Τεχνολογικό Εκπαιδευτικό Ίδρυμα Κρήτης

Σχολή Τεχνολογικών Εφαρμογών
Τμήμα Μηχανικών Πληροφορικής

Πτυχιακή εργασία

«Σχεδιασμός και ανάπτυξη διαδραστικής εφαρμογής για την πρόληψη της νόσου του Alzheimer»

ΟΝΟΜΑΤΕΠΩΝΥΜΟ: **ΚΥΡΙΑΚΟΠΟΥΛΟΥ ΜΑΡΙΝΑ**
ΗΜΕΡΟΜΗΝΙΑ: **05/04/17**

ΕΙΣΗΓΗΤΗΣ: **ΒΙΔΑΚΗΣ ΝΙΚΟΛΑΟΣ**

Ευχαριστίες

Με την ολοκλήρωση αυτής της πτυχιακής θα ήθελα να ευχαριστήσω αρχικά την οικογένεια μου για όλη τους την αγάπη και την ψυχολογική και οικονομική στήριξη που μου παρείχαν καθ' όλη την διάρκεια των φοιτητικών και μαθητικών μου σπουδών όπως και τον φίλο μου Αλέξανδρο για όλη του την αγάπη και καθοδήγηση που μου παρείχε στις δύσκολες στιγμές που αντιμετώπισα. Τέλος θέλω να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή της πτυχιακής μου κ. Νικόλαο Βιδάκη, διότι μου έδωσε την δυνατότητα να ασχοληθώ με ένα θέμα το οποίο θεωρώ εξαιρετικά σημαντικό.

Abstract

Research has shown that the systematic involvement of a person, with games, which are designated to exercise memory and concentration, contributes to long-term preservation of the human memory and therefore leads to the prevention of the most common form of dementia, known as “Alzheimer's disease”.
The aim of this thesis is to design and create an application, which in accordance with educational standards, will help the user to sharpen their memory, and also keep track of its condition, In a user-friendly environment.

Σύνοψη

Σύμφωνα με έρευνες που έχουν διεξαχθεί, έχει αποδειχθεί ότι η συστηματική ενασχόληση ενός ατόμου, με παιχνίδια τα οποία αποσκοπούν στην εξάσκηση της μνήμης και της συγκέντρωσης, συμβάλλει στην μακροχρόνια διατήρηση της μνήμης του ανθρώπου και συνεπώς στην πρόληψη της πιο κοινής μορφής άνοιας, γνωστής και ως «νόσος Alzheimer».

Σκοπός της πτυχιακής αυτής είναι ο σχεδιασμός και η δημιουργία μιας εφαρμογής, η οποία σύμφωνα με τα εκπαιδευτικά πρότυπα, θα βοηθήσει τον χρήστη στην όξυνση της μνήμης του, ενώ παράλληλα θα εξετάζεται και η κατάστασή της, σε ένα φιλικό προς τον ίδιο περιβάλλον.

Δομή τόμου

- Η **εισαγωγή** της παρούσας πτυχιακής εργασίας αποτελείται από ένα κείμενο, το οποίο περιγράφει συνοπτικά ορισμένες βασικές πληροφορίες σχετικά με τη Νόσο Αλτσχάιμερ

- Το **1ο κεφάλαιο** περιέχει πληροφορίες σχετικά με τα υπολογιστικά συστήματα, τις μονάδες τους, τις κατηγορίες στις οποίες κατατάσσονται, καθώς και μια ιστορική αναδρομή με την εξέλιξη των υπολογιστικών συστημάτων από την αρχαιότητα ως και σήμερα.

- Το **2ο κεφάλαιο** αποτελεί μια εισαγωγή στην έννοια των «σοβαρών παιχνιδιών» (Serious Games) . Παρουσιάζονται τα είδη των serious games, διάφορες εφαρμογές τους, η διαδικασία σχεδιασμού ενός παιχνιδιού αυτού του είδους, καθώς και η σύνδεσή τους με τις θεωρίες μάθησης. Στο κεφάλαιο περιλαμβάνεται επίσης μια σύντομη ιστορική αναδρομή, σχετικά με την εξέλιξη των serious games κατά τις τελευταίες δεκαετίες.

- Στο **3ο κεφάλαιο** παρουσιάζονται πληροφορίες σχετικά με τις γλώσσες προγραμματισμού C# και SQL, οι οποίες αποτελούν τις κύριες γλώσσες προγραμματισμού, οι οποίες χρησιμοποιήθηκαν για την ανάπτυξη της εφαρμογής.

- Στο **4ο κεφάλαιο** παρουσιάζονται πληροφορίες σχετικά με το Visual Studio 2015, το οποίο αποτελεί την πλατφόρμα όπου αναπτύχθηκε η παρούσα εφαρμογή.

- Τέλος, το **5ο κεφάλαιο** αποτελεί μια αναλυτική περιγραφή της διαδικασίας που ακολουθήθηκε κατά τον σχεδιασμό και την ανάπτυξη της εφαρμογής.

Πίνακας περιεχομένων

Κατάλογος εικόνων

Κεφάλαιο 1 – Η χρήση των υπολογιστών στην καθημερινή μας ζωή:

Κεφάλαιο 2-Εισαγωγή στην έννοια των serious games:

Κεφάλαιο 3 - Οι γλώσσες προγραμματισμού C# και MySQL:

Κεφάλαιο 4 - Εισαγωγή στο Visual Studio:

Κεφάλαιο 5 - Σχεδιασμός και υλοποίηση της εφαρμογής:

Κατάλογος πινάκων

Κεφάλαιο 1 – Η χρήση των υπολογιστών στην καθημερινή μας ζωή:

Κεφάλαιο 2-Εισαγωγή στην έννοια των serious games:

Κεφάλαιο 3 - Οι γλώσσες προγραμματισμού C# και MySQL:

Η νόσος Αλτσχάιμερ είναι η πιο κοινή μορφή άνοιας. Είναι μια ασθένεια μη θεραπεύσιμη, εκφυλιστική και θανατηφόρα η οποία περιγράφηκε αρχικά από το Γερμανό ψυχίατρο και νευροπαθολόγο Αλοΐσιο Αλτσχάιμερ το 1906 (εξ ού και το όνομα). Γενικά εντοπίζεται στους ανθρώπους πάνω από 65 ετών, αν και το λιγότερο συχνά, πρόωρο Αλτσχάιμερ μπορεί να εμφανιστεί πολύ νωρίτερα, ίσως και πριν τα 50.

Εκτιμάται ότι, το 2006, υπέφεραν από την νόσο Αλτσχάιμερ περίπου 26,6 εκατομμύρια άνθρωποι. Αυτός ο αριθμός ενδέχεται να έχει τετραπλασιαστεί μέχρι το 2050.

Η νόσος Αλτσχάιμερ, αποτελείται από διάφορα στάδια, τα οποία περιγράφονται από ορισμένα κοινά συμπτώματα. Τα πιο πρόωρα συμπτώματα της νόσου, είναι η απώλεια μνήμης, και η δυσκολία επαναφοράς στην μνήμη πρόσφατων γεγονότων. Με την πρόοδο της ασθένειας, τα συμπτώματα περιλαμβάνουν σύγχυση, οξυθυμία, επιθετικότητα, ταλάντευση διάθεσης, διακοπή ομιλίας, απώλεια της μακροπρόθεσμης μνήμης, και τη γενική κοινωνική απόσυρση του πάσχοντος καθώς οι αισθήσεις του μειώνονται. Τελικά, ο ασθενής οδηγείται σε θάνατο, καθώς οι σωματικές λειτουργείες μειώνονται.

Η πρόγνωση της ασθένειας είναι δύσκολο να αξιολογηθεί, καθώς η διάρκεια της ανάπτυξής της διαφέρει σε κάθε περίπτωση. Η μέση περίοδος ανάπτυξης της νόσου εκτιμάται ότι είναι 7 χρόνια.

Η ακριβής αιτία και η πρόοδος της νόσου Αλτσχάιμερ δεν είναι ακόμα γνωστές. Σύμφωνα με τις έρευνες που έχουν διεξαχθεί μέχρι σήμερα, η ασθένεια πιθανόν συνδέεται με πλάκες στον εγκέφαλο.

Αφού η ακριβής αιτία της νόσου δεν είναι γνωστή, συνεπώς δεν είναι δυνατός ο ακριβής προσδιορισμός των μέτρων πρόληψης κατά της νόσου. Η διαρκής διανοητική εγρήγορση, η σωματική άσκηση, και μια ισορροπημένη διατροφή συστήνονται ως πιθανή μέθοδος πρόληψης.

Σκοπός της εφαρμογής η οποία αναπτύχθηκε για την παρούσα πτυχιακή εργασία είναι η προσπάθεια πρόληψης της νόσου Αλτσχάιμερ, βασιζόμενοι στο γεγονός ότι η διανοητική εγρήγορση αποτελεί μια πιθανή μέθοδο πρόληψης. Η εφαρμογή είναι κατάλληλη για ανθρώπους μεγαλύτερης ηλικίας καθώς παρέχει φιλικό περιβάλλον, απλούς χειρισμούς και ενθάρρυνση του χρήστη.

Κεφάλαιο 1: Η χρήση των υπολογιστών στην καθημερινή μας ζωή

1.1 Το υπολογιστικό σύστημα

Με τον όρο υπολογιστικό σύστημα, αναφερόμαστε σε μια πλήρη υπολογιστική συσκευή, η οποία απαρτίζεται από δυο κύρια τμήματα: Το **υλικό (hardware)** και το **λογισμικό (software)**.

Το υλικό ενός υπολογιστικού συστήματος, αποτελεί το σύνολο των μηχανημάτων, τα οποία το απαρτίζουν. Όλα αυτά τα μηχανήματα, ελέγχονται μέσω του λογισμικού, το οποίο ουσιαστικά είναι ένα σύνολο προγραμμάτων.

1.1.1 Υλικό ενός υπολογιστικού συστήματος

Κάθε υπολογιστικό σύστημα αποτελείται από διάφορα μέρη, τα οποία ονομάζονται μονάδες. Κάθε μονάδα είναι επιφορτισμένη με μια ή περισσότερες λειτουργίες και επικοινωνεί με άλλες μονάδες, με τελικό στόχο την επεξεργασία της πληροφορίας.

Ανάλογα με την λειτουργία της, κάθε μονάδα μπορεί να καταταγεί σε μια από τις κατηγορίες του παρακάτω πίνακα (πίνακας 1.1).

Μονάδες υπολογιστικού συστήματος		
Όνομα	Περιγραφή	Παραδείγματα
Μονάδα εισόδου	Επιτρέπει την εισαγωγή δεδομένων στον υπολογιστή.	1. Πληκτρολόγιο 2. Ποντίκι 3. Οθόνη αφής 4. Σαρωτής (scanner) 5. Κάμερα 6. Μικρόφωνο 7. Ιχνόσφαιρα 8. Χειριστήρια παιχνιδιών
Μονάδα εξόδου	Παρουσιάζει το αποτέλεσμα της επεξεργασίας των δεδομένων στη μορφή που έχει ζητήσει ο χρήστης.	1. Οθόνη 2. Εκτυπωτής 3. Ηχεία 4. Βιντεοπροβολέας 5. Σχεδιογράφος
Μονάδα Κύριας Μνήμης	Η μονάδα όπου αποθηκεύονται προσωρινά τα δεδομένα τα οποία χρησιμοποιούνται από τον επεξεργαστή.	-
Κεντρική Μονάδα Επεξεργασίας	Η μονάδα αυτή είναι επιφορτισμένη με τη διαδικασία της επεξεργασίας των δεδομένων και αποτελείται από διάφορες επιμέρους μονάδες, όπως η Μονάδα Αριθμητικών και Λογικών Πράξεων, η Μονάδα Ελέγχου και άλλες.	-

Πίνακας 1.1 Κατηγορίες Μονάδων Υπολογιστικού Συστήματος

1.1.2 Λογισμικό ενός υπολογιστικού συστήματος

Το λογισμικό αποτελείται από προγράμματα, τα οποία χρησιμοποιούνται από τον υπολογιστή. Με άλλα λόγια, πρόκειται για μια συλλογή από προγράμματα, διαδικασίες και οδηγίες χρήσεις, τα οποία είναι υπεύθυνα για την εκτέλεση διάφορων διεργασιών του υπολογιστή.

Το λογισμικό του υπολογιστή διακρίνεται σε κατηγορίες, ανάλογα με τις εργασίες τις οποίες εκτελεί. Οι κατηγορίες αυτές, ιεραρχικά ξεκινώντας από το λογισμικό που είναι πλησιέστερο στη γλώσσα μηχανής, προς το λογισμικό που είναι πιο φιλικό προς το χρήστη, είναι οι εξής:

- Το υλικολογισμικό (Firmware)
- Το ενδιάμεσο λογισμικό
- Το λογισμικό συστήματος
- Το λογισμικό εφαρμογών.

Θα αναφερθούμε συνοπτικά σε κάθε κατηγορία ξεχωριστά.

1.1.2.1 Το υλικολογισμικό (Firmware)

Το υλικολογισμικό είναι ένα είδος λογισμικού, το οποίο είναι γραμμένο σε γλώσσα μηχανής. Κάθε μοντέλο ηλεκτρονικής συσκευής έχει το δικό της ξεχωριστό υλικολογισμικό. Το firmware δημιουργείται αποκλειστικά από τους κατασκευαστές κάθε συσκευής, και είναι σχεδόν αδύνατο κάποιος χρήστης να δημιουργήσει το δικό του firmware.

Για να εγκατασταθεί το υλικολογισμικό, απαιτείται μια συγκεκριμένη διαδικασία. Αρχικά, πρέπει να αφαιρεθεί το ήδη εγκατεστημένο λογισμικό, και στη συνέχεια να εγκατασταθεί και να ελεγχθεί το νέο. Το προηγούμενο λογισμικό αφαιρείται με την εφαρμογή τάσης στην μνήμη, οπότε ορισμένες φορές υπάρχει η πιθανότητα να εφαρμοστεί παραπάνω τάση από όσο πρέπει.

Το αντίστοιχο υλικολογισμικό στους υπολογιστές, είναι το BIOS.

1.1.2.2 Το Ενδιάμεσο λογισμικό

Το ενδιάμεσο λογισμικό αποτελεί το λογισμικό, το οποίο επιτρέπει τη διεπαφή μεταξύ του πελάτη και του διακομιστή. Συχνά, όταν ένας πελάτης απαιτεί μία υπηρεσία από έναν διακομιστή το αίτημα περνά μέσω πολλών ενδιάμεσων στρωμάτων λογισμικού που παρεμβάλλονται μεταξύ των πελατών και των διακομιστών, δηλαδή μέσω του ενδιαμέσου λογισμικού.

Ένα αντιπροσωπευτικό παράδειγμα ενδιάμεσου λογισμικού είναι το λογισμικό διεπαφής ενός περιηγητή και του Παγκόσμιου Ιστού. Όταν γίνεται αίτηση για μια ιστοσελίδα, ας πούμε όταν ένας χρήστης κάνει κλικ σε έναν υπερσύνδεσμο μιας σελίδας, ένα κείμενο, που αποτελεί μέρος ενός πρωτοκόλλου, αποστέλλεται στο ενδιάμεσο λογισμικό ζητώντας για την σελίδα. Στην συνέχεια, το ενδιάμεσο λογισμικό θα εντοπίσει την σελίδα, θα την ανακτήσει και θα την στείλει πίσω στον περιηγητή. Ως τμήμα του ενδιάμεσου λογισμικού υπάρχει κώδικας που ασχολείται

με την ανεύρεση σελίδων, την παρακολούθηση λαθών, την αναφορά λαθών, την μεταβίβαση δεδομένων και την επικοινωνία με τα κατώτερα στρώματα λογισμικού, που είναι μέρος του πιο διαδεδομένου διαδικτυακού πρωτοκόλλου, του TCP/IP.

Το ενδιάμεσο λογισμικό διακρίνεται σε δύο κατηγορίες: το γενικό και το ειδικό λογισμικό.

Το γενικό ενδιάμεσο λογισμικό είναι λογισμικό που σχετίζεται με υπηρεσίες που ζητούνται από όλους τους πελάτες και τους διακομιστές, ενώ το ειδικό ενδιάμεσο λογισμικό συσχετίζεται με μία συγκεκριμένη υπηρεσία όπως η εκτύπωση αρχείων σε έναν απομακρυσμένο υπολογιστή.

1.1.2.3 Το λογισμικό συστήματος

Το λογισμικό συστήματος αποτελεί το λογισμικό, το οποίο διαχειρίζεται τους πόρους ενός υπολογιστικού συστήματος. Χωρίς την ύπαρξη του λογισμικού αυτού, η διεπαφή μεταξύ του χρήστη και του υπολογιστή θα ήταν σχεδόν ακατόρθωτη.

Το λογισμικό αυτού του είδους χωρίζεται στις εξής κατηγορίες:

- **Λειτουργικό Σύστημα**: Αποτελεί μία συλλογή βασικών προγραμμάτων, η οποία ελέγχει τη λειτουργία του υπολογιστή συνολικά και χρησιμοποιείται ως υπόβαθρο για την εκτέλεση όλων των υπόλοιπων προγραμμάτων, τη διαχείριση των περιφερειακών συσκευών και την εξασφάλιση της επικοινωνίας μεταξύ χρήστη και υπολογιστή. Στην πράξη πρόκειται για ένα επίπεδο λογισμικού που μεσολαβεί μεταξύ του υλικού και των εκτελούμενων προγραμμάτων σε έναν ηλεκτρονικό υπολογιστή. Αποτελείται από ένα σύνολο μηχανισμών μέσω των οποίων επιτυγχάνεται αυτόματη διαχείριση των πόρων ενός υπολογιστή και ελεγχόμενη κατανομή τους στις εκτελούμενες εφαρμογές, έτσι ώστε οι τελευταίες να είναι σε θέση να προσπελάσουν εύκολα τους πόρους και τις συσκευές του συστήματος χωρίς να χρειάζεται να γνωρίζουν με ακρίβεια τη δομή του υποκείμενου υλικού, αλλά και ώστε πολλαπλές εφαρμογές να μπορούν να εκτελούνται ταυτόχρονα χωρίς να έρχονται σε διένεξη μεταξύ τους ή με τον υπολογιστή.

- **Λογισμικό Ανάπτυξης ή Λογισμικό Προγραμματισμού**: Για να μπορέσει ο υπολογιστής να ανταποκριθεί στις εντολές του προγραμματιστή, είναι αναγκαία η ύπαρξη ενός λογισμικού, το οποίο θα μεταφράζει την ανθρώπινη γλώσσα σε γλώσσα μηχανής. Για να ικανοποιηθεί αυτή η ανάγκη, αναπτύχθηκε το λογισμικό προγραμματισμού. Το λογισμικό αυτό αποτελείται από μεταγλωττιστές και από βιβλιοθήκες για κάθε προγραμματιστική γλώσσα.

- **Βοηθητικά Προγράμματα**

- **Διασύνδεση χρήστη με γραφικά (graphical user interface GFI)**

1.1.2.4 Το λογισμικό εφαρμογών

Το λογισμικό εφαρμογών αποτελείται από προγράμματα, τα οποία αναπτύσσονται από προγραμματιστές, έτσι ώστε να καλύπτονται διάφορες ανάγκες. Τα προγράμματα αυτά μπορεί να εξυπηρετούν διάφορους σκοπούς, όπως για παράδειγμα επαγγελματικούς, εκπαιδευτικούς, ψυχαγωγικούς, ή και μίξη αυτών. Παραδείγματα τέτοιων εφαρμογών είναι:

- Το λογισμικό επεξεργασίας κειμένου
- Τα λογιστικά φύλλα
- Το λογισμικό διαχείρισης βάσεων δεδομένων
- Τα προγράμματα παρουσίασης
- Τα προγράμματα γραφικών
- Οι εφαρμογές δημιουργίας πολυμέσων
- Τα πακέτα λογισμικού ψυχαγωγίας και εκπαίδευσης
- Τα εργαλεία σχεδίασης, τα προγράμματα αναζήτησης Διαδικτύου καθώς και οι εφαρμογές Internet
- Τα παιχνίδια.

1.2 Είδη υπολογιστικών συστημάτων

1.2.1 Κατηγορίες υπολογιστικών συστημάτων

Τα υπολογιστικά συστήματα χωρίζονται σε δυο βασικές κατηγορίες, ανάλογα με το μέγεθός τους : Στους **μικροϋπολογιστές** και στα **μεγάλα συστήματα**. [6]

1.2.1.1 Μικροϋπολογιστές

Οι μικροϋπολογιστές είναι η πιο γνωστή κατηγορία υπολογιστών στο ευρύ κοινό. Προορίζονται κυρίως για οικιακή χρήση, αν και έχουν ένα μεγάλο εύρος δυνατοτήτων. Αυτή η κατηγορία υπολογιστικών συστημάτων, άρχισε να διαδίδεται κατά τη δεκαετία του 70, από εταιρίες όπως η Apple, η Commodore και η Attari. Πλέον, μπορούμε να πούμε ότι οι μικροϋπολογιστές βρίσκονται σε κάθε σπίτι. Τα δυο πιο διαδεδομένα είδη μικροϋπολογιστών είναι:

- Οι προσωπικοί υπολογιστές (Personal Computers – PCs)
- Οι υπολογιστές της εταιρίας Apple

Οι μικροϋπολογιστές, ανάλογα με το μέγεθος και την χρήση τους, διακρίνονται σε περεταίρω υποκατηγορίες:

1. **Υπολογιστές γραφείου (επιτραπέζιοι υπολογιστές - Desktop)**

Αυτή η υποκατηγορία μικροϋπολογιστών απαρτίζεται, όπως δηλώνει και το όνομά της, από υπολογιστές, οι οποίοι τοποθετούνται πάνω στην επιφάνεια ενός γραφείου. Αποτελούνται από δύο κύρια μέρη, την κεντρική μονάδα (κουτί ή πύργος), την οθόνη και συνήθως διαθέτουν και περιφερειακές συσκευές, όπως ένα πληκτρολόγιο και ένα ποντίκι. Οι επιτραπέζιοι υπολογιστές διαθέτουν σχετικά μεγάλο όγκο και βάρος, οπότε είναι δύσκολο να μετακινηθούν από το σημείο το οποίο εγκαθίστανται αρχικά.

Εικόνα 1.1 Επιτραπέζιος υπολογιστής (Desktop)

2. **Φορητοί υπολογιστές (Laptop / Notebook)**

Οι φορητοί υπολογιστές έχουν παρόμοια λειτουργία με τους υπολογιστές γραφείου. Η κύρια διαφορά τους είναι η φορητότητά τους, καθώς οι φορητοί υπολογιστές είναι μικρότεροι σε μέγεθος από τους υπολογιστές γραφείου.

Στους φορητούς υπολογιστές, η οθόνη είναι ενσωματωμένη. Συνήθως έχει διαγώνιο 11,4 μέχρι 18,4 ίντσες. Ένα laptop μπορεί να διαθέτει είτε σκληρό δίσκο HDD, χωρητικότητας από 500 GB ως 1,5 TB, είτε δίσκο SSD , χωρητικότητας 32, 64, 120, 256 ή 512 GB. Η μνήμη RAM είναι τύπου DDR3 και χωρητικότητας από 2 έως 8 GB. Όλα τα περιφερειακά μηχανήματα είναι επίσης ενσωματωμένα.

Εικόνα 1.2 Φορητός υπολογιστής (laptop/notebook)

Η επιλογή ενός laptop έναντι ενός desktop έχει ορισμένα πλεονεκτήματα καθώς και ορισμένα μειονεκτήματα, τα οποία αναγράφονται στον παρακάτω πίνακα.

Πλεονεκτήματα	Μειονεκτήματα
Η φορητότητα ενός laptop, καθιστά εύκολη τη μεταφορά του.	Ένας φορητός υπολογιστής έχει ελαφρώς μεγαλύτερο κόστος από έναν σταθερό υπολογιστή με τις ίδιες δυνατότητες.
Ο σχεδιασμός του είναι περισσότερο εργονομικός από ενός desktop.	Οι αναβαθμίσεις ενός φορητού υπολογιστή είναι πιο ακριβές και περιορισμένες σε ποικιλία από ότι ενός σταθερού υπολογιστή.
Έχει σχετικά μεγάλη αυτονομία σε περίπτωση διακοπής ρεύματος.	Λόγω της υψηλής συρρίκνωσης των κυκλωμάτων τους, οι φορητοί υπολογιστές είναι αρκετά πιο ευαίσθητοι από τους σταθερούς.
Καταναλώνουν λιγότερη ενέργεια (σχεδόν κατά 50%) από έναν σταθερό υπολογιστή	Ασυμβατότητα υλικού μεταξύ διαφορετικών κατασκευαστών (δηλαδή πολλά εξαρτήματα φορητών υπολογιστών δεν κάνουν σε άλλους φορητούς υπολογιστές)

Πίνακας 1.2 Τα πλεονεκτήματα και τα μειονεκτήματα ενός φορητού έναντι ενός σταθερού υπολογιστή

3. Προσωπικοί ψηφιακοί βοηθοί (PDA – Personal Digital Assistant)

Ο προσωπικός ψηφιακός βοηθός (PDA), είναι μια εύχρηστη και συμπαγής συσκευή μικρού μεγέθους, η οποία χρησιμοποιείται κυρίως για αποθήκευση, ανάκτηση και οργάνωση δεδομένων και διαθέτει δυνατότητα τηλεφωνικής και διαδικτυακής σύνδεσης. Διαθέτει οθόνη αφής, η οποία χρησιμοποιείται με ένα ειδικό στυλό (stylus).

Ένα τυπικό PDA διαθέτει οθόνη αφής (ή απλά ένα συμβατικό πληκτρολόγιο), υποδοχή κάρτας μνήμης για την αποθήκευση δεδομένων και δυνατότητα σύνδεσης με wifi και Bluetooth. Πλέον έχουν αντικατασταθεί από τα «έξυπνα κινητά» (smartphones), όμως χρησιμοποιούνται ακόμα από ορισμένες μεταφορικές εταιρίες για την διαχείριση των παραγγελιών.

Εικόνα 1.3: Προσωπικός ψηφιακός βοηθός (PDA – Personal Digital Assistant)

4. «Έξυπνα τηλέφωνα» (Smartphones) \ Tablets

Τα «έξυπνα τηλέφωνα» (smartphones) αποτελούν μια πιο εξελιγμένη έκδοση των συμβατικών κινητών τηλεφώνων. Βασίζονται σε ένα λειτουργικό σύστημα με μεγάλη υπολογιστική ικανότητα. Αρχικά, απλά αντικαθιστούσαν τους Προσωπικούς Ψηφιακούς Βοηθούς (PDA), αργότερα όμως προστέθηκαν και άλλες λειτουργίες, όπως φορητοί media players, ψηφιακές κάμερες και βιντεοκάμερες τσέπης χαμηλής ανάλυσης. Τα πιο σύγχρονα μοντέλα διαθέτουν λειτουργίες όπως GPS, γρήγορους web browsers και δυνατότητα σύνδεσης στο διαδίκτυο μέσω WIFI.

Τα λειτουργικά συστήματα τα οποία χρησιμοποιούνται στα smartphones είναι τα εξής:

- Το Android της Google
- Το iOS της Apple
- Το Symbian της Nokia
- Το BlackBerry OS της RIM
- Τα Windows Phone της Microsoft
- Το webOS της Hewlett-Packard

Μερικά άλλα επερχόμενα λειτουργικά συστήματα είναι το Firefox OS της Mozilla, το Ubuntu Phone της Canonical Ltd's και το Tizen.

Οι διαστάσεις της οθόνης ενός smartphone, κυμαίνονται γύρω στις 3 με 8 ίντσες, ενώ η αναλύσεις της οθόνης ποικίλουν από 320x240px (QVGA) έως 1920x1080px (Full HD).

Εικόνα 1.4 "Έξυπνα τηλέφωνα" (Smartphones)

Ένα tablet έχει παρόμοια λειτουργία με ένα smartphone, και μεγαλύτερο μέγεθος από αυτό. Διαθέτει οθόνη αφής, ο χειρισμός της οποίας γίνεται με το δάκτυλο ή με ειδικό στυλό (stylus).

Τα tablet διαθέτουν τα ίδια λειτουργικά συστήματα όπως τα smartphones. Το μέγεθος της διαγώνιου της οθόνης τους κυμαίνεται από 7 έως 10.1 ίντσες.

Άλλα χαρακτηριστικά που μπορεί να διαθέτει ένα tablet, είναι τα εξής :

- **Επιταχυνσιόμετρο**: Το επιταχυνσιόμετρο είναι μια συσκευή που ανιχνεύει τις φυσικές κινήσεις της ταμπλέτας. Αυτό επιτρέπει μεγαλύτερη ευελιξία στη χρήση. Το επιταχυνσιόμετρο μπορεί επίσης να χρησιμοποιηθεί για την ανίχνευση του προσανατολισμού της ταμπλέτας σε σχέση με το οριζόντιο επίπεδο, αλλά μπορεί επίσης να ανιχνεύσει την κίνηση, δεδομένα τα οποία μπορούν να χρησιμοποιηθούν ως εναλλακτική διεπαφή ελέγχου για το λογισμικό.

- **Αισθητήρες φωτισμού και εγγύτητας**: είναι συνήθως μια μονάδα που ανιχνεύει το επίπεδο φωτισμού και την απόσταση από ένα αντικείμενο, με αυτόν τον τρόπο η CPU ρυθμίζει αυτόματα την φωτεινότητα της οθόνης και εάν χρειαστεί τη σβήνει για να αποφευχθούν εντολές από ακούσιες επαφές

- **Μέσα αποθήκευσης**: Όλες οι ταμπλέτες (tablets) χρησιμοποιούν για αποθηκευτικό μέσο μνήμες flash στερεάς κατάστασης, και αυτό γιατί καταναλώνουν πολύ λιγότερη ενέργεια για την λειτουργία τους, και έχουν μεγάλη ανεκτικότητα στις ζημίες. Αλλά όσο μεγάλος είναι ο αποθηκευτικός χώρος της ταμπλέτας πότε δεν είναι αρκετός, έτσι λοιπόν για να αντιμετωπιστεί το πρόβλημα αυτό, οι ταμπλέτες (εκτός τα iPad's) έχουν και υποδοχή για κάρτες μνήμης τύπου SD με υποστήριξη μέχρι 32 με 64 GB (διαφέρει από συσκευή σε συσκευή). Επιπλέον εάν υποστηρίζουν το USB-OTG, εξωτερικοί σκληροί δίσκοι και USB flash drives εγγυώνται άφθονο χώρο.

- **Ασύρματο**: Επειδή οι ταμπλέτες από το σχεδιασμό τους είναι σαν φορητός υπολογιστής, οι ασύρματες συνδέσεις είναι λιγότερο περιοριστικές για την κίνηση από τις ενσύρματες συνδέσεις. Η συνδεσιμότητα Wi-Fi έχει γίνει απανταχού παρούσα στις ταμπλέτες, τα πολύ φτηνά tablets έχουν μόνο αυτό τον τρόπο σύνδεσης. Η συνδεσιμότητα Bluetooth χρησιμοποιείται συνήθως για τη σύνδεση περιφερειακών και για την επικοινωνία με τις τοπικές συσκευές στη θέση της ενσύρματου USB σύνδεσης.

- **3D** :Μετά από τα κινητά τηλέφωνα, υπάρχουν και 3D ταμπλέτες με διπλό φακό στο πίσω μέρος της ταμπλέτας και παρέχονται επίσης με μπλε-κόκκινο γυαλί.

- Σταθμός σύνδεσης: Μερικά νεότερα δισκία προσφέρουν ένα προαιρετικό σταθμό σύνδεσης που διαθέτει ένα πλήρες πληκτρολόγιο QWERTY μεγέθους και θύρα USB, παρέχοντας τόσο φορητότητα και ευελιξία.

Εικόνα 1.5: Tablet

1.2.1.2 Μεγάλα συστήματα

Αν και οι σύγχρονοι μικροϋπολογιστές έχουν πλέον πολύ μεγάλες δυνατότητες, δεν επαρκούν για ορισμένες εργασίες, οι οποίες απαιτούν τεράστια υπολογιστική ισχύ. Γι' αυτό τον λόγο υπάρχουν τα μεγάλα υπολογιστικά συστήματα. Ανάλογα με το μέγεθός τους και τις δυνατότητές τους, τα μεγάλα υπολογιστικά συστήματα διακρίνονται σε τρείς κατηγορίες:

1. Τους υπερυπολογιστές (Supercomputers)
2. Τους μεγάλους υπολογιστές (Mainframes)
3. Και τους μεσαίους υπολογιστές

1. Υπερυπολογιστές (Supercomputers)

Οι υπερυπολογιστές είναι οι ισχυρότεροι αλλά και ακριβότεροι υπολογιστές που υπάρχουν. Έχουν την δυνατότητα να εκτελούν έναν τεράστιο αριθμό πράξεων κινητής υποδιαστολής ανά δευτερόλεπτο, και αυτό το οφείλουν στους εκατοντάδες ως και χιλιάδες επεξεργαστές που διαθέτουν. Η ικανότητα υπολογισμών μετριέται συνήθως με τον όρο Flops (FLoating-point Operations Per Second, υπολογισμοί κινητής υποδιαστολής ανά δευτερόλεπτο). Η υπολογιστική ικανότητα των σημερινών υπερυπολογιστών έχει ξεπεράσει το 1 PetaFlop.

