλα του συστήματος που χρησιμοποιεί με την εντολή :

```
14
15  $ node my_first_http_server.js
16
```

Εικόνα 11 : Εντολή αποθήκευσης

- Παράδειγμα 2ο : Δημιουργία αρχείου

```
18
19  var fs = require('fs');
20
21  fs.writeFile("D:\myfile.txt", "Hello World!", function(err, date){
22      if ( err ) throw err;
23      console.log(date);
24  })
25
```

Εικόνα 12 : Παράδειγμα δημιουργίας αρχείου

Στο παράδειγμα αυτό χρησιμοποιούμε την έτοιμη βιβλιοθήκη του node για την επεξεργασία αρχείων με την εντολή "require([library_name])" και δημιουργεί ή

αντικαθιστά αν υπάρχει ήδη στο δίσκο D το αρχείο «myfile.txt» με περιεχόμενο «Hello World». Όταν ολοκληρωθεί η λειτουργία αυτής της εγγραφής, ενεργοποιείται το callback το οποίο ορίστηκε, και αυτό με την σειρά του είτε στέλνει μήνυμα προβλήματος σε περίπτωση αποτυχίας, είτε στέλνει τα δεδομένα του αρχείου στην κονσόλα του χρήστη.

3. Frameworks

3.1 Software Frameworks

Στον υπολογιστικό προγραμματισμό, ένα software framework είναι μία αφηρημένη έννοια κατά την οποία ένα λογισμικό που παρέχει γενική λειτουργία μπορεί να αλλαχθεί επιλεκτικά από έναν επιπλέον χρήστη ή ένα γραπτό κώδικα, παρέχοντας έτσι μία εφαρμογή ή ένα ειδικό λογισμικό. Ένα πλαίσιο λογισμικού είναι ένα παγκόσμιο, επαναχρησιμοποιούμενο λογισμικό περιβάλλον που παρέχει συγκεκριμένη λειτουργικότητα, ως μέρος μιας μεγαλύτερης πλατφόρμας, για τη διευκόλυνση ανάπτυξης λογισμικών εφαρμογών, προϊόντων και λύσεων. Τα software frameworks μπορεί να περιλαμβάνουν υποστηρικτικά προγράμματα, μεταγλωττιστές, βιβλιοθήκες κώδικα, σετ εργαλείων και διεπαφή προγραμματισμού εφαρμογών (APIs) τα οποία συγκεντρώνουν όλα τα διαφορετικά συστατικά προκειμένου να καταστεί δυνατή η ανάπτυξη ενός project ή μιας λύσης.

Τα frameworks περιέχουν βασικά διακριτικά γνωρίσματα που τα διαχωρίζουν από τις κανονικές βιβλιοθήκες:

- αντιστροφή του ελέγχου: Σε ένα framework, σε αντίθεση με τις βιβλιοθήκες ή τις κανονικές εφαρμογές χρηστών, η συνολική ροή ελέγχου του προγράμματος δεν υπαγορεύεται από τον καλούντα, αλλά από το πλαίσιο.
- προεπιλεγμένη συμπεριφορά: Ένα framework έχει μία προεπιλεγμένη συμπεριφορά. Αυτή η προεπιλεγμένη συμπεριφορά πρέπει να είναι κάποια χρήσιμη συμπεριφορά και όχι μία σειρά από no-ops.
- επεκτασιμότητα: Ένα framework μπορεί να επεκταθεί συνήθως από το χρήστη μέσω επιλεκτικής παράκαμψης ή εξειδικευμένου κώδικα από το χρήστη, για την παροχή συγκεκριμένης λειτουργικότητας.
- μη-τροποποιήσιμος κώδικας πλαισίου: Ο framework κώδικας ,σε γενικές γραμμές, δεν πρέπει να τροποποιείται, ενώ γίνονται αποδεκτές εφαρμοσμένες επεκτάσεις του χρήστη. Με άλλα λόγια, οι χρήστες μπορούν να επεκτείνουν το framework, αλλά δεν θα πρέπει να τροποποιούν τον κώδικά του.

3.1.1 Φιλοσοφία

Οι σχεδιαστές των software frameworks έχουν στόχο να διευκολύνουν την ανάπτυξη λογισμικού, επιτρέποντας στους σχεδιαστές και τους προγραμματιστές να αφιερώσουν το χρόνο τους στην ικανοποίηση των αναγκών του λογισμικού παρά να ασχοληθούν με τις πιο στάνταρ, χαμηλού επιπέδου λεπτομέρειες της παροχής ενός λειτουργικού συστήματος, μειώνοντας έτσι το συνολικό χρόνο ανάπτυξης. Για παράδειγμα, μία ομάδα χρησιμοποιώντας ένα δικτυακής εφαρμογής framework για την ανάπτυξη μίας τραπεζικής ιστοσελίδας, μπορεί να επικεντρωθεί στη δημιουργία ενός

ειδικού τραπεζικού κώδικα παρά στους μηχανισμούς χειρισμού των αιτημάτων και της κρατικής διαχείρισης.

Τα frameworks συχνά προσθέτουν στο μέγεθος των προγραμμάτων ένα φαινόμενο που ονομάζεται "φουσκωμένος κώδικας" ("code bloat"). Λόγω των αναγκών των εφαρμογών που καθοδηγούνται από πελατειακές απαιτήσεις, τόσο τα ανταγωνιστικά όσο και τα συμπληρωματικά frameworks καταλήγουν μερικές φορές σε ένα προϊόν. Επιπλέον, λόγω της πολυπλοκότητάς των APIs τους, η προβλεπόμενη μείωση στο συνολικό χρόνο ανάπτυξης μπορεί να μην επιτευχθεί, εξαιτίας της ανάγκης να δαπανηθεί επιπλέον χρόνος για την εκμάθηση της χρήσης του πλαισίου: αυτή η κριτική σαφώς ισχύει όταν ένα ειδικό ή καινούριο framework αντιμετωπίζεται για πρώτη φορά από το αναπτυξιακό προσωπικό. Εάν ένα τέτοιο framework δεν χρησιμοποιηθεί σε επόμενες ανατεθειμένες εργασίες, ο χρόνος που επενδύθηκε για την εκμάθηση του framework μπορεί να κοστίσει περισσότερο από τον επιδιωκόμενο γραπτό κώδικα, γνωστό στο προσωπικό του project· πολλοί προγραμματιστές κρατάνε αντίγραφα χρήσιμων στερεοτύπων για συνηθισμένες ανάγκες.

Όπως φαίνεται σε βάθος χρόνου, τα πιο αποτελεσματικά frameworks έχουν αποδειχτεί ότι είναι εκείνα που εξελίσσονται από την εκ νέου συντέλεση του κοινού κώδικα της επιχείρησης, αντί τη χρήση ενός γενικού, συγκεκριμένο σε μέγεθος framework που ταιριάζει με όλα και αναπτύχθηκε από τρίτους για γενικούς σκοπούς. Ένα παράδειγμα αυτού θα ήταν το πώς η διεπαφή χρήστη σε ένα τέτοιο πακέτο εφαρμογών, όπως είναι το office suite, αναπτύσσεται και έχει συνηθισμένη εμφάνιση, αίσθηση, χαρακτηριστικά και μεθόδους διαμοιρασμού δεδομένων, όπως κάποτε οι διάσπαρτες συνοδευτικές εφαρμογές αναπτύσσονται ενωμένες σε μία σύνθεση που είναι μικρότερη· η καινούρια-εξελιγμένη σύνθεση μπορεί να είναι ένα προϊόν που μοιράζεται αναπόσπαστες χρηστικές βιβλιοθήκες και διεπαφές χρήστη.

Αυτή είναι και η αιτία η οποία φέρνει στην επιφάνεια ένα σημαντικό θέμα με τα frameworks. Η δημιουργία ενός framework που είναι κομψό, σε σχέση με ένα που απλά λύνει ένα πρόβλημα, εξακολουθεί να θεωρείται τέχνη και όχι επιστήμη. Η «λογισμική κομψότητα» υπονοεί σαφήνεια, περιεκτικότητα, και λίγα άχρηστα υλικά (πρόσθετη ή ξένη λειτουργικότητα, η περισσότερη από την οποία καθορίζεται από τον χρήστη). Για εκείνα τα frameworks που δημιουργούν κώδικα, για παράδειγμα, η "κομψότητα" θα υπονοούσε τη δημιουργία κώδικα ο οποίος είναι καθαρός και κατανοητός σε έναν αρκετά πεπειραμένο προγραμματιστή (και ο οποίος είναι, συνεπώς, εύκολα τροποποιήσιμος), σε σχέση με εκείνον που απλά δημιουργεί σωστό κώδικα. Το θέμα της κομψότητας είναι ο λόγος που σχετικά λίγα software frameworks έχουν αντέξει στη δοκιμασία του χρόνου: τα καλύτερα frameworks έχουν καταφέρει να εξελιχθούν ομαλά καθώς η υποβόσκουσα τεχνολογία, πάνω στην οποία αναπτύχθηκαν, προχώρησε. Ακόμα και εκεί, έχοντας εξελιχθεί, πολλά τέτοια πακέτα θα διατηρήσουν δυνατότητες κληρονομιάς φουσκώνοντας το τελικό λογισμικό, όπως άλλες αντικατεστημένες μέθοδοι έχουν διατηρηθεί παράλληλα με τις καινούριες μεθόδους.

3.1.2 Παραδείγματα

Τα software frameworks συνήθως περιέχουν σημαντικό οικιακό και χρηστικό κώδικα προκειμένου να βοηθήσουν την εκκίνηση εφαρμογών χρηστών, αλλά γενικά επικεντρώνονται σε συγκεκριμένα προβλήματα τομέων όπως:

- Καλλιτεχνικό σχέδιο, μουσική σύνθεση και μηχανική CAD

- Εφαρμογές χρηματοοικονομικών μοντέλων
- Εφαρμογές συστημικών μοντέλων σχετικά με τη Γη
- Συστήματα υποστήριξης αποφάσεων
- Πολυμεσικά framework – Αναπαραγωγή πολυμέσων και συγγραφή
- Ajax framework / JavaScript framework
- Web frameworks
- Ενδιάμεσο λογισμικό (Middleware)
- Cactus framework – Υψηλής απόδοσης επιστημονική πληροφορική
- Application framework – Γενικές εφαρμογές γραφικής διεπαφής χρήστη (GUI-Graphical User Interface)
- Enterprise Architecture framework
- Framework Ανάπτυξης Εφαρμογών Oracle

3.1.3 Αρχιτεκτονική

Σύμφωνα με το βιβλίο Pree, τα software frameworks αποτελούνται από παγωμένα σημεία (frozen spots) και καυτά σημεία (hot spots). Τα παγωμένα σημεία καθορίζουν τη συνολική αρχιτεκτονική ενός λογισμικού συστήματος, δηλαδή τα βασικά συστατικά του και τις σχέσεις μεταξύ τους. Αυτά παραμένουν αμετάβλητα (παγωμένα) σε κάθε στιγμιότυπο του application framework. Τα καυτά σημεία αντιπροσωπεύουν εκείνα τα μέρη όπου οι προγραμματιστές χρησιμοποιώντας το πλαίσιο προσθέτουν το δικό τους κώδικα για να προσθέσουν τη λειτουργικότητα που είναι συγκεκριμένη για το δικό τους project.

Σε ένα αντικειμενοστραφή περιβάλλον, ένα πλαίσιο αποτελείται από αφηρημένες και συγκεκριμένες κλάσεις. Η συγκεκριμενοποίηση ενός τέτοιου framework αποτελείται από τη σύνθεση και την υποβάθμιση των υπαρχουσών κλάσεων.

Κατά την ανάπτυξη ενός συγκεκριμένου λογισμικού συστήματος με ένα πλαίσιο λογισμικού, οι κατασκευαστές χρησιμοποιούν τα καυτά σημεία σύμφωνα με τις ιδιαίτερες ανάγκες και απαιτήσεις του συστήματος. Τα software frameworks βασίζονται στην Αρχή του Hollywood: “Μη μας καλέσετε, θα σας καλέσουμε εμείς.” Αυτό σημαίνει ότι οι, καθορισμένες από το χρήστη, κλάσεις (για παράδειγμα, νέες δευτερεύουσες κλάσεις), λαμβάνουν μηνύματα από τις προκαθορισμένες framework κλάσεις. Οι κατασκευαστές χειρίζονται συνήθως αυτό μέσω της εφαρμογής αφηρημένων υπερκλάσεων.

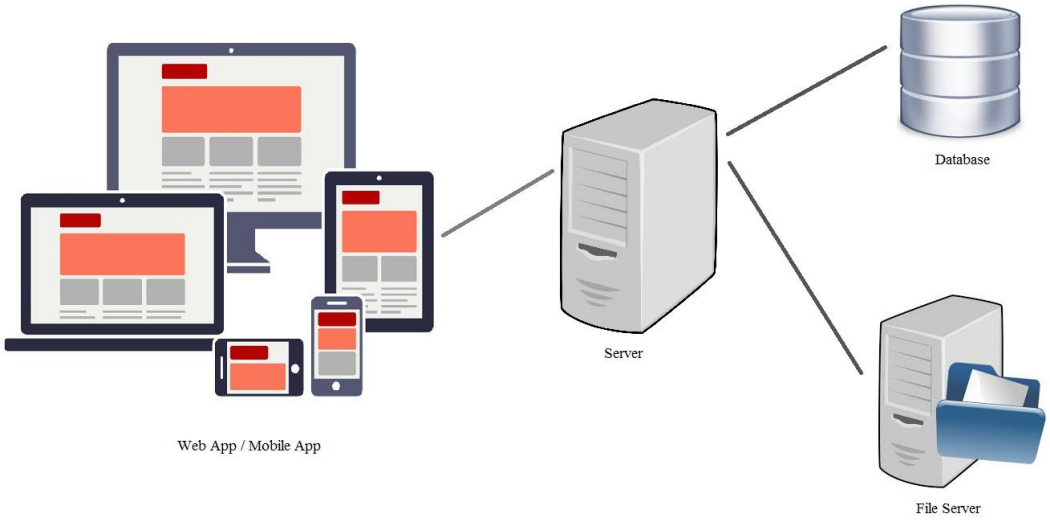
3.2 Javascript Frameworks

Ένα Javascript framework είναι ένα διαδικτυακής εφαρμογής framework γραμμένο σε γλώσσα προγραμματισμού Javascript και διαφέρει από μία Javascript βιβλιοθήκη. Μία βιβλιοθήκη προσφέρει πολλές χρήσιμες προκαθορισμένες λειτουργίες που μπορείς να καλέσεις για να βελτιώσεις και να επεκτείνεις την εφαρμογή σου. Ένα framework που περιγράφει την δομή της αίτησης και σου δίνει ένα τρόπο για να οργανώσεις τον κώδικά σου για να κάνεις την εφαρμογή σου πιο ευέλικτη και επεκτάσιμη. Δεν μπορείς να καλέσεις ένα framework , αλλά είναι το πλαίσιο που θα καλέσει και θα χρησιμοποιήσει τον κωδικό σου με κάποιο συγκεκριμένο τρόπο.