Οι υπερυπολογιστές χρησιμοποιούνται για απαιτητικές προσομοιώσεις. Οι πιο συνήθεις χρήσεις τους γίνονται :

- Σε πανεπιστήμια και διαστημικούς σταθμούς, για την προσομοίωση φαινομένων όπως για παράδειγμα για την προσομοίωση της συμπεριφοράς των αστεριών ενός γαλαξία ή της ατμόσφαιρας σε πλανητική κλίμακα.
- Στις ειδικές δυνάμεις, για την προσομοίωση πολεμικών σεναρίων.

- Για τον σχεδιασμό μέσων μεταφοράς.
- Σε μετεωρολογικούς και σεισμολογικούς σταθμούς.
- Σε δημόσιους οργανισμούς και υπουργεία.

Οι υπερυπολογιστές, αν και έχουν τεράστια υπολογιστική ισχύ, δεν μπορούν να εκτελέσουν πολλές εργασίες ταυτόχρονα, καθώς είναι κατασκευασμένοι ώστε να δίνουν μεγαλύτερη έμφαση στην ταχύτητα εκτέλεσης της εργασίας.

Εικόνα 1.6: Δωμάτιο με υπερυπολογιστές

2. Μεγάλοι \ Κεντρικοί υπολογιστές (Mainframes)

Οι μεγάλοι ή κεντρικοί υπολογιστές (mainframes) είναι ισχυροί και ακριβοί υπολογιστές οι οποίοι διαθέτουν μνήμη RAM πολλών gb, και εκατοντάδες επεξεργαστές. Διατηρούνται σε ειδικά κλιματιζόμενα δωμάτια, όπου έχει πρόσβαση μόνο εξουσιοδοτημένο προσωπικό. Μπορούν να χρησιμοποιηθούν ταυτόχρονα από πολλούς χρήστες, οι οποίοι έχουν πρόσβαση σε αυτούς μέσω τερματικών. Επίσης, έχουν την δυνατότητα να διαχειρίζονται πλήθος εργασιών ταυτόχρονα, σε αντίθεση με τους υπερυπολογιστές. Είναι πολύ αξιόπιστοι και έχουν υψηλή απόδοση.

Οι κεντρικοί υπολογιστές χρησιμοποιούνται:

- από κυβερνητικές υπηρεσίες
- από μεγάλες εταιρίες
- σε εργαστήρια
- από οργανισμούς
- σε ερευνητικά κέντρα

για κρίσιμες εφαρμογές, όπως:

- μαζική επεξεργασία συναλλαγών και δεδομένων σε απογραφή πληθυσμού
- στατιστικές έρευνες βιομηχανιών/καταναλωτών
- σχεδιασμός και διαχείριση πόρων
- Επεξεργασία μεγάλου όγκου δεδομένων

Εικόνα 1.7: Κεντρικοί υπολογιστές (mainframes)

3. Μεσαίοι υπολογιστές

Οι μεσαίοι (ή μίνι) υπολογιστές κατατάσσονται ανάμεσα στους μικροϋπολογιστές και τα μεγαλύτερα συστήματα και ως προς την ισχύ και ως προς το κόστος. Μπορούν να εξυπηρετήσουν μικρά δίκτυα μικροϋπολογιστών ή τερματικά. Χρησιμοποιούνται και ως αυτόνομες μονάδες αλλά και συνδεδεμένοι με μεγαλύτερα συστήματα. Συνήθως χρησιμοποιούνται για την αποθήκευση και διαχείριση δεδομένων μικρών επιχειρήσεων, ή σχολείων. Φυσικά, το κόστος κάθε κατηγορίας υπολογιστικών συστημάτων αυξάνεται ανάλογα με τις δυνατότητες της. Στην παρακάτω εικόνα, παρουσιάζεται ποιοτικά η διακύμανση των τιμών μεταξύ των υπολογιστικών συστημάτων.

Εικόνα 1.8: Σχηματική αναπαράσταση διακύμανσης τιμών μεταξύ των υπολογιστικών συστημάτων

1.2.2 Ιστορική εξέλιξη υπολογιστικών συστημάτων

1.2.2.1 Υπολογιστές κατά την αρχαιότητα

Εκτιμάται ότι τα υπολογιστικά συστήματα υπάρχουν από τα αρχαία χρόνια. Βέβαια δεν είχαν την μορφή την οποία γνωρίζουμε σήμερα, αλλά ήταν απλούστερα.
Μια πρώιμη μορφή υπολογιστικού συστήματος, ήταν ο άβακας, δηλαδή το γνωστό σε όλους Αριθμητήριο. Αναπτύχθηκε γύρω στο 2200 π.Χ. από τους Βαβυλωνίους, καθώς είχαν αναπτύξει το εμπόριο τους σε μεγάλο βαθμό, και χρειαζόταν ένα εργαλείο για να τους βοηθά στους υπολογισμούς τους.

Αρκετούς αιώνες αργότερα, το 130 π.Χ , ανακαλύφθηκε το Κόσκινο του Ερατοσθένη. Η ανακάλυψη αυτή έγινε από τον ομώνυμο μαθηματικό της εποχής εκείνης, ο οποίος είχε ανάγκη ένα μέσο για να υπολογίζει τους πρώτους αριθμούς.
Μερικά χρόνια μετά (κατά το 150 με 100 π.Χ.), δημιουργήθηκε ο μηχανισμός των Αντικυθήρων. Οι αρχαίοι Έλληνες είχαν αναπτύξει τεράστιο πολιτισμό και, φυσικά, ενδιαφέρθηκαν για τις Επιστήμες όπως Μαθηματικά, Αστρονομία κ.α. Οι πληροφορίες που έχουμε για την αρχαία ελληνική τεχνολογία είναι κυρίως γραπτές. Οι μόνοι μηχανισμοί (ή θραύσματά τους) που έχουν μέχρι στιγμής ανακαλυφθεί είναι ο Μηχανισμός των Αντικυθήρων και ο Βυζαντινός μηχανισμός.

Ο Μηχανισμός των Αντικυθήρων είναι συσκευή αστρονομικών υπολογισμών που χαρακτηρίζεται παγκόσμια ως ο «Αρχαιότερος Υπολογιστής». Κατασκευάστηκε γύρω στο 87 π.Χ. -πιθανά στη Ρόδο- και διέθετε 32 οδοντωτά γρανάζια. Κατά τη μεταφορά του στη Ρώμη το πλοίο που τον μετέφερε βυθίστηκε κοντά στα Αντικύθηρα και ανακαλύφθηκε γύρω στα 1900 από ομάδα σφουγγαράδων. Σήμερα βρίσκεται στο Εθνικό Αρχαιολογικό Μουσείο.

Οι διαστάσεις του είναι 16 x 32 x 9 cm (ίδιες με αυτές ενός σύγχρονου φορητού υπολογιστή). Αποτελούνταν από ένα κέλυφος με ενδεικτικούς πίνακες στην εξωτερική του όψη και ένα πολυσύνθετο μηχανισμό 32 τροχών στο εσωτερικό του. Ο πίνακας έδειχνε την ετήσια κίνηση του ήλιου στο ζωδιακό κύκλο καθώς και τις ανατολές και τις δύσεις των λαμπρών άστρων και αστερισμών κατά τη διάρκεια του έτους.

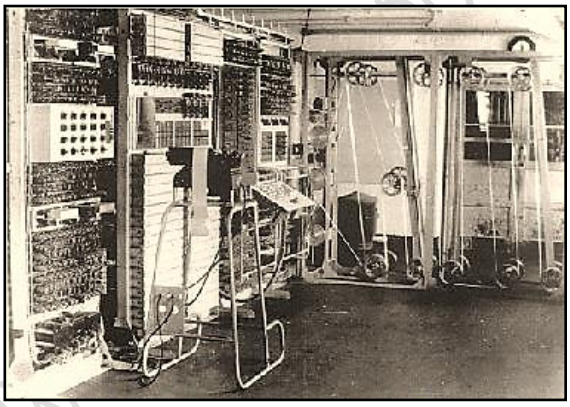
Εικόνα 1.9 Οι δυο αρχαιότεροι υπολογιστές: Ο Άβακας και ο Μηχανισμός των Αντικυθήρων

1.2.2.2 Οι σύγχρονοι υπολογιστές

Ο πρώτος «ηλεκτρονικός» υπολογιστής (ENIAC), δημιουργήθηκε κατά τη δεκαετία του 1940. Από τότε, οι ηλεκτρονικοί υπολογιστές έχουν υποστεί πολλές αλλαγές και έχουν περάσει από πολλά στάδια, πριν καταλήξουν στην μορφή με την οποία τους γνωρίζουμε σήμερα. Καθένα από αυτά τα στάδια αποτελεί μια γενιά υπολογιστών. Συνολικά υπάρχουν πέντε γενιές.

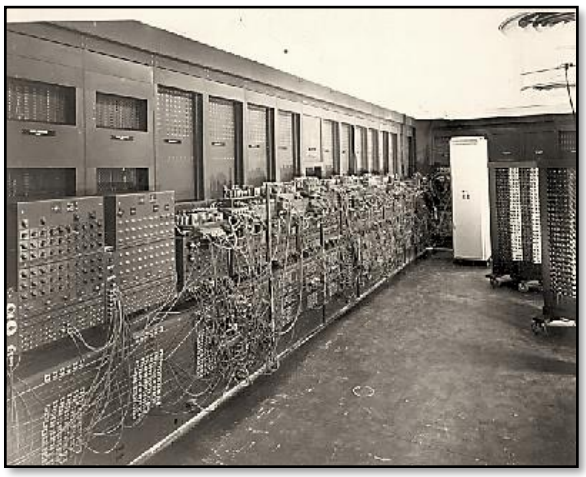
1. Πρώτη γενιά υπολογιστών (1943- 1956)

Κατά τη διάρκεια του Β΄ Παγκοσμίου πολέμου, ο μαθηματικός Άλαν Τιούρινγκ σχεδίασε την μηχανή Τιούρινγκ, η οποία είχε τη δυνατότητα να λύσει οποιοδήποτε πρόβλημα με την μορφή αλγορίθμου. Η μηχανή αυτή δεν ήταν δυνατό να κατασκευαστεί, έδωσε όμως την ιδέα για τον σχεδιασμό του ENIAC. Κατά την ίδια χρονική περίοδο, το 1943, κατασκεύασε τον Colossus Mark I, ο οποίος χρησιμοποιήθηκε για την για την αποκρυπτογράφηση της γερμανικής μηχανής Enigma στον Β' Παγκόσμιο πόλεμο. Ο Colossus Mark I ήταν ο πρώτος προγραμματιζόμενος υπολογιστής που κατασκευάστηκε ποτέ.



Εικόνα 1.10: Ο Colossus Mark I, ο πρώτος προγραματιζόμενος υπολογιστής

Αργότερα, το 1943, η ανάγκη του στρατού των Ηνωμένων Πολιτειών για μια συσκευή η οποία θα βοηθούσε τους στρατιωτικούς στους υπολογισμούς για να βρίσκουν με τα όπλα τους το στόχο με μεγαλύτερη ακρίβεια, οδήγησε στην κατασκευή μιας τεράστιας υπολογιστικής συσκευής, με το όνομα ENIAC. Ο ENIAC, ο οποίος καταλάμβανε έναν ολόκληρο όροφο, αντί για κυκλώματα, διέθετε λυχνίες.



Εικόνα 1.11: Ο υπολογιστής ENIAC

Αποτέλεσε τον πρώτο επαναπρογραμματιζόμενο υπολογιστή, ο κώδικας του οποίου γραφόταν σε γλώσσα μηχανής. Παρ' όλα αυτά, το τεράστιο μέγεθος του ENIAC, η ανάγκη που είχε να ελέγχεται συνεχώς από ειδικούς, και το γεγονός ότι οι λυχνίες του καίγονταν συχνά και απαιτούσαν αντικατάσταση, οδήγησε σύντομα στο σχεδιασμό της δεύτερης γενιάς υπολογιστών.

2. Δεύτερη γενιά υπολογιστών (1956- 1963)

Το 1956 ξεκίνησε να κατασκευάζεται η δεύτερη γενιά υπολογιστών. Οι λυχνίες αντικαταστάθηκαν από τρανζίστορ. Η αλλαγή αυτή επέτρεψε την κατασκευή μικρότερων σε μέγεθος υπολογιστών, με περισσότερες δυνατότητες. ο πρώτος Ηλεκτρονικός Υπολογιστής δεύτερης γενιάς, ήταν ο TX-0, ο οποίος κατασκευάστηκε στο Τεχνολογικό Ινστιτούτο Μασαχουσέτης (M.I.T.), το 1956.

Παράλληλα, εμφανίστηκαν και οι πρώτες γλώσσες προγραμματισμού υψηλού επιπέδου: η FORTRAN, η ALGOL και η COBOL.

3. Τρίτη γενιά υπολογιστών (1964- 1971)

Η δημιουργία της τρίτης γενιάς υπολογιστών, αποτέλεσε ένα σημείο- καμπή για την εξέλιξη των υπολογιστών. Όλα ξεκίνησαν το 1958, όταν ο Τζακ Κίλμπυ (Jack Kilby), της εταιρείας Texas Instruments κατασκεύασε το πρώτο Ολοκληρωμένο Κύκλωμα συνδυάζοντας τρανζίστορς, πυκνωτές, αντιστάτες και άλλα ηλεκτρονικά εξαρτήματα όλα τοποθετημένα στο ίδιο κομμάτι από πυρίτιο. Από το 1964, ξεκίνησαν να κατασκευάζονται υπολογιστές οι οποίοι χρησιμοποιούσαν ως βασικές δομικές μονάδες τα ολοκληρωμένα κυκλώματα μικρής κλίμακας ολοκλήρωσης (SSI) και μέσης κλίμακας ολοκλήρωσης(MSI). Αυτές οι συνθήκες επέτρεψαν την δημιουργία μικρότερων σε μέγεθος υπολογιστικών συστημάτων, που μπορούσαν ακόμα και να μεταφερθούν.

Τεράστια εξέλιξη υπήρξε και στον τομέα του λογισμικού. Εκείνη την χρονική περίοδο, άρχισαν να αναπτύσσονται οι πρώτες γλώσσες προγραμματισμού υψηλού επιπέδου, οι οποίες διέθεταν και έξυπνους μεταφραστές (compilers) , καθώς και τα πρώτα λειτουργικά συστήματα πολυπρογραμματισμού και καταμερισμού χρόνου.

4. Τέταρτη γενιά υπολογιστών (1971 - σήμερα)

Οι σύγχρονοι υπολογιστές τους οποίους χρησιμοποιούμε σήμερα, ανήκουν στην τέταρτη γενιά. Οι υπολογιστές αυτής της γενιάς χρησιμοποιούν ως βασικές δομικές μονάδες ολοκληρωμένα κυκλώματα μεγάλης και πολύ μεγάλης κλίμακας ολοκλήρωσης(LSI και VLSI). Κάθε υπολογιστής τέταρτης γενιάς, είναι εφοδιασμένος με έναν επεξεργαστή (CPU), μνήμη , μονάδα αποθήκευσης πληροφοριών και μονάδες εισόδου (πληκτρολόγιο, ποντίκι, κλπ.)

2.1 Τί είναι τα «σοβαρά παιχνίδια» (serious games)

Ο όρος serious games, αναφέρεται σε παιχνίδια, των οποίων πρωταρχικός σκοπός είναι η εκπαίδευση, και όχι η ψυχαγωγία. Τα παιχνίδια αυτά είναι σχεδιασμένα ώστε να λειτουργούν σε υπολογιστές, κονσόλες, ή φορητές συσκευές, όπως είναι τα tablet και τα κινητά τηλέφωνα. Χρησιμοποιούνται κυρίως στους τομείς της εκπαίδευσης και της υγείας, και συνήθως πρόκειται για παιχνίδια προσομοίωσης διαφόρων καταστάσεων ανάλογα με το εκπαιδευτικό αντικείμενο, ή παιχνίδια γρίφων, τα οποία συμβάλλουν στη βελτίωση της νοητικής κατάστασης του χρήστη και στην όξυνση συγκεκριμένων δεξιοτήτων του.

Επομένως, τα serious games έχουν δύο βασικά χαρακτηριστικά:

1. Έναν **μηχανισμό μάθησης** , ο οποίος ουσιαστικά αποτελεί τις τεχνικές οι οποίες χρησιμοποιούνται για να ενεργοποιηθούν οι γνωστικές ικανότητες του παίκτη ώστε να επιτευχθεί η μάθηση.

2. Το **σενάριο του παιχνιδιού**, το οποίο καλείται να αναπτυχθεί με βάση τα ενδιαφέροντα της ομάδας ατόμων στην οποία απευθύνεται το παιχνίδι, έτσι ώστε να διατηρήσει το ενδιαφέρον του παίκτη.

2.1.1 Είδη serious games.

Ο διαχωρισμός των serious games σε κατηγορίες, γίνεται με βάση τον σκοπό τον οποίο εξυπηρετούν. Τα πιο γνωστά είδη serious games είναι:

1. **Παιχνίδια εξάσκησης (Training games):** Πρόκειται συνήθως για παιχνίδια προσομοίωσης, τα οποία έχουν ως σκοπό να εκπαιδεύσουν το χρήστη για κάποιο επάγγελμα, προσομοιώνοντας τις συνθήκες εργασίας και το αντικείμενο στο οποίο καλείται να εργαστεί. Σε αυτήν την κατηγορία υπάγονται και τα εκπαιδευτικά παιχνίδια, τα οποία μπορούν να βοηθήσουν τους μαθητές να κατανοήσουν κάποια εκπαιδευτικά αντικείμενα. Τα παιχνίδια εξάσκησης είναι συνήθως χαμηλού κόστους, καθώς δίνουν προτεραιότητα σε ένα συγκεκριμένο στόχο, και απευθύνονται σε ειδικό κοινό.

2. **Παιχνίδια που ασχολούνται με την υγεία (Health games):** Τα παιχνίδια αυτά έχουν ως σκοπό την διατήρηση ή την μερική αποκατάσταση της ψυχικής ή φυσικής υγείας του χρήστη. Συνήθως, πετυχαίνουν τον στόχο τους, καθοδηγώντας τον χρήστη στην εκτέλεση των ενεργειών που θα βοηθήσουν στη θεραπεία του. Τα παιχνίδια αυτά παρέχουν στον παίκτη αρκετή ενθάρρυνση, αλλά και επιβράβευση όταν αυτός καταφέρει να κάνει το σωστό βήμα.

3. **<u>Παιχνίδια ευαισθητοποίησης σε κοινωνικά / περιβαλλοντικά ζητήματα</u>:** Πρόκειται για παιχνίδια τα οποία προσομοιώνουν κάποιο κοινωνικό ή περιβαλλοντικό ζήτημα, και ο παίκτης καλείται να εκτελέσει κάποιες ενέργειες για να το αντιμετωπίσει. Συνεπώς, ο χρήστης σκέφτεται και ευαισθητοποιείται σχετικά με το θέμα το οποίο αφορά το παιχνίδι.

2.1.2 Προϋποθέσεις που πρέπει να πληροί ένα serious game

Για να θεωρηθεί ένα παιχνίδι serious game, πρέπει να πληροί κάποιες βασικές προϋποθέσεις:

- Θα πρέπει να έχει έναν σαφή και ξεκάθαρο στόχο, πέραν της ψυχαγωγίας. Ο στόχος αυτός μπορεί είτε να σχετίζεται με την εκπαίδευση, είτε να αποσκοπεί στη βελτίωση της υγείας του χρήστη.

- Είναι απαραίτητο το παιχνίδι να σχεδιάζεται με βάση την ηλικία της ομάδας ατόμων για την οποία προορίζεται. Για παράδειγμα, όσο αναφορά το γραφικό περιβάλλον, αν το παιχνίδι προορίζεται για μικρότερα παιδιά, θα πρέπει να έχει ζωντανά χρώματα και απλές εικόνες. Για μεγαλύτερα παιδιά, καλό θα ήταν να υπάρχουν εντυπωσιακά γραφικά , ώστε να διατηρηθεί το ενδιαφέρον τους αμείωτο και να επιτευχθεί ευκολότερα ο σκοπός του παιχνιδιού. Όσο για τους ενήλικες και τους ηλικιωμένους, θα πρέπει να προτιμώνται απλούστερα γραφικά και ζεστά χρώματα.

- Κατά τον σχεδιασμό του παιχνιδιού θα πρέπει να λαμβάνονται υπόψιν και οι ειδικές ανάγκες της ομάδας ατόμων για την οποία προορίζεται. Για παράδειγμα, αν ο σκοπός του παιχνιδιού είναι η διευκόλυνση της εκπαίδευσης ατόμων με διάσπαση προσοχής, τότε το παιχνίδι θα πρέπει να παρέχει αρκετά ερεθίσματα (οπτικά ή ακουστικά) , ώστε ο χρήστης να παραμείνει συγκεντρωμένος στο παιχνίδι. Αν πάλι η εφαρμογή προορίζεται για άτομα με προβλήματα όρασης, θα πρέπει να υπάρχουν και άλλες μέθοδοι εξόδου, εκτός από την κλασσική οπτική έξοδο στην οθόνη.

- Αν το παιχνίδι είναι διαδικτυακό, τότε η πλατφόρμα του πρέπει να παρέχει όσο το δυνατόν περισσότερη διαδραστικότητα μεταξύ του επιβλέποντα (δάσκαλος, γιατρός) και του χρήστη (μαθητής, ασθενής), έτσι ώστε να εξασφαλίζεται η επίτευξη των βέλτιστων αποτελεσμάτων.

2.1.3 Πλεονεκτήματα των serious games

Τα serious games μπορούν να διευκολύνουν πολλούς τομείς της εκπαίδευσης και της υγείας, λόγω των πλεονεκτημάτων, τα οποία προσφέρουν.

- **Καθιστούν την εκπαιδευτική δραστηριότητα πιο ελκυστική στο χρήστη:** Χάρη στα διαδραστικά μέσα τα οποία χρησιμοποιούν, τα serious games καταφέρνουν να κεντρίσουν το ενδιαφέρον του χρήστη, και να το διατηρήσουν αμείωτο. Έτσι, το ποσοστό επιτυχίας της εκπαιδευτικής δραστηριότητας αυξάνεται.

- **Παρέχουν τη δυνατότητα εκπαίδευσης σε ένα ασφαλές περιβάλλον:** Μέσω των παιχνιδιών αυτών, οι χρήστες έχουν την δυνατότητα να εκπαιδευτούν σε διάφορα αντικείμενα, με μεγαλύτερη ασφάλεια, και συνεπώς να αποκτήσουν περισσότερες εμπειρίες από ότι θα αποκτούσαν εκτός προσομοιώσεως.

- **Διευκολύνουν την διαδικασία εκπαίδευσης για ομάδες ατόμων με ειδικές ανάγκες:** Η σύγχρονη τεχνολογία διευκολύνει την χρήση των υπολογιστών από άτομα με ειδικές ανάγκες. Συνεπώς, η χρήση των πολυμέσων στην εκπαίδευση καθιστά πιο εύκολη την διδασκαλία προς άτομα που διαφορετικά θα αντιμετώπιζαν δυσκολίες.

- **Δυνατότητα εκπαίδευσης σε πολλά διαφορετικά αντικείμενα:** Με την χρήση των serious games, είναι δυνατόν να προσομοιωθεί οποιαδήποτε δραστηριότητα, χωρίς να είναι απαραίτητη η κατασκευή ειδικών εγκαταστάσεων. Επομένως οι εκπαιδευόμενοι μπορούν να εξασκηθούν σε οποιοδήποτε αντικείμενο επιθυμούν, στο χώρο μιας αίθουσας, ενώ το μόνο υλικό που απαιτείται είναι ένας υπολογιστής.

- **Το κόστος ανάπτυξης ενός serious game:** Πρέπει να είναι σημαντικά μικρότερο από το κόστος των εγκαταστάσεων και των υλικών που θα χρειαζόταν για την εκπαίδευση του χρήστη με τον συμβατικό τρόπο.

2.2 Διαδικασία σχεδιασμού και ανάπτυξης ενός serious game.

Η διαδικασία ανάπτυξης ενός serious game, είναι παρόμοια με αυτήν ενός απλού παιχνιδιού. Η μόνη διαφορά είναι ότι, καθώς ένα serious game έχει έναν ειδικό σκοπό, απαιτείται μελέτη και έρευνα του αντικειμένου με το οποίο σχετίζεται ο σκοπός του παιχνιδιού. Με άλλα λόγια, η διαδικασία για τον σχεδιασμό και την ανάπτυξη ενός serious game σε βήματα είναι η εξής:

1. **Καθορισμός του σκοπού του παιχνιδιού**. Όπως αναφέρθηκε παραπάνω, για να θεωρηθεί ένα παιχνίδι ως serious game, θα πρέπει να έχει έναν συγκεκριμένο σκοπό. Ο σκοπός του παιχνιδιού λοιπόν είναι απαραίτητο να έχει αποφασιστεί πριν την εκκίνηση του σχεδιασμού και της ανάπτυξης της εφαρμογής. Αφού αποφασιστεί ο σκοπός του παιχνιδιού, θα πρέπει να γίνει μελέτη σχετικά με το αντικείμενο του σκοπού του παιχνιδιού, ώστε να γίνει ένας σωστός σχεδιασμός.

2. **Καθορισμός του μηχανισμού μάθησης που θα χρησιμοποιηθεί.** Ο μηχανισμός μάθησης καθορίζεται ανάλογα με τον σκοπό του παιχνιδιού, και την ομάδα των χρηστών στην οποία απευθύνεται το παιχνίδι.

3. **Καθορισμός της πλατφόρμας και της γλώσσας προγραμματισμού που θα χρησιμοποιηθούν.** Για να αποφασιστεί σε ποια πλατφόρμα πρέπει να αναπτυχθεί το παιχνίδι, πρέπει πρώτα να ληφθούν υπόψιν κάποιοι σημαντικοί παράγοντες, όπως ο τύπος του παιχνιδιού (αν δηλαδή θα είναι στατικό, δισδιάστατο ή τρισδιάστατο), καθώς και σε ποιες συσκευές επιθυμούμε να λειτουργεί. Επίσης, πρέπει να σκεφτούμε αν θα χρησιμοποιηθούν και άλλα περιφερειακά μέσα για τη λειτουργία της εφαρμογής (αισθητήρες κλπ.).

4. **Σχεδιασμός Storyboard (προσχεδίου) για το παιχνίδι.** Πριν ξεκινήσει η ανάπτυξη του παιχνιδιού, είναι απαραίτητο να γίνει πρώτα ο σχεδιασμός του σεναρίου και των υπόλοιπων χαρακτηριστικών του. Με αυτόν τον τρόπο διευκολύνεται κατά μεγάλο βαθμό η ανάπτυξη της εφαρμογής, και μας δίνεται η δυνατότητα να εντοπίσουμε σφάλματα αλλά και να προσθέσουμε περισσότερα ή να αφαιρέσουμε περιττά στοιχεία από το παιχνίδι. Με άλλα λόγια, το Storyboard αποτελεί τον σχεδιασμό του βασικού σκελετού της εφαρμογής, και μας καθοδηγεί καθ' όλη την διάρκεια της ανάπτυξης της. Μια επιτυχημένη εφαρμογή χρειάζεται ένα σωστά δομημένο προσχέδιο. Στον πίνακα 3.1 που βρίσκεται παρακάτω, αναφέρονται ορισμένα στοιχεία τα οποία πρέπει να αποτελούν τη δομή του προσχεδίου (storyboard).

5. **Ανάπτυξη κώδικα για την εφαρμογή.** Το τελικό, αλλά και το πιο σημαντικό βήμα για την ολοκλήρωση της ανάπτυξης ενός παιχνιδιού είναι η γραφή κώδικα. Ο κώδικας πρέπει να είναι ευανάγνωστος και λειτουργικός, έτσι ώστε να είναι δυνατές τυχόν αλλαγές που μπορεί να προκύψουν.

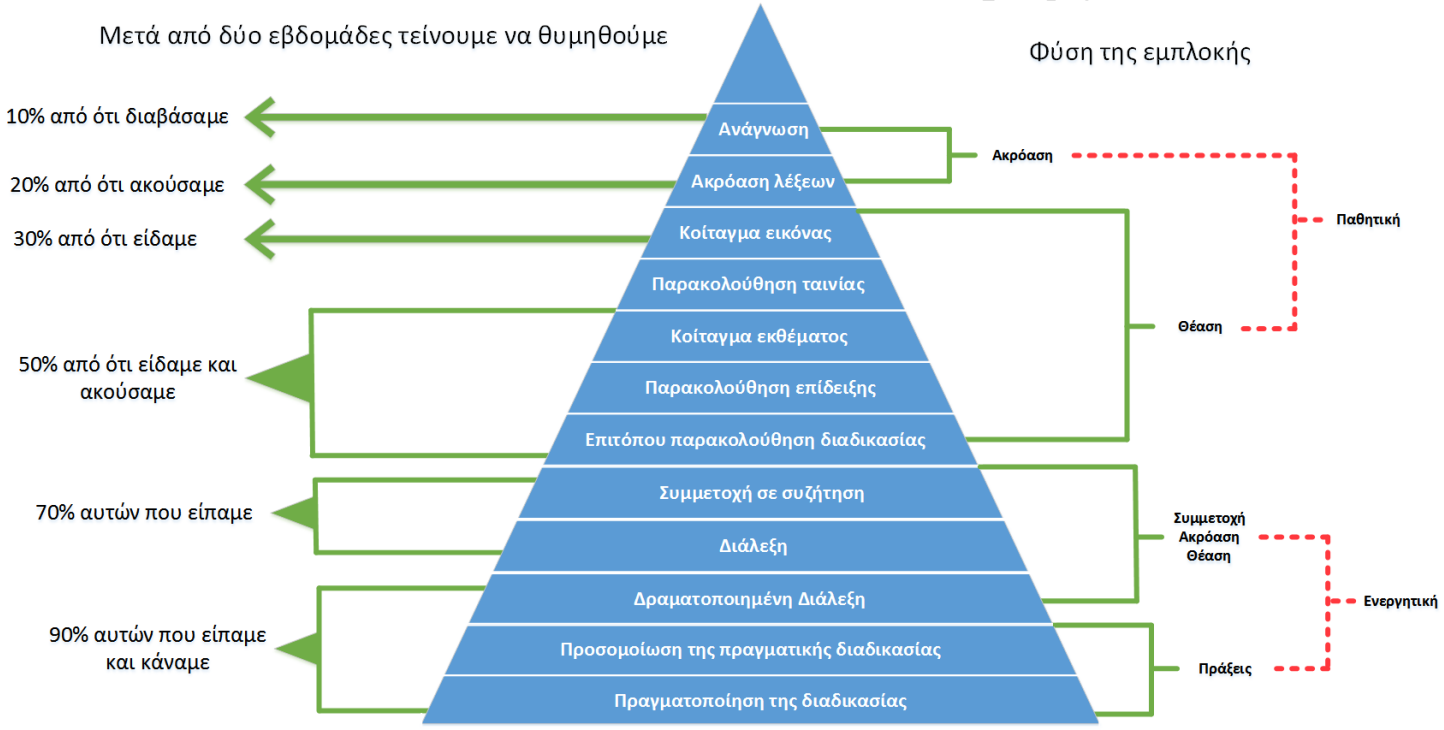
Στοιχεία που περιέχονται σε ένα προσχέδιο (storyboard)	
Κύριος στόχος	Κάθε serious game έχει έναν συγκεκριμένο στόχο, ο οποίος θα πρέπει να είναι ξεκάθαρος προτού ξεκινήσει η ανάπτυξη του παιχνιδιού.
Γενική περιγραφή	Η γενική περιγραφή αποτελεί μια «σύνοψη» του παιχνιδιού. Περιέχει γενικές πληροφορίες για τις ενέργειες που πρέπει να εκτελέσει ο χρήστης, ώστε να φτάσει τελικά στον στόχο του παιχνιδιού.
Προσδιορισμός επιπέδων (level)	Σε αυτό το σημείο, προσδιορίζεται ο αριθμός των επιπέδων που θα περιέχει το παιχνίδι, ο βαθμός δυσκολίας κάθε επιπέδου, και προαιρετικά κάποια επιπλέον στοιχεία για κάθε επίπεδο ξεχωριστά.
Χαρακτήρες παιχνιδιού (προαιρετικά)	Στο προσχέδιο, πρέπει να καθοριστεί η ύπαρξη ή όχι χαρακτήρων στο παιχνίδι, ο αριθμός τους (αν τελικά υπάρχουν), κάποιο όνομα και μια μικρή περιγραφή για τον καθένα.
Ξεχωριστή περιγραφή κάθε σκηνής	Πρόκειται για μια σύντομη περιγραφή των λειτουργειών που περιέχονται σε κάθε σκηνή του επιπέδου.
Οπτικό προσχέδιο κάθε σκηνής	Για κάθε σκηνή που διαφοροποιείται από τις υπόλοιπες, θα πρέπει να υπάρχει ένα προσχέδιο στο οποίο θα καθορίζονται οι θέσεις των οπτικών στοιχείων κάθε σκηνής (κουμπιά, textboxes, θέσεις εικόνων κλπ.).
Διαγράμματα περιπτώσεων (use case diagrams)	Το σημαντικότερο στοιχείο που πρέπει να βρίσκεται σε κάθε storyboard. Το διάγραμμα περιπτώσεων περιέχει πληροφορίες για την εξέλιξη του παιχνιδιού μετά από οποιαδήποτε ενέργεια του χρήστη. Σε αυτό καθορίζεται τι θα συμβεί αν για παράδειγμα ο χρήστης εισάγει λάθος χαρακτήρες στο πεδίο κειμένου, αν δώσει λανθασμένη ή σωστή απάντηση, πόσες φορές δικαιούται να εισάγει λανθασμένη απάντηση κλπ.

Πίνακας 2.1:Δομικά στοιχεία ενός storyboard

2.3 *Serious games και θεωρίες μάθησης.*

Όπως αναφέρθηκε παραπάνω, τα serious games είναι παιχνίδια τα οποία πέραν της ψυχαγωγίας, έχουν και κάποιον ειδικό σκοπό, ο οποίος συνήθως είναι εκπαιδευτικός. Συνεπώς, ο τελικός στόχος ενός serious game, είναι να αφομοιώσουν οι μαθητές όσα διδάχθηκαν και να είναι σε θέση να τα θυμούνται για όσο το δυνατόν μεγαλύτερο χρονικό διάστημα. Για αυτόν τον λόγο άλλωστε περιέχουν και τον μηχανισμό μάθησης. Το ερώτημα που προκύπτει όμως είναι, ποιος μηχανισμός μάθησης είναι ο καταλληλότερος για να πετύχει ο σκοπός του παιχνιδιού;

Σύμφωνα με τον κώνο της εμπειρίας του Dale (εικόνα 2.1), ο άνθρωπος έχει την δυνατότητα να θυμηθεί το 90% αυτών που έκανε κατά τη διάρκεια μιας διαδικασίας, η της προσομοίωσης αυτής, αφού περάσει ένα διάστημα δυο εβδομάδων από την στιγμή που πραγματοποιήθηκε η διαδικασία ή η προσομοίωση.



Εικόνα 2.1 Ο κώνος της εμπειρίας του Dale

Για να πετύχει λοιπόν τον εκπαιδευτικό του σκοπό με όσο το δυνατόν μεγαλύτερο ποσοστό επιτυχίας, κάθε serious game διαθέτει έναν μηχανισμό μάθησης, ο οποίος βασίζεται σε μια θεωρία μάθησης, η οποία καθορίζεται από παράγοντες όπως το είδος και ο σκοπός του παιχνιδιού. Οι τέσσερεις βασικές θεωρίες μάθησης που χρησιμοποιούνται ως βάση για την ανάπτυξη του μηχανισμού μάθησης των serious games είναι οι εξής:

1. Η μπιχεβιοριστική ή συμπεριφοριστική θεωρία
2. Η ανακαλυπτική θεωρία
3. Ο εποικοδομισμός
4. Η γνωσιοκρατία

2.3.1 Η μπιχεβιοριστική ή συμπεριφοριστική θεωρία

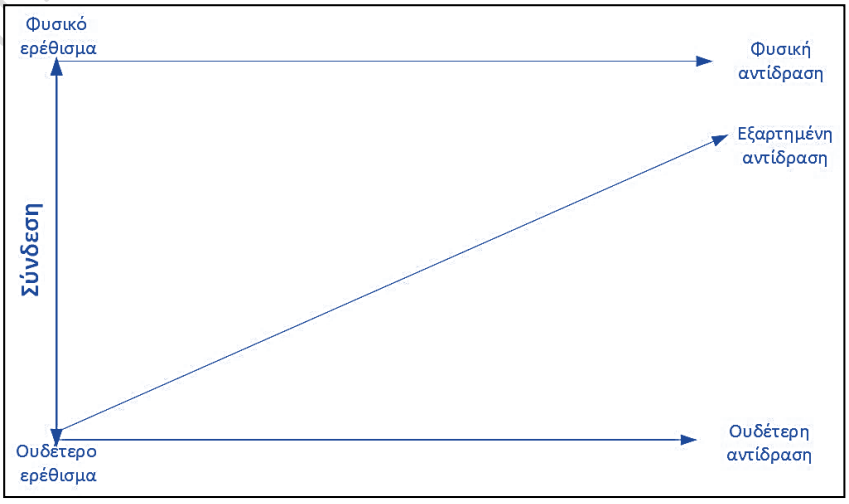
Η μπιχεβιοριστική (ή συμπεριφοριστική) θεωρία υποστηρίζει ότι ο βαθμός και ο ρυθμός μάθησης ενός ατόμου καθορίζεται από εξωτερικούς παράγοντες. Με άλλα λόγια, η ανθρώπινη συμπεριφορά διαμορφώνεται και ελέγχεται από περιβαλλοντικούς παράγοντες, και συνεπώς ο οργανισμός θεωρείται ως εξαρτημένη μεταβλητή των περιβαλλοντικών συνθηκών.

Σύμφωνα με τον κύριο εκφραστή της μπιχεβιοριστικής θεωρίας, τον Pawlow, η μάθηση είναι επιτυχής, όταν επιτευχθεί η συσχέτιση ενός ουδέτερου ερεθίσματος, με κάποια αντίδραση εκ' μέρους του εκπαιδευόμενου. Για να επιτευχθεί η μάθηση, συνήθως προαπαιτείται μια διαδικασία. Αρχικά, ο εκπαιδευόμενος δέχεται φυσικά ερεθίσματα, τα οποία σίγουρα προκαλούν αντίδραση. Στη συνέχεια, τα ερεθίσματα ουδετεροποιούνται σταδιακά, ώσπου να γίνουν τελείως ουδέτερα. Αν υπάρξει η επιθυμητή αντίδρασή στο ουδέτερο ερέθισμα, ο παίκτης επιβραβεύεται. Όταν πλέον προκληθεί αντίδραση αντίστοιχη με την αρχική σε ουδέτερο ερέθισμα, τότε έχει επιτευχθεί και η μάθηση. Αυτή η διαδικασία μπορεί να αναπαρασταθεί και σχηματικά, όπως βλέπουμε στην εικόνα 2.2.

Κατά την μπιχεβιοριστική θεωρία, είναι δυνατό να καταλήξουμε σε συμπεράσματα σχετικά με την μάθηση, παρατηρώντας την συμπεριφορά των εκπαιδευόμενων. Αυτό συμβαίνει διότι η συμπεριφορά του ατόμου μεταβάλλεται λόγω της απόκτησης εμπειριών του.

Το συμπέρασμα που προκύπτει είναι ότι κατά τον μπιχεβιορισμό, το γνωστικό αντικείμενο πρέπει να συσχετιστεί με τις αισθήσεις του εκπαιδευόμενου για να επιτευχθεί η μάθηση.

Η συμπεριφοριστική θεωρία χρησιμοποιείται κυρίως σε serious games τα οποία έχουν την μορφή τεστ. Για να έχει το επιθυμητό αποτέλεσμα ένα παιχνίδι του οποίου ο μηχανισμός μάθησης έχει ως βάση τη συγκεκριμένη θεωρία, θα πρέπει η χρήση του να επαναλαμβάνεται ανά τακτά χρονικά διαστήματα, ώστε να ελέγχεται το ποσοστό της πληροφορίας την οποία αφομοίωσε ο χρήστης.



Εικόνα 2.2: Σχηματική αναπαράσταση της διαδικασίας μάθησης, σύμφωνα με τον μπεχίβιορισμό

2.3.2 Η ανακαλυπτική θεωρία

Η ανακαλυπική θεωρία υποστηρίζει ότι το υποκείμενο (δηλαδή ο εκπαιδευόμενος), θα πρέπει να αποκτήσει την γνώση εξάγοντας τα δικά του συμπεράσματα, αφού δράσει σε συγκεκριμένα αντικείμενα. Με άλλα λόγια, ο εκπαιδευόμενος καλείται να ανακαλύψει τη γνώση, και να κατανοήσει το αντικείμενο μετά από δική του προσπάθεια. Αυτή η μέθοδος έχει μεγαλύτερο ποσοστό επιτυχίας όταν εφαρμόζεται σε ομάδες ατόμων, και όχι μεμονωμένα, καθώς οι εκπαιδευόμενοι μπορούν να ανταλλάξουν απόψεις σχετικά με τα συμπεράσματά τους. Το γεγονός αυτό, καθιστά αυτήν την μέθοδο ως την πιο κατάλληλη για ανάπτυξη serious games, τα οποία απευθύνονται σε σχολεία, καθώς επιτρέπουν σε όλους τους μαθητές της τάξης να συμμετέχουν ταυτόχρονα στην εκπαιδευτική διδασκαλία, αλλά και ενθαρρύνουν την συνεργασία μεταξύ των μαθητών και των δασκάλων / καθηγητών ως ομάδα.

Στην ανακαλυπτική θεωρία στηρίζονται συνήθως εκπαιδευτικά παιχνίδια, τα οποία απευθύνονται σε μαθητές σχολείου, και χρησιμοποιούνται κατά κύριο λόγο μέσα στη σχολική αίθουσα. Η πιο συχνή χρήση της ανακαλυπτικής θεωρίας συναντάται στα εκπαιδευτικά παιχνίδια τα οποία έχουν ως σκοπό να εκπαιδεύσουν τους μαθητές σχετικά με τα φυσικά φαινόμενα. Παραδείγματα τέτοιων serious games είναι τα παιχνίδια τα οποία αφορούν την κατανόηση των νόμων της Φυσικής, των Μαθηματικών εννοιών, των καιρικών φαινομένων κλπ.

2.3.3 Ο εποικοδομητισμός

Σύμφωνα με την θεωρία του εποικοδομητισμού, ο άνθρωπος μπορεί να οικοδομήσει τη γνώση σταδιακά, αλληλοεπιδρώντας με τα δεδομένα στο υπόβαθρο των εμπειριών του. Με άλλα λόγια, η μάθηση επιτυγχάνεται καθώς το άτομο εξασκείται σε προσομοιώσεις καταστάσεων, αποκτώντας εμπειρία, έτσι ώστε να είναι σε θέση να αντιμετωπίσει παρόμοιες καταστάσεις στην πραγματική ζωή.

Τα «σοβαρά» παιχνίδια τα οποία στηρίζονται στην θεωρία του εποικοδομητισμού, είναι συνήθως παιχνίδια προσομοίωσης. Ανάλογα με το σενάριο του παιχνιδιού, μπορούν να εκπαιδεύσουν τον χρήστη σε τομείς όπως:

1. **Τον επαγγελματικό τομέα:** Κάποιες εταιρίες χρησιμοποιούν παιχνίδια προσομοίωσης, ώστε να εκπαιδεύσουν τους εργαζομένους σχετικά με το επάγγελμα. Αυτό συμβαίνει διότι τα παιχνίδια αυτά παρέχουν συνθήκες ασφάλειας, οπότε οποιοσδήποτε μπορεί να εκπαιδευτεί χωρίς να διατρέχει κανέναν απολύτως κίνδυνο.

2. **Τον κοινωνικό τομέα:** Μέσω αυτού του είδους παιχνιδιών, ο χρήστης μπορεί να αναπτύξει τις κοινωνικές του δεξιότητες ακόμα και στο περιβάλλον του σπιτιού του. Τα serious games αυτού του είδους είναι ιδανικά για την

εκπαίδευση των παιδιών σε κοινωνικές δεξιότητες, η για την ομαλή επανένταξη ατόμων με ψυχικές διαταραχές στην κοινωνία.

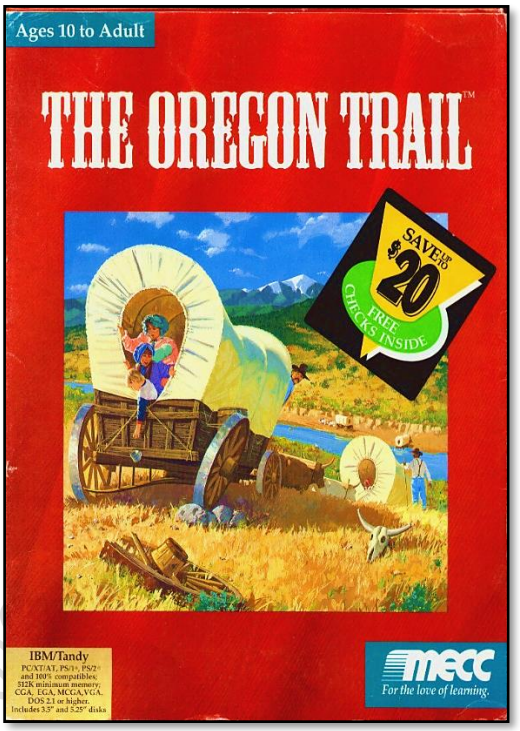
3. **Την κατανόηση διάφορων σχολικών μαθημάτων:** Στην εικόνα 2.1, είδαμε ότι στον κώνο εμπειρίας του Dale, δηλώνεται ότι είμαστε ικανοί να θυμηθούμε το 90% αυτών που είπαμε και κάναμε κατά τη διάρκεια μιας πράξης ή μιας προσομοίωσης, μετά την πάροδο δυο εβδομάδων από τη στιγμή που πραγματοποιήθηκε η δραστηριότητα. Το γεγονός αυτό μπορεί να χρησιμοποιηθεί προς όφελος της εκπαιδευτικής διαδικασίας, όσο αναφορά την διδασκαλία ορισμένων μαθημάτων στα σχολεία. Για παράδειγμα, για το μάθημα της ιστορίας, ένα παιχνίδι προσομοίωσης το οποίο θα επέτρεπε στους μαθητές να παίξουν τον ρόλο του κύριου ιστορικού χαρακτήρα, θα συνέβαλλε στην αποτελεσματικότερη αφομοίωση των ιστορικών γεγονότων. Επιπλέον, οι μαθητές θα είχαν μεγαλύτερο κίνητρο να ασχοληθούν με την εκμάθηση των ιστορικών γεγονότων, αφού τα serious games συνδυάζουν την μάθηση με την ψυχαγωγία.

2.3.4 Η γνωσιοκρατία

Η γνωσιοκρατία είναι η θεωρία μάθησης, η οποία εξετάζει τον τρόπο με τον οποίο ο ανθρώπινος εγκέφαλος αποθηκεύει την καινούρια πληροφορία, την συνδυάζει με την ήδη υπάρχουσα, και χρησιμοποιεί αυτήν την συνδυασμένη πληροφορία σε καταστάσεις στις οποίες κρίνεται αναγκαία. Η συγκεκριμένη θεωρία, δεν θεωρεί την συμπεριφορά του ατόμου σημαντικό παράγοντα στον τομέα της μάθησης, αλλά δίνει μεγαλύτερη έμφαση στην εξασφάλιση συνθηκών, οι οποίες θα είναι ευνοϊκές προς τη διαδικασία της μάθησης.

2.4 Ιστορική αναδρομή

Η χρήση των παιχνιδιών στον τομέα της εκπαίδευσης έγινε δημοφιλής κατά τον 20ο αιώνα (συγκεκριμένα, κατά το έτος 2002). Όμως, η αρχή της χρήσης παιχνιδιών για εκπαιδευτικούς σκοπούς τοποθετείται γύρω στο 1960 με 1970, οπότε και ξεκίνησαν να χρησιμοποιούνται στις τάξεις εκπαιδευτικά παιχνίδια, κατασκευασμένα από χαρτί. Η χρήση των παιχνιδιών αυτών είχε ως σκοπό να βοηθήσει τους μαθητές να βελτιώσουν τις επιδόσεις τους στα μαθήματα της ανάγνωσης, της γραφής και της αριθμητικής. Αυτά τα παιχνίδια αποτέλεσαν την αρχική μορφή των serious games.



Εικόνα 2.3: Το παιχνίδι "Oregon Trail"

Ο πιο γνωστός και αξιοσημείωτος «πρόγονος» των serious games, ήταν ένα ηλεκτρονικό παιχνίδι το οποίο ονομαζόταν «The Oregon Trail». Δημιουργήθηκε το 1971 στην Μινεσότα της Αμερικής από τρείς καθηγητές Ιστορίας, τους Don Rawitsch, Bill Heinemann, και Paul Dillenberger.

Στόχος του ήταν να διδάξει τους μαθητές σχετικά με την ζωή των προσκυνητών (pilgrims) του 19ου αιώνα. Το παιχνίδι έγινε δημοφιλές πολύ σύντομα, και τελικά, το 1978 εκδόθηκε σε open-source μορφή. Το παιχνίδι αυτό έθεσε τα θεμέλια για την εξέλιξη των serious games, στην μορφή την οποία γνωρίζουμε σήμερα.

Η ανάπτυξη των «σοβαρών παιχνιδιών» (serious games), με την μορφή την οποία γνωρίζουμε σήμερα, ξεκίνησε κατά την δεκαετία του 2000, όταν δηλαδή άρχισαν να εξελίσσονται σημαντικά η σύγχρονη τεχνολογία, τα σύγχρονα πολυμέσα και το διαδίκτυο.

3.1 Η γλώσσα C#

Η C# είναι μια πολύ-πρότυπη, εξελίξιμη και ολοκληρωμένη γλώσσα προγραμματισμού, βασιζόμενη σε αντικειμενοστραφείς αρχές, η οποία παρέχει ασφάλεια των τύπων της. Δημιουργήθηκε από τη Microsoft, μέσω της πλατφόρμας .NET. Αργότερα αναγνωρίστηκε επισήμως από την Ecma (ECMA-334) και την ISO (ISO/IEC 2327:2006). Είναι μια από τις γλώσσες προγραμματισμού που δημιουργήθηκαν για την Common Language Infrastructure.

Η C# είναι μια διαδεδομένη γλώσσα προγραμματισμού, καθώς είναι απλή, και παρέχει άμεση πρόσβαση σε τεράστιες βιβλιοθήκες της .NET. Το γεγονός αυτό την καθιστά ιδανική γλώσσα για ανάπτυξη εφαρμογών στο Visual Studio, καθώς και εκεί χρησιμοποιούνται .NET βιβλιοθήκες.

Χρησιμοποιώντας την C#, μπορούμε να δημιουργήσουμε μεγάλο πλήθος εφαρμογών, όπως:

- Εφαρμογές για κινητά (windows phone)
- Εφαρμογές για tablet (windows store)
- Desktop εφαρμογές (WPF, EXE)
- Εφαρμογές για το διαδίκτυο (asp.Net , WPF BrowserApplication)

3.1.1 Τα χαρακτηριστικά της C#

Η C# είναι μια δημοφιλής γλώσσα προγραμματισμού. Αυτό το οφείλει στα χαρακτηριστικά της. Η C# χαρακτηρίζεται από:

1. **Απλότητα**

   Η C# είναι μια απλή και εύχρηστη γλώσσα προγραμματισμού, η οποία δεν απαιτεί πολύ χρόνο για την εκμάθηση της. Επίσης, έχει σχεδιαστεί έτσι ώστε να μειώνεται η πολυπλοκότητα του κώδικα, αφού η C# έχει απλουστευμένο συντακτικό.

2. **Αντικειμενοστρέφεια**

   Η C# σχεδιάστηκε για να είναι αντικειμενοστραφής. Γι' αυτό και είναι τόσο δημοφιλής. Δυο βασικά χαρακτηριστικά του αντικειμενοστραφούς προγραμματισμού είναι η ενθυλάκωση και η ανταλλαγή μηνυμάτων. Αυτά τα δυο χαρακτηριστικά καλύπτουν την ανάγκη για την ύπαρξη κατανεμημένων συστημάτων πελάτη – εξυπηρετητή. Η C# παρέχει μια ξεκάθαρη και αποδοτική αντικειμενοστραφή πλατφόρμα παρέχοντας στους προγραμματιστές μια συλλογή βιβλιοθηκών δοκιμασμένων αντικειμένων που παρέχουν λειτουργικότητα που ποικίλει από απλούς τύπους δεδομένων, σε διεπαφές εισόδου/εξόδου ή δικτυακές και εργαλεία για τη δημιουργία παραθυρικών

εφαρμογών. Αυτές οι βιβλιοθήκες μπορούν να προσαρμοστούν στις ανάγκες του προγραμματιστή.

Άλλο ένα προτέρημα της C#, είναι το γεγονός ότι υποστηρίζει τον προγραμματισμό βασιζόμενο σε components (component-based programming), ο οποίος υποστηρίζει τον προσδιορισμό αυτόνομων μονάδων λειτουργικότητας. Κάθε component, διαθέτει τα δικά του χαρακτηριστικά, όπως ιδιότητες, μεθόδους, events και μεταδεδομένα.

3. **Οικειότητα**

Η C# έχει αρκετά χαρακτηριστικά της C++ και της Java, χωρίς όμως την πολυπλοκότητά τους. Δεδομένου ότι οι αυτές οι γλώσσες προγραμματισμού είναι οι περισσότερο διδασκόμενες, η μετάβαση στην χρήση της C# είναι εύκολη για κάθε σχεδόν προγραμματιστή.

3.1.2 Οι εκδόσεις της C#

1. **Έκδοση 1.0**

- ✓ Η πρώτη έκδοση της C# ήταν η έκδοση 1.0. Η έκδοση αυτή αναπτύχθηκε το 2000. Αποτέλεσε την βάση για την ανάπτυξη των επόμενων εκδόσεων.

2. **Έκδοση 2.0**

- ✓ Σε αυτήν την έκδοση, ουσιαστικότερη προσθήκη ήταν η υποστήριξη των generics. Το συντακτικό ήταν παρόμοιο με τις C++ templates, αλλά η κύρια διαφορά είναι ότι οι τύποι που δημιουργούνται από τα generics είναι δυναμικοί στη φύση τους, δηλαδή κατασκευάζονται κατά την εκτέλεση, σε αντίθεση με άλλες γλώσσες προγραμματισμού, όπως τη C, που τους κατασκευάζουν κατά τη μεταγλώττιση.

- ✓ Στη C# 2.0 προστέθηκε και η υποστήριξη των anonymous μεθόδων. Μια anonymous μέθοδος, η οποία μερικές φορές καλείται και lambda function είναι αρκετά χρήσιμη στη διαχείριση των events.

- ✓ Μια άλλη σημαντική προσθήκη αποτέλεσε η υποστήριξη των iterator blocks. Τέτοιες δομές κώδικα χρησιμεύουν στην εύκολη προσπέλαση ενός αντικειμένου που συμπεριφέρεται σαν collection. Τέτοια αντικείμενα πρέπει να υλοποιούν το IEnumerable interface το οποίο περιλαμβάνει την μέθοδο GetEnumerator.

- ✓ Τέλος ,προστέθηκε η υποστήριξη των partial types. Στις προηγούμενες εκδόσεις κάθε κλάση ήταν υποχρεωτικό να ορίζεται στο δικό της αρχείο

(το οποίο ονομαζόταν compilation unit). Η απαίτηση αυτή χαλάρωσε με την εισαγωγή των partial τύπων που διευκολύνουν την επέκταση χρήσιμων τύπων δεδομένων που παράγονται αυτόματα από οδηγούς όπως για παράδειγμα κατά τη δημιουργία Windows Forms.

3. Έκδοση 3.0

- ✓ Σε αυτήν την έκδοση έγιναν πολλές σημαντικές προσθήκες και αλλαγές. Αρχικά, έγινε εφικτή η χρήση μεταβλητών τύπου var. Οι μεταβλητές var έχουν τύπο δεδομένων, ο οποίος προσδιορίζεται κατά την εκτέλεση του κώδικα, και όχι κατά τη μεταγλώττιση, όπως συμβαίνει στους υπόλοιπους τύπους μεταβλητών.

- ✓ Επίσης, έγινε εφικτή η αυτόματη δημιουργία των properties μιας κλάσης, συνεπώς μειώθηκαν και τα τυχόν σφάλματα που προκύπταν από τη χειροκίνητη δημιουργία τους.

- ✓ Το γεγονός αυτό, σε συνδυασμό με την προσθήκη νέων χαρακτηριστικών που βοηθούν στην αρχικοποίηση νέων αντικειμένων με μια σύνθετη πρόταση, αποτέλεσε μεγάλη διευκόλυνση για τους προγραμματιστές που χρησιμοποιούν τη C#.

- ✓ Στη C# 3.0 εισήχθησαν οι partial methods σαν επέκταση των partial κλάσεων της C# 2.0. Έτσι λοιπόν είναι δυνατόν να δηλωθεί η signature μιας μεθόδου και να αφήνεται στο χρήστη-προγραμματιστή η επιλογή της υλοποίησής της.

- ✓ Οι extension methods είναι άλλη μια χρήσιμη προσθήκη στη C# 3.0. Οι extension methods επιτρέπουν την προσθήκη μεθόδων σε υπάρχοντες τύπους δεδομένων χωρίς τη δημιουργία νέων τύπων δεδομένων (μέσω της κληρονομικότητας) ή την ανάγκη μεταγλώττισης και κατά συνέπεια αλλαγής του αρχικού τύπου δεδομένων. Οι extension methods είναι μια ειδική περίπτωση μιας static μεθόδου, αλλά καλούνται σαν να ήταν μέθοδοι στιγμιοτύπου.

- ✓ Ακόμα πιο αξιοπρόσεκτη είναι η υποστήριξη των lambda expressions. Μια lambda expression είναι μια anonymous function που μπορεί να περιέχει expressions και statements που μπορούν να χρησιμοποιηθούν για τη δημιουργία delegates ή τύπους expression tree.

- ✓ Τέλος η σημαντικότερη προσθήκη στη C# 3.0, είναι η LINQ που βασίζεται πάνω στα προηγούμενα χαρακτηριστικά και κυριότερα στις extension methods, τις lambda expressions και τους anonymous τύπους. Εισάγει νέες

δεσμευμένες λέξεις για το γράψιμο προτάσεων-ερωτήσεων (queries), συνενώνοντας έτσι τον αντικειμενοστρεφή κόσμο με τον κόσμο των δεδομένων. Η LINQ χρησιμοποιείται για την υποβολή ερωτήσεων με τον ίδιο τρόπο σε πολλαπλές και συχνά ετερογενείς πηγές δεδομένων, όπως σε βάσεις δεδομένων, XML εγγράφων ή ακόμα και συλλογές αντικειμένων, ενώ προστίθενται συνεχώς και η δυνατότητα υποβολής ερωτημάτων σε νέες πηγές δεδομένων όπως το Document Object Model.

4. **Έκδοση 4.0**

- ✓ Αποτελεί την πιο σύγχρονη έκδοση της C#, την οποία χρησιμοποιούμε μέχρι και σήμερα. Σε αυτήν την έκδοση, δόθηκε ιδιαίτερη σημασία στην εξασφάλιση της διαλειτουργικότητας.

- ✓ Αρχικά, έγινε η προσθήκη του τύπου dynamic που επιτρέπει την παράκαμψη του ελέγχου τύπου κατά τη μεταγλώττιση, διευκολύνοντας έτσι την πρόσβαση σε COM APIs όπως τα APIs για το Office.

- ✓ Άλλο ένα νέο χαρακτηριστικό αυτής της έκδοσης είναι τα named arguments, η δυνατότητα δηλαδή περάσματος παραμέτρων χωρίς να παίζει ρόλο η σειρά εμφάνισής τους.

- ✓ Συχνά οι παράμετροι μιας μεθόδου είναι προαιρετικές. Έτσι με τη χρήση των named παραμέτρων είναι δυνατόν να επιλεχθούν ποιες από τις παραμέτρους θα χρησιμοποιηθούν.

- ✓ Τέλος εισήχθησαν τα χαρακτηριστικά της covariance και contravariance τα οποία επιτρέπουν την έμμεση μετατροπή αναφορών για παραμέτρους τύπου πινάκων, delegates και generics. Με την covariance διατηρούμε την συμβατότητα ανάθεσης ενώ με την contravariance την αντιστρέφουμε.

3.1.3 Δομή του κώδικα σε ένα πρόγραμμα της C#

Όπως συμβαίνει και σε κάθε άλλη γλώσσα προγραμματισμού, η C# έχει ορισμένους κανόνες, όσο αναφορά τη δομή του κώδικα της.

```
1  using System;
2   class Myprogram {
3     static void Main(String[] args) {
4        Console.WriteLine("Hello World!!!");
5      }
6    }
```

Εικόνα 3.1: Παράδειγμα κώδικα ενός απλού προγράμματος σε c#

- Στην εικόνα 3.1 της σελίδας 39 βλέπουμε ένα παράδειγμα κώδικα ενός απλού προγράμματος στην C#. Όπως φαίνεται στην εικόνα 3.1, στην πρώτη γραμμή του κώδικα ενός προγράμματος στην C#, ορίζεται ότι το πρόγραμμα χρησιμοποιεί το namespace “System”.

- Στη C# ένα namespace διατηρεί ένα σύνολο ονομάτων ξεχωριστά από ένα άλλο. Κατά συνέπεια τα ονόματα που ορίζονται στο ένα namespace δεν θα συγχέονται με τα ονόματα που ορίζονται σε ένα άλλο namespace, οπότε στην περίπτωση που είναι ίδια τα ξεχωρίζουμε με το namespace. Το namespace System είναι ένα δεσμευμένο namespace για αντικείμενα της βιβλιοθήκης κλάσεων του .NET. Η δεσμευμένη λέξη using απλώς ορίζει ότι θα χρησιμοποιηθεί το namespace που ακολουθεί.

- Στη δεύτερη γραμμή του κώδικα, ορίζεται το όνομα της κλάσης, το οποίο στην περίπτωσή μας είναι “Myprogram”. Η δεσμευμένη λέξη class, υποδηλώνει την δημιουργία μιας νέας κλάσης. Κάθε κλάση περιέχει μια ομάδα από μεταβλητές και μεθόδους, οι οποίες περιγράφουν τις ιδιότητες και τις συμπεριφορές του αντικειμένου. Τα περιεχόμενα μιας κλάσης, τα οποία εσωκλείονται σε αγκύλες, ονομάζονται μέλη της κλάσης.

- Στην τρίτη γραμμή, ξεκινάει η μέθοδος Main. Η μέθοδος Main είναι η κύρια μέθοδος, και καλείται πρώτη σε κάθε εφαρμογή της C#, αφού σηματοδοτεί την έναρξη του προγράμματος .

- Μια μέθοδος είναι η βασική μονάδα ομαδοποίησης εντολών και αντιστοιχεί στη συμπεριφορά μιας αφηρημένης έννοιας. Η παράμετρος της μεθόδου String[] args περιέχει τα ορίσματα γραμμών εντολών (command line arguments). Η δεσμευμένη λέξη static προσδιορίζει ότι η μέθοδος Main μπορεί να κληθεί χωρίς την αρχικοποίηση της κλάσης. Οι περισσότερες μέθοδοι στη C# λειτουργούν μέσα σε αντικείμενα και οι static μέθοδοι δεν είναι συχνές σε μεγάλα προγράμματα. Ωστόσο η main πρέπει να είναι πάντα static γιατί τρέχει πριν το πρόγραμμα μπορέσει να δημιουργήσει αντικείμενα.

- Οι εντολές που βρίσκονται ενδιάμεσα στις αγκύλες της μεθόδου main, αποτελούν το κύριο μέρος της main. Οι εντολές αυτές εκτελούνται ξεχωριστά, μια προς μια, και χωρίζονται μεταξύ τους με semicolons (ελληνικά ερωτηματικά).

- Στο παράδειγμά μας, το κύριο μέρος αποτελείται από μια και μόνο εντολή, η οποία είναι Console.Writeline(“Hello World !!!”).

- Το περιεχόμενο της παρένθεσης βρίσκεται εντός εισαγωγικών, καθώς αποτελεί συμβολοσειρά (String). Η εντολή Writeline “τυπώνει’’ το περιεχόμενο της παρένθεσης.

- Η λέξη Console, στην C# υποδηλώνει την κονσόλα. Επομένως, η εντολή μεταφράζεται ως : «Τύπωσε το περιεχόμενο της παρένθεσης στην κονσόλα».

Το output του προγραμματός μας θα είναι:

3.1.4 Παρατηρήσεις σχετικά με την C#

1. Το όνομα ενός προγράμματος C# είναι αυθαίρετο. Αντίθετα με άλλες γλώσσες (όπως η Java) στις οποίες το όνομα του αρχείου είναι σημαντικό, στη C# δεν παίζει κάποιο ρόλο. Ωστόσο καλό είναι να είναι όσο το δυνατόν περιγραφικό για το περιεχόμενό του. Πολλοί προγραμματιστές ακολουθούν τη σύμβαση της Java και ονομάζουν το αρχείο σύμφωνα με την κύρια κλάση που περιέχει. Επίσης τα προγράμματα C# έχουν κατάληξη .cs.

2. Η C# είναι Case sensitive Για να δηλώσουμε την Main() χρησιμοποιήσαμε κεφαλαίο M! Θα ήταν λάθος να λέγαμε static void main(). ο compiler εξ ορισμού καταλαβαίνει ότι το βασικό πρόγραμμα βρίσκεται στην μέθοδο Main() και όχι στην main().

3. Στη C# υπάρχουν δεσμευμένες λέξεις-κλειδιά. Δεν μπορούμε να δηλώσουμε πχ μια μεταβλητή με όνομα Console/int/void/class, όπως επίσης δεν μπορούμε να βάζουμε αριθμό μπροστά από μεταβλητή.

4. Η δήλωση των σταθερών στο C# πρόγραμμα όπως πχ. Η τιμή του π=3.14. γίνεται με αυτόν τον τρόπο : const double p = 3.14;

3.2 Η γλώσσα SQL

Η SQL (αγγλ. αρκτ. από το Structured Query Language) είναι μία γλώσσα υπολογιστών στις βάσεις δεδομένων, που σχεδιάστηκε για τη διαχείριση δεδομένων, σε ένα σύστημα διαχείρισης σχεσιακών βάσεων δεδομένων (Relational Database Management System, RDBMS) και η οποία, αρχικά, βασίστηκε στη σχεσιακή άλγεβρα. Η γλώσσα περιλαμβάνει δυνατότητες ανάκτησης και ενημέρωσης δεδομένων, δημιουργίας και τροποποίησης σχημάτων και σχεσιακών πινάκων, αλλά και ελέγχου πρόσβασης στα δεδομένα. Η SQL ήταν μία από τις πρώτες γλώσσες για το σχεσιακό μοντέλο του Edgar F. Codd, στο σημαντικό άρθρο του το 1970, και έγινε η πιο ευρέως χρησιμοποιούμενη γλώσσα για τις σχεσιακές βάσεις δεδομένων.