Τα Javascript Frameworks είναι συνήθως βασισμένα στο αρχιτεκτονικό πρότυπο MVC αρχιτεκτονικής (Model-view-controller) , επειδή έχουν επίσης σχεδιαστεί για να διαχωρίζουν τις διάφορες πτυχές μιας διαδικτυακής εφαρμογή για τη βελτίωση της ποιότητας του κωδικό σου και για να κάνουν την ανάπτυξη ευκολότερη.

Μερικά διάσημα παραδείγματα ενός Javascript Frameworks με MVC αρχιτεκτονική είναι το AngularJS και Ember.js.

3.3 Front-end και Back-end



Εικόνα 13 : Front-end / Back-end

Ανοίγεις μία καινούρια καρτέλα στο πρόγραμμα περιήγησης, πληκτρολογείς μία διεύθυνση URL και πατάς enter. Η ιστοσελίδα φορτώνει αμέσως. Βλέπεις απευθείας μία καλά κατασκευασμένη σελίδα με εξαιρετικά καθαρή δόμηση και εντυπωσιακά γραφικά. Τα άτομα που είναι υπεύθυνα για κάθε μέρος αυτής της εμπειρίας; Οι προγραμματιστές του διαδικτύου.

Από το Νοέμβριο του 2014, το Διαδίκτυο περιέχει περισσότερες από 680 εκατομμύρια σελίδες και συνεχώς αυξάνονται. Μιλάμε για σοβαρή εργασιακή ασφάλεια για τους προγραμματιστές του διαδικτύου, τους ανθρώπους που είναι υπεύθυνοι για την κωδικοποίηση, την κατασκευή, την ανάλυση και τη συντήρηση όλων αυτών των ιστοσελίδων.

Όμως ας δούμε τα πράγματα απ’ την προγραμματιστική πλευρά. Ο προγραμματισμός του διαδικτύου τείνει να διαχωρίζεται σε τρεις κύριες συγκεντρώσεις: το front-end, το back-end και το full-stack όμως οι δύο πρώτες θα μας απασχολήσουν. Έτσι, στον προγραμματισμό λογισμικού , οι όροι ”front-end” και “back-end” διαφέρουν όσον αφορά ότι το front-end αναφέρεται στο στρώμα παρουσίασης ενώ το back end αναφέρεται στο στρώμα πρόσβασης δεδομένων. Το front end από την πλευρά του είναι μία διεπαφή μεταξύ του χρήστη και του back-end. Το

front-end και το back-end μπορούν να κατανεμηθούν μεταξύ ενός ή περισσότερων συστημάτων. Ας τα δούμε πιο αναλυτικά παρακάτω.

3.3.1 Front-end

Με τον όρο “front-end” μιας ιστοσελίδας εννοούμε το κομμάτι με το οποίο αλληλοεπιδρούν οι χρήστες. Οτιδήποτε βλέπεις όταν σερφάρεις στο Διαδίκτυο, από τις γραμματοσειρές και τα χρώματα μέχρι τα αναπτυσσόμενα μενού και τις μπάρες, είναι μία σύνθεση από HTML, CSS και JavaScript που ελέγχονται από το πρόγραμμα περιήγησης του υπολογιστή σου. Το front end αποτελείται συνήθως από δύο μέρη : το σχεδιασμό και την ανάπτυξη front end.

Οι front-end προγραμματιστές είναι υπεύθυνοι για τον κώδικα της ιστοσελίδας με τον οποίο έρχονται σε επαφή οι χρήστες και με την αρχιτεκτονική των καθηλωτικών εμπειριών του χρήστη. Για την εκτέλεση των εν λόγω στόχων, οι front-end προγραμματιστές πρέπει να είναι έμπειροι σε τρεις βασικές γλώσσες: στον προγραμματισμό HTML, CSS και JavaScript. Εκτός από την άνεση σε αυτές τις γλώσσες, οι front-end προγραμματιστές χρειάζεται να είναι εξοικειωμένοι με πλαίσια όπως είναι το Bootstrap, το Backbone και το AngularJS, τα οποία εξασφαλίζουν εξαιρετικά εμφανίσιμο περιεχόμενο, ασχέτως της συσκευής, και βιβλιοθήκες όπως η jQuery και η LESS, οι οποίες συσκευάζουν τον κώδικα σε μία πιο χρήσιμη και γρηγορότερη φόρμα. Πολλές αγγελίες για front-end προγραμματιστές ζητάνε επίσης εμπειρία με Ajax, μία ευρέως χρησιμοποιημένη τεχνική για χρήση της JavaScript που επιτρέπει στις σελίδες να φορτώνονται δυναμικά μέσω της λήψης δεδομένων του διακομιστή στο παρασκήνιο.

Με άλλα λόγια , ό,τι βλέπεις αυτή τη στιγμή σε αυτή την ιστοσελίδα κατέστη δυνατό από έναν front-end προγραμματιστή. Ένας σχεδιαστής φιλοτέχνησε το λογότυπο και τα γραφικά, ένας φωτογράφος τράβηξε τις φωτογραφίες και ένας κειμενογράφος έγραψε το κείμενο. Αλλά ένας front-end προγραμματιστής συναρμολόγησε όλα αυτά τα κομμάτια, τα μετέφρασε σε διαδικτυακή ομιλία και έχτισε την εμπειρία που έχεις με κάθε σελίδα.

3.3.2 Back-end

Οπότε, τι είναι αυτό που κάνει ένα front-end σε μία ιστοσελίδα εφικτό; Πού αποθηκεύονται όλα αυτά δεδομένα; Το back-end μία ιστοσελίδας αποτελείται από ένα διακομιστή(server), μία εφαρμογή(application) και μία βάση δεδομένων(database). Ένας back-end προγραμματιστής χτίζει και διατηρεί την τεχνολογία που τροφοδοτεί αυτά τα εξαρτήματα τα οποία, ενωμένα, επιτρέπουν στην πλευρά της ιστοσελίδας που φαίνεται στο χρήστη να υπάρχει ακόμα και στην πρώτη θέση.

Προκειμένου να μπορέσουν να επικοινωνήσουν ο διακομιστής, η εφαρμογή και η βάση δεδομένων μεταξύ τους, οι back-end προγραμματιστές χρησιμοποιούν γλώσσες από την πλευρά του διακομιστή, όπως είναι η PHP, η Python, η Java και η .Net, για να οικοδομήσουν μία εφαρμογή, και εργαλεία, όπως το MySQL, το Oracle και ο SQL Server, για να βρουν, να αποθηκεύσουν και να αλλάξουν δεδομένα και να τα «σερβίρουν» πίσω στο χρήστη σε κώδικα front-end. Οι back-end προγραμματιστές χρειάζονται επίσης εμπειρία με πλαίσια PHP, όπως είναι το Zend, το Symfony και το CakePHP· εμπειρία με λογισμικό ελέγχου έκδοσης όπως είναι το SVN, το CVS ή το Git· και εμπειρία με Linux ως σύστημα ανάπτυξης και εγκατάστασης. Έτσι λοιπόν, οι

back-end προγραμματιστές χρησιμοποιούν αυτά τα εργαλεία για να δημιουργήσουν ή να συνεισφέρουν σε διαδικτυακές εφαρμογές με καθαρό και καλά τεκμηριωμένο κώδικα.

Όταν πλοηγήθηκες σε μία ιστοσελίδα, οι διακομιστές έστειλαν πληροφορίες στον υπολογιστή ή την κινητή σου συσκευή, οι οποίες μετατράπηκαν στην σελίδα που βλέπεις εκείνη τη στιγμή. Αυτή η διαδικασία είναι το αποτέλεσμα της δουλειάς ενός back-end προγραμματιστή. Επιπλέον, η αποθήκευση των προσωπικών σου πληροφοριών – και το γεγονός ότι κάθε φορά που επιστρέφεις στην ιστοσελίδα και συνδέεσαι, τα δεδομένα σου επικαλούνται – οφείλεται σε έναν back-end προγραμματιστή.

3.4 JavaScript Βιβλιοθήκες

Μία βιβλιοθήκη Javascript είναι μία βιβλιοθήκη προ-γραμμένου κώδικα JavaScript που επιτρέπει την ευκολότερη ανάπτυξη εφαρμογών που βασίζονται σε Javascript. Κάποια παραδείγματα τέτοιων βιβλιοθηκών είναι η GUI , DOM, Pure Javascript/ Ajax, Web-application,Template system και άλλες.

3.4.1 DOM

3.4.1.1 jQuery

Εικόνα 14 : Λογότυπο jQuery

Η jQuery είναι μία πολύ-πλατφορμική βιβλιοθήκη JavaScript σχεδιασμένη να απλοποιεί την υλοποίηση σεναρίου HTML από την πλευρά του πελάτη. Η jQuery είναι η πιο δημοφιλής βιβλιοθήκη JavaScript σε χρήση σήμερα, με εγκατάσταση στο 65% των 10 εκατομμυρίων κορυφαίων ιστοσελίδων στον Παγκόσμιο Ιστό με την περισσότερη κίνηση. Η jQuery είναι ένα δωρεάν, ανοιχτού κώδικα λογισμικό με άδεια από την MIT License.

Η σύνταξη της jQuery είναι σχεδιασμένη να κάνει ευκολότερη την πλοήγηση ενός εγγράφου, την επιλογή στοιχείων DOM, τη δημιουργία κινουμένων σχεδίων, τον χειρισμό γεγονότων και τον σχεδιασμό εφαρμογών Ajax. Η jQuery παρέχει επίσης στους προγραμματιστές δυνατότητες για να δημιουργήσουν πρόσθετα εξαρτήματα λογισμικού στην κορυφή της βιβλιοθήκης JavaScript. Αυτό επιτρέπει στους προγραμματιστές να δημιουργήσουν αφαιρέσεις για χαμηλού επιπέδου αλληλεπίδραση και κινούμενα σχέδια, προηγμένα εφέ και υψηλού επιπέδου θεματικά widgets. Η αρθρωτή προσέγγιση της βιβλιοθήκης jQuery επιτρέπει τη δημιουργία ισχυρών δυναμικών ιστοσελίδων και διαδικτυακών εφαρμογών.

Η Microsoft και η Nokia δεσμεύουν την jQuery στις πλατφόρμες τους.

3.4.1.1.1 Επισκόπηση

Η jQuery, στον πυρήνα της, είναι μία βιβλιοθήκη χειρισμού DOM (Document Object Model). Το DOM είναι μία δομική αναπαράσταση δέντρου όλων των στοιχείων μίας σελίδας Διαδικτύου, και η jQuery απλοποιεί τη σύνταξη για εύρεση, επιλογή και χειρισμό αυτών των στοιχείων DOM. Για παράδειγμα, η jQuery μπορεί να χρησιμοποιηθεί για εύρεση ενός στοιχείου μέσα στο αρχείο με μία συγκεκριμένη ιδιότητα (π.χ. όλα τα στοιχεία με μία h1 ετικέτα), για αλλαγή ενός ή περισσοτέρων από τα χαρακτηριστικά του (π.χ. χρώμα, ορατότητα), ή για να το καταστήσει ικανό να ανταποκριθεί σε ένα γεγονός (π.χ. ένα κλικ του ποντικιού).

Η jQuery παρέχει επίσης ένα παράδειγμα για χειρισμό γεγονότων που ξεπερνούν τη βασική επιλογή και χειρισμό DOM. Η ανάθεση γεγονότων και ο ορισμός της λειτουργίας callback γίνονται σε ένα μοναδικό βήμα σε μία μοναδική τοποθεσία στον κώδικα. Η jQuery έχει επίσης ως στόχο να ενσωματώσει άλλες λειτουργίες υψηλής χρήσης της JavaScript (π.χ. το fade in και το fade out κατά το κρύψιμο στοιχείων, κινούμενα σχέδια μέσω της χρήσης ιδιοτήτων CSS).

Τα πλεονεκτήματα της χρήσης jQuery είναι:

- Ενθαρρύνει τον διαχωρισμό μεταξύ της JavaScript και της HTML: Η βιβλιοθήκη jQuery παρέχει απλό συντακτικό για την προσθήκη χειριστών των γεγονότων στο DOM χρησιμοποιώντας την JavaScript, παρά της προσθήκης HTML χαρακτηριστικών των γεγονότων στις JavaScript λειτουργίες. Επομένως, ενθαρρύνονται οι κατασκευαστές να διαχωρίσουν πλήρως τον κώδικα JavaScript από την HTML.
- Συντομία και σαφήνεια: Η jQuery προωθεί τη συντομία και τη σαφήνεια με χαρακτηριστικά όπως είναι οι αλυσιδωτές λειτουργίες και λειτουργία στενογραφίας ονομάτων.
- Εξαλείφει τις ασυμβατότητες μεταξύ των προγραμμάτων περιήγησης**:** Οι μηχανές JavaScript διαφορετικών προγραμμάτων περιήγησης διαφέρουν ελαφρώς, έτσι ώστε ο κώδικας JavaScript που λειτουργεί για ένα πρόγραμμα περιήγησης ίσως να μην λειτουργεί για ένα άλλο. Όπως και άλλα εργαλεία της JavaScript, η jQuery χειρίζεται όλες αυτές τις αντιφάσεις μεταξύ των προγραμμάτων περιήγησης και παρέχει μία συνεπή διεπαφή που λειτουργεί σε διαφορετικά προγράμματα περιήγησης.
- Επεκτασιμότητα**:** Νέα γεγονότα, στοιχεία και μέθοδοι μπορούν εύκολα να προστεθούν και μετά να ξαναχρησιμοποιηθούν ως ένα πρόσθετο εξάρτημα λογισμικού.