Με τον όρο σχεσιακή βάση δεδομένων εννοείται μία συλλογή δεδομένων οργανωμένη σε συσχετισμένους πίνακες που παρέχει ταυτόχρονα ένα μηχανισμό για ανάγνωση, εγγραφή, τροποποίηση ή και πιο πολύπλοκες διαδικασίες πάνω στα δεδομένα. Ο σκοπός μιας βάσης δεδομένων είναι η οργανωμένη αποθήκευση πληροφορίας και η δυνατότητα εξαγωγής της πληροφορίας αυτής, ιδίως σε πιο οργανωμένη μορφή, σύμφωνα με ερωτήματα που τίθενται στη σχεσιακή βάση δεδομένων. Τα δεδομένα είναι δυνατόν να αναδιοργανώνονται με πολλούς διαφορετικούς τρόπους, σε νοητούς πίνακες, χωρίς να είναι απαραίτητη η αναδιοργάνωση των φυσικών πινάκων που τα αποθηκεύουν. Τη σχεσιακή βάση δεδομένων επινόησε ο Έντγκαρ Κοντ το 1970.

Οι ερωτήσεις, είτε από το χρήστη είτε από λογισμικό, προς τη βάση δεδομένων, γίνονται συνήθως μέσω της διαδεδομένης διαλογικής γλώσσας SQL (Structured Query Language). Εκτελώντας ερωτήματα ο χρήστης (ή το λογισμικό που εκπροσωπεί το χρήστη) είναι δυνατόν, ανάλογα με τα δικαιώματά του, να δημιουργήσει, να μεταβάλλει και να διαγράψει δεδομένα στη βάση, ή να ανασύρει πληροφορίες με σύνθετα κριτήρια αναζήτησης. Η βάση δεδομένων είναι οργανωμένη σε πίνακες. Ο σχεδιασμός αυτών των πινάκων βασίζεται στο μοντέλο οντοτήτων – συσχετίσεων.

3.2.1 Μοντέλο οντοτήτων – συσχετίσεων

Το μοντέλο οντοτήτων – συσχετίσεων (μοντέλο Ο/Σ - ER model) είναι ένα αφαιρετικό ιδεατό μοντέλο δεδομένων, τα οποίο έχει καθορισμένη δομή. Χρησιμοποιείται κατά το πρώτο στάδιο του σχεδιασμού μιας βάσης δεδομένων, ως μοντέλο για τη σχηματική αναπαράσταση της βάσης. Το διάγραμμα που προκύπτει από αυτή την φάση, ονομάζεται διάγραμμα οντοτήτων – συσχετίσεων.

Σκοπός του μοντέλου οντοτήτων – συσχετίσεων είναι να περιγράφει τις αναγκαίες πληροφορίες οι οποίες πρέπει να αποθηκευτούν στην βάση, καθώς και τον τύπο τους. Η μοντελοποίηση είναι πολύ σημαντική, καθώς γίνεται για την περιγραφή των χρησιμοποιούμενων όρων και των σχέσεων τους σε έναν ορισμένο τομέα ενδιαφέροντος.

Προτού ασχοληθούμε περαιτέρω με την έννοια του μοντέλου οντοτήτων – συσχετίσεων, πρέπει να αναφερθούμε σε ορισμένους σχετικούς όρους:

- **Οντότητα:** Ως οντότητα ορίζεται οτιδήποτε υπάρχει αυτοτελώς και έχει συγκεκριμένα χαρακτηριστικά τα οποία το ορίζουν ως αυτοτελή ύπαρξη. Στιγμιότυπο μιας οντότητας είναι μια συγκεκριμένη περίπτωση ενός τύπου οντότητας. Ο τύπος της οντότητας είναι μια συλλογή χαρακτηριστικών που περιγράφουν την οντότητα. Μια οντότητα αναπαρίσταται σε ένα διάγραμμα οντοτήτων – συσχετίσεων με ένα ορθογώνιο.

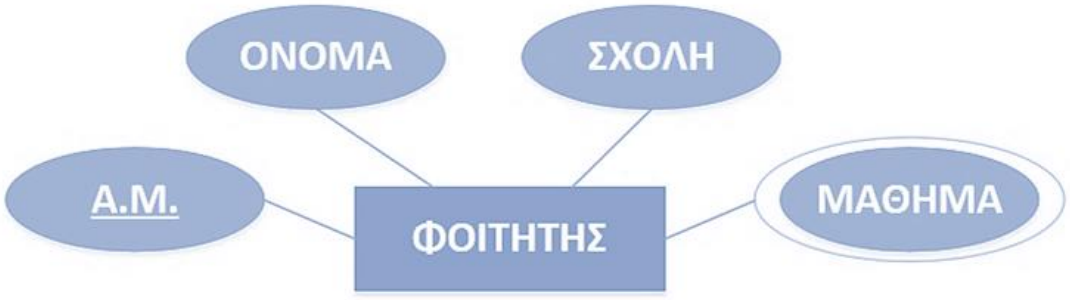
Εικόνα 3.2: Αναπαράσταση οντότητας στο διάγραμμα οντοτήτων - Συσχετίσεων

- **Χαρακτηριστικό:** Χαρακτηριστικά ή ιδιότητες μιας οντότητας ονομάζονται τα στοιχεία τα οποία την προσδιορίζουν. Τα χαρακτηριστικά, ανάλογα με τον αριθμό των τιμών τις οποίες μπορούν να πάρουν, χωρίζονται σε δύο κατηγορίες:

  1. **Μονότιμα**, τα οποία παίρνουν μόνο μια τιμή

  2. **Πλειότιμα**, τα οποία μπορούν να πάρουν περισσότερες από μια τιμές.

Στο διάγραμμα οντοτήτων - συσχετίσεων οι ιδιότητες που έχει μια οντότητα παριστάνονται μέσα σε έλλειψη, με υπογραμμισμένο το πρωτεύον κλειδί. Τα πλειότιμα χαρακτηριστικά μιας οντότητας παριστάνονται μέσα σε έλλειψη με διπλό περίγραμμα.



Εικόνα 3.3: Αναπαράσταση οντότητας και των χαρακτηριστικών της στο διάγραμμα οντοτήτων - Συσχετίσεων

- **Συσχέτιση:** Είναι η συσχέτιση μεταξύ δύο οντοτήτων ή χαρακτηριστικών τους. Ένας τύπος συσχέτισης (σύνολο συσχετίσεων) παριστάνεται με ρόμβο. Στο εσωτερικό αναγράφεται το όνομα με μικρά γράμματα. Υποδεικνύουμε τα όρια της συσχέτισης με ένα δείκτη. Ως όρια μπούμε να συναντήσουμε:

    - **0 έως άπειρο**
      (κατώτατο όριο 0, ανώτατο όριο άπειρο)
    - **τουλάχιστον 1**
      (κατώτατο όριο 1, ανώτατο όριο άπειρο)
    - **ακριβώς 1**
      (κατώτατο όριο 1, ανώτατο όριο 1)
    - **το πολύ 1**
      (κατώτατο όριο 0, ανώτατο όριο 1)

Εικόνα 3.4: Αναπαράσταση συσχέτισης "πολλά προς πολλά" στο διάγραμμα οντοτήτων - Συσχετίσεων

- **Ασθενής οντότητα:** Αδύναμη ή ασθενής οντότητα λέγεται μια οντότητα που εξαρτάται από την ύπαρξη κάποιας άλλης. Οι αδύναμες οντότητες συμμετέχουν σε συσχετίσεις μέσω ταυτοποιητικών συσχετίσεων με ισχυρή οντότητα. Αναπαρίσταται στο διάγραμμα οντοτήτων – συσχετίσεων με διπλό ορθογώνιο.

- **Ταυτοποιητική συσχέτιση:** Είναι η συσχέτιση στην οποία το πρωτεύον κλειδί της ισχυρής οντότητας χρησιμοποιείται ως μέρος του πρωτεύοντος κλειδιού της αδύναμης οντότητας. Αναπαρίσταται στο διάγραμμα οντοτήτων – συσχετίσεων με διπλό ρόμβο.

- **Διακριτικό ή μερικό κλειδί:** Διακριτικό ή μερικό κλειδί ονομάζεται το χαρακτηριστικό της αδύναμης οντότητας το οποίο μαζί με το πρωτεύον κλειδί της ισχυρής οντότητας είναι το πρωτεύον κλειδί της αδύναμης. Αναπαρίσταται στο διάγραμμα οντοτήτων – συσχετίσεων με διακεκομμένη γραμμή.

3.2.2 Γλωσσικά στοιχεία της SQL

Η γλώσσα SQL αποτελείται από διάφορα γλωσσικά στοιχεία. Αυτά τα γλωσσικά στοιχεία περιλαμβάνουν:

- **Όρους (Clauses)**: Όπως μια γραμματική πρόταση, έτσι και μια πρόταση SQL αποτελείται από συγκεκριμένους όρους. Οι πιο συνηθισμένοι όροι της SQL παρατίθενται στον παρακάτω πίνακα.

Όρος SQL	**Λειτουργία**
SELECT	Παραθέτει τα πεδία που περιέχουν δεδομένα τα οποία σας ενδιαφέρουν. (Απαιτούμενη)
FROM	Παραθέτει τους πίνακες που περιέχουν τα πεδία τα οποία παρατίθενται στον όρο SELECT. (Απαιτούμενη)
WHERE	Καθορίζει τα κριτήρια πεδίου που πρέπει να πληρούνται από κάθε εγγραφή για να συμπεριληφθεί στα αποτελέσματα. (Μη απαιτούμενη)
ORDER BY	Καθορίζει τον τρόπο ταξινόμησης των αποτελεσμάτων. (Μη απαιτούμενη)
GROUP BY	Σε μια πρόταση SQL που περιέχει συναρτήσεις συγκεντρωτικών αποτελεσμάτων, παραθέτει πεδία τα οποία δεν συνοψίζονται στον όρο SELECT. (Μη απαιτούμενη)
HAVING	Σε μια πρόταση SQL που περιέχει συναρτήσεις συγκεντρωτικών αποτελεσμάτων, καθορίζει τις συνθήκες που ισχύουν για τα πεδία τα οποία συνοψίζονται στην πρόταση SELECT. (Μη απαιτούμενη)

Πίνακας 3.1 : Οι πιο συνηθισμένοι όροι (clauses) της SQL

- **Εκφράσεις (Expressions):** Παράγουν πίνακες που αποτελούνται από στήλες και σειρές στοιχείων. Οι εκφράσεις μπορούν να χωριστούν σε δύο κατηγορίες, ανάλογα με το αντικείμενο το οποίο διαχειρίζονται: σε εκφράσεις διαχείρισης βάσης και σε εκφράσεις διαχείρισης δεδομένων. Οι εκφράσεις της SQL αναφέρονται συνοπτικά στον παρακάτω πίνακα:

Έκφραση	Λειτουργία	Σύνταξη
Εκφράσεις διαχείρισης βάσης		
CREATE DATABASE	Χρησιμοποιείται για τη δημιουργία μιας νέας, κενής βάσης δεδομένων.	CREATE [OR REPLACE] {DATABASE | SCHEMA} [IF NOT EXISTS] db_name [create_specification] ... create_specification: [DEFAULT] CHARACTER SET [=] charset_name | [DEFAULT] COLLATE [=] collation_name
DROP DATABASE	Χρησιμοποιείται για την καταστροφή μιας βάσης δεδομένων.	DROP {DATABASE | SCHEMA} [IF EXISTS] db_name
USE	Χρησιμοποιείται για την επιλογή μιας βάσης.	USE db_name
CREATE TABLE	Χρησιμοποιείται για τη δημιουργία ενός πίνακα στη βάση.	CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name (create_definition,...) [table_options]... [partition_options] CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name [(create_definition,...)] [table_options]... [partition_options] select_statement CREATE [OR REPLACE] [TEMPORARY] TABLE [IF NOT EXISTS] tbl_name { LIKE old_table_name | (LIKE old_table_name) } select_statement: [IGNORE | REPLACE] [AS] SELECT ... (Some legal select statement)
ALTER TABLE	Χρησιμοποιείται για την τροποποίηση ενός πίνακα που υπάρχει ήδη στη βάση.	ALTER [ONLINE] [IGNORE] TABLE tbl_name alter_specification[,alter_specification] ...
DROP TABLE	Χρησιμοποιείται για την καταστροφή ενός πίνακα που υπάρχει ήδη στη βάση.	DROP [TEMPORARY] TABLE [IF EXISTS] [/*COMMENT TO SAVE*/] tbl_name [, tbl_name] ...

		[RESTRICT | CASCADE]
DESCRIBE	Χρησιμοποιείται για την περιγραφή της δομής ενός πίνακα που υπάρχει ήδη στη βάση.	{DESCRIBE | DESC} tbl_name [col_name | wild]
Εκφράσεις διαχείρισης δεδομένων		
SELECT	Χρησιμοποιείται για την επιλογή δεδομένων από έναν πίνακα.	SELECT [ALL | DISTINCT | DISTINCTROW] [HIGH_PRIORITY] [STRAIGHT_JOIN] [SQL_SMALL_RESULT] [SQL_BIG_RESULT] [SQL_BUFFER_RESULT] [SQL_CACHE | SQL_NO_CACHE] [SQL_CALC_FOUND_ROWS] select_expr [, select_expr ...] [FROM table_references [PARTITION (partition_list)] [{USE|FORCE|IGNORE} INDEX [FOR {JOIN|ORDER BY|GROUP BY}] ([index_list])] [WHERE where_condition]
INSERT	Χρησιμοποιείται για την εισαγωγή δεδομένων σε έναν πίνακα.	INSERT [LOW_PRIORITY | DELAYED | HIGH_PRIORITY] [IGNORE] [INTO] tbl_name [PARTITION (partition_list)] [(col,...)] {VALUES | VALUE} ({expr | DEFAULT},...),(...),... [ON DUPLICATE KEY UPDATE col=expr
UPDATE	Χρησιμοποιείται για την ανανέωση δεδομένων ενός πίνακα.	UPDATE [LOW_PRIORITY] [IGNORE] table_reference [PARTITION (partition_list)] SET col1={expr1|DEFAULT} [,col2={expr2|DEFAULT}] ... [WHERE where_condition] [ORDER BY ...] [LIMIT row_count]

DELETE	Χρησιμοποιείται για την διαγραφή δεδομένων από έναν πίνακα.	DELETE [LOW_PRIORITY] [QUICK] [IGNORE] FROM tbl_name [PARTITION (partition_list)] [WHERE where_condition] [ORDER BY ...] [LIMIT row_count] [RETURNING select_expr [, select_expr ...]]
REPLACE	Χρησιμοποιείται για την αντικατάσταση των δεδομένων ενός πίνακα.	REPLACE [LOW_PRIORITY | DELAYED] [INTO] tbl_name [PARTITION (partition_list)] [(col,...)] {VALUES | VALUE} ({expr | DEFAULT},...),(...),...
TRUNCATE	Χρησιμοποιείται για το άδειασμα ενός πίνακα.	TRUNCATE [TABLE] tbl_name
Εκφράσεις διαχείρισης συναλλαγών		
START TRANSACTION	Χρησιμοποιείται για την εκκίνηση μιας συναλλαγής.	START TRANSACTION [transaction_property [, transaction_property] ...] | BEGIN [WORK] COMMIT [WORK] [AND [NO] CHAIN] [[NO] RELEASE] ROLLBACK [WORK] [AND [NO] CHAIN] [[NO] RELEASE] SET autocommit = {0 | 1} transaction_property: WITH CONSISTENT SNAPSHOT | READ WRITE | READ ONLY
COMMIT	Χρησιμοποιείται για την αποθήκευση αλλαγών και τον τερματισμό μιας συναλλαγής.	COMMIT [WORK] [AND [NO] CHAIN] [[NO] RELEASE]
ROLLBACK	Χρησιμοποιείται για τον τερματισμό μιας συναλλαγής, χωρίς την αποθήκευση των αλλαγών.	ROLLBACK [WORK] [AND [NO] CHAIN] [TO [SAVEPOINT] {<savepoint name> | <simple target specification>}]

Πίνακας 3.2: Συνοπτικός πίνακας εκφράσεων της SQL

- **Κατηγορήματα** (**Predicates**): Διευκρινίζουν τους όρους που μπορούν να αξιολογηθούν σαν σωστό ή λάθος. Έχουν ως έξοδο TRUE, FALSE ή UNKNOWN. Χρησιμοποιούνται στην περίπτωση που χρησιμοποιείται η έκφραση αναζήτησης SEARCH , μαζί με τα clauses WHERE ή HAVING, ή γενικά οπουδήποτε η έξοδος έχει τύπο Boolean. Ορισμένα predicates είναι τα εξής:

    - ✓ AND
    - ✓ OR
    - ✓ LIKE
    - ✓ BETWEEN
    - ✓ AS
    - ✓ TOP(LIMIT)

- **Επερωτήματα** (**Queries**): Ανακτούν στοιχεία βασισμένες σε ειδικά κριτήρια.

- **Δηλώσεις** (**Statements**): ελέγχουν τη ροή του προγράμματος και τις συνδέσεις από άλλα προγράμματα.

Κεφάλαιο 4: Εισαγωγή στο Visual Studio

Το Microsoft Visual Studio είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (integrated development environment - IDE), το οποίο δημιουργήθηκε από την Microsoft. Χρησιμοποιείται για την ανάπτυξη υπολογιστικού λογισμικού για το λειτουργικό σύστημα Windows της Microsoft, καθώς και για άλλες εφαρμογές, όπως ιστοσελίδες, διαδικτυακές εφαρμογές, διαδικτυακές υπηρεσίες και εφαρμογές για κινητά. Το Visual Studio χρησιμοποιεί πλατφόρμες ανάπτυξης λογισμικού, φτιαγμένες από τη Microsoft, όπως τις Windows API, Windows Forms, Windows Presentation Foundation, Windows Store και Microsoft Silverlight. Έχει τη δυνατότητα να παράγει και εγγενή και διαχειριζόμενο κώδικα.

Το Visual Studio περιέχει έναν επεξεργαστή κώδικα, ο οποίος υποστηρίζει το IntelliSense (το συστατικό το οποίο ολοκληρώνει τον κώδικα) όπως επίσης και δυνατότητα επανασχεδιασμού κώδικα. Ο ενσωματωμένος αποσφαλματωτής (debugger) λειτουργεί και ως αποσφαλματωτής πηγαίου κώδικα (source-level debugger) και ως αποσφαλματωτής κώδικα μηχανής (machine-level debugger). Άλλα ενσωματωμένα εργαλεία περιλαμβάνουν: έναν σχεδιαστή για φόρμες ο οποίος χρησιμοποιείται για την δημιουργία GUI εφαρμογών, έναν σχεδιαστή διαδικτυακών εφαρμογών, σχεδιαστή κλάσεων, και σχεδιαστή βάσεων δεδομένων. Δίνει την δυνατότητα χρήσης επεκτάσεων (plug-ins),τα οποία βελτιώνουν τη λειτουργικότητα της πλατφόρμας σε μεγάλο βαθμό—συμπεριλαμβανομένης και της επιπλέον υποστήριξης για συστήματα ελέγχου πηγής (για παράδειγμα το Subversion) και την πρόσθεση νέων εργαλειοθηκών ,όπως editors και οπτικούς σχεδιαστές για γλώσσες συγκεκριμένου τομέα ή εργαλειοθήκες για άλλους τομείς του κύκλου ζωής ανάπτυξης μιας εφαρμογής.

Το Visual Studio υποστηρίζει διάφορες γλώσσες προγραμματισμού και επιτρέπει στον αποσφαλματωτή να υποστηρίξει σχεδόν οποιαδήποτε γλώσσα προγραμματισμού υπάρχει. Οι ενσωματωμένες γλώσσες που περιλαμβάνονται στο Visual Studio είναι οι: C, C++ και C++/CLI (με την μορφή της Visual C++), VB.NET (με την μορφή της Visual Basic .NET), C# (με την μορφή της Visual C#), και η F#. Η υποστήριξη για άλλες γλώσσες όπως η Python, η Ruby, και η Node.js, είναι διαθέσιμη μέσω των γλωσσικών υπηρεσιών (language services) οι οποίες εγκαθίστανται ξεχωριστά. Το Visual Studio υποστηρίζει επίσης XML/XSLT, HTML/XHTML, JavaScript και CSS. Η Java (και η J#) υποστηρίζονταν στο παρελθόν.

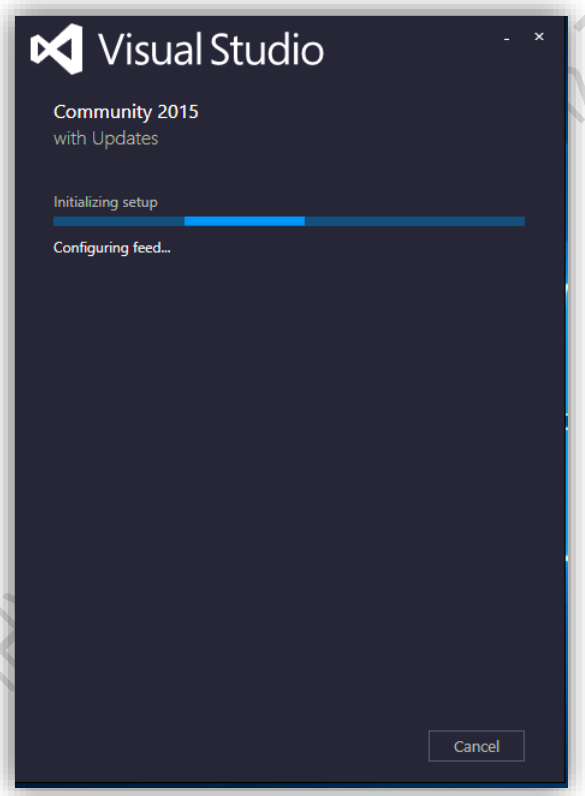
4.1 Εγκατάσταση του Visual Studio.

Μπορούμε να κατεβάσουμε το Visual Studio 2015 Community edition, χρησιμοποιώντας τον εξής σύνδεσμο:

https://www.visualstudio.com/downloads/

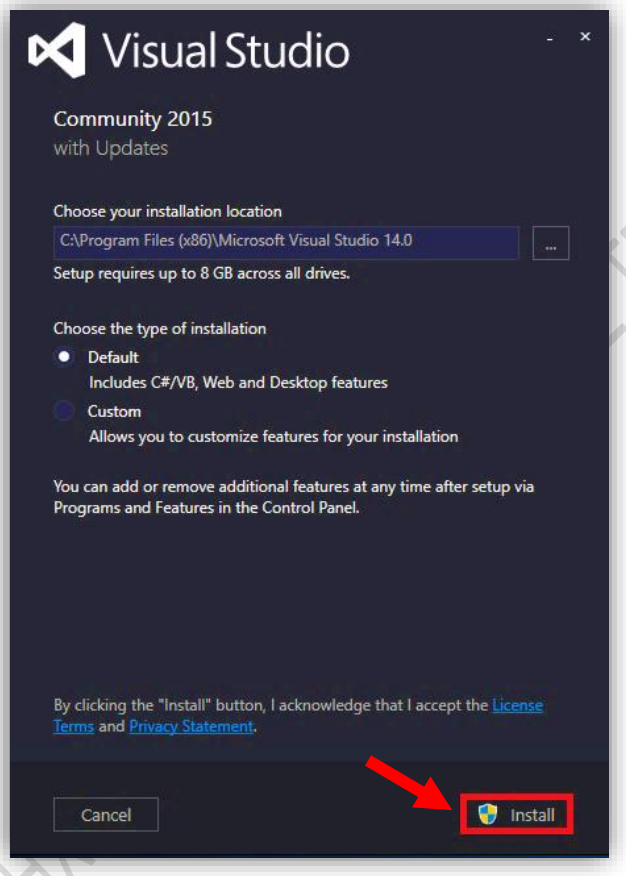
Παρακάτω, περιγράφονται τα βήματα που πρέπει να ακολουθηθούν για την σωστή εγκατάσταση του Visual Studio 2015:

- Βήμα 1ο : Προτού ξεκινήσει η διαδικασία της εγκατάστασης, το Visual Studio πρέπει να ρυθμίσει ορισμένες παραμέτρους, έτσι ώστε να εξασφαλιστεί η ομαλή εγκατάσταση και λειτουργεία του προγράμματος. Συνεπώς, εμφανίζεται η παρακάτω αρχική οθόνη.



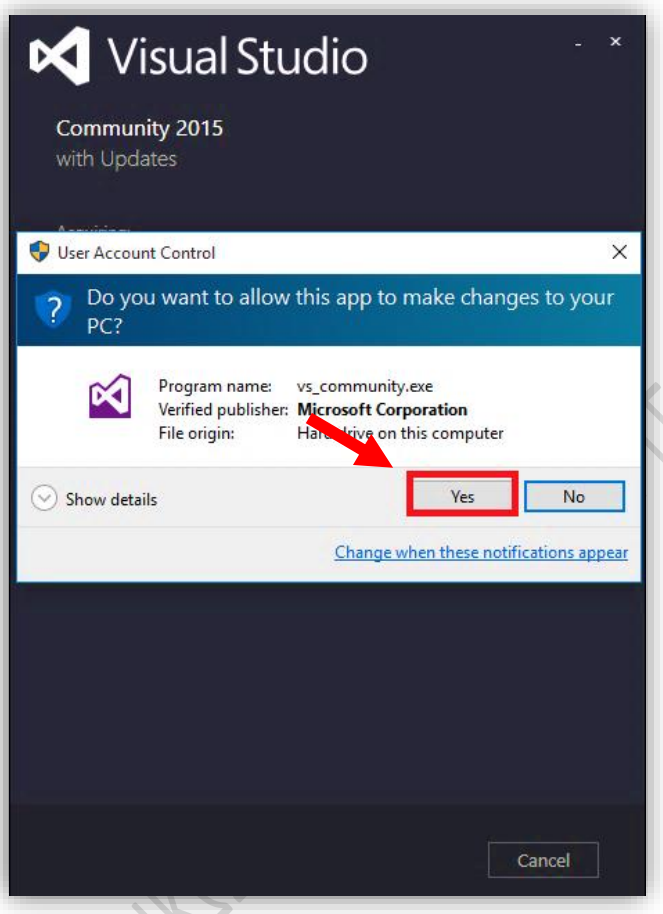
Εικόνα 4.1: Αρχική οθόνη εγκατάστασης του Visual Studio 2015.

➢ Βήμα 2ο : Αφού το Visual Studio ρυθμίσει τις παραμέτρους του, εμφανίζεται στον χρήστη μια οθόνη, η οποία του επιτρέπει να επιλέξει σε ποιο directory θα εγκατασταθεί το πρόγραμμα και τι είδους εγκατάσταση επιθυμεί (default - προκαθορισμένη ή custom -προσαρμοσμένη , ενδείκνυται να παραμείνει η επιλογή default, ώστε να μην υπάρξουν σφάλματα ή ελλείψεις στην λειτουργεία του Visual Studio). Αφού ο χρήστης επιλέξει τις ρυθμίσεις που επιθυμεί, πρέπει να πατήσει το κουμπί Install, έτσι ώστε να ξεκινήσει η εγκατάσταση του Visual Studio.



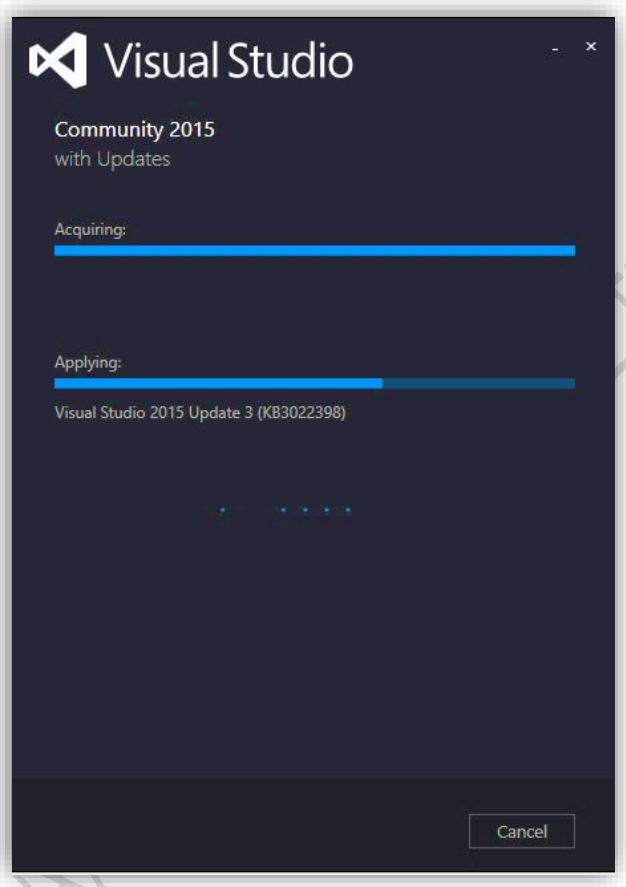
Εικόνα 4.2: Οθόνη επιλογής ρυθμίσεων από τον χρήστη.

- Βήμα 3ο : Μόλις ο χρήστης πατήσει το κουμπί install, θα εμφανιστεί ένα παράθυρο διαλόγου των Windows, έτσι ώστε ο χρήστης να δώσει δικαίωμα εγκατάστασης στο Visual Studio.



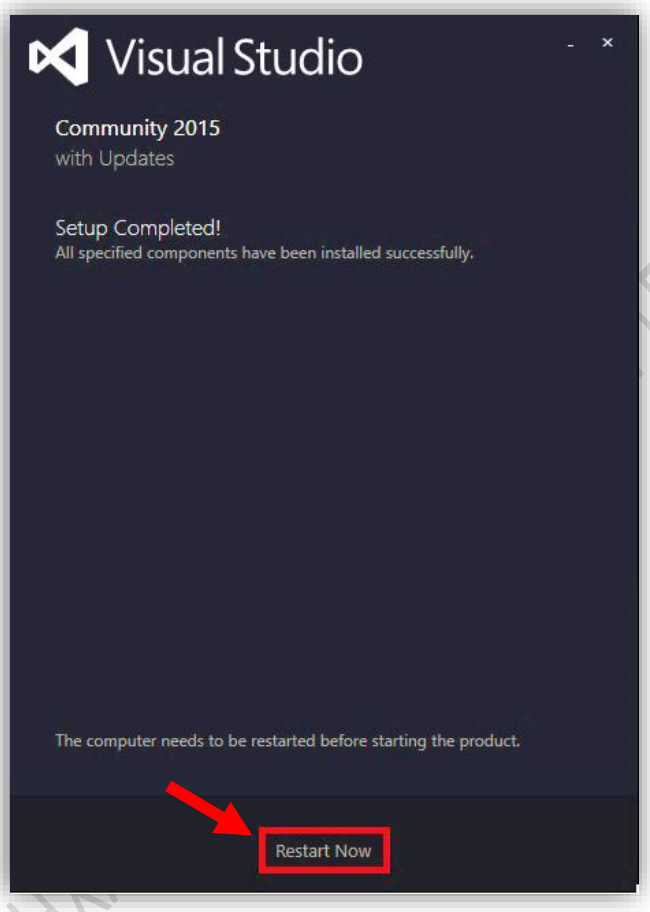
Εικόνα 4.3: Παράθυρο διαλόγου δικαιοδοσίας της εγκατάστασης από τον χρήστη.

- Βήμα 4ο : Αφού δοθούν τα δικαιώματα εγκατάστασης στην εφαρμογή από τον χρήστη, ξεκινά η διαδικασία ανάκτησης και εγκατάστασης των απαιτούμενων αρχείων του Visual Studio. Αυτό το βήμα είναι απλό, καθώς ο χρήστης δεν καλείται να πραγματοποιήσει κάποια ενέργεια, όμως χρειάζεται να περιμένει αρκετή ώρα μέχρι την ολοκλήρωσή του.



Εικόνα 4.4: Οθόνη ανάκτησης και εγκατάστασης απαιτούμενων αρχείων του Visual Studio

- Βήμα 5ο : Μόλις ολοκληρωθεί η εγκατάσταση όλων των απαραίτητων αρχείων του Visual Studio, εμφανίζεται στον χρήστη μια οθόνη, η οποία ενημερώνει τον χρήστη ότι η εγκατάσταση του Visual Studio ολοκληρώθηκε με επιτυχία. Προτού όμως χρησιμοποιηθεί το Visual Studio για πρώτη φορά, απαιτείται η επανεκκίνηση των Windows. Όταν ο χρήστης πατήσει το κουμπί Restart Now, θα γίνει επανεκκίνηση των Windows.



Εικόνα 4.5: Οθόνη επιτυχούς εγκατάστασης του Visual Studio

Για την δημιουργία της βάσης, απαιτείται η εγκατάσταση ενός πρόσθετου extension, της εργαλειοθήκης SQLite / SQL Server Compact Toolbox. Την εργαλειοθήκη αυτή μπορούμε να την προμηθευτούμε, αν επισκεφτούμε τον εξής σύνδεσμο:

https://marketplace.visualstudio.com/items?itemName=ErikEJ.SQLServerCompactSQLiteToolbox

Κεφάλαιο 5: Σχεδιασμός και υλοποίηση της εφαρμογής

5.1 Σχεδιασμός της εφαρμογής

Σε αυτήν την παράγραφο περιγράφεται η διαδικασία που ακολουθήθηκε για τον σχεδιασμό της εφαρμογής. Η διαδικασία σχεδιασμού της εφαρμογής έχει δύο σκέλη τα οποία είναι:

1. Το πρώτο σκέλος αφορά την διαδικασία της καταγραφής και η τεκμηρίωσης των σχετικών αποφάσεων όπως:

    I. Ποιες είναι οι ανάγκες του καταναλωτικού κοινού στο οποίο απευθυνόμαστε

    II. Τον ηλικιακό πληθυσμό στον οποίο στοχεύουμε για το προϊόν μας

    III. Απλότητα και αμεσότητα στην εφαρμογή με βάση πάντα τις ανάγκες του καταναλωτικού κοινού στο οποίο απευθυνόμαστε

    Πέραν των εν λόγω βασικών αποφάσεων που θα πρέπει να λάβουμε υπόψιν, είναι αναγκαίο να γίνει σωστή τήρηση αυτών όπως και σωστής διαφημίσεως του εν λόγω προϊόντος προς σε αυτό το καταναλωτικό κοινό μετά από κάποιο σχετικό χρονικό διάστημα δοκιμασίας του προϊόντος μας κι εφόσον μόνο φέρει τα επιθυμητά αποτελέσματα.

2. Το δεύτερο σκέλος αφορά την διαδικασία σχεδιασμού του γραφικού περιβάλλοντος (GUI) της εφαρμογής έτσι ώστε να εξυπηρετεί τους εξής παράγοντες:

    I. Φιλικό γραφικό περιβάλλον

    II. Καταγραφή των χρηστών μέσω με δημιουργία ηλεκτρονικού λογαριασμού

    III. Συλλογή στοιχείων με βάση τα αποτελέσματα που θα δώσει ο χρήστης προκειμένου να σχεδιάσουμε από αυτά τα αντίστοιχα στατιστικά δεδομένα.

    Θα πρέπει να λάβουμε υπόψιν ότι για να εξάγουμε σωστά στατιστικά δεδομένα θα πρέπει ο χρήστης να απαντάει με ειλικρίνεια και σαφήνεια στις εικόνες που του παρουσιάζονται.

5.1.1 Παράγοντες που λάβαμε υπόψιν κατά το σχεδιασμό της εφαρμογής

Η νόσος Alzheimer εμφανίζεται περισσότερο σε άτομα προχωρημένης ηλικίας, τα οποία είναι άνω των 65. Η συχνότητα εμφάνισης της ασθένειας διαφοροποιείται με την ηλικία: κάθε πέντε έτη μετά από την ηλικία των 65, ο κίνδυνος εμφάνισης της ασθένειας περίπου διπλασιάζεται. [ref. 1]

Επομένως, υπάρχει η ανάγκη για τη δημιουργία μιας εκπαιδευτικής εφαρμογής, η οποία θα είναι κατάλληλη για την ομάδα την οποία προορίζεται, δηλαδή τους ηλικιωμένους. Για να επιτευχθεί αυτό, θα πρέπει να ληφθούν υπόψιν ορισμένοι παράγοντες. [ref. 5]

Έπειτα από μελέτη σχετικής βιβλιογραφίας, προέκυψαν ως συμπέρασμα ορισμένες προϋποθέσεις, οι οποίες πρέπει να ακολουθηθούν, έτσι ώστε να είναι η εφαρμογή κατάλληλη για ηλικιωμένους. Με άλλα λόγια:

- **Η εφαρμογή πρέπει να είναι απλή, φιλική προς το χρήστη και να παρέχει απλό χειρισμό. [ref. 2]** Καθώς το μεγαλύτερο ποσοστό των ηλικιωμένων δεν είναι εξοικειωμένοι με την σύγχρονη τεχνολογία, το παιχνίδι θα πρέπει να παρέχει ένα απλό και φιλικό περιβάλλον, έτσι ώστε να μην υπάρξει σύγχυση.

- **Κάθε επίπεδο του παιχνιδιού, θα πρέπει να είναι σύντομο και να απαιτεί όσο το δυνατόν λιγότερα βήματα για την ολοκλήρωσή του. [ref. 3]** Ο αριθμός των βημάτων που απαιτείται για την ολοκλήρωση ενός επιπέδου, ενδείκνυται να είναι μεταξύ 8 με 12 βήματα, ώστε ο χρήστης να μην απογοητευτεί και εγκαταλείψει το παιχνίδι.

- **Το παιχνίδι δεν θα πρέπει να παρέχει πολλά και περιττά ερεθίσματα στον χρήστη. [ref. 3]** Τα πολλά και περιττά ερεθίσματα, ενδέχεται να προκαλέσουν σύγχυση και απογοήτευση στον χρήστη.

- **Τα χρώματα που θα χρησιμοποιηθούν για το γραφικό περιβάλλον της εφαρμογής, θα πρέπει να είναι κατά προτίμηση θερμά. [ref. 2]** Σύμφωνα με έρευνες που έχουν διεξαχθεί, έχει αποδειχθεί ότι τα θερμά χρώματα τραβούν περισσότερο την προσοχή των ανθρώπων μεγαλύτερης ηλικίας. Συνεπώς, για να πετύχει η εφαρμογή το βέλτιστο αποτέλεσμα, ενδείκνυται το γραφικό περιβάλλον να αποτελείται ως επί το πλείστων από θερμά χρώματα.

- **Κάθε χρήστης θα πρέπει να διαθέτει το δικό του, εξατομικευμένο προφίλ, έτσι ώστε να καταχωρούνται περιοδικά στοιχεία σχετικά με την κατάσταση της μνήμης και την πρόοδό του.** Για να επιτύχει η εφαρμογή το σκοπό της, δηλαδή την όξυνση της μνήμης του χρήστη, θα πρέπει να διατηρούνται και να καταγράφονται στη βάση και σε αρχείο, κάποια βασικά στοιχεία σχετικά με τις επιδόσεις κάθε χρήστη ξεχωριστά σε ένα αρχείο (log file). Το αρχείο αυτό θα συμβάλλει στην πιο εξατομικευμένη αντιμετώπιση των αναγκών κάθε χρήστη ξεχωριστά, αλλά και στην συλλογή και καταγραφή δεδομένων, με σκοπό την εκτίμηση της προόδου της κατάστασης της μνήμης του.

5.1.2 Σχεδιασμός storyboard του παιχνιδιού

Προτού προχωρήσουμε στην υλοποίηση της εφαρμογής, είναι απαραίτητος ο σχεδιασμός ενός προσχεδίου, του storyboard. (βλ. παράγραφο 2.2)

Το αρχικό στάδιο του σχεδιασμού του storyboard, αποτελείται από τον καθορισμό του σκοπού της εφαρμογής. Αφού πραγματοποιηθεί αυτό, πρέπει να δοθεί μια γενική περιγραφή για το παιχνίδι. Στην περίπτωσή μας, το παιχνίδι αποτελείται από διάφορα επίπεδα δυσκολίας. Το κάθε επίπεδο αποτελείται από 2 φάσεις. Στην πρώτη φάση, προβάλλονται διαδοχικά φωτογραφίες ατόμων με το όνομα του κάθε ατόμου. Ο χρήστης θα έχει τη δυνατότητα να κοιτάξει την κάθε φωτογραφία για όσο χρονικό διάστημα επιθυμεί και να περιηγηθεί στις φωτογραφίες όσες φορές επιθυμεί. Μόλις τελειώσει η προβολή των φωτογραφιών, το παιχνίδι περνάει στη δεύτερη φάση, όπου ο χρήστης, αφού πατήσει και το τελευταίο κουμπί, καλείται να συμπληρώσει το όνομα του κάθε ατόμου στη φωτογραφία, οι οποίες τώρα προβάλλονται με διαφορετική σειρά και φυσικά χωρίς το όνομα του ατόμου το οποίο απεικονίζεται σε αυτήν. Αφού ολοκληρώσει το πρώτο επίπεδο, μπορεί να ξεκινήσει το δεύτερο κλπ. Σε κάθε επίπεδο, ο αριθμός φωτογραφιών που προβάλλονται θα αυξάνεται.

Το παιχνίδι αποτελείται από 4 επίπεδα, τα οποία είναι δομημένα ως εξής:

- Στο 1ο επίπεδο, (το οποίο είναι κυρίως για εξάσκηση), θα προβάλλονται 3 φωτογραφίες, και το κάθε όνομα θα αποτελείται από 3 γράμματα.

- Στο 2ο επίπεδο, θα προβάλλονται 5 φωτογραφίες, και το κάθε όνομα θα αποτελείται από 4 γράμματα. Για να έχει πρόσβαση σε αυτό το επίπεδο, ο παίκτης θα πρέπει να έχει συγκεντρώσει 100 πόντους ή περισσότερους.

- Στο 3ο επίπεδο, θα προβάλλονται 10 φωτογραφίες, και το κάθε όνομα θα αποτελείται από 5 γράμματα. Για να έχει πρόσβαση σε αυτό το επίπεδο, ο παίκτης θα πρέπει να έχει συγκεντρώσει 300 πόντους ή περισσότερους.

- Στο 4ο και τελευταίο επίπεδο, θα προβάλλονται 20 φωτογραφίες, και το κάθε όνομα θα αποτελείται από 6 γράμματα. Για να έχει πρόσβαση σε αυτό το επίπεδο, ο παίκτης θα πρέπει να έχει συγκεντρώσει 700 πόντους ή περισσότερους.

Πόντοι και βοήθειες:

Η σωστή συμπλήρωση του ονόματος σε μία φωτογραφία, δίνει στον χρήστη 50 πόντους.

Στον χρήστη θα δίνεται η δυνατότητα της παροχής κάποιας υπόδειξης (Hint) , η οποία θα είναι να εμφανίζεται κάποιο γράμμα του ονόματος στην οθόνη. Όμως, κάθε φορά που ο χρήστης χρησιμοποιεί κάποιο hint, θα αφαιρούνται 5 πόντοι από τη βαθμολογία (εκτός αν η βαθμολογία είναι 0, όπου δεν θα αφαιρούνται πόντοι).

Οι πόντοι θα προστίθενται και θα αφαιρούνται από τη βάση.

Κάθε φορά που παίζει ο χρήστης το παιχνίδι, μαζί με τη βαθμολογία, θα καταχωρείται στη βάση και ο χρόνος που χρειάστηκε για να ολοκληρώσει το κάθε επίπεδο.

Αναλυτικός σχεδιασμός κάθε φόρμας:

Κυρίος σκοπός ενός storyboard, είναι ο σχεδιασμός της εφαρμογής και η διευκόλυνση της υλοποίησης της. Συνεπώς, είναι απαραίτητος ο αναλυτικός σχεδιασμός και ο προσδιορισμός της θέσης κάθε στοιχείου της εφαρμογής.

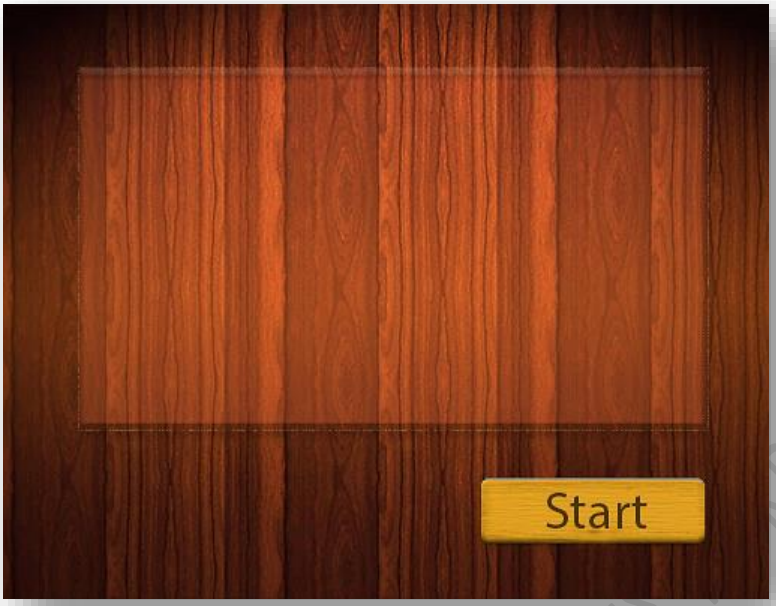
Το παιχνίδι το οποίο περιγράφεται στην παρούσα πτυχιακή εργασία, αποτελείται από φόρμες. Κατά συνέπεια, πρέπει να σχεδιάσουμε κάθε φόρμα ξεχωριστά, έτσι ώστε να προσδιορίσουμε τις ανάγκες που πρέπει να καλυφθούν, ξεκινώντας από τις κεντρικές, και καταλήγοντας στις «πιο εξειδικευμένες» φόρμες.

- Αρχική οθόνη παιχνιδιού: Η αρχική οθόνη του παιχνιδιού αποτελείται από τον τίτλο του παιχνιδιού, και δυο κουμπιά: το κουμπί «Start» και το κουμπί «Exit».

Εικόνα 5.1 : Αρχική οθόνη παιχνιδιού

Αν ο χρήστης πατήσει το κουμπί «Exit», τότε η εφαρμογή θα τερματιστεί, ενώ αν πατήσει το κουμπί «Start», τότε θα εμφανιστεί μια οθόνη, η οποία θα δίνει οδηγίες για το παιχνίδι.

- Οθόνη οδηγιών παιχνιδιού: Εδώ προβάλλονται οι οδηγίες του παιχνιδιού, δηλαδή τι πρέπει να κάνει ο χρήστης για να ολοκληρώσει το παιχνίδι.

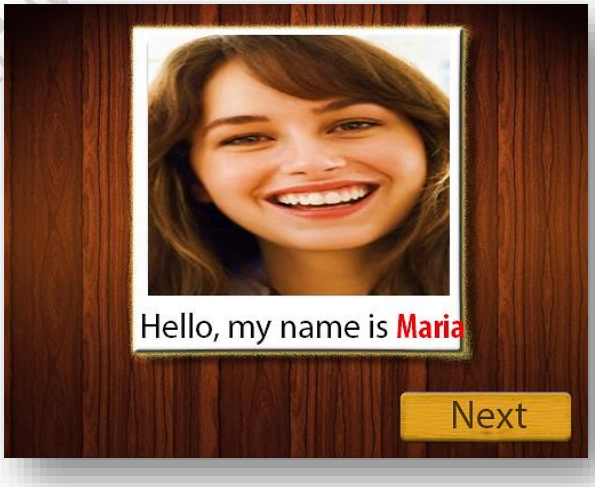


Εικόνα 5.2: Οθόνη οδηγιών του παιχνιδιού

Όταν ο χρήστης πατήσει το κουμπί «Start», μεταφέρεται αυτόματα στο level που αντιστοιχεί στους πόντους που έχει συγκεντρώσει.

- Κύρια οθόνη παιχνιδιού (επίπεδα παιχνιδιού): Το κάθε επίπεδο αποτελείται από 2 φάσεις. Στην πρώτη φάση, προβάλλονται διαδοχικά φωτογραφίες ατόμων με το όνομα του κάθε ατόμου και στη δεύτερη φάση ο χρήστης καλείται να συμπληρώσει το όνομα κάθε ατόμου στην αντίστοιχη φωτογραφία. Γι' αυτόν τον λόγο, κάθε φάση του παιχνιδιού πρέπει να σχεδιαστεί ξεχωριστά.

  - Πρώτη φάση (Προβολή φωτογραφιών):



Εικόνα 5.3: Στιγμιότυπο πρώτης φάσης παιχνιδιού

Αυτή είναι η πρώτη φάση του παιχνιδιού. Κάτω από κάθε φωτογραφία, υπάρχει το κείμενο “Hello, my name is “ και το όνομα του εικονιζόμενου ατόμου. Ο χρήστης μπορεί να περιηγηθεί στις φωτογραφίες, πατώντας το πλήκτρο “ Next”,για να προβάλλει την επόμενη φωτογραφία, ή το πλήκτρο “Previous” για να προβάλλει την προηγούμενη.

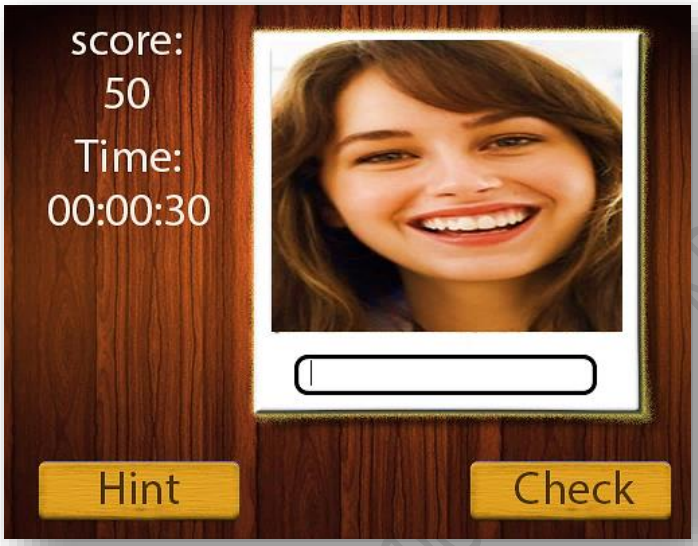
Όταν ο χρήστης φτάσει στην τελευταία φωτογραφία, το κουμπί “Next”, θα αντικατασταθεί από το κουμπί “Finish”.

Εικόνα 6.4: Στιγμιότυπο από το παιχνίδι κατά την προβολή της τελευταίας φωτογραφίας

Μόλις ο χρήστης πατήσει το κουμπί “Finish” , θα ξεκινήσει η δεύτερη φάση του παιχνιδιού.

Εικόνα 5.5: Ροή πρώτης φάσης πρώτου επιπέδου

- Δεύτερη φάση (Αναγνώριση προσώπων):
  Σε αυτήν την φάση, προβάλλονται οι φωτογραφίες που προβλήθηκαν και στην πρώτη φάση, αλλά με διαφορετική σειρά. Επίσης, στη θέση του κειμένου, υπάρχει ένα textbox, όπου ο χρήστης καλείται να πληκτρολογήσει το όνομα του εικονιζόμενου ατόμου.



Εικόνα 5.6: Στιγμιότυπο από τη δεύτερη φάση του παιχνιδιού

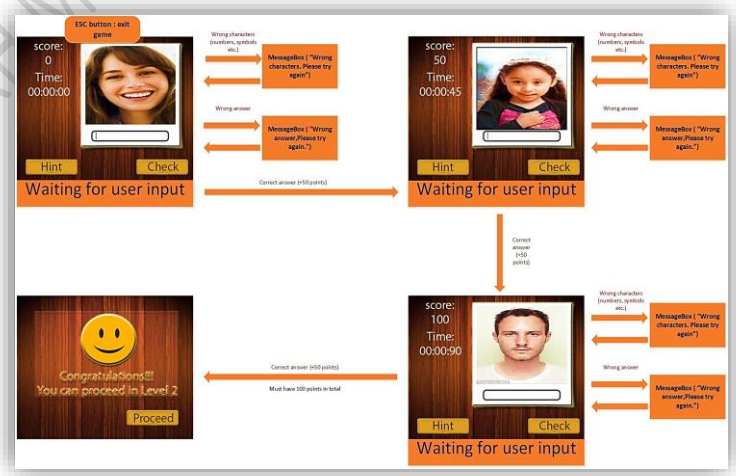
Ο χρήστης πρέπει να πληκτρολογήσει το όνομα του εικονιζόμενου ατόμου (δηλαδή μόνο γράμματα, αλλιώς θα είναι exception) στο textbox στο κάτω μέρος της φωτογραφίας.

Όταν ο χρήστης πατήσει το κουμπί “Check”, εάν το όνομα που πληκτρολόγησε είναι σωστό, θα προχωρήσει στην επόμενη φωτογραφία. Εάν όμως είναι λάθος, θα εμφανιστεί ένα messageBox στην οθόνη με το μήνυμα “Wrong answer. Please, try again”.

Μόλις ο χρήστης πατήσει το κουμπί “Hint” για πρώτη φορά , θα εμφανιστεί στο textbox το πρώτο γράμμα του ονόματος του εικονιζόμενου στη φωτογραφία, και θα αφαιρεθούν 5 πόντοι από τη βαθμολογία του. Αν το πατήσει δεύτερη φορά , θα εμφανιστεί το δεύτερο γράμμα του ονόματος και θα αφαιρεθούν άλλοι 5 πόντοι κ.ο.κ.

Εικόνα 5.7 : Στιγμιότυπο παιχνιδιού αφού ο χρήστης πατήσει το κουμπί "Hint"

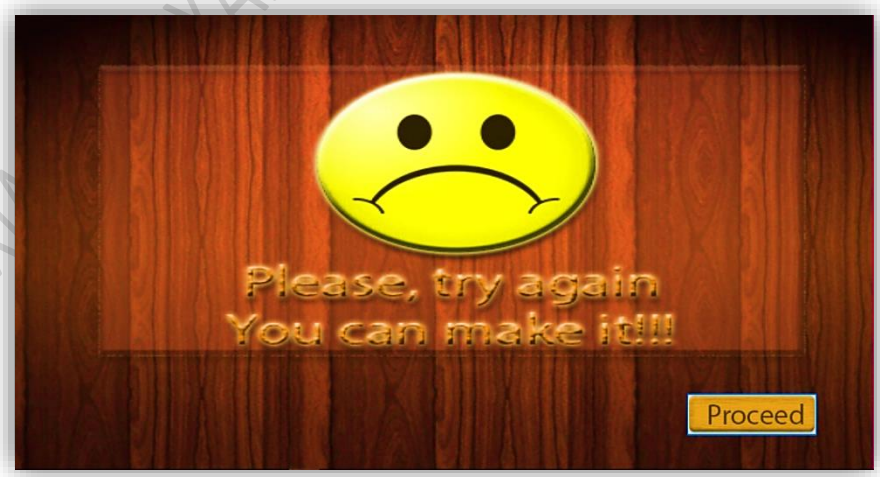
Εικόνα 6.8 : Ροή "Hint" παιχνιδιού



Εικόνα 5.9 : Ροή δεύτερης φάσης παιχνιδιού

Αν ο χρήστης συγκεντρώσει αρκετούς πόντους σε ένα επίπεδο, οδηγείται στην « οθόνη επιτυχίας», μέσω της οποίας μπορεί να προχωρήσει στο επόμενο επίπεδο. Αν δεν συγκεντρώσει αρκετούς πόντους, οδηγείται στην οθόνη επανάληψης, και πρέπει να επαναλάβει το επίπεδο.

Εικόνα 5.10 : Οθόνη επιτυχίας



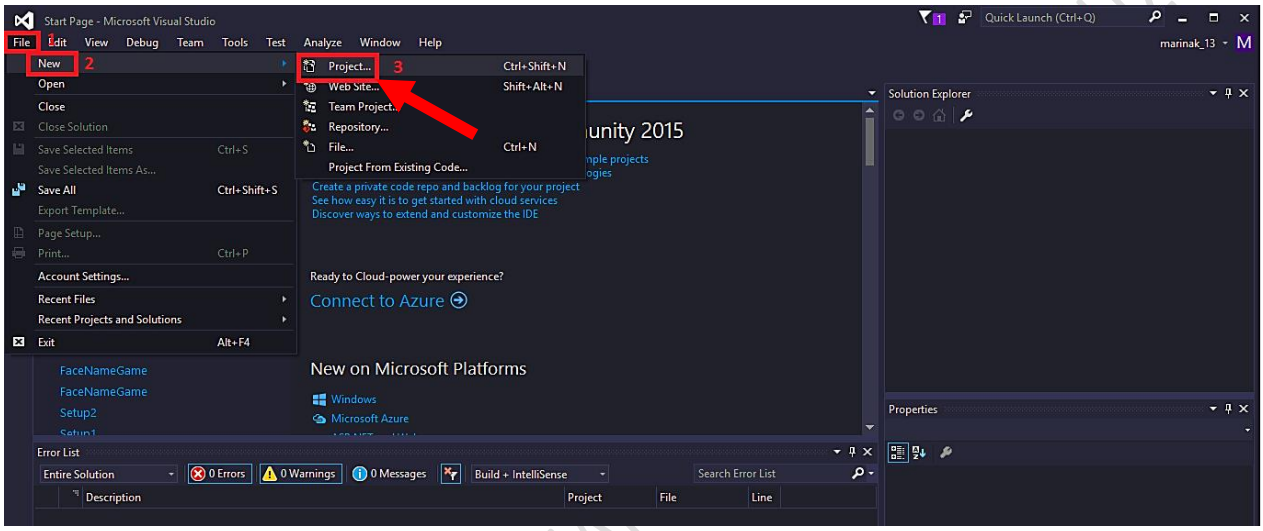
Εικόνα 5.11 : Οθόνη επανάληψης

5.2 Υλοποίηση της εφαρμογής

5.2.1 Δημιουργία νέου Project

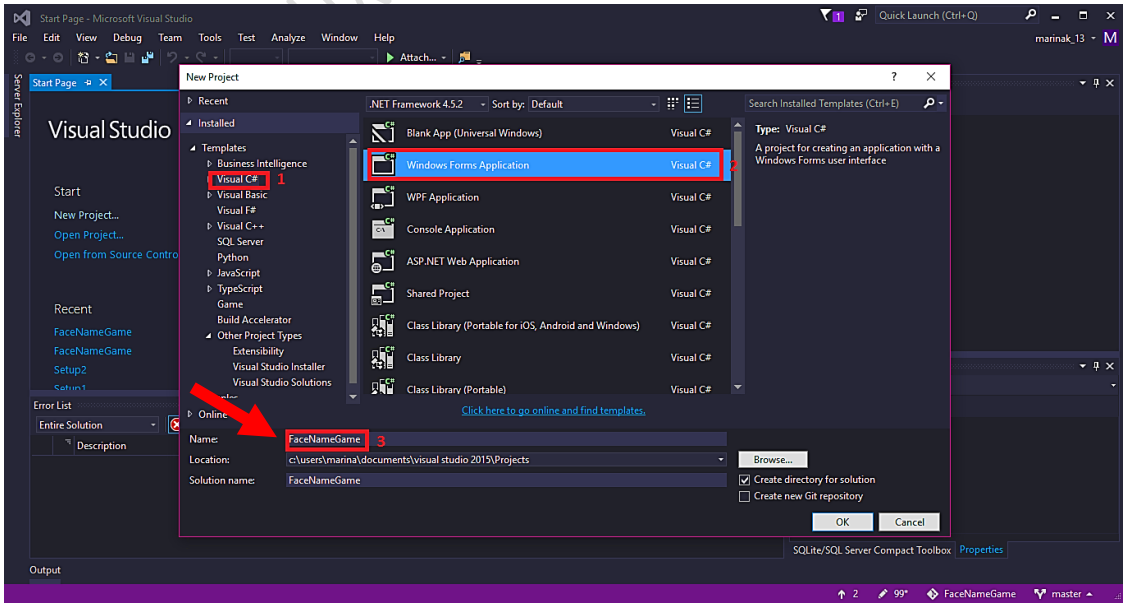
Το πρώτο βήμα για την υλοποίηση της εφαρμογής, είναι η δημιουργία ενός νέου project στο visual studio.

Για να δημιουργήσουμε ένα νέο project στο Visual Studio, ακολουθούμε την εξής διαδρομή: File -> New -> Project



Εικόνα 5.12 : Αρχική οθόνη Visual Studio 2015

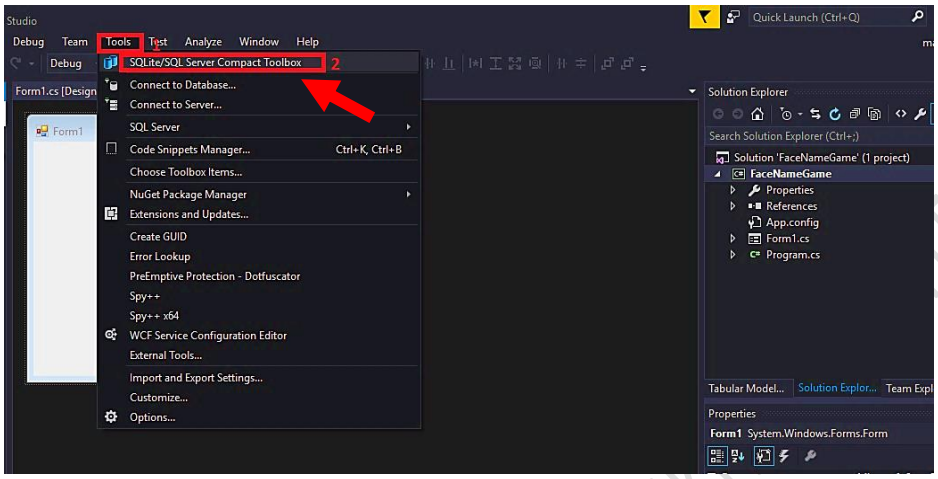
Αφού ανοίξει η καρτέλα «New Project», επιλέγουμε Visual C# -> Windows Forms Applications και εισάγουμε το όνομα του project στο πλαίσιο Name.



Εικόνα 5.13: Καρτέλα "New Project"

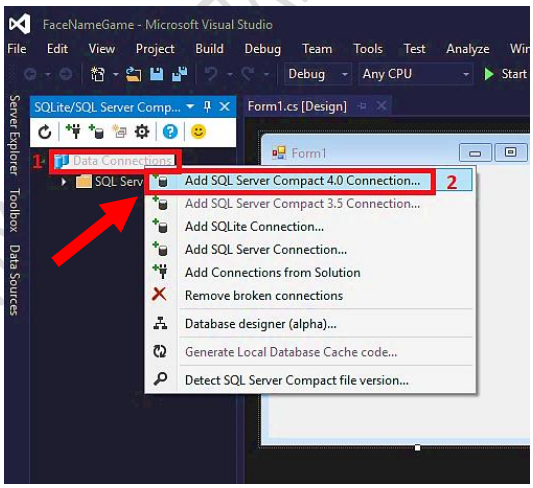
5.2.2 *Δημιουργία της βάσης SQL της εφαρμογής*

Για τη δημιουργία της βάσης SQL, η διαδικασία είναι η εξής: Tools -> SQLite/SQL Server Compact Box.



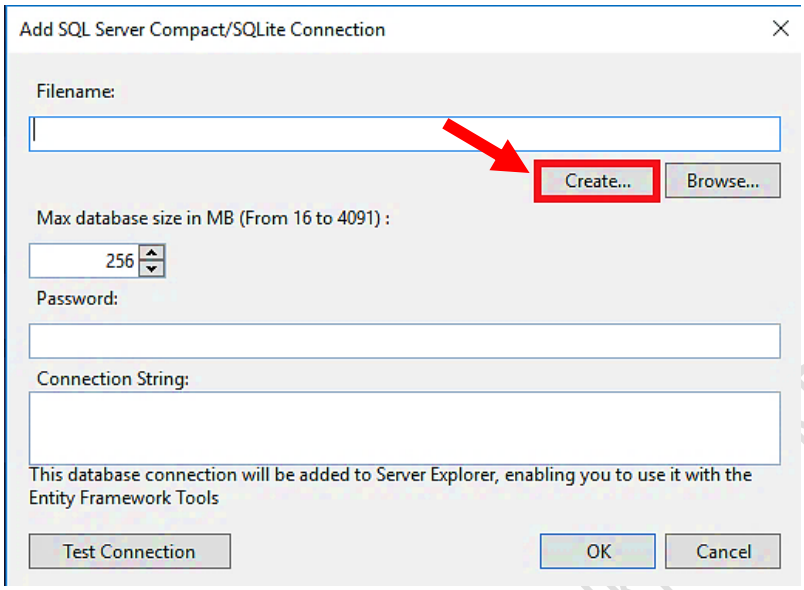
Εικόνα 5.14: Έναρξη διαδικασίας δημιουργίας βάσης SQL

Αφού εμφανιστεί η καρτέλα «SQLite/ SQL Server Compact Toolbox» , πατάμε δεξί κλικ στο «Data Connections» και επιλέγουμε « Add SQL Server Compact 4.0 Connection» από το μενού το οποίο θα εμφανιστεί.



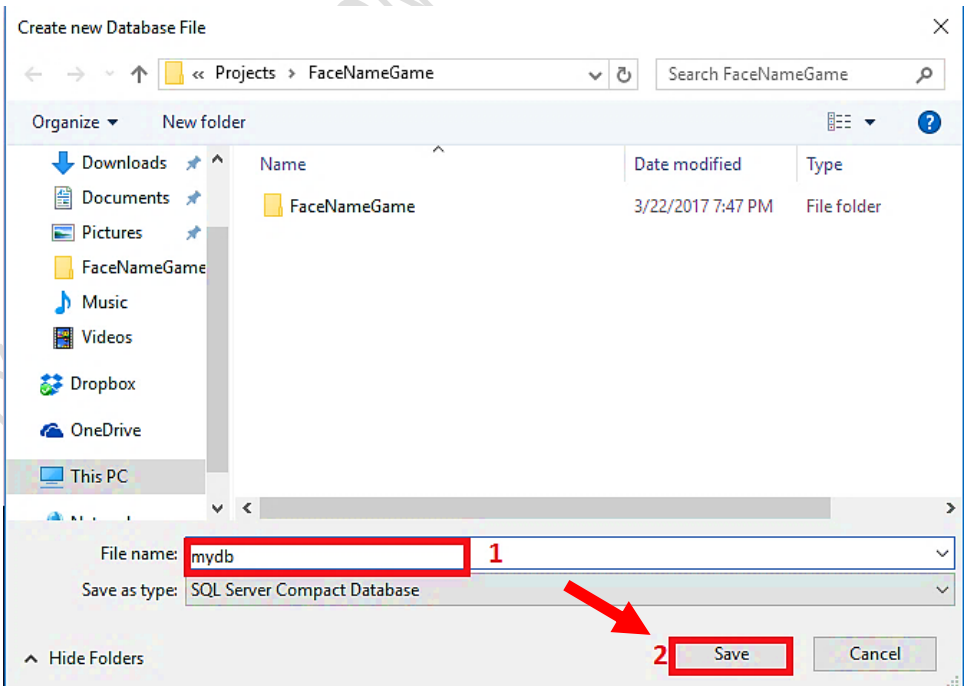
Εικόνα 5.15: Καρτέλα SQLite / SQL Server Compact Toolbox

Στην καρτέλα την οποία θα εμφανιστεί , πατάμε το κουμπί «Create».