3.4.1.1.2 Χαρακτηριστικά

Η jQuery περιλαμβάνει τα εξής χαρακτηριστικά:

- DOM element επιλογές χρησιμοποιώντας την ανοιχτού κώδικα μηχανή επιλογής πολλαπλών φυλλομετρητών Sizzle.
- DOM διάσχιση και τροποποίηση
- χειρισμός DOM βασισμένος σε CSS επιλογείς που χρησιμοποιεί τα id και class σαν κριτήρια για να κατασκευάσει επιλογείς.

- Events
- Εφέ και κινητά στοιχεία
- AJAX
- Επεκτασιμότητα μέσω plug-ins
- Εργαλεία όπως πληροφορίες user-agent, ανίχνευση χαρακτιριστικώ.
- Μεθόδους συμβατότητας που είναι εγγενώς διαθέσιμα σε σύγχρονα προγράμματα περιήγησης.
- Υποστήριξη πολλαπλών φυλλομετρητών.

▪ Υποστήριξη προγραμμάτων περιήγησης

Και οι δύο εκδόσεις, 1.x και 2.x, της jQuery υποστηρίζουν τις «τρέχουσες-1 εκδόσεις» (εννοώντας τις τρέχουσες σταθερές εκδόσεις του προγράμματος περιήγησης και της έκδοσης που προηγήθηκε αυτού) του Firefox, Google Chrome, Safari και Opera. Η έκδοση 1.x υποστηρίζει επίσης το Internet Explorer 6 ή και μεγαλύτερο. Ωστόσο, η έκδοση 2.x της jQuery απέρριψε την υποστήριξη του Internet Explorer 6-8 (που αντιπροσωπεύει το 2% όλων των προγραμμάτων περιήγησης σε χρήση) και υποστηρίζει μόνο το IE 9 και μετέπειτα εκδόσεις.

3.4.1.1.3 Χρήση

▪ Συμπεριλαμβανομένης της βιβλιοθήκης

Η βιβλιοθήκη jQuery είναι ένα μεμονωμένο αρχείο JavaScript που περιέχει όλα τα συνηθισμένα DOM, γεγονότα, εφέ και λειτουργίες Ajax. Μπορεί να περιληφθεί μέσα σε μία ιστοσελίδα μέσω της σύνδεσης σε ένα τοπικό αντίγραφο ή σε ένα από τα πολλά αντίγραφα που είναι διαθέσιμα από τους δημόσιους διακομιστές. Η jQuery έχει περιεκτικότητα σε δίκτυο αναμονής (Content Delivery Network – CDN) που φιλοξενείται από τη MaxCDN (που μετακινήθηκε από τη MediaTemple και, πριν από αυτή, από την Amazon). Η Google και η Microsoft την φιλοξενούν επίσης.

```
2
3   <script src = "jquery.js"></script>
4
```

Εικόνα 15 : Σύνδεση σε τοπικό αντίγραφο jQuery

Είναι επίσης δυνατό η jQuery να περιληφθεί απευθείας από το περιεχόμενο των δικτύων διανομής (CDNs) .

```
7
8   <script src="https://code.jquery.com/jquery-latest.min.js"></script>
9
```

Εικόνα 16 : Σύνδεση από το περιεχόμενο των δικτύων διανομής

- Στυλ χρήσης

Η jQuery έχει δύο στυλ χρήσης:

- Μέσω της **$** λειτουργίας, η οποία είναι μια βασική μέθοδος για το αντικείμενο jQuery. Αυτές οι λειτουργίες, που συχνά ονομάζονται διαταγές, είναι αλυσιδωτές καθώς όλες επιστρέφουν αντικείμενα jQuery.
- Μέσω των **$.**-prefixed λειτουργιών. Αυτές είναι πρακτικές λειτουργίες, οι οποίες δεν ενεργούν άμεσα σύμφωνα με το αντικείμενο jQuery.

Η πρόσβαση και η χειραγώγηση πολλαπλών κόμβων DOM στην jQuery ξεκινάει συνήθως με την κλήση της $ λειτουργίας με μεταβλητή διαλογέα CSS. Αυτή επιστρέφει ένα αντικείμενο jQuery που αναφέρεται σε όλα τα στοιχεία που ταιριάζουν στην ιστοσελίδα HTML. Η $("div.test"), για παράδειγμα, επιστρέφει ένα αντικείμενο jQuery με όλα τα στοιχεία div της κλάσης test.

- Τυπικό σημείο εκκίνησης

Η τυπική χρήση της jQuery είναι να θέσει τον κώδικα αρχικοποίησης και τις λειτουργίες χειρισμού γεγονότων στην .ready(). Αυτή ενεργοποιείται όταν το πρόγραμμα περιήγησης έχει κατασκευάσει το DOM και στέλνει ένα φορτωμένο γεγονός.

```
3
4   <script type="text/javascript">
5       $(document).ready(function(){
6
7       });
8   </script>
9
```

Εικόνα 5 : Χρήση .ready()

Οι λειτουργίες callbacks για τον χειρισμό των γεγονότων συμπεριλαμβάνονται επίσης μέσα στην .ready() ως ανώνυμες λειτουργίες, αλλά καλούνται όταν το γεγονός για την ανάκληση ενεργοποιείται. Για παράδειγμα, ο παρακάτω κώδικας jQuery προσθέτει έναν χειριστή γεγονότων για το κλικ του ποντικιού πάνω σε ένα στοιχείο img εικόνας.

Οι ακόλουθες συντάξεις είναι ισοδύναμες:

- $(document).ready(handler)
- $(handler)

- Λειτουργίες Χρησιμότητας

Οι λειτουργίες με το πρόθεμα $., είναι λειτουργίες χρησιμότητας ή λειτουργίες που επηρεάζουν τις παγκόσμιες ιδιότητες και συμπεριφορά. Η παρακάτω, για παράδειγμα, είναι μια λειτουργία που χρησιμοποιείται για την επανάληψη πάνω σε πίνακες, που ονομάζεται each στην jQuery.

```
3
4   <script type="text/javascript">
5       $(document).ready(function(){
6           $('img').click(function() {
7
8           });
9       });
10  </script>
11
```

Εικόνα 17 : Χειρισμός γεγονότων για το κλικ ποντικιού σε μία εικόνα

```
6
7           $.each([4,5,6],function(){
8               console.log(this + 1);
9           });
10
```

Εικόνα 18 : Λειτουργία επανάληψης σε πίνακες

Αυτό γράφει '5', '6', '7' στην κονσόλα.

- Ασύγχρονη

Σημειώστε ότι το παραπάνω παράδειγμα χρησιμοποιεί την αναβαλλόμενη φύση τής $.ajax() για να χειριστεί την ασύγχρονη φύση της: η .done() και .fail() δημιουργούν ανακλήσεις (callbacks) που λειτουργούν μόνο όταν η ασύγχρονη διαδικασία έχει ολοκληρωθεί.

- Πρόσθετα εξαρτήματα λογισμικού της jQuery (jQuery plug-ins)

Η αρχιτεκτονική της jQuery επιτρέπει στους προγραμματιστές να δημιουργήσουν κώδικα πρόσθετου εξαρτήματος λογισμικού (plug-in code) για την επέκταση της λειτουργίας του. Υπάρχουν χιλιάδες plug-ins της jQuery διαθέσιμα στον Παγκόσμιο Ιστό που καλύπτουν ένα μεγάλο μέρος λειτουργιών, όπως είναι οι Ajax helpers, οι υπηρεσίες Διαδικτύου (Web services), τα datagrids, οι δυναμικές λίστες, τα XML και XSLT εργαλεία, το drag and drop, τα events, ο χειρισμός των cookies και τα παράθυρα μεταφορών.

Μία σημαντική πηγή των plug-ins της jQuery είναι ο υποτομέας plug-ins της ιστοσελίδας jQuery Project. Τα plug-ins σε αυτόν τον υποτομέα, ωστόσο, διαγράφηκαν καταλάθος τον Δεκέμβριο του 2011 σε μία προσπάθεια απαλλαγής από την σελίδα των spam. Η νέα σελίδα θα περιλαμβάνει ένα φιλοξενούμενο αποθετήριο GitHub, το οποίο θα απαιτεί από τους προγραμματιστές να υποβάλλουν εκ νέου τα plug-ins και να συμμορφώνονται με νέες απαιτήσεις υποβολής. Υπάρχουν εναλλακτικές μηχανές αναζήτησης plug-in, όπως είναι η jquer.in που αναλαμβάνει πιο εξειδικευμένες

προσεγγίσεις, όπως είναι η δημιουργία λιστών μόνο με plug-ins που πληρούν ορισμένα κριτήρια (π.χ. εκείνα που έχουν ένα αποθετήριο δημόσιου κώδικα).

3.4.1.1.4 Ιστορία

Η jQuery αρχικά κυκλοφόρησε τον Ιανουάριο του 2006 στο BarCamp της Νέας Υόρκης από τον John Resig και επηρεάστηκε από την παλαιότερη βιβλιοθήκη cssQuery του Dean Edwards. Αυτή τη στιγμή διατηρείται από μια ομάδα προγραμματιστών με επικεφαλή τον Timmy Willison (με τη διαλογική μηχανή της jQuery, τη Sizzle, να καθοδηγείται από τον Richard Gibson).

Από το 2015, η jQuery παραμένει η πιο διαδεδομένη βιβλιοθήκη JavaScript στο Διαδίκτυο. Σύμφωνα με την αναλυτική υπηρεσία της βιβλιοθήκης JavaScript, η jQuery χρησιμοποιείται σε πάνω από το 63% των κορυφαίων και πιο διαδεδομένων εκατομμυρίων σελίδων με την περισσότερη κίνηση. Αξιοσημείωτες σελίδες την χρησιμοποιούν, συμπεριλαμβανομένων του Twitter, του LinkedIn, του Pinterest και του eBay.

- Πλαίσιο ελέγχου

Το QUnit είναι ένα πλαίσιο ελέγχου αυτοματοποίησης που χρησιμοποιείται για τον έλεγχο του jQuery project. Η ομάδα jQuery το ανέπτυξε ως μία μονάδα βιβλιοθήκης εσωτερικού ελέγχου. Η ομάδα jQuery το χρησιμοποιεί για να ελέγξει τους κώδικες και τα plug-ins της, αλλά μπορεί να ελέγξει οποιονδήποτε γενικό κώδικα JavaScript, συμπεριλαμβανομένου και του κώδικα της JavaScript από τη πλευρά του διακομιστή.

Από το 2011, η Ομάδα Ελέγχου της jQuery χρησιμοποιεί το QUnit με το TestSwarm για να ελέγξει κάθε μία βάση κώδικα της jQuery που κυκλοφορεί.

3.4.1.2 Mootools

Εικόνα 19 : Λογότυπο Mootools

Το MooTools (My Object-Oriented Tools) είναι ένα ελαφρύ, αντικειμενοστραφές πλαίσιο JavaScript. Έχει διατεθεί υπό το ελεύθερο, ανοιχτού κώδικα MIT License. Χρησιμοποιείται σε περισσότερες από το 4% όλων των ιστοσελίδων και είναι μία από τις πιο δημοφιλείς βιβλιοθήκες JavaScript.

2.3.4.2.1 Ιστορία

Πρώτος έγραψε το πλαίσιο ο Valerio Proietti και το κυκλοφόρησε τον Σεπτέμβριο του 2006, λαμβάνοντας ως έμπνευση το Prototype και το base2 του Dean Edward. Το MooTools προήλθε από το Moo.fx, ένα δημοφιλές πρόσθετο εξάρτημα λογισμικού(plug-in) του Proietti που παράχθηκε για το Prototype τον Οκτώβριο του 2005, το οποίο διατηρείται και χρησιμοποιείται ακόμη.

Αν και το επεκταμένο Prototype λειτούργησε ως πρότυπο για πολλά από τα μητρικά αντικείμενα String (Συμβολοσειρά), Array (Πίνακας) και Function (Λειτουργία) της JavaScript με πρόσθετες μεθόδους, ο Proietti θέλησε ένα πλαίσιο που (εκείνη τη στιγμή) θα διεύρυνε περαιτέρω το μητρικό Αντικείμενο Element (Στοιχείο) καθώς και θα προσέφερε μεγαλύτερο έλεγχο του DOM.

2.3.4.2.2 Εξαρτήματα

Το MooTools περιλαμβάνει αρκετά εξαρτήματα, αλλά δεν χρειάζονται όλα να φορτωθούν για κάθε εφαρμογή. Μερικές από τις κατηγορίες των εξαρτημάτων είναι:

- Πυρήνας (Core): Μία συλλογή από λειτουργίες χρησιμότητας που όλα τα άλλα εξαρτήματα απαιτούν.
- Περισσότερο (More): Μία επίσημη συλλογή add-ons που επεκτείνουν τον Πυρήνα (Core) και παρέχουν βελτιωμένη λειτουργικότητα.
- Κλάση (Class): Η βασική βιβλιοθήκη για το στιγμιότυπο του αντικειμένου Class.
- Μητρικά (Natives): Μία συλλογή από βελτιώσεις του Αντικειμένου Μητρικό (Native) της JavaScript. Τα Μητρικά (Natives) προσθέτουν λειτουργικότητα, συμβατότητα και νέες μεθόδους που απλοποιούν την κωδικοποίηση.
- Στοιχείο (Element): Περιλαμβάνει έναν μεγάλο αριθμό από βελτιώσεις και συμβατότητα τυποποίησης στο Αντικείμενο Στοιχείο HTML.
- Fx: Ένα API προηγμένων εφέ για την εμψύχωση των στοιχείων της σελίδας.
- Αίτημα (Request): Περιλαμβάνει διεπαφή XHR, Cookie, JSON και ειδικά εργαλεία ανάκτησης HTML για να εκμεταλλευτούν οι προγραμματιστές.
- Παράθυρο (Window): Παρέχει μία διεπαφή μεταξύ των προγραμμάτων περιήγησης για συγκεκριμένες πληροφορίες του πελάτη, όπως είναι οι διαστάσεις του παραθύρου

- Συμβατότητα του Προγράμματος Περιήγησης

Το MooTools είναι συμβατό και έχει δοκιμαστεί με:

- Safari 3+
- Internet Explorer 6+
- Mozilla Firefox 2+
- Opera 9+
- Chrome 4+

2.3.4.2.3 Πλεονεκτήματα

Το MooTools παρέχει στον χρήστη έναν αριθμό πλεονεκτημάτων έναντι της μητρικής JavaScript. Αυτά περιέχουν:

- Ένα επεκτάσιμο και αρθρωτό πλαίσιο που επιτρέπει στους προγραμματιστές να επιλέξουν τον δικό τους προσαρμοσμένο συνδυασμό συστατικών.
- Το MooTools ακολουθεί αντικειμενοστραφείς πρακτικές και την αρχή DRY.
- Ένα εξάρτημα προηγμένων εφέ, με βελτιστοποιημένες μεταβάσεις όπως οι εύκολες εξισώσεις που χρησιμοποιούνται από πολλούς προγραμματιστές του Flash.
- Βελτιώσεις για το DOM, που επιτρέπει στους προγραμματιστές να προσθέτουν, να τροποποιούν, να επιλέγουν και να διαγράφουν εύκολα στοιχεία DOM. Η αποθήκευση και η ανάκτηση πληροφοριών με την αποθήκευση Στοιχείου (Element) υποστηρίζεται επίσης.

Το πλαίσιο περιλαμβάνει ενσωματωμένες λειτουργίες για χειρισμό CSS, στοιχείων DOM, μητρικά αντικείμενα JavaScript, αιτήματα Ajax, DOM επιδράσεις και άλλα πολλά. Το MooTools παρέχει επίσης μία λεπτομερή, συνεπή διεπαφή προγραμματισμού εφαρμογών (API), καθώς επίσης και μια προσαρμοσμένη ενότητα λήψεων, που επιτρέπει στους προγραμματιστές να κατεβάζουν μόνο ενότητες και εξαρτήσεις που χρειάζονται για μια συγκεκριμένη εφαρμογή.

- Έμφαση στην αρθρώτητα και την επαναχρησιμοποίηση

Κάθε πλαίσιο JavaScript έχει την δικιά του φιλοσοφία, και το MooTools ενδιαφέρεται για την πλήρη αξιοποίηση της ευελιξίας και τη δύναμη της JavaScript με τρόπο που δίνει έμφαση στην μεγαλύτερη αρθρώτητα και στην επαναχρησιμοποίηση του κώδικα. Το MooTools επιτυγχάνει αυτούς τους στόχους με τρόπο διαισθητικό για έναν προγραμματιστή που προέρχεται από μία κληρονομική γλώσσα βασισμένη σε κλάσεις, όπως είναι η Java, με το αντικείμενο Class του MooTools.

Η Class είναι ένα αντικείμενο από ζευγάρια κλειδιών/τιμών που μπορούν να περιέχουν είτε ιδιότητες είτε μεθόδους (λειτουργίες). Η Class αβίαστα ανακατεύεται και επεκτείνεται με άλλα στιγμιότυπα Class επιτρέποντας τη μεγαλύτερη εστίαση στο MooTools: η επαναχρησιμοποίηση του κώδικα που επιτυγχάνεται μέσω της μεγιστοποίησης της δύναμης της πρωτότυπης κληρονομικότητας της JavaScript, αλλά σε μία σύνταξη αντικειμένου Class πιο εξοικειωμένη με κλασικά κληρονομικά μοντέλα.

- Αντικειμενοστραφής Προγραμματισμός

Το MooTools περιέχει ένα ισχυρό σύστημα δημιουργίας Class και κληρονομικότητας που μοιάζει με τις περισσότερες κλασικές αντικειμενοστραφείς γλώσσες προγραμματισμού.

3.4.2 GUI

3.4.2.1 AngularJS

Εικόνα 20 : Λογότυπο AngularJS

Η AngularJS (που συνήθως αναφέρεται ως «Angular» ή «Angular.js») είναι ένα ανοιχτού κώδικα διαδικτυακού πλαίσιο εφαρμογής που διατηρείται κυρίως από την Google και από μία κοινότητα μεμονωμένων προγραμματιστών και εταιρειών για να διευθετήσουν πολλές από τις προκλήσεις που αντιμετωπίζουν κατά την ανάπτυξη εφαρμογών μονής σελίδας(single-page). Οι JavaScript συνιστώσες συμπληρώνουν τη PhoneGap, το πλαίσιο που χρησιμοποιήθηκε για την ανάπτυξη cross-platform εφαρμογών για κινητά. Έχει ως στόχο να απλοποιήσει τόσο την ανάπτυξη και τον έλεγχο τέτοιων εφαρμογών για αρχιτεκτονικές από την πλευρά του client, model-view-controller /MVC και model-view-view model/MVVM, μαζί με συνιστώσες που χρησιμοποιούνται συνήθως σε πλούσιες εφαρμογές του Διαδικτύου.

Η AngularJS πλαίσιο λειτουργεί αφού πρώτα διαβαστεί η σελίδα HTML, στην οποία έχουν ενσωματωθεί επιπλέον προσαρμοσμένα χαρακτηριστικά ετικετών. Η Angular ερμηνεύει αυτά τα χαρακτηριστικά ως οδηγίες για να ενώσει τα κομμάτια εισόδου και εξόδου της σελίδας σε ένα μοντέλο που αντιπροσωπεύεται από πρότυπες μεταβλητές JavaScript. Οι τιμές για αυτές τις μεταβλητές JavaScript μπορούν να ρυθμιστούν χειροκίνητα μέσα στον κώδικα ή να ανακτηθούν από στατικούς ή δυναμικούς JSON πόρους.

Σύμφωνα με την αναλυτική υπηρεσία Libscore της JavaScript, η AngularJS χρησιμοποιείται στις ιστοσελίδες του NBC, Intel, Spirit, ABC News και περίπου 8.400 άλλους διαδικτυακούς τόπους από τους 1 εκατομμύριο που ελέγχθηκαν τον Ιούλιο του 2015.

Η AngularJS είναι το front-end κομμάτι της στοίβας MEAN, μαζί με τον χρόνο εκτέλεσης του Node.js, το back-end πλαίσιο Express.js και τη βάση δεδομένων MongoDB.

3.4.2.1.1 Φιλοσοφία

Η AngularJS βασίζεται στην πεποίθηση ότι ο δηλωτικός προγραμματισμός θα έπρεπε να χρησιμοποιείται για να δημιουργεί διεπαφές χρηστών και να συνδέει εξαρτήματα λογισμικού, ενώ ο προγραμματισμός επιτακτικής ανάγκης ταιριάζει καλύτερα στον καθορισμό της επιχειρηματικής λογικής μιας εφαρμογής. Το πλαίσιο προσαρμόζει και επεκτείνει παραδοσιακές HTML για να παρουσιάσει δυναμικό περιεχόμενο μέσω μίας αμφίδρομης σύνδεσης δεδομένων που επιτρέπει τον αυτόματο συγχρονισμό των μοντέλων (models) και των οπτικών πεδίων (views). Ως αποτέλεσμα,

η AngularJS δεν δίνει έμφαση στη ρητή χειραγώγηση DOM με στόχο την βελτίωση της ελεγξιμότητας και της απόδοσης.

Οι στόχοι σχεδιασμού του AngularJS περιλαμβάνουν:

- να αποσυνδεθεί η χειραγώγηση DOM από τη λογική της εφαρμογής. Η δυσκολία αυτού επηρεάζεται δραματικά από τον τρόπο που ο κώδικας είναι δομημένος.
- να αποσυνδεθεί η πλευρά του client(πελάτη) από μία εφαρμογή από την πλευρά του server(διακομιστή). Αυτό επιτρέπει στο προγραμματιστικό έργο να προχωράει παράλληλα, και επιτρέπει την επαναχρησιμοποίηση και των δύο πλευρών.
- να παρέχει δομή για το ταξίδι της οικοδόμησης μιας εφαρμογής: από τον σχεδιασμό της UI, μέσω της συγγραφής της επιχειρηματικής λογικής, μέχρι τον έλεγχο.

Η Angular υλοποιεί το μοτίβο MVC για να διαχωρίσει τα δεδομένα και τα εξαρτήματα της λογικής. Χρησιμοποιώντας dependency injection(πρότυπο σχεδιασμού λογισμικού που υλοποιεί αναστροφή του ελέγχου για την επίλυση εξαρτήσεων), η Angular φέρνει παραδοσιακά υπηρεσίες από την πλευρά του διακομιστή, όπως οι controllers που εξαρτώνται από το οπτικό πεδίο (view), σε διαδικτυακές εφαρμογές από την πλευρά του πελάτη. Κατά συνέπεια, ένα μεγάλο μέρος της επιβάρυνσης στον διακομιστή μπορεί να μειωθεί.

3.4.2.1.2 Εντολή «Scope»

Η Angular χρησιμοποιεί τον όρο «scope» με έναν τρόπο παρόμοιο με τις βασικές αρχές της επιστήμης των υπολογιστών.

Το scope στην επιστήμη των υπολογιστών περιγράφει όταν μέσα στο πρόγραμμα μία συγκεκριμένη δέσμευση είναι έγκυρη. Στην Angular, το «scope» είναι ένα συγκεκριμένο είδος αντικειμένου που από μόνο του μπορεί να βρίσκεται μέσα στο πεδίο ή εκτός του πεδίου εφαρμογής σε οποιοδήποτε δεδομένο μέρος του προγράμματος, ακολουθώντας τους συνήθεις κανόνες του πεδίου εφαρμογής των μεταβλητών στη JavaScript όπως και σε οποιοδήποτε άλλο αντικείμενο. Όταν ο όρος «scope» χρησιμοποιείται παρακάτω, αναφέρεται στο scope αντικειμένων του Angular και όχι στο scope μιας ονοματικής δέσμευσης.

3.4.2.1.3 Bootstrap

Οι εργασίες που εκτελούνται από το πρόγραμμα εκκίνησης της AngularJS συμβαίνουν σε τρεις φάσεις αφού πρώτα έχει φορτωθεί το DOM:

1. Δημιουργία ενός νέου Injector
2. Κατάρτιση των οδηγιών που κοσμούν το DOM
3. Σύνδεση όλων των οδηγιών με το scope

Οι οδηγίες της AngularJS επιτρέπουν στον προγραμματιστή να καθορίσει προσαρμοσμένα και επαναχρησιμοποιούμενα HTML – όπως στοιχεία και χαρακτηριστικά που καθορίζουν τις συνδέσεις των δεδομένων και την συμπεριφορά παρουσίασης των εξαρτημάτων.

Μερικές από τις πιο χρησιμοποιούμενες οδηγίες είναι οι εξής:

- ng-app: Δηλώνει το στοιχείο ρίζας μιας εφαρμογής AngularJS, σύμφωνα με το οποίο μπορούν να χρησιμοποιηθούν οι οδηγίες για να δηλώσουν δεσμεύσεις και να καθορίσουν συμπεριφορά.
- ng-bind: Ορίζει το κείμενο ενός στοιχείου DOM σε μία τιμή μιας έκφρασης. Για παράδειγμα, το δηλώνει την τιμή ενός name στο εσωτερικό του στοιχείου span. Οποιεσδήποτε αλλαγές στη μεταβλητή name στο scope της εφαρμογής αντανακλά άμεσα στο DOM.
- ng-model: Παρόμοιο με το ng-bind, αλλά καθιερώνει μία διπλή δέσμευση δεδομένων μεταξύ του view και του scope.
- ng-model-options: Παρέχει ρύθμιση για το πώς γίνονται οι ενημερώσεις του μοντέλου.
- ng-class : Επιτρέπει στα χαρακτηριστικά κλάσης να φορτωθούν δυναμικά.
- ng-controller: Καθορίζει μία κλάση controller JavaScript που αξιολογεί εκφράσεις HTML.
- ng-repeat: Ενσαρκώνει ένα στοιχείο τη φορά ανά αντικείμενο από μία συλλογή.
- ng-show & ng-hide: Υπό όρους εμφανίζει ή κρύβει ένα στοιχείο, ανάλογα με την τιμή μιας boolean έκφρασης. Η εμφάνιση και η απόκρυψη επιτυγχάνεται με τον καθορισμό του στυλ εμφάνισης CSS.
- ng-switch: Ενσαρκώνει υπό όρους ένα πρότυπο από μια σειρά από επιλογές, ανάλογα με την τιμή μιας έκφρασης επιλογής.
- ng-view: Η βασική οδηγία, υπεύθυνη για τον χειρισμό διαδρομών που επιλύουν JSON πριν καταστήσουν πρότυπα, οδηγούμενη από καθορισμένους controllers.
- ng-if: Η βασική δήλωση οδηγίας if που επιτρέπει την εμφάνιση του παρακάτω στοιχείου εάν οι συνθήκες είναι αληθείς. Όταν η συνθήκη είναι ψευδής, το στοιχείο αφαιρείται από το DOM. Όταν είναι αληθής, ένας κλώνος από το μεταγλωττισμένο στοιχείο εισάγεται εκ νέου.