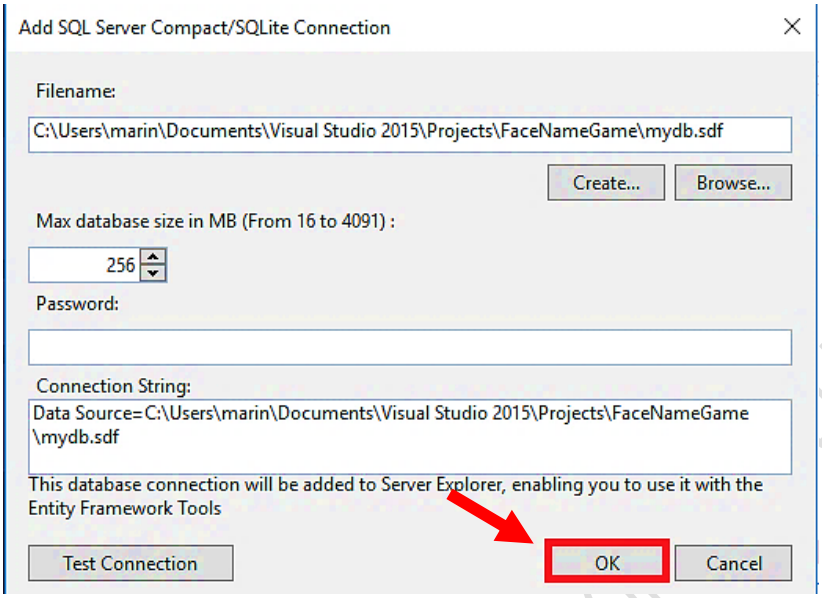
Εικόνα 5.16: Καρτέλα δημιουργίας αρχείου βάσης .sdf

Στην συνέχεια, εμφανίζεται το παράθυρο «Create New Database File». Στο πλαίσιο File Name εισάγουμε το όνομα αρχείου βάσης το οποίο επιθυμούμε, και πατάμε «Save».



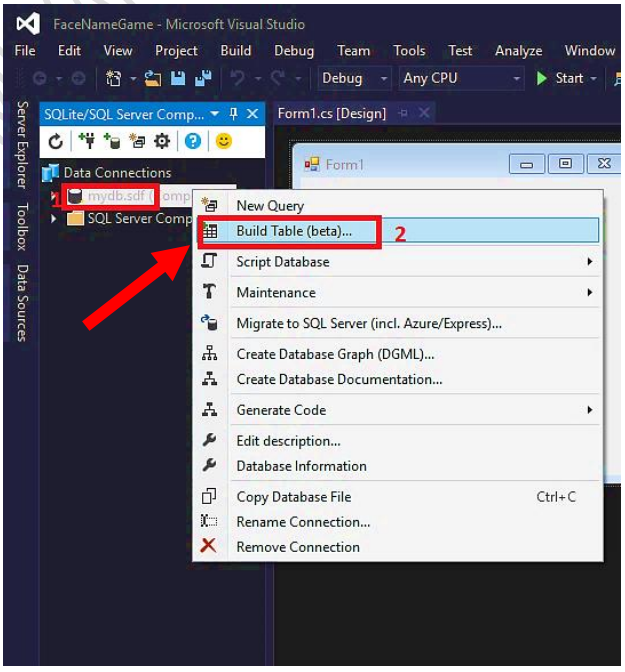
Εικόνα 5.17: Δημιουργία αρχείου βάσης .sdf

Αφού πατήσουμε το κουμπί «Save», επιστρέφουμε στην προηγούμενη καρτέλα. Πατάμε το κουμπί «OK». Το αρχείο βάσης είναι έτοιμο.



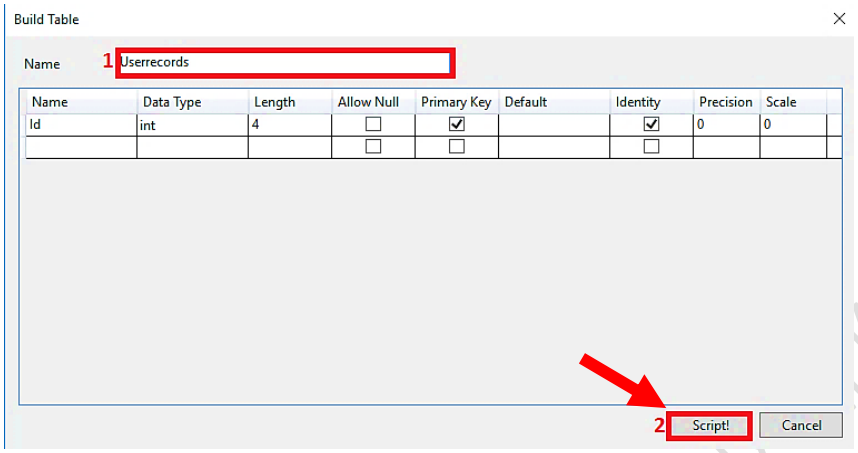
Εικόνα 5.18: Τελικό βήμα δημιουργίας αρχείου βάσης

Στην καρτέλα «SQLite/ SQL Server Compact Toolbox», πατάμε δεξί κλικ στην βάση που δημιουργήσαμε, και επιλέγουμε «Build Table» από το μενού το οποίο θα εμφανιστεί, για να προσθέσουμε έναν νέο πίνακα στη βάση δεδομένων.

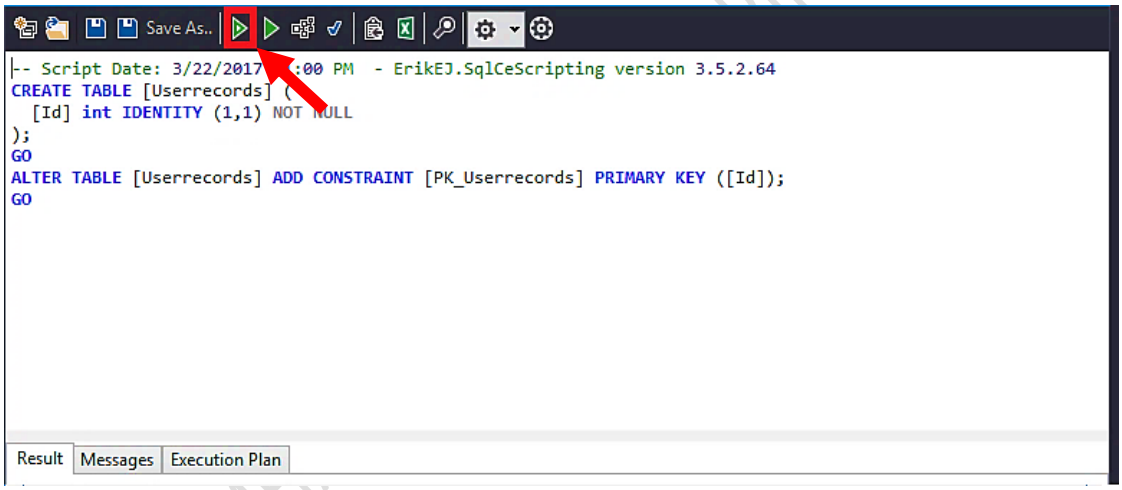


Εικόνα 5.19: Δημιουργία πίνακα στη βάση

Στη συνέχεια, προκύπτει ένα νέο παράθυρο. Αφού εισάγουμε το όνομα του πίνακα στο πεδίο κειμένου «Name» και τα χαρακτηριστικά για αυτόν τον πίνακα, πατάμε το κουμπί «Script!».



Εικόνα 5.20: Εισαγωγή χαρακτηριστικών στον πίνακα



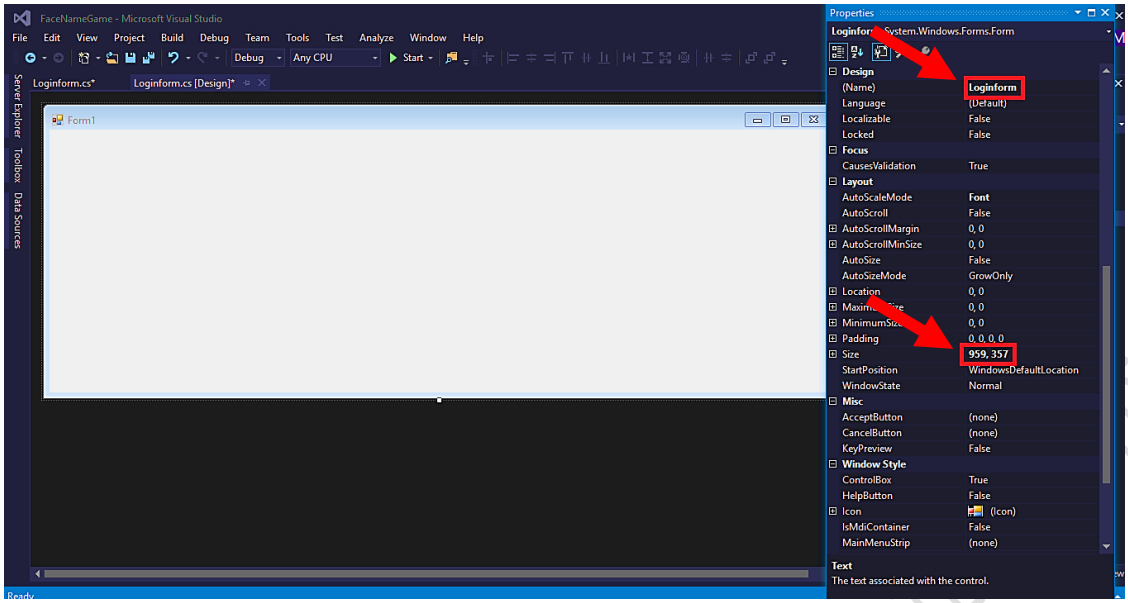
Εικόνα 5.21: Εκτέλεση queries για την δημιουργία του πίνακα

Για την παρούσα εφαρμογή ήταν απαραίτητη η δημιουργία τριών πινάκων.

5.2.3 Δημιουργία φόρμας εισαγωγής (log-in form)

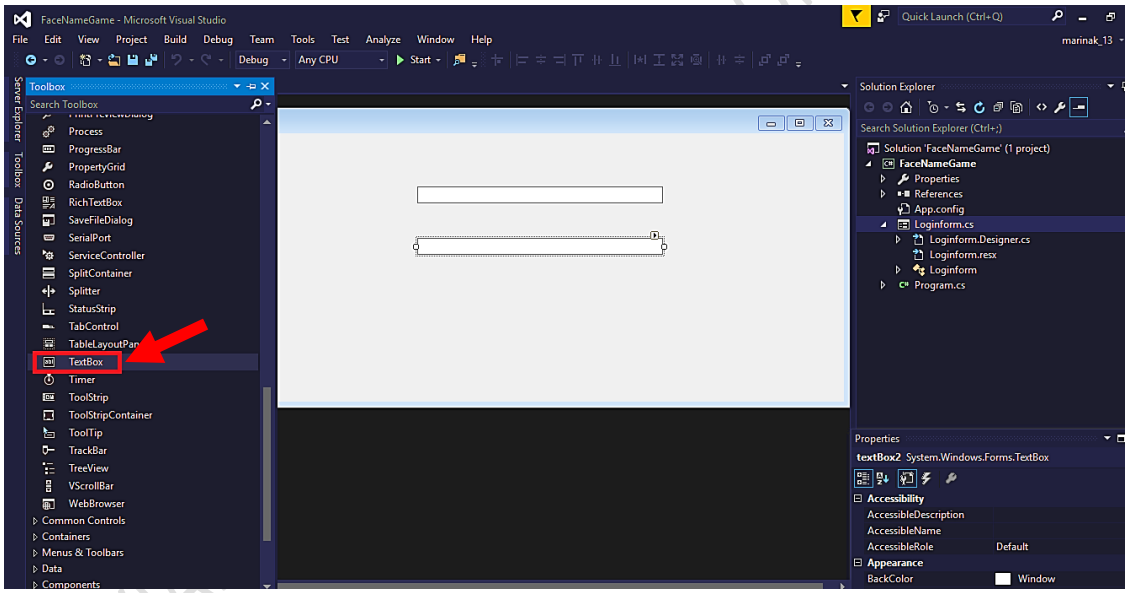
Αποτελεί την φόρμα στην οποία ο χρήστης εισάγει τα στοιχεία του, προκειμένου να αποκτήσει πρόσβαση στο παιχνίδι.

Αρχικά, μεταβάλλουμε το μέγεθος της φόρμας σε 959, 357 pixels και αλλάζουμε το όνομα σε «Loginform».



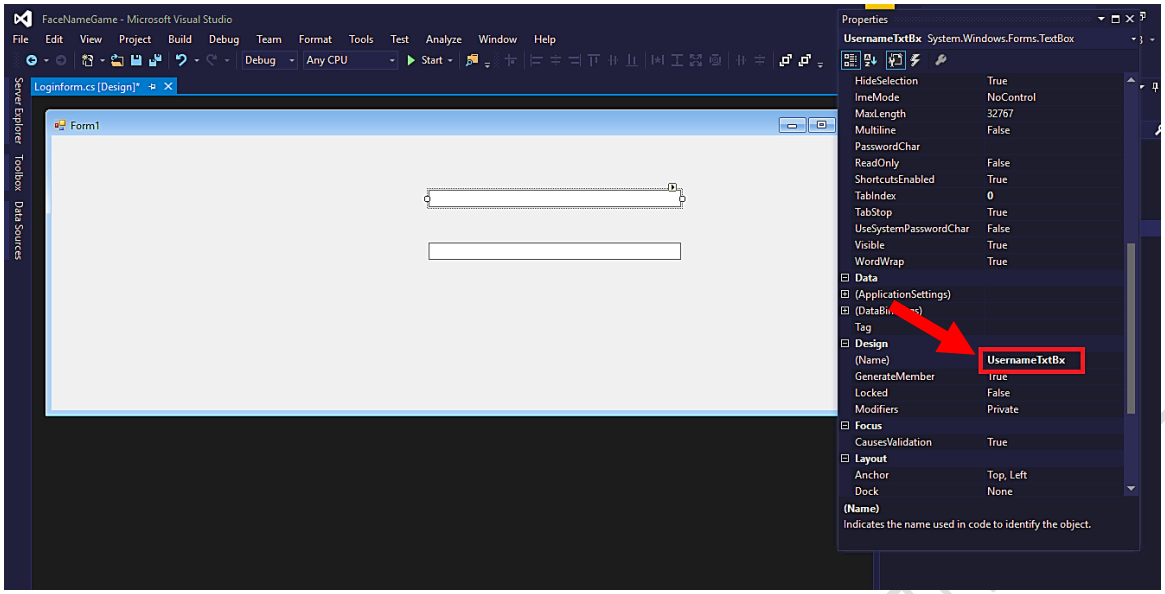
Εικόνα 5.22: Μεταβολή μεγέθους και ονόματος φόρμας Loginform

Έπειτα εισάγουμε δύο textboxes, χρησιμοποιώντας το toolbox.



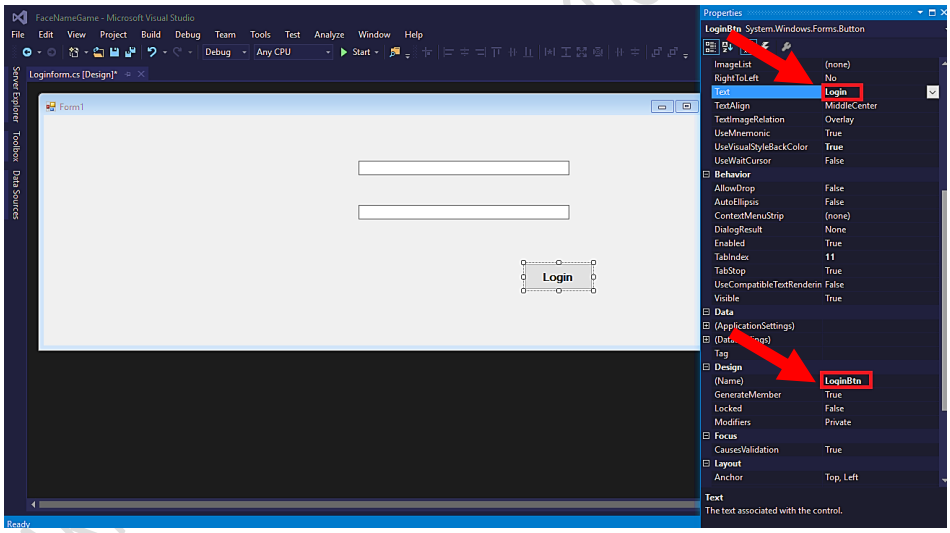
Εικόνα 5.23: Εισαγωγή textboxes στην φόρμα

Αλλάζουμε τα ονόματα των textboxes σε «UsernameTxtBx» και «PasswordTxtBx» αντίστοιχα.



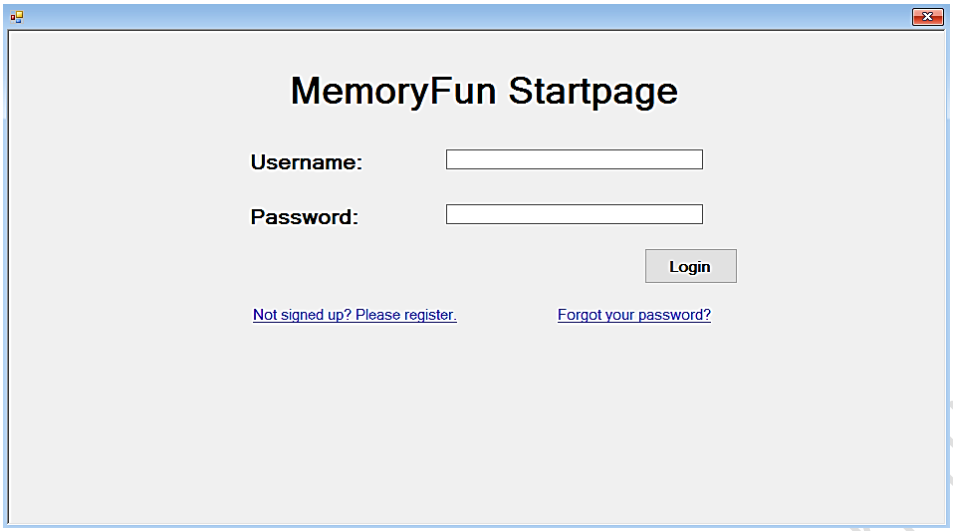
Εικόνα 5.24: Μεταβολή ονομάτων των textboxes

Ακολουθεί η εισαγωγή ενός Button στη φόρμα. Μεταβάλλουμε το κείμενο του κουμπιού σε «Login», και το όνομά του σε «LoginBtn».



Εικόνα 5.25: Εισαγωγή Login button

Με τον ίδιο τρόπο που αναφέρθηκε παραπάνω, εισάγουμε στην φόρμα labels, και άλλα δύο buttons («RegisterLinkBtn» και «ForgotPasswordLinkBtn»). Η τελική μορφή της φόρμας Login είναι η εξής:



Εικόνα 5.26: Φόρμα LoginForm

Στο πλαίσιο κειμένου Username (UsernameTxtBx), ο χρήστης εισάγει το όνομα χρήστη (Username) και στο Password (PasswordTxtBx). Μόλις πατήσει το κουμπί Login, τα στοιχεία που εισήγαγε ταυτοποιούνται με τα στοιχεία τα οποία βρίσκονται αποθηκευμένα στον πίνακα Registrations, στην βάση δεδομένων. Αν τα στοιχεία ταυτοποιηθούν σωστά, ο παίκτης αποκτά πρόσβαση στο παιχνίδι. Στην αντίθετη περίπτωση, ένα μήνυμα στην οθόνη πληροφορεί τον χρήστη ότι εισήγαγε λάθος στοιχεία. Αυτό οφείλεται στο εξής τμήμα του κώδικα:

```
    private void LoginBtn_Click(object sender, EventArgs e)
{

    string ConnectionString = @"Data Source =" + Application.StartupPath + @"\mydb.sdf";
    SqlCeConnection con = new SqlCeConnection(@"Data Source=" + Application.StartupPath +
@"\mydb.sdf");
    System.Data.SqlServerCe.SqlCeCommand cmd = new
System.Data.SqlServerCe.SqlCeCommand(ConnectionString);
    cmd.CommandType = System.Data.CommandType.Text;
    string userid = UsernameTxtBx.Text;
    LoginInfo.UserID = userid;
    string password = PasswordTxtBx.Text;
    cmd = new SqlCeCommand("select * from registrations where Username= '" + userid +
"'and Password='" + password + "'", con);
    SqlCeDataAdapter da = new SqlCeDataAdapter(cmd);
    DataTable dt = new DataTable();
    da.Fill(dt);


    if (dt.Rows.Count > 0)
    {
        MessageBox.Show("Login success !");
        Directionsscreen main = new Directionsscreen();
        main.Show();
        this.Hide();
    }
    else
    {
```

```
            MessageBox.Show("Invalid Login please check username and password");
        }


    }
```

Εκτός από το «LoginBtn» , στην φόρμα υπάρχουν και άλλα δυο κουμπιά, το «RegisterLinkBtn» και το « ForgotPasswordLinkBtn».

Το «RegisterLinkBtn», οδηγεί στην φόρμα εγγραφής (RegistrationForm), όπου ο χρήστης έχει τη δυνατότητα να εγγραφεί στο παιχνίδι, καταχωρώντας τα στοιχεία τα οποία ζητούνται. Το τμήμα κώδικα το οποίο εκτελεί τη συγκεκριμένη λειτουργία είναι το εξής:

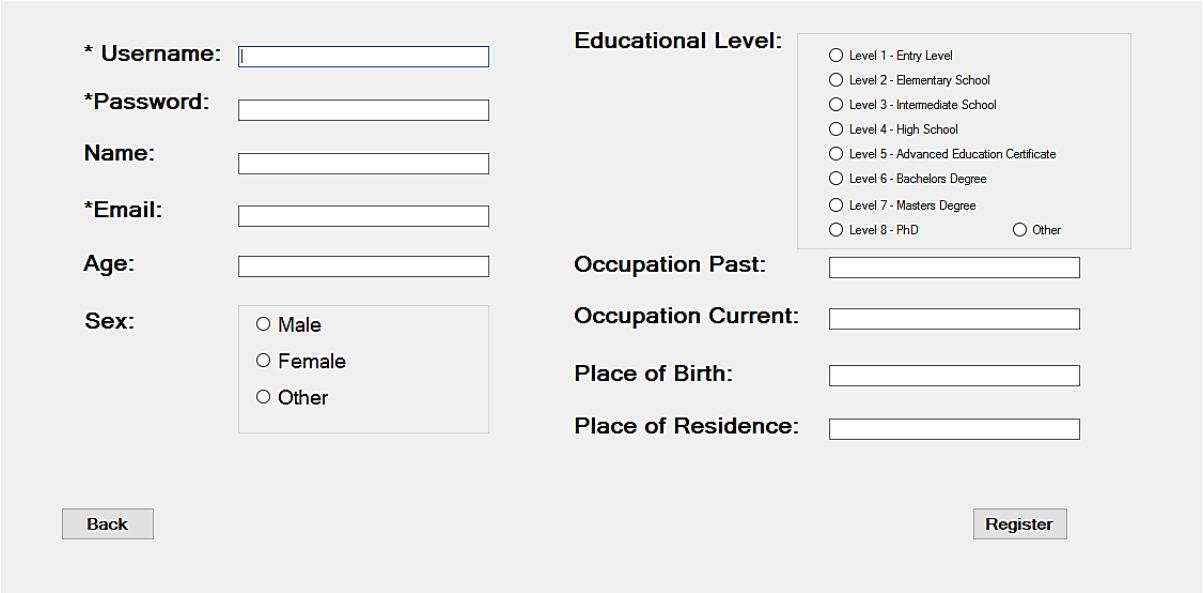
```
private void RegisterLinkBtn_Click(object sender, EventArgs e)
        {
            RegisterForm registerfrm = new RegisterForm();
            registerfrm.Show();
            this.Hide();
                                                }
```

Το «ForgotPasswordLinkBtn», οδηγεί στην φόρμα υπενθύμισης κωδικού πρόσβασης (ForgotPasswordForm), όπου ο χρήστης έχει τη δυνατότητα να ανακτήσει τον κωδικό πρόσβασής του, καταχωρώντας τα στοιχεία τα οποία ζητούνται. Το τμήμα κώδικα το οποίο εκτελεί τη συγκεκριμένη λειτουργία είναι το εξής:

```
private void ForgotPasswordLinkBtn_Click(object sender, EventArgs e)
        {
            ForgotPasswordForm forgotpassform = new ForgotPasswordForm();
            forgotpassform.Show();
            this.Hide();
        }
```

5.2.4 *Δημιουργία φόρμας εγγραφής (registration form)*

Η φόρμα εγγραφής συμπληρώνεται από τον χρήστη όταν εκείνος αποκτήσει πρόσβαση στο παιχνίδι για πρώτη φορά.



Εικόνα 5.27: Φόρμα εγγραφής χρήστη (RegisterForm)

Η φόρμα αυτή αποτελείται από εννιά textboxes. Ένα για την εισαγωγή του Username του χρήστη (RegUsernameTxtBx), για την εισαγωγή του κωδικού πρόσβασης (RegPasswordTxtBx), του πραγματικού ονόματος του χρήστη (RegNameTxtBx), της διεύθυνσης ηλεκτρονικού ταχυδρομείου (RegEmailTxtBx), της ηλικίας του χρήστη (AgeTxtBx), της προηγούμενης (OccupationpastTxtBx) και της τρέχουσας (OccupationcurrentTxtBx) απασχόλησης του χρήστη, καθώς και του τόπου γέννησης (PlaceofbirthTxtBx) και του τόπου κατοικίας του (PlaceofrecidenseTxtBx). Επιπλέον, στην φόρμα υπάρχουν δύο ομάδες από radiobuttons. Η πρώτη εξ' αυτών, δίνει στον χρήστη τη δυνατότητα να επιλέξει το φύλο του (Sexgroup), ενώ η δεύτερη αφορά το επίπεδο ακαδημαϊκής του μόρφωσης (Educationgroup). Ο χρήστης έχει την δυνατότητα να δηλώσει ψευδή στοιχεία αν το επιθυμεί για λόγους ανωνυμίας, όμως ακόμα και αν δηλώσει τα αληθή του στοιχεία, εκείνα δεν θα χρησιμοποιηθούν για διαφορετικό σκοπό πέραν της συλλογής στατιστικών για ερευνητικούς σκοπούς.

Επιπλέον στην φόρμα υπάρχουν δύο Buttons. Το ένα εξ' αυτών είναι το κουμπί καταχώρησης στοιχείων στην βάση (RegRegisterBtn). Μόλις ο χρήστης συμπληρώσει όλα τα πεδία και πατήσει το κουμπί, ξεκινάει η διαδικασία καταχώρησης των στοιχείων στην βάση. Με άλλα λόγια , αφού ελεγχθούν όλα τα πεδία, τα στοιχεία καταχωρούνται στον πίνακα Registrations της βάσης δεδομένων. Έπειτα, εμφανίζεται στην οθόνη ένα μήνυμα, το οποίο πληροφορεί τον χρήστη ότι η εγγραφή του πραγματοποιήθηκε επιτυχώς, και στη συνέχεια «καθαρίζονται» όλα τα πεδία. Αυτή η διαδικασία περιγράφεται στο παρακάτω τμήμα κώδικα:

```
private void RegRegisterBtn_Click(object sender, EventArgs e)
        {


            bool errors = false;
            Regex namerx = new Regex(@"^[A-Za-zÀ-ú]+$");
            Regex emailrx = new Regex(@"^([\w\.\-]+)@([\w\-]+)((\.(\w){2,3})+)$");


            if (string.IsNullOrWhiteSpace(RegUsernameTxtBx.Text))
            {
                MessageBox.Show("Please insert a valid username!");
                errors = true;
            }


            if (string.IsNullOrWhiteSpace(RegPasswordTxtBx.Text))
            {
                MessageBox.Show("Please insert a password!");
                errors = true;
            }



           Match match2 = emailrx.Match(RegEmailTxtBx.Text);
            if (!match2.Success)
            {
                MessageBox.Show("not valid email address! Please check again.");
                errors = true;
            }


                if (errors)
                {
                    return;
                }
```

```
            string ConnectionString = @"Data Source ="+ Application.StartupPath +
@"\mydb.sdf";
            SqlCeConnection sqlConnection1 = new SqlCeConnection(@"Data
Source="+Application.StartupPath + @"\mydb.sdf");
            System.Data.SqlServerCe.SqlCeCommand cmd = new
System.Data.SqlServerCe.SqlCeCommand(ConnectionString);
            cmd.CommandType = System.Data.CommandType.Text;
            cmd.CommandText = "insert into registrations (Username, Password, Name, Email,
Educationallevel, Sex, Occupationpast, Occupationcurrent, Placeofbirth, Placeofresidence,
Age) values(@Username, @Password, @Name, @Email, @Educationallevel, @Sex, @Occupationpast,
@Occupationcurrent, @Placeofbirth, @Placeofresidence, @Age)";
            cmd.Connection = sqlConnection1;




            {


                cmd.Parameters.AddWithValue("@Username", RegUsernameTxtBx.Text);
                cmd.Parameters.AddWithValue("@Password", RegPasswordTxtBx.Text);
                cmd.Parameters.AddWithValue("@Name", RegNameTxtBx.Text);
                cmd.Parameters.AddWithValue("@Email", RegEmailTxtBx.Text);
                cmd.Parameters.AddWithValue("@Age", AgeTxtBx.Text);
                cmd.Parameters.AddWithValue("@Sex", GetSelectedRadioButtonText(Sexgroup));
                cmd.Parameters.AddWithValue("@Educationallevel",
GetSelectedRadioButtonText(EducationGroup));
                cmd.Parameters.AddWithValue("@Occupationpast", OccupationpastTxtBx.Text);
                cmd.Parameters.AddWithValue("@Occupationcurrent",
OccupationcurrentTxtBx.Text);
                cmd.Parameters.AddWithValue("@Placeofbirth", PlaceofbirthTxtBx.Text);
                cmd.Parameters.AddWithValue("@Placeofresidence",
PlaceofresidenceTxtBx.Text);



                sqlConnection1.Open();
                cmd.ExecuteNonQuery();


               if (!System.IO.Directory.Exists("" + Application.StartupPath +
@"\registrationlogs\" + RegUsernameTxtBx.Text + ""))
               {
                   System.IO.Directory.CreateDirectory("" + Application.StartupPath +
@"\registrationlogs\" + RegUsernameTxtBx.Text + "");
               }


               string strSQL = "SELECT * FROM registrations where Username= '" +
RegUsernameTxtBx.Text + "'";
               SqlCeDataAdapter dt = new SqlCeDataAdapter(strSQL, sqlConnection1);
               DataSet ds = new DataSet();
               dt.Fill(ds, "registrationlogs");
               ds.WriteXml(("" + Application.StartupPath + @"\registrationlogs\" +
RegUsernameTxtBx.Text + @"\" + RegUsernameTxtBx.Text+ "_"+"userinfo"));

               sqlConnection1.Close();
            }
```

Αφού πραγματοποιηθεί η εγγραφή, τα στοιχεία του χρήστη αποθηκεύονται στον φάκελο registrationlogs, σε ξεχωριστό φάκελο για κάθε χρήστη. Η μόνη χρήση αυτού του φακέλου, είναι η σύγκριση των δεδομένων του χρήστη (όπως ηλικία, εκπαίδευση, τόπος διαμονής κλπ), με σκοπό την εξαγωγή στατιστικών στοιχείων, τα οποία θα συμβάλλουν στην έρευνα.

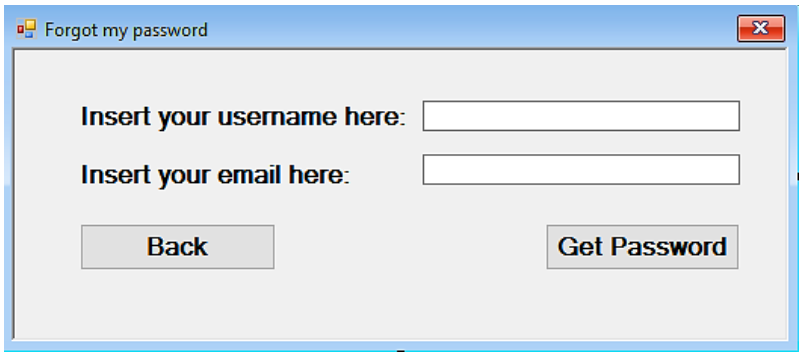
```
            RegUsernameTxtBx.Clear();
            RegPasswordTxtBx.Clear();
            RegNameTxtBx.Clear();
            RegEmailTxtBx.Clear();
            AgeTxtBx.Clear();
            OccupationcurrentTxtBx.Clear();
            OccupationcurrentTxtBx.Clear();
            PlaceofresidenceTxtBx.Clear();
            PlaceofbirthTxtBx.Clear();



    }
```

Στη φόρμα βρίσκεται επίσης και το κουμπί εξόδου (RegBackBtn), το οποίο δίνει την δυνατότητα στον χρήστη να επιστρέψει στην οθόνη log-in.

```
private void RegBackBtn_Click(object sender, EventArgs e)
        {
            LoginForm loginfrm = new LoginForm();
            this.Hide();
            loginfrm.Show();
        }
```

5.2.5 Δημιουργία φόρμας ανάκτησης κωδικού πρόσβασης (forgot password form)

Η φόρμα ανάκτησης κωδικού πρόσβασης δίνει στον χρήστη την δυνατότητα να ανακτήσει τον κωδικό πρόσβασής του, σε περίπτωση που τον ξεχάσει.