Καθώς τα χαρακτηριστικά ng-* δεν είναι έγκυρα στις HTML προδιαγραφές, το data-ng-* μπορεί επίσης να χρησιμοποιηθεί ως πρόθεμα. Για παράδειγμα, τόσο το ng-app όσο και το data-ng-app είναι έγκυρα στη AngularJS.

Αμφίδρομη δέσμευση δεδομένων

Η αμφίδρομη δέσμευση της AngularJS είναι το πιο αξιοσημείωτο χαρακτηριστικό του, ανακουφίζοντας σε μεγάλο βαθμό το back-end του διακομιστή από τις πρότυπες ευθύνες. Αντ' αυτού, τα πρότυπα παρέχονται σε μορφή απλού HTML

σύμφωνα με τα δεδομένα που περιέχονται σε ένα scope που ορίζεται από το μοντέλο. Η $scope υπηρεσία στην Angular ανιχνεύει τις αλλαγές στην ενότητα του μοντέλου και τροποποιεί τις εκφράσεις HTML στο οπτικό πεδίο μέσω ενός controller. Ομοίως, οποιεσδήποτε μεταβολές στο οπτικό πεδίο αντικατοπτρίζονται στο μοντέλο. Αυτό παρακάμπτει την ανάγκη να χειραγωγηθεί ενεργά το DOM και ενθαρρύνει την εκκίνηση και την ταχεία προτυποποίηση των διαδικτυακών εφαρμογών. Η AngularJS ανιχνεύει αλλαγές στα μοντέλα συγκρίνοντας τις τρέχουσες τιμές με τιμές που αποθηκεύονται νωρίτερα σε μία διαδικασία dirty-checking, σε αντίθεση με την Ember.js και Backbone.js που ενεργοποιούν τους ακροατές όταν οι τιμές των μοντέλων αλλάζουν.

3.4.2.1.4 Ιστορία Ανάπτυξης

Η AngularJS αναπτύχθηκε αρχικά το 2009 από τον Misko Hevery στο Brat Tech LLC ως το λογισμικό πίσω από μία διαδικτυακή JSON υπηρεσία αποθήκευσης, το οποίο θα είχε τιμολογηθεί από τα megabyte, για εύκολες στην κατασκευή εφαρμογές για την επιχείρηση. Αυτή η επιχείρηση βρισκόταν στον διαδικτυακό τόπο «GetAngular.com», και είχε αρκετούς συνδρομητές, πριν οι δυο τους αποφασίσουν να εγκαταλείψουν την επιχειρησιακή ιδέα και να κυκλοφορήσουν την Angular ως μία ανοιχτού κώδικα βιβλιοθήκη.

- Κυκλοφορίες

Angular 1

Από τις 19 Νοεμβρίου του 2015, η κυκλοφορία 1.4.8 (κωδικό όνομα ice-manipulation) είναι η τρέχουσα σταθερή έκδοση.

Angular 2

Η AngularJS 2.0 ανακοινώθηκε στο ng- ευρωπαϊκό συνέδριο στις 22-23 Σεπτεμβρίου του 2014. Οι δραστικές αλλαγές στην έκδοση 2.0 δημιούργησαν σημαντικές αντιπαραθέσεις μεταξύ των προγραμματιστών. Στις 30 Απριλίου του 2015, οι προγραμματιστές της AngularJS ανακοίνωσαν ότι η AngularJS 2 μετακινήθηκε από την Alpha στην Developer Preview.

- Πρόσθετο εξάρτημα λογισμικού Chrome (Chrome plugin)

Τον Ιούλιο του 2012, η ομάδα Angular κατασκεύασε ένα plugin για το πρόγραμμα περιήγησης του Google Chrome, ονομαζόμενο Batarang, που βελτιώνει την εμπειρία αποσφαλμάτωσης για τις διαδικτυακές εφαρμογές που είναι κατασκευασμένες με Angular.

3.4.2.1.5 Εκτέλεση

Η AngularJS αποδύει το παράδειγμα ενός κύκλου αφομίωσης. Αυτός ο κύκλος μπορεί να θεωρηθεί ως βρόγχος, κατά τον οποίο η AngularJS ελέγχει εάν υπάρχουν αλλαγές σε όλες τις μεταβλητές που παρακολουθούνται από όλα τα $scopes. Οπότε, εάν το $scope.myVar ορίζεται σε έναν controller και αυτή η μεταβλητή σημειώθηκε για παρακολούθηση, η AngularJS θα παρακολουθεί τις αλλαγές στο myVar σε κάθε επανάληψη του βρόγχου. Αυτή η προσέγγιση οδηγεί ενδεχομένως σε επιβράδυνση της απόδοσης, επειδή η AngularJS ελέγχει πάρα πολλές μεταβλητές στο $scope κάθε κύκλου.

3.4.2.2 Bootstrap

Εικόνα 21 : Λογότυπο Bootstrap

Το Bootstrap είναι μία ελεύθερη και ανοιχτού κώδικα συλλογή από εργαλεία για τη δημιουργία ιστοσελίδων και διαδικτυακών εφαρμογών. Περιέχει HTML και CSS πρότυπα σχεδίασης για τυπογραφία, φόρμες, κουμπιά πλοήγησης και άλλα στοιχεία διεπαφής, καθώς επίσης και προαιρετικές επεκτάσεις JavaScript. Στόχο έχει να διευκολύνει την ανάπτυξη δυναμικών ιστοσελίδων και διαδικτυακών εφαρμογών. Το Bootstrap είναι ένα front-end πλαίσιο, δηλαδή, μία διεπαφή για τον χρήστη, σε αντίθεση με τον κώδικα από την πλευρά του διακομιστή ο οποίος βρίσκεται στο back-end του διακομιστή. Το Bootstrap είναι το πιο δημοφιλές πρόγραμμα του GitHub και έχει χρησιμοποιηθεί από τη NASA, μεταξύ άλλων.

3.4.2.2.1 Προέλευση

Το Bootstrap, που αρχικά ονομάστηκε Twitter Blueprint, αναπτύχθηκε από τον Mark Otto και τον Jacob Thornton στο Twitter ως ένα πλαίσιο για την προώθηση της συνοχής μεταξύ των εσωτερικών εργαλείων. Πριν το Bootstrap, ποικίλες βιβλιοθήκες χρησιμοποιήθηκαν για την ανάπτυξη της διεπαφής, κάτι το οποίο οδήγησε σε ασυνέπειες και υψηλό φόρτο συντήρησης.

Η πρώτη εγκατάσταση υπό πραγματικές συνθήκες συνέβη κατά τη διαρκεια της πρώτης Hackweek του Twitter. Μετονομάστηκε από Twitter Blueprint σε Bootstrap, και κυκλοφόρησε ως ένα λογισμικό ανοιχτού κώδικα στις 19 Αυγούστου του 2011. Έχει συνεχίσει να διατηρείται από τον Mark Otto, τον Jacob Thornton και μια μικρή ομάδα κορυφαίων προγραμματιστών, καθώς επίσης και μία μεγάλη κοινότητα από συνεισφέροντες.

Στις 31 Ιανουαρίου του 2012, ανακοινώθηκε το Bootstrap 2. Αυτή η έκδοση πρόσθεσε το διατακτικό πλέγμα δώδεκα στηλών και ανταποκρίσιμα σχεδιαστικά στοιχεία, καθώς επίσης και αλλαγές σε πολλά από τα υπάρχοντα στοιχεία. Η έκδοση Bootstrap 3 ανακοινώθηκε στις 19 Αυγούστου του 2013, μετακινούμενη προς μία φορητή πρώτη προσέγγιση και χρησιμοποιώντας έναν επίπεδο σχεδιασμό.

Στις 29 Οκτωβρίου του 2014, ο Mark Otto ανακοίνωσε ότι κατασκευάζεται το Bootstrap 4. Η πρώτη alpha έκδοση του Bootstrap 4 δρομολογήθηκε στις 19 Αυγούστου του 2015.

3.4.2.2.2 Χαρακτηριστικά

Το Bootstrap είναι συμβατό με τις τελευταίες εκδόσεις των προγραμμάτων περιήγησης του Google Chrome, Firefox, Internet Explorer, Opera, και Safari, αν και μερικά από αυτά τα προγράμματα περιήγησης δεν υποστηρίζονται σε όλες τις πλατφόρμες.

Από την κυκλοφορία της έκδοσης 2.0, υποστηρίζει επίσης τον ανταποκρίσιμο διαδικτυακό σχεδιασμό. Αυτό σημαίνει ότι η διάταξη των ιστοσελίδων προσαρμόζεται δυναμικά, λαμβάνοντας υπόψη τα χαρακτηριστικά της συσκευής που χρησιμοποιείται (επιτραπέζιος υπολογιστής, tablet, κινητό τηλέφωνο). Ξεκινώντας με την έκδοση 3.0, το Bootstrap υιοθέτησε μία πρώτη φιλοσοφία σχεδιασμού για κινητά, δίνοντας έμφαση στον ανταποκρίσιμο σχεδιασμό από προεπιλογή. Η κυκλοφορία της έκδοσης 4.0 alpha πρόσθεσε την υποστήριξη Sass και Flexbox.

Το Bootstrap είναι ανοιχτού κώδικα και διαθέσιμο στο Github. Οι προγραμματιστές ενθαρρύνονται να συμμετέχουν στο project και να κάνουν την δική τους συνεισφορά στη πλατφόρμα.

3.4.2.2.3 Δομή και Λειτουργία

Το Bootstrap είναι σπονδυλωτό και αποτελείται ουσιαστικά από μία σειρά στυλ stylsheets φύλλων LESS που εφαρμόζουν τα ποικίλα στοιχεία της εργαλειοθήκης. Ένα στυλ φύλλων που ονομάζεται bootstrap.less περιέχει αυτά τα συστατικά stylesheets. Οι προγραμματιστές μπορούν να προσαρμόσουν το ίδιο το αρχείο Bootstrap, επιλέγοντας τα συστατικά που επιθυμούν να χρησιμοποιήσουν στο project τους.

Οι προσαρμογές είναι δυνατές μέχρι έναν βαθμό, μέσω ενός κεντρικού στυλ διαμόρφωσης των φύλλων. Βαθύτερες αλλαγές είναι εφικτές μέσω των δηλώσεων LESS. Η χρήση γλώσσας στυλ φύλλων LESS επιτρέπει τη χρήση για μεταβλητές, λειτουργίες και φορείς, ένθετους επιλογείς, γνωστά και ως μείγματα mixins.

Από την έκδοση 2.0, η διαμόρφωση του Bootstrap έχει επίσης μια ειδική επιλογή «Προσαρμογή» στην τεκμηρίωση. Επιπλέον, ο προγραμματιστής επιλέγει μία φόρμα, τα επιθυμητά συστατικά και προσαρμόζει, εάν είναι απαραίτητο, τις τιμές ποικίλων επιλογών σύμφωνα με τις ανάγκες τους. Το πακέτο που δημιουργείται στη συνέχεια περιέχει ήδη το προκατασκευασμένο CSS στυλ φύλλων.

- Σύστημα πλέγματος(Grid system) και ο ανταποκρίσιμος σχεδιασμός(responsive design)

Προσφέρονται με προδιαγραφές ένα διατακτικό πλέγμα των 1170 pixel σε εύρος. Εναλλακτικά, ο προγραμματιστής μπορεί να χρησιμοποιήσει μία μεταβλητού πλάτους διάταξη. Και στις δύο περιπτώσεις, η εργαλειοθήκη έχει τέσσερις παραλλαγές για την χρήση διαφορετικών αναλύσεων και τύπους συσκευών κινητά τηλέφωνα, κατακόρυφη και οριζόντια διάταξη, ταμπλέτες και υπολογιστές με χαμηλή και υψηλή ανάλυση. Κάθε παραλλαγή αλλάζει το πλάτος των στηλών.

- Η κατανόηση του CSS στυλ

Το Bootstrap παρέχει ένα σύνολο στυλ φύλλων που παρέχει βασικούς ορισμούς στυλ για όλα τα βασικά HTML στοιχεία. Αυτά παρέχουν μία ενιαία, σύγχρονη εμφάνιση για μορφοποίηση κειμένου, πίνακες και στοιχεία μίας φόρμας.

- Επαναχρησιμοποιήσιμα στοιχεία

Εκτός από τα βασικά HTML στοιχεία, το Bootstrap περιέχει άλλα στοιχεία περιβάλλοντος που χρησιμοποιούνται συχνά. Αυτά περιλαμβάνουν κουμπιά με προηγμένα χαρακτηριστικά (π.χ. ομαδοποίηση κουμπιών ή κουμπιά με επιλογή drop-down, λίστες πλοήγησης, οριζόντιες και κάθετες καρτέλες, πλοήγηση, πλοήγηση breadcrumb, σελιδοποίηση, κ.τ.λ.), ετικέτες, προηγμένες τυπογραφικές δυνατότητες, εικονίδια, προειδοποιητικά μηνύματα και μία γραμμή προόδου. Τα συστατικά εφαρμόζονται ως κλάσεις CSS, οι οποίες πρέπει να εφαρμοστούν σε συγκεκριμένα στοιχεία HTML μιας σελίδας.

- JavaScript στοιχεία

Το Bootstrap έρχεται με πολλά συστατικά JavaScript σε μία μορφή jQuery plugins. Αυτά παρέχουν επιπλέον στοιχεία διεπαφή χρήστη, όπως είναι τα παράθυρα διαλόγου, οι επεξηγήσεις (tooltips) και τα καρουσέλ (carousels). Επίσης,μπορούν να επεκτείνουν τη λειτουργικότητα ορισμένων υφιστάμενων στοιχείων της διασύνδεσης, συμπεριλαμβανομένης για παράδειγμα μιας λειτουργίας αυτόματης συμπλήρωσης πεδίων εισαγωγής. Στην έκδοση 2.0, υποστηρίζονται τα ακόλουθα plugins: Modal, Dropdown, Scrollspy, Tab, Tooltip, Popover, Alert, Button, Collapse, Carousel και Typeahead.

3.4.2.2.4 Χρήση

Το Bootstrap,για να χρησιμοποιηθεί σε μία σελίδα HTML, ο προγραμματιτής θα πρέπει να κάνει λήψη του στυλ CSS Bootstrap το οποίο περιλαμβάνει μία σύνδεση στο αρχείο HTML. Αν ο προγραμματιστής θέλει να χρησιμοποιήσει τα στοιχεία Javascript, θα πρέπει να αναφέρονται μαζί με τη βιβλιοθήκη jQuery στο HTML έγγραφο.

3.4.3 Web-App

3.4.3.1 BackboneJS

Εικόνα 22 : Λογότυπο BackboneJS

Το Backbone.js είναι μία JavaScript βιβλιοθήκη με μία RESTful JSON διεπαφή και είναι βασισμένο στο model – view – presenter (MVP). Το Backbone είναι γνωστό ότι είναι ελαφρύ, καθώς η μόνη βαριά εξάρτησή του είναι σε μία βιβλιοθήκη JavaScript, την Underscore.js, συν η jQuery για χρήση ολόκληρης της βιβλιοθήκης. Είναι σχεδιασμένο για την κατασκευή μονών διαδικτυακών εφαρμογών, και την διατήρηση του συγχρονισμού διαφόρων μερών των διαδικτυακών εφαρμογών (π.χ. πολλαπλούς πελάτες και διακομιστές). Το Backbone δημιουργήθηκε από τον Jeremy Ashkenas στις 13 Οκτωμβρίου του 2010, που επίσης είναι γνωστός για την CoffeScript και την Underscore.js.

- Χρήση

Κάποιες από τις ακόλουθες διαδικτυακές εφαρμογές είναι κατασκευασμένες με Backbone.js:

- BitTorrent.com
- Foursquare
- Linkedln Mobile
- Pinterest
- Reddit

3.4.3.2 React

Εικόνα 23 : Λογότυπο React

Η React (μερικές φορές ονομαζόμενη ως React.js ή ReactJS) είναι μία ανοιχτού τύπου βιβλιοθήκη JavaScript που παρέχει ένα οπτικό πεδίο για δεδομένα που παρέχονται ως HTML. Τα οπτικά πεδία της React αποδίδονται τυπικά μέσω της χρήσης στοιχείων που περιέχουν επιπλέον στοιχεία που ορίζονται ως ειδικές ετικέτες HTML.

Η React υπόσχεται στους προγραμματιστές ένα μοντέλο στο οποίο τα δευτερεύοντα στοιχεία δεν μπορούν να επηρεάσουν άμεσα τα περιβάλλοντα στοιχεία (τα δεδομένα ρέουν προς τα κάτω), αποτελεσματική κωδικοποίηση για ενημέρωση όταν τα δεδομένα αλλάζουν· και ένας ξεκάθαρος διαχωρισμός ανάμεσα στα στοιχεία μιας σύγχρονης εφαρμογής μονής σελίδας.

Συντηρείται από το Facebook, το Instagram και μία κοινότητα από μεμονωμένους προγραμματιστές και εταιρείες. Σύμφωνα με την αναλυτική υπηρεσία Libscore της JavaScript, η React χρησιμοποιείται αυτή τη στιγμή στις αρχικές ιστοσελίδες των Imgur, Bleacher Report, Feedly, Airbnb, SeatGeek, HelloSign και άλλες.

Από τον Μάρτιο του 2016, η React και React Native είναι τα δύο κορυφαία open-source projects του Facebook από τον αριθμό των αστερίων στο GitHub, και η React είναι το 6^{ο} από τα projects που πρωταγωνίστησαν περισσότερο στο GitHub.

- Ιστορία

Η React δημιουργήθηκε από τον Jordan Walke, έναν μηχανικό λογισμικού του Facebook. Επηρεάστηκε από το XHP, ένα HTML πλαίσιο στοιχείων για PHP.

3.4.3.2.1 Χαρακτηριστικά

Μονή ροή δεδομένων

Οι ιδιότητες, ένα σετ από αναλλοίωτες τιμές, περνάνε στη παροχή ενός στοιχείου ως ιδιότητες της HTML ετικέτας του. Ένα στοιχείο δεν μπορεί να τροποποιήσει καμία από τις ιδιότητες που περνάνε σε αυτό, αλλά μπορούν να περάσουν λειτουργίες ανάκλησης (callback functions) που όντως τροποποιούν τις τιμές. Η υπόσχεση αυτού του μηχανισμού εκφράζεται ως «οι ιδιότητες ρέουν προς τα κάτω και οι ενέργειες ρέουν προς τα πάνω» («properties flow down and actions flow up»).

Εικονικό DOM

Ένα άλλο αξιοσημείωτο χαρακτηριστικό είναι η χρήση ενός «εικονικού DOM». Η React παρέχει δεδομένα που υπάρχουν στη κρυφή μνήμη, υπολογίζει τις διαφορές που προκύπτουν και στη συνέχεια ενημερώνει αποτελεσματικά το εμφανιζόμενο DOM στο πρόγραμμα περιήγησης. Αυτό επιτρέπει στον προγραμματιστή να γράψει κώδικα σαν να παρέχεται ολόκληρη η σελίδα σε κάθε αλλαγή, καθώς οι βιβλιοθήκες React παρέχουν μόνο δευτερεύοντα στοιχεία που πραγματικά αλλάζουν.

JSX

Τα στοιχεία της React είναι συνήθως γραμμένα σε JSX, μία επεκταμένη σύνταξη της JavaScript που επιτρέπει την εύκολη αναφορά σε HTML και τη χρήση συντακτικού HTML ετικέτα για την παροχή δευτερευόντων στοιχείων. Η σύνταξη

HTML μεταποιείται σε κλήσεις JavaScript (JavaScript calls) της βιβλιοθήκης React. Οι προγραμματιστές μπορούν επίσης να γράψουν σε καθαρή JavaScript.

Αρχιτεκτονική πέραν του HTML

Η βασική αρχιτεκτονική της React εφαρμόζεται πέραν της παροχής HTML στο πρόγραμμα περιήγησης. Για παράδειγμα, το Facebook έχει δυναμικά διαγράμματα που παρέχουν τα <canvas> tags, και το Netflix και το Paypal χρησιμοποιούν ισόμορφη φόρτωση για να προσφέρουν πανομοιότυπα HTML, τόσο στον διακομιστή όσο και στον πελάτη.

Οι Μητρικές (Native) βιβλιοθήκες της React ανακοίνωσαν από το Facebook το 2015 παρέχουν την αρχιτεκτονική React στο μητρικό IOS και στις εφαρμογές Android.

3.4.3.3 EmberJS

Εικόνα 24 : Λογότυπο EmberJS

Η Ember.js είναι ένα ανοιχτού κώδικα πλαίσιο εφαρμογών JavaScript βασισμένο στο μοτίβο Model–view–viewmodel (MVVM). Επιτρέπει στους προγραμματιστές να δημιουργήσουν κλιμακούμενες διαδικτυακές εφαρμογές μονής σελίδας, μέσω της ενσωμάτωσης στο πλαίσιο συνηθισμένων ιδιωμάτων και βέλτιστων πρακτικών.

Η Ember χρησιμοποιείται σε πολλές δημοφιλείς ιστοσελίδες, συμπεριλαμβανομένων των Discourse, Groupon, Vine, Live Nation, Nordstrom, and Chipotle. Παρόλο που κατά κύριο λόγο θεωρείται ένα πλαίσιο για το διαδίκτυο, είναι επίσης δυνατό να κατασκευαστούν στην Ember εφαρμογές τόσο για σταθερό υπολογιστή όσο και για κινητό. Το πιο αξιοσημείωτο παράδειγμα μιας εφαρμογής επιτραπέζιου υπολογιστή Ember είναι το Apple Music, χαρακτηριστικό της εφαρμογής επιφάνειας εργασίας iTunes.

Το Νοέμβριο του 2015, η Ember «κατεβάστηκε» πάνω από 200.000 φορές από το npm(διαχειριστής προεπιλεγμένου πακέτου για τη Javascript σε περιβάλλον χρόνου εκτέλεσης Node.js) αποθετήριό της.

3.4.3.3.1 Φιλοσοφία και σχεδιασμός

Από την αρχή, η Ember σχεδιάστηκε γύρω από διάφορες βασικές ιδέες:

- Έμφαση σε φιλόδοξες διαδικτυακές εφαρμογές

  Η Ember έχει ως στόχο να παρέχει μια χονδρική λύση στο πρόβλημα της εφαρμογής από την πλευρά του πελάτη. Αυτό έρχεται σε σύγκρουση με πολλά πλαίσια JavaScript που ξεκινάνε παρέχοντας μία λύση στο V του MVC(model- view-controller), και επιχειρούν να προχωρήσουν από εκεί.

- Πιο παραγωγική και έτοιμη για χρήση

  Η Ember είναι ένα συστατικό από ένα σύνολο εργαλείων που λειτουργούν μαζί για να παρέχουν ένα ολοκληρωμένο προγραμματιστικό stack. Ο στόχος αυτών των εργαλείων είναι να κάνουν τον προγραμματιστή άμεσα παραγωγικό. Για παράδειγμα, η Ember CLI παρέχει μία πρότυπη δομή εφαρμογής και ένα χτίσμα αγωγού. Έχει επίσης μία αρχιτεκτονική που δέχεται πρόσθετα εξαρτήματα λογισμικού και πάνω από χίλια επτακόσια πρόσθετα εξαρτήματα (addons) για την ενίσχυση και επέκτασή της.

- Σταθερότητα χωρίς στασιμότητα

  Αυτή είναι η ιδέα ότι η καθυστερημένη συμβατότητα είναι σημαντική και μπορεί να διατηρηθεί, ενώ εξακολουθεί να καινοτομεί και να εξελίσσει το πλαίσιο.

- Μελλοντική πρόβλεψη των διαδικτυακών προτύπων

  Η Ember έχει υιοθετήσει και έχει υπάρξει πρωτοπόρος πολλών προτύπων γύρω από την JavaScript και το διαδίκτυο, συμπεριλαμβανομένων των υποσχέσεων, Web Components και του συντακτικού ES6. Ο Yehuda Katz, ένας από τους συνιδρυτές της Ember, είναι μέλος της TC39, η οποία είναι η υπεύθυνη επιτροπή για τις μελλοντικές εκδόσεις της γλώσσας JavaScript.

Όπως η Ruby στο Rails, η Ember ακολουθεί την αρχή «Convention over Configuration» (CoC – Σύμβαση πάνω από Διαμόρφωση) και την αρχή «Don't Repeat Yourself» (DRY – Μην Επαναλαμβάνεσαι). Έχει περιγραφτεί ως ένα πολύ δογματικό πλαίσιο που έχει κατασκευαστεί για να είναι πολύ ευέλικτο.

3.4.3.3.2 Βασικές έννοιες

Το Ember αποτελείται από πέντε βασικές έννοιες:

- Διαδρομές

  Στην Ember, η κατάσταση μιας εφαρμογής αντιπροσωπεύεται από ένα URL. Κάθε URL έχει ένα αντικείμενο αντίστοιχης διαδρομής που ελέγχει το τι είναι ορατό στον χρήστη.

- Μοντέλα

  Κάθε διαδρομή έχει ένα σχετικό μοντέλο, που περιέχει τα δεδομένα που σχετίζονται με την τρέχουσα κατάσταση της εφαρμογής. Καθώς κάποιος μπορεί να χρησιμοποιήσει την jQuery για να φορτώσει αντικείμενα JSON από έναν διακομιστή και να χρησιμοποιήσει αυτά τα αντικείμενα ως μοντέλα, οι

περισσότερες εφαρμογές χρησιμοποιούν μία βιβλιοθήκη μοντέλων όπως είναι η Ember Data για να το χειρίζεται αυτό.

- Πρότυπα

  Τα πρότυπα χρησιμοποιούνται για την κατασκευή HTML εφαρμογών και γράφονται με την πρότυπη γλώσσα HTMLBars. (Η HTMLBars είναι μία παραλλαγή του Handlebars που κατασκευάζει στοιχεία DOM και όχι ένα String).

- Συστατικά

  Ένα συστατικό είναι μια ειδική ετικέτα HTML. Η συμπεριφορά εφαρμόζεται χρησιμοποιώντας JavaScript και η εμφάνισή του ορίζεται χρησιμοποιώντας HTMLBars πρότυπα. Στα πρότυπα «ανήκουν» τα δεδομένα τους. Μπορούν επίσης να χρησιμοποιηθούν ως ένθετα και να επικοινωνήσουν με τα αρχικά συστατικά τους μέσω ενεργειών (συμβάντων).

- Υπηρεσίες

  Οι υπηρεσίες είναι απλά αντικείμενα singleton για να κρατάνε μακρόχρονα δεδομένα.

Η Ember επίσης παρέχει ένεση εξάρτησης (dependency injection), δηλωτική αμφίδρομη δέσμευση δεδομένων, υπολογιστικές ιδιότητες και αυτόματες ενημερώσεις προτύπων.