Εικόνα 5.28: Φόρμα ανάκτησης κωδικού πρόσβασης (Forgot my password form)

Η φόρμα ανάκτησης κωδικού πρόσβασης αποτελείται από δυο textboxes, στα οποία ο χρήστης πρέπει να εισάγει τα στοιχεία που ζητούνται, έτσι ώστε να επαληθευτεί η ταυτότητά του, και να ανακτήσει τον κωδικό πρόσβασής του. Στο πρώτο textbox (ForgotPasswordUsernameTxtBx), ο χρήστης καλείται να εισάγει το όνομα χρήστη του, ενώ στο δεύτερο (ForgotPasswordEmailTxtBx), πρέπει να εισάγει το email το οποίο έχει δηλώσει στην φόρμα εγγραφής.

Στη φόρμα επίσης συμπεριλαμβάνονται δύο κουμπιά. Το πρώτο κουμπί (BackBtn), οδηγεί τον χρήστη πίσω στην φόρμα log- in.

```
private void BackBtn_Click(object sender, EventArgs e)
        {
            LoginForm loginfrm = new LoginForm();
            this.Hide();
            loginfrm.Show();
        }
```

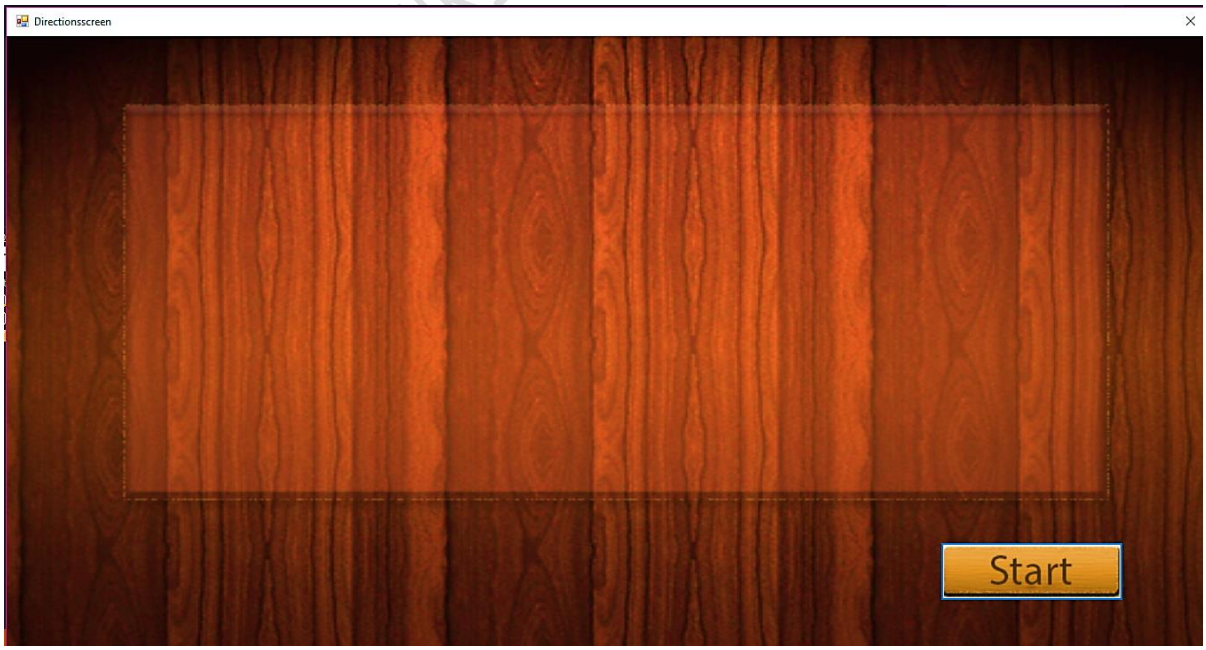
Το δεύτερο κουμπί (GetPasswordBtn), είναι το κουμπί το οποίο χρησιμοποιείται για την ανάκτηση κωδικού πρόσβασης και περιέχει το τμήμα του κώδικα το οποίο είναι υπεύθυνο για την ανάκτηση του password, και πραγματοποιεί όλους τους απαραίτητους ελέγχους.

```
private void GetPasswordBtn_Click(object sender, EventArgs e)
        {
            bool errors = false;
            Regex emailrx = new Regex(@"^([\w\.\-]+)@([\w\-]+)((\.(\w){2,3})+)$");

            if (string.IsNullOrWhiteSpace(ForgotPasswordUsernameTxtBx.Text))
            {
                MessageBox.Show("Please insert your username!");
                errors = true;
            }

            Match match = emailrx.Match(ForgotPasswordEmailTxtBx.Text);
            if (!match.Success)
            {
                MessageBox.Show("not walid email address! Please check again.");
                errors = true;
            }


            if (errors)
            {
                return;
            }

            string ConnectionString = @"Data Source =" + Application.StartupPath + @"\mydb.sdf";
            SqlCeConnection con = new SqlCeConnection(@"Data Source=" + Application.StartupPath
+ @"\mydb.sdf");
            string userid = ForgotPasswordUsernameTxtBx.Text;
            string email = ForgotPasswordEmailTxtBx.Text;


            SqlCeCommand cmd = new SqlCeCommand("select Password from registrations where
Username= '" + userid + "'and Email='" + email + "'", con);
```

```
SqlCeDataAdapter da = new SqlCeDataAdapter(cmd);
DataTable dt = new DataTable();

da.Fill(dt);


if (dt.Rows.Count > 0)
{
    MessageBox.Show("Login success !");
    Directionsscreen main = new Directionsscreen() ;
    main.Show();
    this.Hide();
}
else
{
    MessageBox.Show("Invalid Login please check username and password");
}
```

5.2.6 Δημιουργία φόρμας οθόνης οδηγιών (directions screen form)

Η φόρμα οθόνης οδηγιών, εκτελεί δυο λειτουργείες. Ο προφανής της ρόλος, είναι η πληροφόρηση του χρήστη σχετικά με τις οδηγίες του παιχνιδιού.



Εικόνα 5.29: Η οθόνη οδηγιών (Directions screen)

Η δεύτερη λειτουργία της φόρμας, οφείλεται στο τμήμα του κώδικα, το οποίο βρίσκεται στο κουμπί εκκίνησης «Start» (StartBtn).

```
private void StartBtn_Click(object sender, EventArgs e)
        {


            string ConnectionString = @"Data Source =" + Application.StartupPath +
@"\mydb.sdf";
            SqlCeConnection con = new SqlCeConnection(@"Data Source=" +
Application.StartupPath + @"\mydb.sdf");

            var sql = @"SELECT TOP 1 Points FROM Userrecords WHERE Username =
@UserName ORDER BY ID DESC";
            SqlCeCommand cmd = new SqlCeCommand(sql, con);
            cmd.Parameters.Add("@UserName", LoginInfo.UserID);



            con.Open();
            point = Convert.ToInt32(cmd.ExecuteScalar());
            con.Close();



            if ((point >= 0) && (point < 130)) {
                LevelOnePartOne levelonepartone = new LevelOnePartOne();
                this.Hide();
                levelonepartone.Show();
            }


           else if ((point >= 130) && (point < 325))
            {
                LevelTwoPartOne leveltwopartone = new LevelTwoPartOne();
                this.Hide();
                leveltwopartone.Show();
            }

            else if ((point >= 325) && (point < 700))
            {
                LevelThreePartOne levelthreepartone = new LevelThreePartOne();
                this.Hide();
                levelthreepartone.Show();
            }


            else if (point <= 700)
            {
                LevelFourPartOne levelfourpartone = new LevelFourPartOne();
                this.Hide();
                levelfourpartone.Show();
            }
        }
```

5.2.7 Δημιουργία φορμών κυρίως μέρους παιχνιδιού (leveltwopartone / leveltwoparttwo)

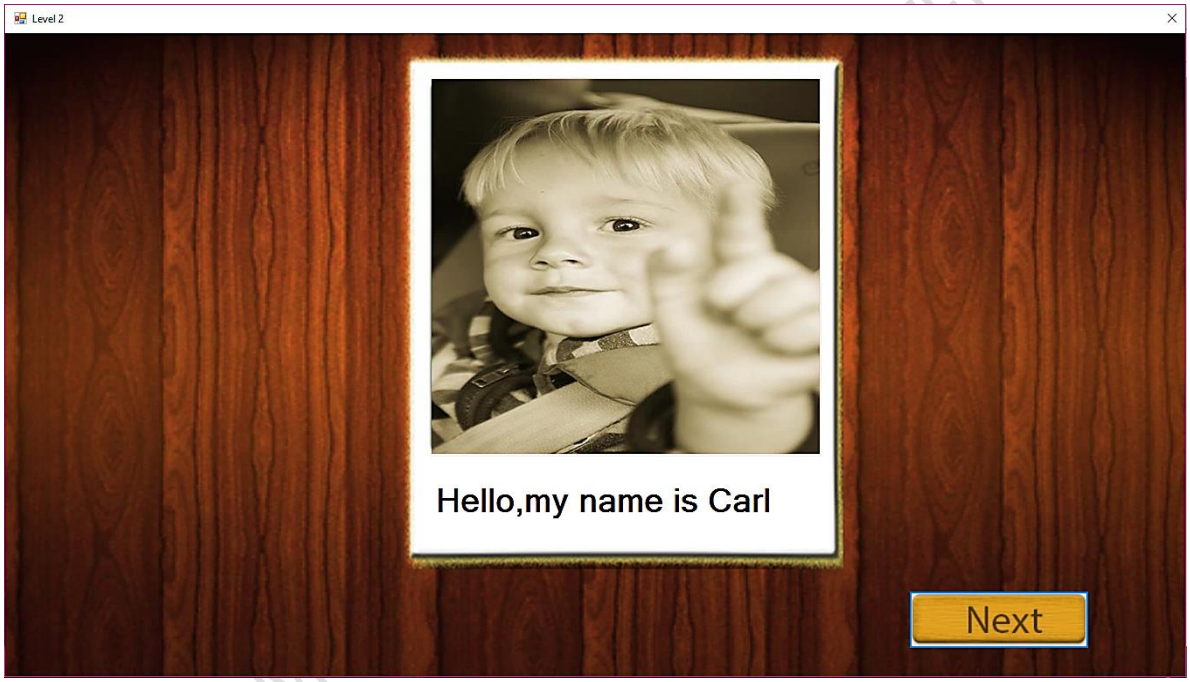
Το παιχνίδι αποτελείται από τέσσερα επίπεδα. Καθώς όμως οι φόρμες σε κάθε επίπεδο δεν διαφέρουν πολύ μεταξύ τους, θα αναλύσουμε την διαδικασία δημιουργίας των φορμών του δεύτερου επιπέδου (level 2) του παιχνιδιού.

Κάθε επίπεδο αποτελείται από δύο φάσεις:

- Την πρώτη φάση της προβολής των φωτογραφιών
- Την δεύτερη φάση της αναγνώρισης προσώπων

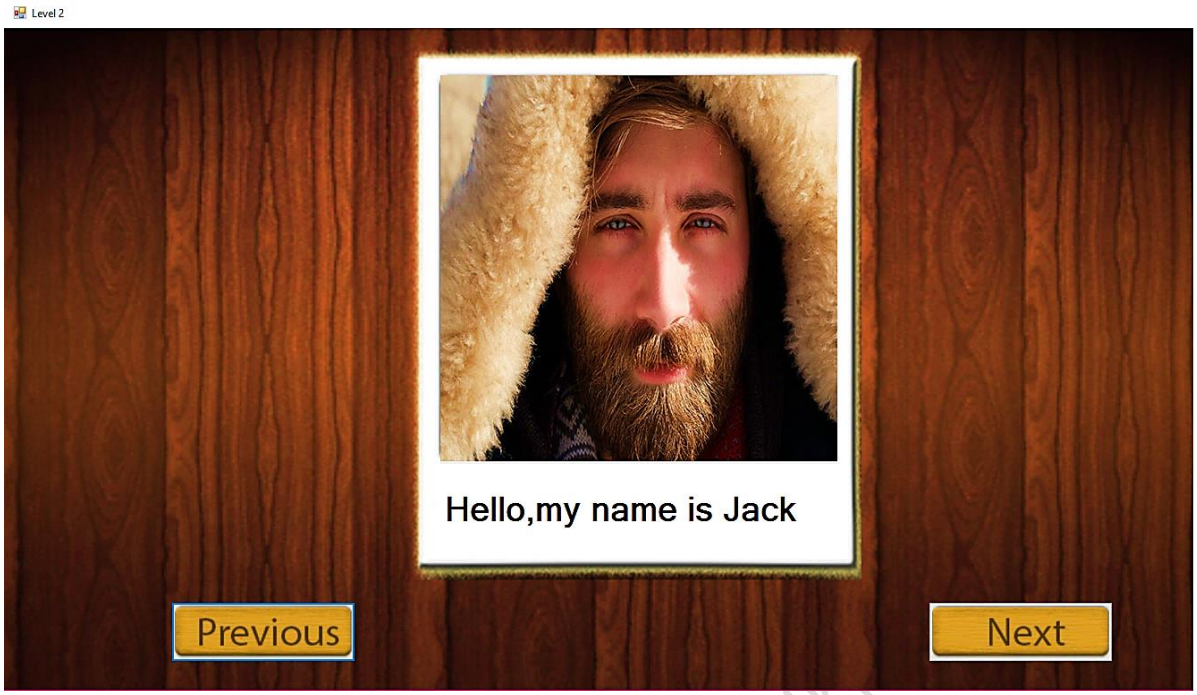
Καθώς η πρώτη και η δεύτερη φάση διαφέρουν σημαντικά μεταξύ τους, ήταν απαραίτητη η δημιουργία μιας εντελώς ξεχωριστής φόρμας για την κάθε μια.

Αρχικά παρουσιάζεται η φόρμα για την πρώτη φάση, δηλαδή την προβολή φωτογραφιών.



Εικόνα 5.30: Προβολή της πρώτης στη σειρά φωτογραφίας της πρώτης φάσης του 2ου επιπέδου

Η φόρμα αποτελείται από ένα Picturebox (pictureBox1). Όταν προβάλλεται η πρώτη φωτογραφία, υπάρχει στην φόρμα και ένα κουμπί, το οποίο φέρει το κείμενο «Next» (NextBtn) και επιτρέπει στο χρήστη να προβάλλει την επόμενη σε σειρά φωτογραφία.



Εικόνα 5.31: Προβολή μιας ενδιάμεσης φωτογραφίας της πρώτης φάσης του 2ου επιπέδου

Όταν προβάλλεται κάποια ενδιάμεση φωτογραφία, εμφανίζεται στη φόρμα ένα επιπλέον κουμπί, το οποίο φέρει το κείμενο «Previous» (PreviousBtn), το οποίο δίνει την δυνατότητα στον χρήστη να προβάλλει την προηγούμενη σε σειρά εικόνα.

Εικόνα 5.32: Προβολή της τελευταίας σε σειρά φωτογραφίας της πρώτης φάσης του 2ου επιπέδου

Όταν προβάλλεται η τελευταία φωτογραφία, το κουμπί NextBtn αντικαθίσταται από ένα διαφορετικό κουμπί, το οποίο φέρει το κείμενο «Proceed» (ProceedBtn). Το κουμπί αυτό δίνει τη δυνατότητα στο χρήστη να προχωρήσει στη δεύτερη φάση του επιπέδου.

Παρακάτω, περιγράφεται αναλυτικά κάθε τμήμα του κώδικα της πρώτης φάσης του δεύτερου επιπέδου.

Αρχικά περιγράφονται οι δηλώσεις μεταβλητών.

```
        int m, piccount=0, k=0;
        Random rd = new Random();
        public string[] picturenames= new string[6];
        string sourcePath = "" + Application.StartupPath + @"\images\Level2";
        string targetPath = "" + Application.StartupPath + @"\images\Level2\used";
        bool directoryexists = false;
        string[] files = Directory.GetFiles("" + Application.StartupPath +
@"\images\Level2", "*.jpg", SearchOption.AllDirectories);
```

- Οι μεταβλητές m, piccount και k, ορίζουν ακαίρους αριθμούς. Οι μεταβλητές m και k λειτουργούν ως δείκτες θέσεων, ενώ η piccount ως μετρητής για τις εικόνες που έχουν προβληθεί.
- Η rd δηλώνει μια random (τυχαία) μεταβλητή.
- Ο πίνακας συμβολοσειρών (string []) picturenames, ορίζει έναν πίνακα με έξι θέσεις, για τα ονόματα των φωτογραφιών.
- Οι συμβολοσειρές Sourcepath και Targetpath, ορίζουν την διαδρομή προέλευσης και προορισμού των εικόνων αντίστοιχα.
- Η Boolean μεταβλητή directoryexists, αποτελεί ένα flag, το οποίο υποδεικνύει αν υπάρχει ή όχι, ο φάκελος προορισμού των φωτογραφιών.
- Ο πίνακας συμβολοσειρών (string []) files συλλέγει τα ονόματα όλων των εικόνων με επέκταση .jpg, οι οποίες βρίσκονται στον φάκελο.

Έπειτα, αναλύεται ο κώδικας, ο οποίος τίθεται σε εφαρμογή αμέσως μόλις φορτώσει η φόρμα

```
         private void LevelTwoPartOne_Load(object sender, EventArgs e)
        {


            ProceedBtn.Hide();
            PreviousBtn.Hide();


            if (!System.IO.Directory.Exists(targetPath))
            {
                System.IO.Directory.CreateDirectory(targetPath);
                directoryexists = true;

            }

            else
            {
                clearFolder(targetPath);
                System.IO.Directory.CreateDirectory(targetPath);
                directoryexists = true;
            }

            Random rand = new Random();
            string[] files =
Directory.GetFiles(""+Application.StartupPath+@"\images\Level2", "*.jpg",
SearchOption.AllDirectories);


             for (m = 0; m < 5; m++)

            {

               Loop:
                picturenames[m] = files[rand.Next(files.Length)];
                string name1 = picturenames[m].ToString();
                string name2 = name1.Substring(name1.LastIndexOf("\\"));
                string name3 = name2.Remove(name2.Length - 4, 4);
                string name = name3.Remove(0, 1);
                string filename = name2.TrimStart('\\');


                if (!IsDirectoryEmpty(targetPath))
                {


                    if (File.Exists("" + Application.StartupPath +
@"images\Level2\used\" + filename))
                    {

                        goto Loop;



                    }

                }
```

```
            string sourceFile = System.IO.Path.Combine(sourcePath, filename);
            string destFile = System.IO.Path.Combine(targetPath, filename);


             System.IO.File.Copy(sourceFile, destFile, true);

        }

        ShowCurrentImage(k);

    }
```

Μόλις φορτώσει η φόρμα, τα κουμπιά PreviousBtn και ProceedBtn μετατρέπονται σε κρυφά.

Έπειτα, ελέγχεται αν υπάρχει ο φάκελος προορισμού των εικόνων. Αν δεν υπάρχει, απλά δημιουργείται, και το flag directoryexists ορίζεται σε true.

Αν όμως υπάρχει, τότε το παλιό directory διαγράφεται μαζί με τα περιεχόμενά του, και στη θέση του δημιουργείται ένα νέο. Το flag directoryexists ορίζεται σε true.

Στη συνέχεια, επιλέγονται τυχαία πέντε εικόνες. Τα ονόματά τους, αντιγράφονται στον πίνακα picturenames, και υφίστανται την κατάλληλη επεξεργασία, ώστε να απομονωθούν τα ονόματα των ατόμων, από τον σύνδεσμο του directory στο οποίο βρίσκεται η φωτογραφία.

Τέλος, ελέγχεται αν κάποια φωτογραφία έχει επιλεχθεί περισσότερες από μια φορές. Αν ναι, τότε ο κώδικας επιστρέφει στο σημείο με την ετικέτα Loop, και επιλέγεται κάποια άλλη εικόνα. Εάν όχι, οι φωτογραφίες αντιγράφονται από τον φάκελο προέλευσης, στον φάκελο προορισμού. Τελικά, καλείται η συνάρτηση ShowCurrentImage, η οποία έχει ως όρισμα τον δείκτη θέσης k. Ο ρόλος αυτής της συνάρτησης, είναι να προβάλλει στο PictureBox1 την εικόνα την οποία βρίσκεται στην θέση k του πίνακα filestouse.

```
protected void ShowCurrentImage(int picindex)
        {

            string[] filestouse = Directory.GetFiles("" + Application.StartupPath +
@"\images\Level2\used\", "*.jpg", SearchOption.AllDirectories);
            pictureBox1.Load(filestouse[picindex]);
            string nameused1 = pictureBox1.ImageLocation.ToString();
            string nameused2 = nameused1.Substring(nameused1.LastIndexOf("\\"));
            string nameused3 = nameused2.Remove(nameused2.Length - 4, 4);
            string nameused = nameused3.Remove(0, 1);
            namelbl.Text = "Hello,my name is" + " " + nameused;
```

Στη συνέχεια παρουσιάζεται το τμήμα του κώδικα το οποίο αφορά την λειτουργία του κουμπιού NextBtn..

```
        private void NextBtn_Click(object sender, EventArgs e)
         {

             piccount = k - 1;
             piccount++;

                 if (k<5)
                 {

                 k = k + 1;
                 ShowCurrentImage(k);



                 }



             if (k == 4 || piccount == 4)
             {
                 NextBtn.Hide();
                 ProceedBtn.Show();
             }

             if (k >= 1)
             {
                 PreviousBtn.Show();
             }

         }
```

Ο ρόλος του NextBtn, είναι να δίνει την δυνατότητα στον χρήστη να προβάλλει την επόμενη εικόνα. Για να το επιτύχει αυτό, απλά αυξάνει τον αριθμό k κατά 1. Όταν ο αριθμός k γίνει ίσος με 4 (δηλαδή όταν προβάλλεται η πέμπτη και τελευταία εικόνα του επιπέδου) , το NextBtn γίνεται κρυφό, ενώ εμφανίζεται το ProceedBtn, το οποίο επιτρέπει στον χρήστη να προχωρήσει στο επόμενο στάδιο του επιπέδου. Αν το k έχει τιμή μεγαλύτερη ή ίση του 1 , δηλαδή αν προβάλλεται οποιαδήποτε άλλη εικόνα εκτός από την πρώτη, τότε το PreviousBtn γίνεται ορατό

```
private void PreviousBtn_Click(object sender, EventArgs e)
        {
            piccount--;

            if (k >= 1 && k <= 5)
                {
                    k=k-1;
                    ShowCurrentImage(k);
                }
                if ((k == 0 && piccount < 0) || (k > 0 && piccount < 0))
                {
                    PreviousBtn.Hide();
                }
        }
```

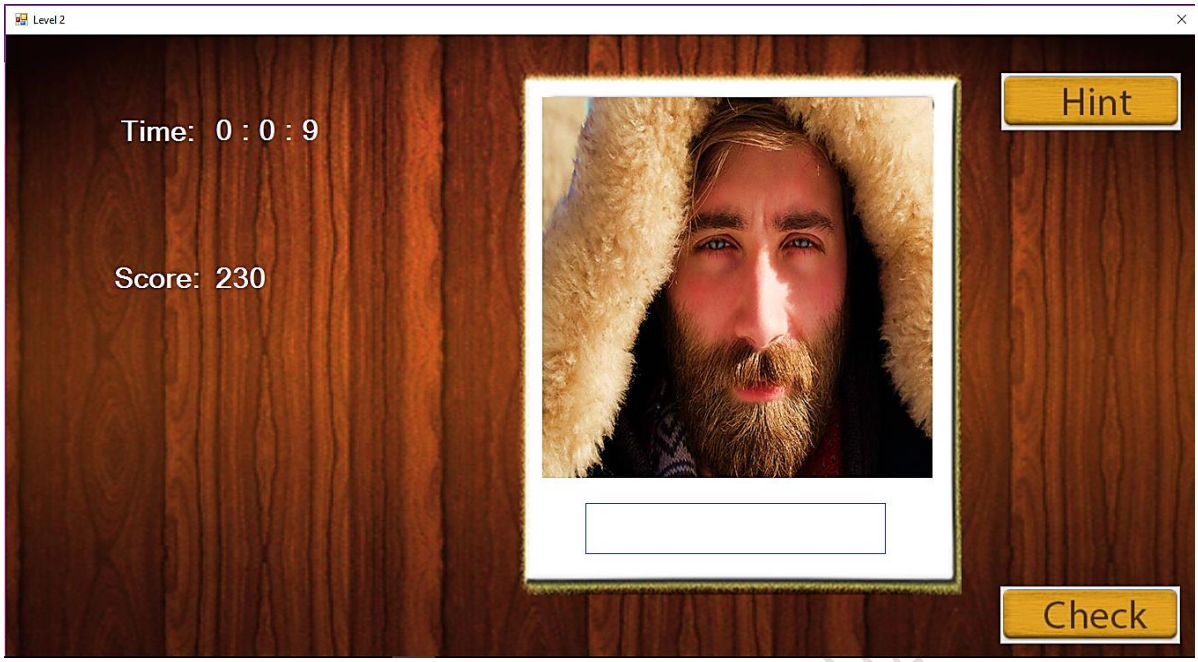
```
if (k < 4)
            {
                ProceedBtn.Hide();
                NextBtn.Show();
            }
        }

        }
```

Ο ρόλος του PreviousBtn, είναι να δίνει την δυνατότητα στον χρήστη να προβάλλει την προηγούμενη σε σειρά εικόνα. Για να το επιτύχει αυτό, απλά μειώνει τον αριθμό k κατά 1. Όταν ο αριθμός k γίνει ίσος με 4 (δηλαδή όταν προβάλλεται η πέμπτη και τελευταία εικόνα του επιπέδου) , το NextBtn γίνεται κρυφό, ενώ εμφανίζεται το ProceedBtn, το οποίο επιτρέπει στον χρήστη να προχωρήσει στο επόμενο στάδιο του επιπέδου. Αν το k έχει τιμή μεγαλύτερη ή ίση του 1 , δηλαδή αν προβάλλεται οποιαδήποτε άλλη εικόνα εκτός από την πρώτη, τότε το PreviousBtn γίνεται ορατό.

```
private void ProceedBtn_Click(object sender, EventArgs e)
        {
            LevelTwoPartTwo lvltwoprttwo = new LevelTwoPartTwo();
            this.Hide();
            lvltwoprttwo.Show();
        }
```

Ο ρόλος του ProceedBtn, είναι να δίνει την δυνατότητα στον χρήστη να μεταβεί στο επόμενο στάδιο του επιπέδου του παιχνιδιού.

Η δεύτερη φάση, δηλαδή η αναγνώριση προσώπων, αποτελεί το κυριότερο σημείο του παιχνιδιού. Ο χρήστης καλείται να αναγνωρίσει τα πρόσωπα που απεικονίζονται στις φωτογραφίες, και να θυμηθεί κάθε όνομα και τη συσχέτισή του με κάθε άτομο που προβλήθηκε στην προηγούμενη φάση. Χρήσιμα στοιχεία όπως: η βαθμολογία, ο χρόνος που χρειάστηκε ο χρήστης για να αναγνωρίσει κάθε πρόσωπο, καθώς και ο αριθμός των λανθασμένων απαντήσεων και των υποδείξεων που χρησιμοποίησε σε κάθε φωτογραφία, αποθηκεύονται σε μια βάση SQL, αλλά και σε ένα ξεχωριστό αρχείο εγγραφών σε μορφή XML. Το αρχείο αυτό καταχωρείται σε καθημερινή βάση, έτσι ώστε να υπάρχει μια σαφής εκτίμηση σύμφωνα με την πρόοδο της μνήμης του χρήστη.



Εικόνα 5.33: Στιγμιότυπο από την δεύτερη φάση του 2ου επιπέδου του παιχνιδιού

Η φόρμα διαθέτει ένα PictureBox (PhotoPicBx) στο οποίο προβάλλονται οι φωτογραφίες, ένα TextBox (NameTxtBx), στο οποίο ο χρήστης καλείται να συμπληρώσει το όνομα του εικονιζόμενου στη φωτογραφία, ένα label για την επίδειξη των πόντων (Pointlbl) και ένα το οποίο δείχνει πόσο χρόνο καταναλώνει ο χρήστης σε κάθε φωτογραφία (Timelbl), καθώς και έναν Timer (Timer1), ο οποίος υπολογίζει αυτόν τον χρόνο.

Η ανάπτυξη της φόρμας έγινε με βάση τον σχεδιασμό που έγινε κατά την διαδικασία του storyboarding. Συνεπώς ισχύει ότι, όταν ο χρήστης χρησιμοποιήσει μια βοήθεια, αφαιρούνται πέντε πόντοι από τη συνολική του βαθμολογία, ενώ αν συμπληρώσει σωστά το όνομα σε μια εικόνα, τότε προχωράει στην επόμενη, και κερδίζει 50 πόντους, οι οποίοι προστίθενται στη συνολική του βαθμολογία.

Παρακάτω, περιγράφεται αναλυτικά κάθε τμήμα του κώδικα της δεύτερης φάσης του δεύτερου επιπέδου.

Αρχικά περιγράφονται οι δηλώσεις μεταβλητών.

```
int point=0, buttoncount = 0, i = 0, h = 4, hbuttoncount = 0, errorcount=0;
```

Όλες οι μεταβλητές περιγράφουν ακεραίους.

- Η μεταβλητή point αφορά τους βαθμούς που ο παίκτης συγκεντρώνει στο παιχνίδι
- Η μεταβλητή buttoncount αφορά τον αριθμό από φορές που ο χρήστης πάτησε το κουμπί «Next».

- Η μεταβλητή hbuttoncount αποτελεί έναν μετρητή ο οποίος καταγράφει τον αριθμό από φορές που ο χρήστης πάτησε το κουμπί βοήθειας «Hint».

- Η μεταβλητή errorcount αποτελεί έναν μετρητή ο οποίος καταγράφει τον αριθμό από φορές που ο χρήστης έδωσε λανθασμένη απάντηση σε μια φωτογραφία.

Έπειτα, αναλύεται ο κώδικας, ο οποίος τίθεται σε εφαρμογή αμέσως μόλις φορτώσει η φόρμα:

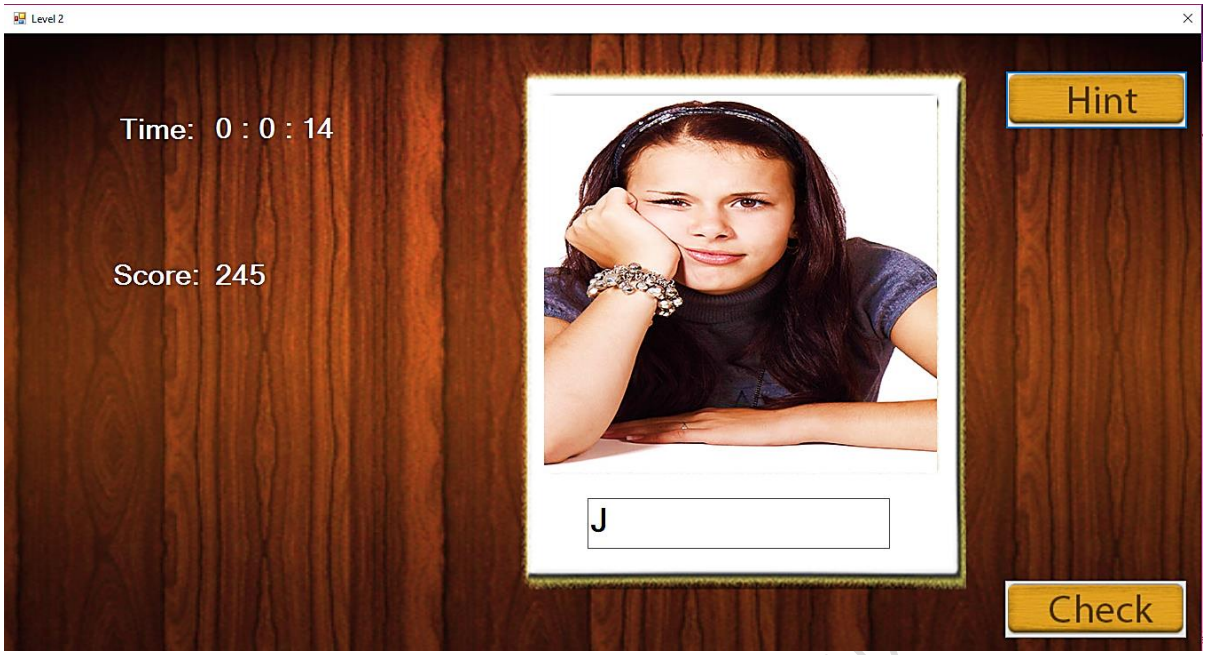
```
private void LevelTwoPartTwo_Load(object sender, EventArgs e)
        {
            string ConnectionString = @"Data Source =" + Application.StartupPath +
@"\mydb.sdf";
            SqlCeConnection con = new SqlCeConnection(@"Data Source=" +
Application.StartupPath + @"\mydb.sdf");

            var sql= @"SELECT TOP 1 Points FROM Userrecords WHERE Username = @UserName
ORDER BY ID DESC";
            SqlCeCommand cmd = new SqlCeCommand(sql, con);
            cmd.Parameters.Add("@UserName", LoginInfo.UserID);



            con.Open();

            var result = cmd.ExecuteScalar();

            string resultstr = Convert.ToString(result);

            con.Close();




            string[] files = Directory.GetFiles("" + Application.StartupPath +
@"\images\Level2\used", "*.jpg", SearchOption.AllDirectories);
            PhotoPicBx.Load(files[i]);
            NameTxtBx.Focus();
            Pointlbl.Text = resultstr;
            point = Convert.ToInt32(resultstr);


        }
```

Αρχικά, εγκαθιδρύεται μια σύνδεση με τη βάση της εφαρμογής. Αναζητείται στον πίνακα Userrecords η τελευταία εγγραφή του πίνακα, η οποία περιλαμβάνει το όνομα του χρήστη ο οποίος είναι συνδεδεμένος στο παιχνίδι. Με αυτόν τον τρόπο ανακτάται από τη βάση η βαθμολογία η οποία συγκεντρώθηκε από τον χρήστη κατά την τελευταία συνεδρία. Στη συνέχεια, φορτώνεται η πρώτη φωτογραφία από τον πίνακα files στο picurebox PhotoPicBx και ο κέρσορας κεντράρεται στο πλαίσιο κειμένου NameTxtBx, όπου ο παίκτης καλείται να δώσει την απάντηση. Οι πόντοι ανανεώνονται με κάθε αντίστοιχη ενέργεια του χρήστη και αποθηκεύονται προσωρινά στη μεταβλητή point.