3.4.3.3.3 Ιστορία

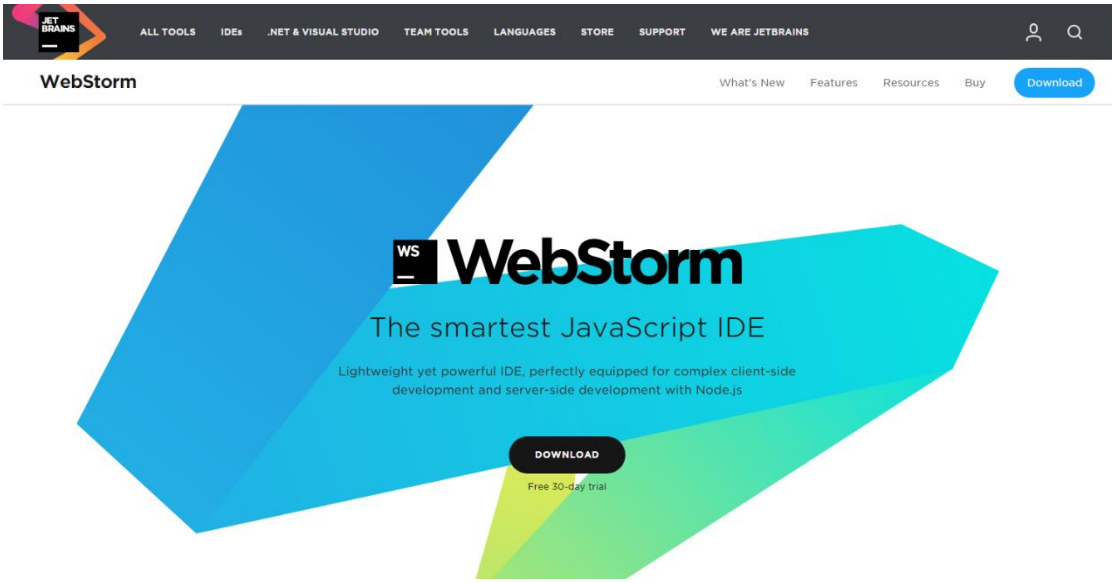
Τον Δεκέμβριο του 2011, το πλαίσιο SproutCore 2.0 μετονομάστηκε σε Ember.js, προκειμένου να μειώσει τη σύγχυση μεταξύ του πλαισίου εφαρμογής και της widget βιβλιοθήκης του SproutCore 1.0. Το πλαίσιο δημιουργήθηκε από τον Yehuda Katz, μέλος των ομάδων της jQuery, της Ruby και της SproutCore. Όπως και πολλά άλλα από τα projects του Katz, υποστηρίζει τη σύμβαση πάνω από τη διαμόρφωση.

4. Σχεδίαση και Υλοποίηση Διαδικτυακής Εφαρμογής

4.1 Προετοιμασία Υλοποίησης

4.1.1 Επιλογή Προγράμματος Υλοποίησης

Για την υλοποίηση της διαδικτυακής εφαρμογής χρειαζόμαστε ένα ολοκληρωμένο περιβάλλον ανάπτυξης λογισμικού(IDE). Υπάρχουν πολλές επιλογές όπως είναι το Eclipse, όμως εμείς επιλέξαμε το WebStorm το οποίο μπορείτε να κατεβάσετε δωρεάν στο https://www.jetbrains.com/webstorm/. Είναι ένα ισχυρό και έξυπνο IDE που δίνει την καλύτερη κωδικοποίηση για JavaScript, HTML και CSS με αυτόματη συμπλήρωση, χαρακτηριστικά refactoring, διορατικότητα κώδικα αλλά και υποστήριξη δημοφιλών frameworks που ήταν μεταξύ άλλων οι λόγοι οι οποίοι μας ώθησαν να το επιλέξουμε.



Εικόνα 25 : Πρόγραμμα υλοποίησης WebStorm

Αφού επιλέξουμε DOWNLOAD, στα έγγραφα μας εμφανίζεται το αρχείο WebStorm-2016.2.2.exe, το οποίο με διπλό κλικ και κάποια επιπλέον βήματα εγκαθίσταται πολύ εύκολα το πρόγραμμα στον υπολογιστή μας.

4.1.2 Εγκατάσταση NodeJS

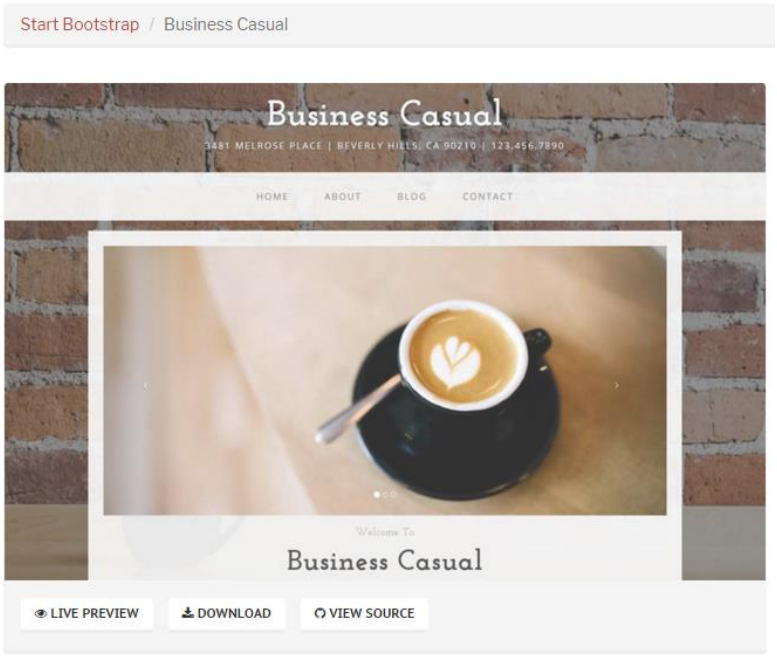
Η ιστοσελίδα https://nodejs.org/en/ παρέχει δωρεάν το NodeJS στην έκδοση v4.5.0 .

Εικόνα 26 : Πλατφόρμα ανάπτυξης λογισμικού NodeJS

Αφού επιλέξουμε Download v4.5.0 LTS, στα έγγραφα μας εμφανίζεται το αρχείο node-v4.5.0-x64.msi, το οποίο με διπλό κλικ και κάποια επιπλέον βήματα εγκαθίσταται πολύ εύκολα το πρόγραμμα στον υπολογιστή μας.

4.1.3 Επιλογή template

Με μία απλή αναζήτηση «bootstrap templates» στο google, μας εμφανίζει πάνω από ένα εκατομμύριο αποτελέσματα. Εμείς επιλέξαμε από τη https://startbootstrap.com/template-categories/all/ σελίδα το template https://startbootstrap.com/template-overviews/business-casual/ .



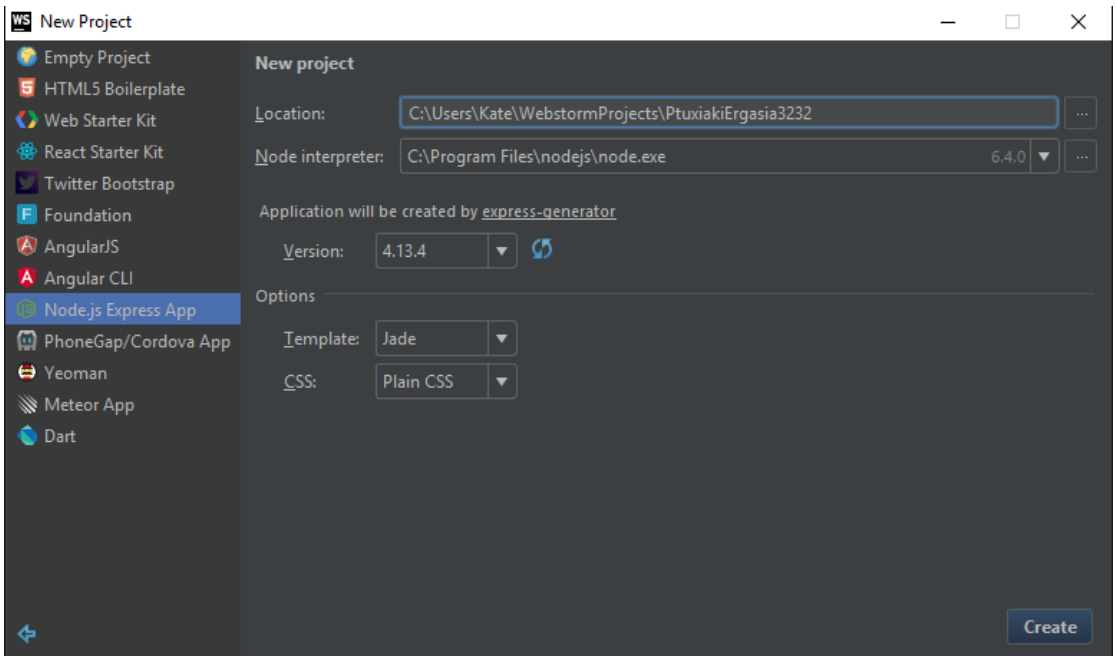
Εικόνα 27 : Το template που επιλέξαμε

Εφόσον πατήσουμε DOWNLOAD, εμφανίζεται στα έγγραφα του υπολογιστή μας το αρχείο startbootstrap-business-casual-gh-pages.zip και κάνουμε αποσυμπίεση του φακέλου.

4.3 Σχεδίαση Διαδικτυακής Εφαρμογής

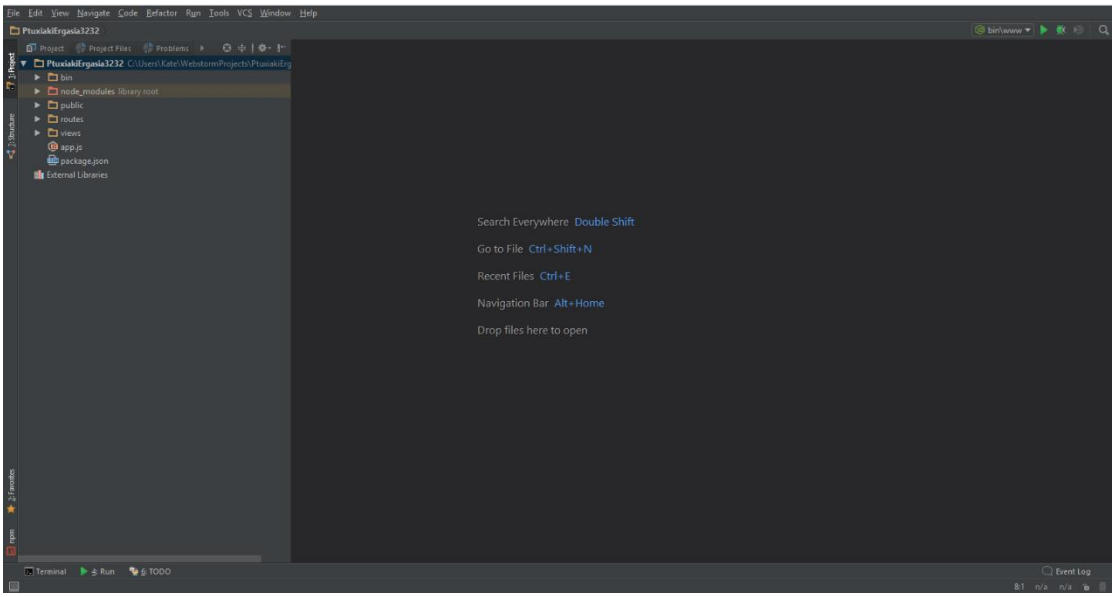
Μετά την εγκατάσταση του WebStorm και του NodeJS είμαστε έτοιμη για την σχεδίαση της εφαρμογής μας.

Δημιουργούμε ένα νέο Node.js Express App Project με όνομα PtuxiakiErgasia3232.



Εικόνα 28 : Δημιουργία Node.js Express App Project

Και το αποτέλεσμα είναι η εικόνα που ακολουθεί :



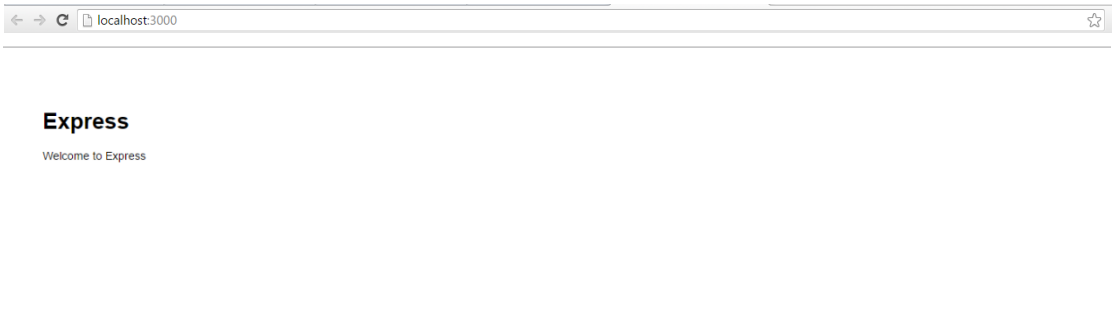
Εικόνα 29 : Αρχική οθόνη WebStorm

Για να ξεκινήσουμε την εφαρμογή μας, στο Terminal που βρίσκεται τέρμα κάτω στο project μας θα πρέπει να πληκτρολογήσουμε την ακόλουθη εντολή:

- npm install : για την εγκατάσταση των εξαρτήσεων στον τοπικό φάκελο node_modules.
- npm start
- npm install ejs

Το επόμενο βήμα είναι να επιλέξουμε από το μενού Run -> Edit Configurations και στην καρτέλα Browser/ Live Edit να κάνουμε τσεκ το After launch. Πατάμε Apply και έπειτα OK.

Τώρα είμαστε έτοιμοι να τρέξουμε την απλή εφαρμογή που ήδη έχει δημιουργήσει το WebStorm. Πατάμε Run -> Run ΄/bin/www΄ .



Εικόνα 30 : Αποτέλεσμα εκτέλεσης παραδείγματος

Σε αυτό το σημείο, μπορούμε μετακινούμε τα αρχεία του template στο project που ήδη δημιουργήσαμε και να κάνουμε τις αλλαγές που επιθυμούμε.

Έτσι τα αρχεία της εφαρμογής μας είναι τα ακόλουθα με τον ακόλουθο κώδικα :

- app.js

```
var express = require('express');
var path = require('path');
var favicon = require('serve-favicon');
var logger = require('morgan');
var cookieParser = require('cookie-parser');
var bodyParser = require('body-parser');

var routes = require('./routes/index');
var users = require('./routes/users');

var app = express();

// view engine setup
app.set('views', path.join(__dirname, 'views'));
app.set('view engine', 'ejs');

// uncomment after placing your favicon in /public
//app.use(favicon(path.join(__dirname, 'public', 'favi-
con.ico')));
app.use(logger('dev'));
app.use(bodyParser.json());
app.use(bodyParser.urlencoded({ extended: false }));
app.use(cookieParser());
app.use(express.static(path.join(__dirname, 'public')));
```

```
app.use('/', routes);
app.use('/users', users);

// catch 404 and forward to error handler
app.use(function(req, res, next) {
  var err = new Error('Not Found');
  err.status = 404;
  next(err);
});

// error handlers

// development error handler
// will print stacktrace
if (app.get('env') === 'development') {
  app.use(function(err, req, res, next) {
    res.status(err.status || 500);
    res.render('error', {
      message: err.message,
      error: err
    });
  });
}

// production error handler
// no stacktraces leaked to user
app.use(function(err, req, res, next) {
  res.status(err.status || 500);
  res.render('error', {
    message: err.message,
    error: {}
  });
});


module.exports = app;
```

- view / error.ejs

```
<h1><%= message %></h1>
<h2><%= error.status %></h2>
<pre><%= error.stack %></pre>
```

- view / index.ejs

```
<!DOCTYPE html>
<html lang="en">

<head>

<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-
scale=1">
<meta name="description" content="">
<meta name="author" content="">
```

```
<title>Welcome to My Web Application</title>

<!-- Bootstrap Core CSS -->
<link href="../css/bootstrap.min.css" rel="stylesheet">

<!-- Custom CSS -->
<link href="../css/business-casual.css" rel="stylesheet">

<!-- Fonts -->
<link href="https://fonts.googleapis.com/css?fam-
ily=Open+Sans:300italic,400italic,600italic,700italic,800italic
,400,300,600,700,800" rel="stylesheet" type="text/css">
<link href="https://fonts.googleapis.com/css?family=Jose-
fin+Slab:100,300,400,600,700,100italic,300italic,400italic,600i
talic,700italic" rel="stylesheet" type="text/css">

<!-- HTML5 Shim and Respond.js IE8 support of HTML5 elements
and media queries -->
<!-- WARNING: Respond.js doesn't work if you view the page via
file:// -->
<!--[if lt IE 9]>
<script
src="https://oss.maxcdn.com/libs/html5shiv/3.7.0/html5shiv.js">
</script>
<script src="https://oss.maxcdn.com/libs/respond.js/1.4.2/re-
spond.min.js"></script>
<![endif]-->

</head>

<body>

<div class="brand">My Web Application</div>

<!-- Navigation -->
<nav class="navbar navbar-default" role="navigation">
<div class="container">
<!-- Brand and toggle get grouped for better mobile display -->
<div class="navbar-header">
<button type="button" class="navbar-toggle" data-toggle="col-
lapse" data-target="#bs-example-navbar-collapse-1">
<span class="sr-only">Toggle navigation</span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
<span class="icon-bar"></span>
</button>
<!-- navbar-brand is hidden on larger screens, but visible when
the menu is collapsed -->
<a class="navbar-brand" href="index.html">My Web Applica-
tion</a>
</div>
<!-- Collect the nav links, forms, and other content for tog-
gling -->
<div class="collapse navbar-collapse" id="bs-example-navbar-
collapse-1">
</div>
<!-- /.navbar-collapse -->
</div>
<!-- /.container -->
</nav>
```

```
<div class="container">

<div class="row">
<div class="box">
<div class="col-lg-12 text-center">
<div id="carousel-example-generic" class="carousel slide">
<!-- Indicators -->
<ol class="carousel-indicators hidden-xs">
<li data-target="#carousel-example-generic" data-slide-to="0"
class="active"></li>
</ol>

<!-- Wrapper for slides -->
<div class="carousel-inner">
<div class="item active">
<img class="img-responsive img-full img-center" src="../../im-
ages/slide-1.png" alt="">
</div>
</div>

</div>
<h2 class="brand-before">
<small>Welcome to</small>
</h2>
<h1 class="brand-name">My Web Application</h1>
<hr class="tagline-divider">
<h2>
<small>By
<strong>zervoudaki aikaterini</strong>
</small>
</h2>
</div>
</div>
</div>
<div class="row">
<div class="box">
<div class="col-lg-12">
<hr>
<h2 class="intro-text text-center">Search </h2>


<img class="img-responsive img-border img-left" src="" alt="">
<hr class="visible-xs">
<p>.</p>
</div>
</div>
</div>

</div>
<!-- /.container -->


<!-- jQuery -->
<script src="../../js/jquery.js"></script>

<!-- Bootstrap Core JavaScript -->
<script src="../../js/bootstrap.min.js"></script>


</body>
```

```
</html>
```

- public / css / bootstrap-casual.css

```
/*!
 * Start Bootstrap - Business Casual (http://startboot-
strap.com/)
 * Copyright 2013-2016 Start Bootstrap
 * Licensed under MIT (https://github.com/BlackrockDig-
ital/startbootstrap/blob/gh-pages/LICENSE)
 */

body {
    font-family: "Open Sans","Helvetica Neue",Helvet-
ica,Arial,sans-serif;
    background: url('../images/bg.jpg') no-repeat center center
fixed;
    -webkit-background-size: cover;
    -moz-background-size: cover;
    background-size: cover;
    -o-background-size: cover;
}

h1,
h2,
h3,
h4,
h5,
h6 {
    text-transform: uppercase;
    font-family: "Josefin Slab","Helvetica Neue",Helvet-
ica,Arial,sans-serif;
    font-weight: 700;
    letter-spacing: 1px;
}

p {
    font-size: 1.25em;
    line-height: 1.6;
    color: #999999;
}

hr {
    max-width: 400px;
    border-color: #999999;
}

.brand,
.address-bar {
    display: none;
}

.navbar-brand {
    text-transform: uppercase;
    font-weight: 900;
    letter-spacing: 2px;
}

.navbar-nav {
```

```
        text-transform: uppercase;
        font-weight: 400;
        letter-spacing: 3px;
    }

    .img-full {
        min-width: 30%;
    }

    .img-center {
        display: block;
        margin: 0 auto;
    }

    .brand-before,
    .brand-name {
        text-transform: capitalize;
    }

    .brand-before {
        margin: 15px 0;
    }

    .brand-name {
        margin: 0;
        font-size: 4em;
    }

    .tagline-divider {
        margin: 15px auto 3px;
        max-width: 250px;
        border-color: #999999;
    }

    .box {
        margin-bottom: 20px;
        padding: 30px 15px;
        background: #fff;
        background: rgba(255,255,255,0.9);
    }

    .intro-text {
        text-transform: uppercase;
        font-size: 1.25em;
        font-weight: 400;
        letter-spacing: 1px;
    }

    .img-border {
        float: none;
        margin: 0 auto 0;
        border: #999999 solid 1px;
    }

    .img-left {
        float: none;
        margin: 0 auto 0;
    }


    @media screen and (min-width:768px) {
```

```
        .brand {
            display: inherit;
            margin: 0;
            padding: 30px 0 10px;
            text-align: center;
            text-shadow: 1px 1px 2px rgba(0,0,0,0.5);
            font-family: "Josefin Slab","Helvetica Neue",Helvet-
    ica,Arial,sans-serif;
            font-size: 5em;
            font-weight: 700;
            line-height: normal;
            color: #404040;
        }

        .top-divider {
            margin-top: 0;
        }

        .img-left {
            float: left;
            margin-right: 25px;
        }

        .address-bar {
            display: inherit;
            margin: 0;
            padding: 0 0 40px;
            text-align: center;
            text-shadow: 1px 1px 2px rgba(0,0,0,0.5);
            text-transform: uppercase;
            font-size: 1.25em;
            font-weight: 400;
            letter-spacing: 3px;
            color: #2F4F4F;
        }

        .navbar {
            border-radius: 0;
        }

        .navbar-header {
            display: none;
        }

        .navbar {
            min-height: 0;
        }

        .navbar-default {
            border: none;
            background: #fff;
            background: rgba(255,255,255,0.9);
        }

        .nav>li>a {
            padding: 35px;
        }

        .navbar-nav>li>a {
            line-height: normal;
        }
```

```
    .navbar-nav {
        display: table;
        float: none;
        margin: 0 auto;
        table-layout: fixed;
        font-size: 1.25em;
    }
}

@media screen and (min-width:1200px) {
    .box:after {
        content: '';
        display: table;
        clear: both;
    }
}
```

- public / stylesheets / style.css

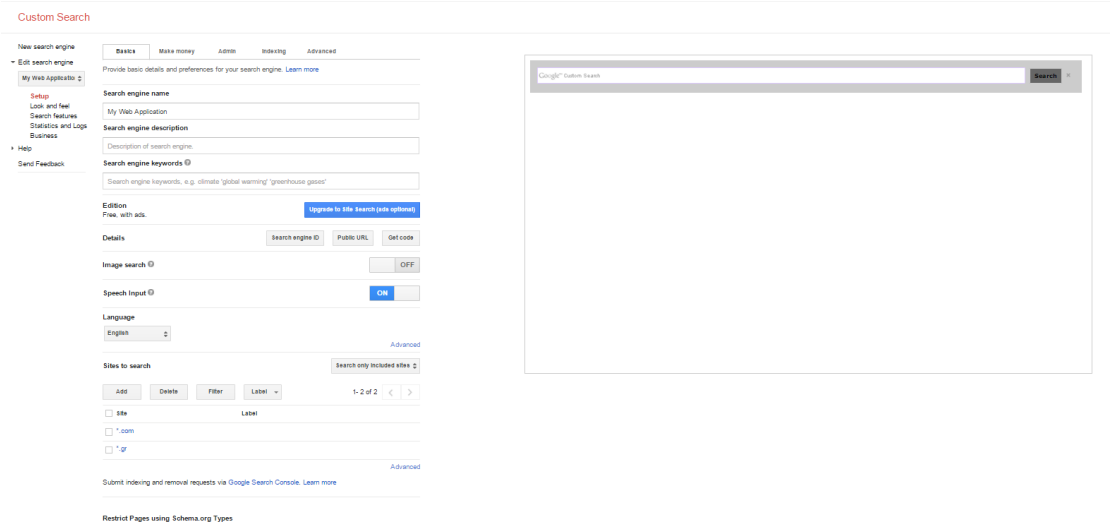
```
body {
  padding: 50px;
  font: 14px "Lucida Grande", Helvetica, Comic Sans MS, sans-
serif;
}

a {
  color: #000100;
}
```

Και τώρα είμαστε έτοιμη να τρέξουμε την εφαρμογή μας.

Εικόνα 31 : Αποτέλεσμα διαδικτυακής εφαρμογής

4.4 Google Custom Search

Για την άντληση δεδομένων από το διαδίκτυο θα χρησιμοποιήσουμε την πλατφόρμα Google Custom Search που μπορούμε να την βρούμε στο https://cse.google.com/cse/all. Εκεί από το μενού επιλέγουμε New search engine και δημιουργούμε τη δική μας μηχανή αναζήτησης.



Εικόνα 32 : Η μηχανή αναζήτησης μας

Τέλος, πατάμε Get code και κάνουμε επικόλληση στο Search.

```
<h2 class="intro-text text-center">Search </h2>
<hr>

    <!-- google custom search -->
    <script>
        (function() {
            var cx = '000942921358891182181:-2-i5ruu8xg';
            var gcse = document.createElement('script');
            gcse.type = 'text/javascript';
            gcse.async = true;
            gcse.src = 'https://cse.google.com/cse.js?cx=' + cx;
            var s = document.getElementsByTagName('script')[0];
            s.parentNode.insertBefore(gcse, s);
        })();
    </script>
    <gcse:search></gcse:search>
    <!-- /.google custom search -->
```

4.5 Αποτέλεσμα Υλοποίησης

Με βάση την έρευνα και εφαρμογή των frameworks, με τη χρήση του WebStorm και της πλατφόρμας ανάπτυξης λογισμικού NodeJS, υλοποιήσαμε την διαδικτυακή εφαρμογή στην ακόλουθη εικόνα.

Εικόνα 33 : Παρουσίαση υλοποίησης εφαρμογής

5. Συμπεράσματα

Με την παρούσα πτυχιακή εργασία γίνεται έρευνα και μελέτη των σύγχρονων JavaScript frameworks, της πλατφόρμας ανάπτυξης λογισμικού NodeJS με σκοπό την δημιουργία μίας βέλτιστης διαδικτυακής εφαρμογής. Με την ανάλυση που γίνεται στο κεφάλαιο 4 δίνεται η δυνατότητα στον αναγνώστη, να δημιουργήσει και αυτός με την σειρά του μία αντίστοιχη εφαρμογή έχοντας τις βασικές γνώσεις για το πρόγραμμα WebStorm, της πλατφόρμας NodeJS και τη χρήση του Google Custom Search.

6. Βιβλιογραφία

1. https://www.jetbrains.com/help/webstorm/2016.2/node-js.html
2. https://www.npmjs.com/
3. http://blog.udacity.com/2014/12/front-end-vs-back-end-vs-full-stack-web-developers.html
4. https://en.wikipedia.org/wiki/JavaScript
5. http://noeticforce.com/best-Javascript-frameworks-for-single-page-modern-web-applications
6. https://en.wikipedia.org/wiki/Node.js
7. https://www.lullabot.com/articles/choosing-the-right-javascript-framework-for-the-job
8. https://en.wikipedia.org/wiki/Backbone.js
9. https://en.wikipedia.org/wiki/Bootstrap_(front-end_framework)
10. https://en.wikipedia.org/wiki/React_(JavaScript_library)
11. https://nodejs.org/en/about/
12. https://www.jetbrains.com/webstorm/
13. https://startbootstrap.com/template-categories/all/
14. http://slidedeck.io/westmonroe/modern-javascript-frameworks
15. https://en.wikipedia.org/wiki/List_of_JavaScript_libraries#Web-application_related_.28MVC.2C_MVVM.29