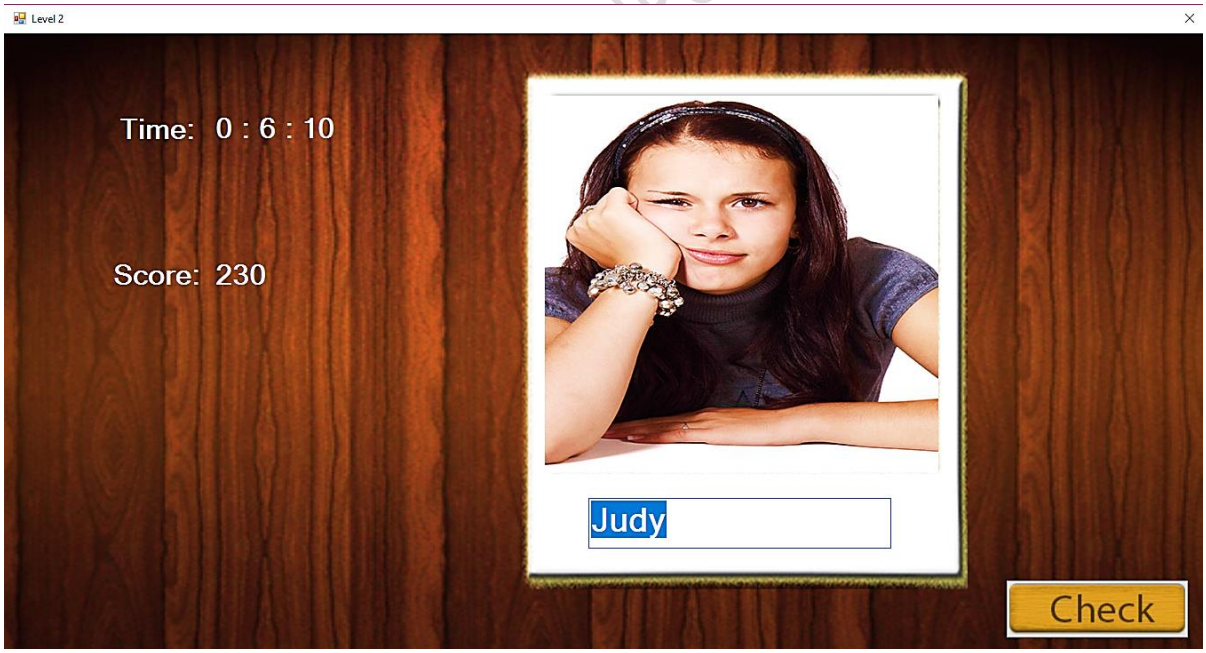
Ο χρήστης έχει τη δυνατότητα να ζητήσει βοήθεια, εάν αντιμετωπίσει δυσκολία, πατώντας το κουμπί, το οποίο φέρει το κείμενο «Hint» (HintBtn).

```
private void Hintbtn_Click(object sender, EventArgs e)
        {
            hbuttoncount++;
            h--;
            string nameused21 = PhotoPicBx.ImageLocation.ToString();
            string nameused22 = nameused21.Substring(nameused21.LastIndexOf("\\"));
            string nameused23 = nameused22.Remove(nameused22.Length - 4, 4);
            string nameused24 = nameused23.Remove(0, 1);
            string nameused2h = nameused24.Remove(nameused24.Length - h, h);
            NameTxtBx.Text = nameused2h;

            if (point > 0)
            {
                point = point - 5;
                Pointlbl.Text = point.ToString();
            }

            if (h == -1)
            {
                h = 0;
            }

            if (hbuttoncount == 4)
            {
                Hintbtn.Hide();
            }
        }
```

Κάθε φορά που ο χρήστης χρησιμοποιεί το κουμπί Hintbtn, ο μετρητής hbuttoncount αυξάνεται κατά ένα, και ένα επιπλέον γράμμα της απάντησης εμφανίζεται και αφαιρούνται πέντε βαθμοί από τη συνολική βαθμολογία του χρήστη. Όταν ο χρήστης έχει χρησιμοποιήσει και τις τέσσερεις βοήθειες, και συνεπώς έχει εμφανιστεί ολόκληρη η απάντηση ,καθώς στο δεύτερο επίπεδο οι απαντήσεις αποτελούνται από τέσσερα γράμματα, τότε, το Hintbtn καθίσταται σε μη ορατό.



Εικόνα 5.34: Στιγμιότυπο του επιπέδου αφού ο χρήστης χρησιμοποιήσει την πρώτη βοήθεια



Εικόνα 5.35: Στιγμιότυπο του παιχνιδιού όπου ο χρήστης έχει χρησιμοποιήσει όλες τις διαθέσιμες βοήθειες

Αφού ο χρήστης δώσει μια απάντηση, πρέπει να πατήσει το πλήκτρο «Confirm» (Checkbtn), για επιβεβαίωση της απάντησης.

```
private void Checkbtn_Click(object sender, EventArgs e)
        {

            string nameused1 = PhotoPicBx.ImageLocation.ToString();
            string nameused2 = nameused1.Substring(nameused1.LastIndexOf("\\"));
            string nameused3 = nameused2.Remove(nameused2.Length - 4, 4);
            string nameused = nameused3.Remove(0, 1);

            int comp = NameTxtBx.Text.CompareTo(nameused);


            if (comp == 0)
            {
                point = point + 50;
                Pointlbl.Text = point.ToString();
                NameTxtBx.Text = "";
                buttoncount++;

                string ConnectionString = @"Data Source =" + Application.StartupPath +
@"\mydb.sdf";
                SqlCeConnection sqlConnection1 = new SqlCeConnection(@"Data Source=" +
Application.StartupPath + @"\mydb.sdf");
                System.Data.SqlServerCe.SqlCeCommand cmdt = new
System.Data.SqlServerCe.SqlCeCommand(ConnectionString);
                cmdt.CommandText = "insert into logs (Sessionid, score, Minutes, Hintcount,
Errorcount, Level, Picname, Username,Datetime, Day) values(@ID, @Score, @Minutes,
@Hintcount, @Errorcount, 'Level 2', @Picname, @username,@Datetime, @Day)";
                cmdt.Connection = sqlConnection1;

                sqlConnection1.Open();



                {


                    var sql = @"SELECT MAX(ID) As Id FROM Userrecords WHERE Username =
@UserName";

                    SqlCeCommand cmd = new SqlCeCommand(sql, sqlConnection1);
                    cmd.Parameters.Add("@UserName", LoginInfo.UserID);



                    cmd.UpdatedRowSource = UpdateRowSource.OutputParameters;

                    object val = cmd.ExecuteScalar();
                    DateTime timelog = DateTime.Now;

                    cmdt.Parameters.AddWithValue("@Score", point);
                    cmdt.Parameters.AddWithValue("@Minutes", time);
```

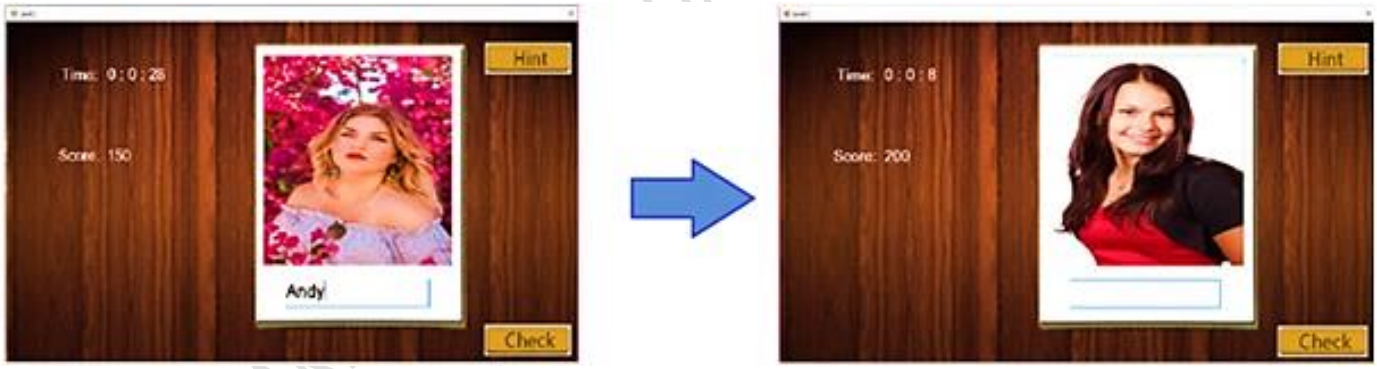
```
                cmdt.Parameters.AddWithValue("@Hintcount", hbuttoncount);
                    cmdt.Parameters.AddWithValue("@Errorcount", errorcount);
                    cmdt.Parameters.AddWithValue("@Picname", nameused2);
                    cmdt.Parameters.AddWithValue("@ID", val);
                    cmdt.Parameters.AddWithValue("@username", LoginInfo.UserID);
                    cmdt.Parameters.AddWithValue("@Datetime", timelog);
                    cmdt.Parameters.AddWithValue("@Day",
DateTime.Now.ToString("d/M/yyyy"));



                    cmdt.UpdatedRowSource = UpdateRowSource.OutputParameters;
                    cmdt.ExecuteNonQuery();

                    string timelogstr = timelog.ToString(("dd MMMM yyyy"));


                    if (!System.IO.Directory.Exists("" + Application.StartupPath +
@"\logs\" + LoginInfo.UserID + ""))
                    {
                        System.IO.Directory.CreateDirectory("" + Application.StartupPath +
@"\logs\" + LoginInfo.UserID + "");
                    }


                    string strSQL = "SELECT  score, Minutes, Hintcount, Errorcount, Level,
Picname, Datetime FROM logs where Username= '" + LoginInfo.UserID + "' and Day= '" +
DateTime.Now.ToString("d/M/yyyy") + "'";
                    SqlCeDataAdapter dt = new SqlCeDataAdapter(strSQL, sqlConnection1);
                    DataSet ds = new DataSet();
                    dt.Fill(ds, "logs");
                    ds.WriteXml(("" + Application.StartupPath + @"\logs\" +
LoginInfo.UserID + @"\" + LoginInfo.UserID + " " + timelogstr + " " + "log" + ".xml"));


                    sqlConnection1.Close();


                }


                if (buttoncount <= 4)
                {
                    if (i < 4)
                    {
                        i++;
                        string[] files = Directory.GetFiles("" + Application.StartupPath +
@"\images\Level2\used", "*.jpg", SearchOption.AllDirectories);
                        PhotoPicBx.Load(files[i]);
                        NameTxtBx.Focus();
                        timer1.Stop();
                        time = 0;
                        timer1.Start();
                        h = 4;
                        hbuttoncount = 0;
                        Hintbtn.Show();
                        errorcount = 0;
                    }

                }
            }
```

```
            else
            {
                MessageBox.Show("Wrong answer! Please try again!");
                errorcount++;
                NameTxtBx.Clear();
                NameTxtBx.Focus();
            }


            int pointcount = Int32.Parse(Pointlbl.Text);
            if (buttoncount > 4)
            {2

                if (pointcount >= 325)
                {
                    LevelTwoSuccessScreen lvltwosucscreen = new LevelTwoSuccessScreen();


                    string ConnectionString = @"Data Source =" + Application.StartupPath +
@"\mydb.sdf";
                    SqlCeConnection con = new SqlCeConnection(@"Data Source=" +
Application.StartupPath + @"\mydb.sdf");
                    System.Data.SqlServerCe.SqlCeCommand cmd = new
System.Data.SqlServerCe.SqlCeCommand(ConnectionString);
                    DateTime time = DateTime.Now;              // Use current time
                    cmd.CommandText = "insert into Userrecords (Username, Points,
Logindate, Level) values(@Username, @Points, @Time, 'Level 2')";
                    cmd.Connection = con;




                    {


                        cmd.Parameters.AddWithValue("@Username", LoginInfo.UserID);
                        cmd.Parameters.AddWithValue("@Points", pointcount);
                        cmd.Parameters.AddWithValue("@Time", time);



                        con.Open();
                        cmd.ExecuteNonQuery();
                        con.Close();
                    }
                    lvltwosucscreen.Show();
                    this.Close();
                }
                if (pointcount < 325)
                {
                    PhotoPicBx.Image.Dispose();
                    LevelTwoRepeatScreen lvltworepscreen = new LevelTwoRepeatScreen();
                    lvltworepscreen.Show();
                    this.Close();

                }

            }
        }



    }
}
```

Αρχικά, το όνομα του αρχείου που περιέχει την εικόνα, υφίσταται επεξεργασία, έτσι ώστε να απομονωθεί το όνομα του ατόμου, το οποίο εικονίζεται στη φωτογραφία. Έπειτα, συγκρίνεται με την απάντηση την οποία έδωσε ο χρήστης στο NameTxtBx, μέσω της συνάρτησης NameTxtBx.Text.CompareTo(nameused).

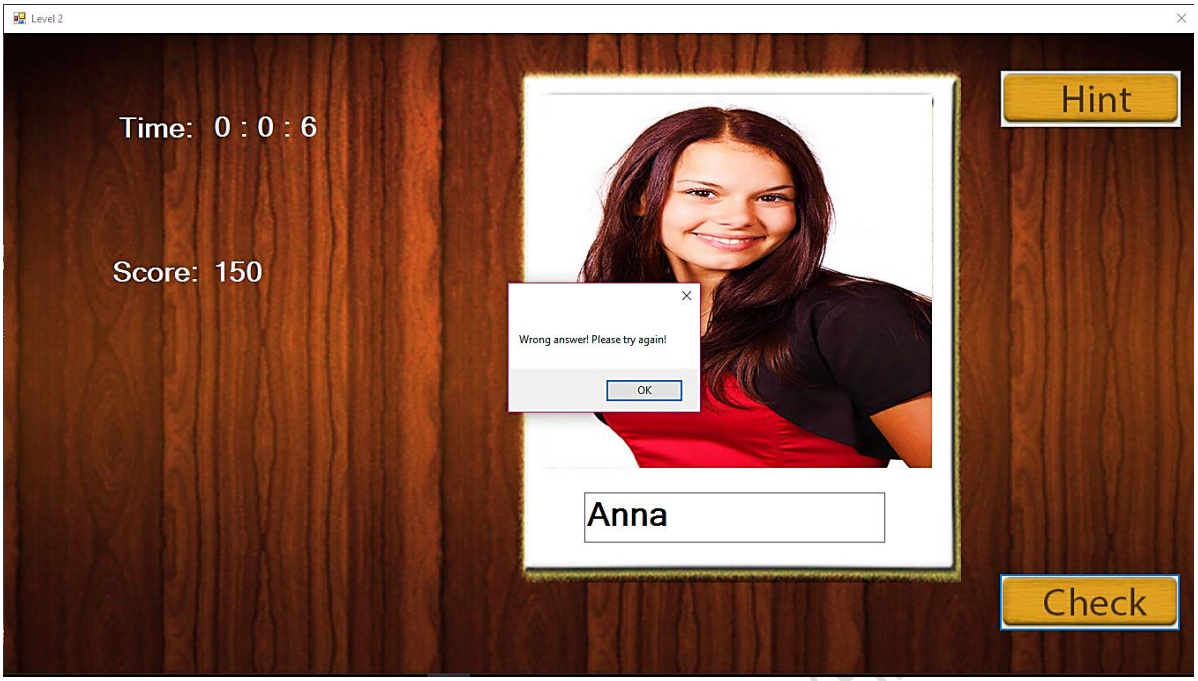
Εάν το αποτέλεσμα είναι ίσο με μηδέν, δηλαδή αν η απάντηση η οποία έδωσε ο χρήστης είναι ίδια με τη συμβολοσειρά που προέκυψε από το όνομα του αρχείου, τότε προστίθενται 50 πόντοι στη συνολική βαθμολογία του χρήστη. Στη συνέχεια, δημιουργείται μια σύνδεση με τη βάση και εισάγονται στον πίνακα logs τα στοιχεία του παιχνιδιού τα οποία θα καταγραφούν στο αρχείο εγγραφών (log). Κάθε αρχείο εγγραφών, εισάγεται σε ξεχωριστό φάκελο, ο οποίος φέρει το όνομα του χρήστη. Αν αυτός ο φάκελος δεν υπάρχει ήδη, τότε δημιουργείται κατά την πρώτη χρήση του παιχνιδιού. Θα αναφερθούμε αναλυτικά στο αρχείο εγγραφών στην επόμενη υποπαράγραφο.

Εάν η τιμή της μεταβλητής buttoncount είναι μικρότερη ή ίση του 4, δηλαδή αν ο παίκτης έχει απαντήσει σε 4 φωτογραφίες ή λιγότερες και πριν προβληθεί η πέμπτη και τελευταία φωτογραφία, μόλις πραγματοποιηθεί ο έλεγχος της απάντησης και διαπιστωθεί ότι η απάντηση είναι σωστή, προβάλλεται η επόμενη φωτογραφία. Εκείνη τη στιγμή, οι μετρητές hbuttoncount και errorbuttoncount (δηλαδή οι μετρητές οι οποίοι καταγράφουν τον αριθμό των βοηθειών που χρησιμοποιήθηκαν και των λανθασμένων απαντήσεων που έδωσε ο χρήστης αντίστοιχα) και ο Timer, ο οποίος χρησιμοποιείται για την καταγραφή του χρόνου τον οποίο χρειάζεται ο χρήστης για να απαντήσει σε μια φωτογραφία, μηδενίζονται. Αυτό συμβαίνει έτσι ώστε να καταγραφούν ακριβή στοιχεία για την επίδοση του χρήστη σε κάθε τμήμα του επιπέδου.



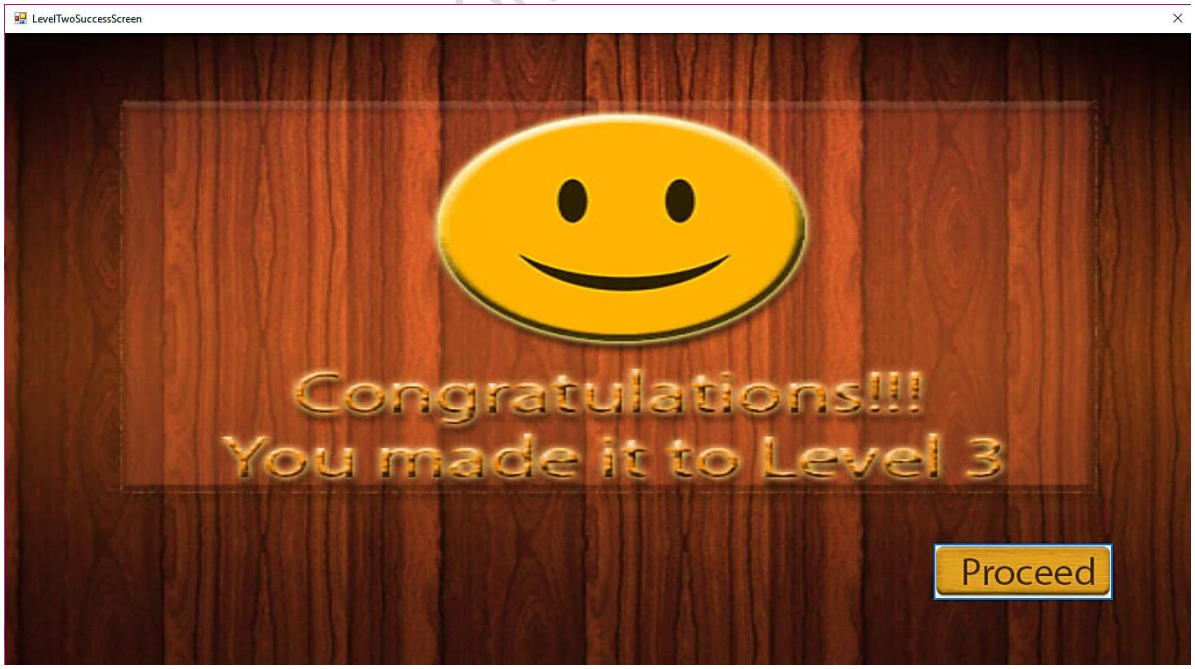
Εικόνα 6.36: Αλλαγή φωτογραφίας μετά από σωστή απάντηση του παίκτη

Εάν ο χρήστης δώσει λανθασμένη απάντηση, τότε θα εμφανιστεί ένα MessageBox, το οποίο θα πληροφορεί τον χρήστη για το γεγονός ότι έδωσε λανθασμένη απάντηση. Επίσης, ο μετρητής errorcount θα αυξηθεί κατά 1.



Εικόνα 5.37: Περίπτωση λανθασμένης απάντησης

Αφού προβληθεί και η τελευταία φωτογραφία, και ο χρήστης δώσει σωστή απάντηση, μόλις πατηθεί το πλήκτρο Check, θα ελεγχθούν οι πόντοι που έχει συγκεντρώσει ο χρήστης. Αν είναι ίσοι ή ξεπερνούν τους 325, τότε εμφανίζεται η «οθόνη επιτυχίας» και ο χρήστης έχει τη δυνατότητα να προχωρήσει στο επόμενο επίπεδο.



Εικόνα 5.38: Οθόνη επιτυχίας του 2ου επιπέδου

Εάν όμως ο χρήστης δεν καταφέρει να συγκεντρώσει τους απαιτούμενους 325 πόντους, τότε οδηγείται στην «οθόνη επανάληψης» και υποχρεώνεται να επαναλάβει το δεύτερο επίπεδο, έτσι ώστε να εξασκηθεί περισσότερο η μνήμη του και να μπορεί να ανταπεξέλθει στα επόμενα και δυσκολότερα επίπεδα.

Εικόνα 6.39: Οθόνη επανάληψης 2ου επιπέδου

5.3 Δημιουργία αρχείου εγγραφών (log files)

Για να είναι η εφαρμογή αποτελεσματική στο σκοπό της, είναι απαραίτητο ορισμένα βασικά στοιχεία του χρήστη, να καταγράφονται σε ένα αρχείο εγγραφών (log file).

Για την παρούσα εφαρμογή, θεωρήθηκε χρήσιμο να καταγράφονται σε ένα αρχείο μορφής XML τα εξής στοιχεία:

- Η βαθμολογία που έχει συγκεντρωθεί από τον χρήστη
- Ο χρόνος (σε δευτερόλεπτα) που χρειάστηκε ο παίκτης για να συμπληρώσει σωστά το όνομα του εικονιζόμενου ατόμου της φωτογραφίας.
- Ο αριθμός των βοηθειών που χρησιμοποίησε ο χρήστης για την κάθε εικόνα.
- Ο αριθμός των λανθασμένων απαντήσεων που δόθηκαν σε κάθε εικόνα.
- Το τρέχον επίπεδο στο οποίο βρίσκεται ο παίκτης
- Το όνομα της εικόνας την οποία αφορούν τα συγκεκριμένα στοιχεία
- Ένα timestamp που δηλώνει σε ποια ακριβώς χρονική στιγμή συνέβησαν όλα αυτά.

Ο φάκελος όπου αποθηκεύονται τα αρχεία εγγραφών (logs), δημιουργείται στο directory στο οποίο βρίσκεται το εκτελέσιμο αρχείο (.exe) της εφαρμογής.

```
<?xml version="1.0" standalone="true"?>
- <NewDataSet>
  - <logs>
      <score>50</score>
      <Minutes>5</Minutes>
      <Hintcount>0</Hintcount>
      <Errorcount>0</Errorcount>
      <Level>Level 1</Level>
      <Picname>\Ivy.jpg</Picname>
      <Datetime>2017-03-22T18:28:32.843+02:00</Datetime>
    </logs>
  - <logs>
      <score>100</score>
      <Minutes>4</Minutes>
      <Hintcount>0</Hintcount>
      <Errorcount>0</Errorcount>
      <Level>Level 1</Level>
      <Picname>\Joe.jpg</Picname>
      <Datetime>2017-03-22T18:28:37.907+02:00</Datetime>
    </logs>
  - <logs>
      <score>150</score>
      <Minutes>13</Minutes>
      <Hintcount>0</Hintcount>
      <Errorcount>0</Errorcount>
      <Level>Level 1</Level>
      <Picname>\Tia.jpg</Picname>
      <Datetime>2017-03-22T18:28:51.647+02:00</Datetime>
    </logs>
```

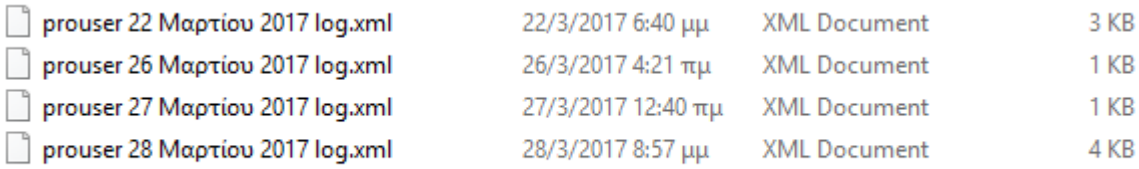
Εικόνα 5.40: Τμήμα από το αρχείο εγγραφών (log file)

Στον φάκελο logs, βρίσκεται ένας ξεχωριστός φάκελος για κάθε χρήστη, ο οποίος ονομάζεται με το username του χρήστη.

Name	Date modified	Type
marina	9/3/2017 2:59 μμ	File folder
me	9/3/2017 9:31 μμ	File folder
newtestuser	11/3/2017 3:14 μμ	File folder
prouser	28/3/2017 8:01 μμ	File folder
testnewuser	10/3/2017 12:27 μμ	File folder
testuser	9/3/2017 9:58 μμ	File folder
theuser	9/3/2017 9:43 μμ	File folder

Εικόνα 5.41: Ο κάθε χρήστης διαθέτει τον δικό του, εξατομικευμένο φάκελο εγγραφών

Σε κάθε έναν από τους φακέλους, βρίσκονται αναλυτικά τα αρχεία εγγραφών (log files) για κάθε χρήστη. Τα αρχεία αυτά καταγράφονται σε ημερήσια βάση. Το κάθε αρχείο καταχωρείται με το username του χρήστη και την ημερομηνία δημιουργίας του ως όνομα.



Εικόνα 5.42: Τα αρχεία εγγραφών (log files) του χρήστη prouser

Κάθε log file είναι μορφής XML, και διαθέτει τη δομή που εικονίζεται στην Εικόνα 5.41.

Συμπεράσματα

Σύμφωνα με έρευνες που έχουν διεξαχθεί, έχει αποδειχθεί ότι η συστηματική ενασχόληση ενός ατόμου, με παιχνίδια τα οποία αποσκοπούν στην εξάσκηση της μνήμης και της συγκέντρωσης, συμβάλλει στην μακροχρόνια διατήρηση της μνήμης του ανθρώπου και συνεπώς στην πρόληψη της πιο κοινής μορφής άνοιας, γνωστής και ως «νόσος Alzheimer».

Η νόσος Alzheimer εμφανίζεται περισσότερο σε άτομα προχωρημένης ηλικίας, τα οποία είναι άνω των 65. Η συχνότητα εμφάνισης της ασθένειας διαφοροποιείται με την ηλικία: κάθε πέντε έτη μετά από την ηλικία των 65, ο κίνδυνος εμφάνισης της ασθένειας περίπου διπλασιάζεται.

Επομένως, υπάρχει η ανάγκη για τη δημιουργία μιας εκπαιδευτικής εφαρμογής, η οποία θα είναι κατάλληλη για την ομάδα την οποία προορίζεται, δηλαδή τους ηλικιωμένους. Για να επιτευχθεί αυτό, θα πρέπει να ληφθούν υπόψιν ορισμένοι παράγοντες.

- ✓ Καθώς το μεγαλύτερο ποσοστό των ηλικιωμένων, δεν είναι εξοικειωμένο με την σύγχρονη τεχνολογία, η εφαρμογή θα πρέπει να είναι απλή, φιλική προς τον χρήστη και να διαθέτει απλό χειρισμό.

- ✓ Δεν πρέπει να παρέχει πολλά και περιττά ερεθίσματα, καθώς είναι πιθανόν να προκληθεί σύγχυση.

- ✓ Επιπλέον, το κάθε επίπεδο, θα πρέπει να είναι σύντομο, και να απαιτεί όσο το δυνατόν λιγότερα βήματα για την ολοκλήρωσή του, έτσι ώστε ο χρήστης να μην αποθαρρύνεται λόγω της αυξημένης πολυπλοκότητας.

- ✓ Και τέλος, κάθε χρήστης θα πρέπει να διαθέτει το δικό του, εξατομικευμένο προφίλ, έτσι ώστε να καταχωρούνται περιοδικά στοιχεία σχετικά με την κατάσταση της μνήμης και την πρόοδό του.

Η εφαρμογή η οποία δημιουργήθηκε για την παρούσα πτυχιακή εργασία πληροί όλες τις παραπάνω προϋποθέσεις.

Βιβλιογραφία

[1] Chengxuan Qiu, Miia Kivipelto, Eva von Strauss *Epidemiology of Alzheimer's disease: occurrence, determinants, and strategies toward intervention*
Dialogues Clin Neurosci. 2009 Jun; 11(2): 111–128.PMCID: PMC3181909

[2] Barnard, Y., Bradley, M. D., Hodgson, F., & Lloyd, A. D. (2013). *Learning to use new technologies by older adults: Perceived difficulties, experimentation behaviour and usability. Computers in Human Behavior*, 29(4), 1715-1724. doi:10.1016/j.chb.2013.02.006

[3] Farber, I., Fua, K. C., Gupta, S., & Pautler, D. (2016). *MoCHA. Proceedings of the 13th International Conference on Advances in Computer Entertainment Technology* - ACE2016.

[4] Bouchard B, Imbeault F, Bouzouane A, Menelas B-AJ. *Developing Serious Games Specifically Adapted to People Suffering from Alzheimer*. Serious Games Development and Applications Lecture Notes in Computer Science. 2012. doi:10.1007/978-3-642-33687-4_21.

[5] Robert, P.h., V. Manera, and A. König. "Recommendations for the development and use of serious games in patients with AD and other dementia-related disorders." Gerontechnology 13.2 (2014): n. pag. Web.

[6] Γ. ΠΑΠΑΚΩΝΣΤΑΝΤΙΝΟΥ, Π. ΤΣΑΝΑΚΑΣ, Ν. ΚΟΖΥΡΗΣ, Α. ΜΑΝΟΥΣΟΠΟΥΛΟΥ, Π. ΜΑΤΖΑΚΟΣ, "Τεχνολογία Υπολογιστικών Συστημάτων και Λειτουργικά Συστήματα Βιβλίο Μαθητή Γ΄ Γενικού Λυκείου", εκδόσεις: ΙΝΣΤΙΤΟΥΤΟ ΤΕΧΝΟΛΟΓΙΑΣ ΥΠΟΛΟΓΙΣΤΩΝ ΚΑΙ ΕΚΔΟΣΕΩΝ «ΔΙΟΦΑΝΤΟΣ».

Ηλεκτρονικές πηγές

Google scholar: https://scholar.google.gr/

Αντίστοιχη πτυχιακή τμήματος Μηχανικών Πληροφορικής ΤΕΙ Κρήτης 2015:
http://nefeli.lib.teicrete.gr/browse/stef/epp/2015/BritzolakisAlexandros/document-1434443335-642563-10369.tkl

Βικιπαίδεια: https://el.wikipedia.org

Dsepwiki: https://dsepwiki.wikispaces.com/

Παιδαγωγικό τμήμα Δημοτικής Εκπαίδευσης πανεπιστημίου Αθηνών – Θεωρίες μάθησης:
http://old.primedu.uoa.gr/sciedu/new_ant/new_theories.htm

Εκπαιδευτική κοινότητα: http://blogs.sch.gr/

Ηλεκτρονικό σύγγραμμα "Εισαγωγή στη C# και το .NET 4.0" (Επιμέλεια: Βασίλης Κόλιας):
http://www.dga.gr/web/publications/files/csharp.pdf

Τμήμα Μηχανικών Η/Υ και Πληροφορικής Πανεπιστημίου Ιωαννίνων – Η γλώσσα SQL:
http://www.cs.uoi.gr/~pitoura/courses/db/db02/slides/sql-part1.pdf