Πανεπιστήμιο Πειραιώς – Τμήμα Πληροφορικής

Πρόγραμμα Μεταπτυχιακών Σπουδών
«Πληροφορική»

Μεταπτυχιακή Διατριβή

Τίτλος Διατριβής	**Διαδικτυακή Εφαρμογή με MVC και C# και εκμάθηση μαθηματικών εξισώσεων** **Web Application with MVC and C# and learning mathematical equations**
Ονοματεπώνυμο Φοιτητή	**Παπαδοπούλου Χρηστίνα**
Πατρώνυμο	**Βασίλειος**
Αριθμός Μητρώου	**ΜΠΠΛ 16019**
Επιβλέπων	**Ευθύμιος Αλέπης, Αναπληρωτής Καθηγητής**

Ημερομηνία Παράδοσης **Σεπτέμβριος 2020**

Τριμελής Εξεταστική Επιτροπή

(υπογραφή)	(υπογραφή)	(υπογραφή)
Ευθύμιος Αλέπης Αναπληρωτής Καθηγητής	Μαρία Βίρβου Καθηγήτρια	Κωνσταντίνος Πατσάκης Επίκουρος Καθηγητής

ΠΕΡΙΛΗΨΗ

Στη παρούσα μεταπτυχιακή διατριβή αναπτύχθηκε μία εφαρμογή ιστού για εκπαιδευτικούς σκοπούς. Ο χρήστης εκπαιδεύεται και μέσω θεωρίας αλλά και μέσω διαγωνισμάτων με στόχο να εμπλουτίσει τις γνώσεις του στις εξισώσεις.Ο κάθε χρήστης αφού κάνει εγγραφή έχει τη δυνατότητα να κάνει διαγωνίσματα από τρία διαφορετικά επίπεδα και να δει τη βαθμολογία του. Επίσης υπάρχει η δυνατότητα να δει τα σκορ όλων των παιχτών γενικά. Καθώς, επίσης δίνεται και ένας εκπαιδευτικός calculator για να μπορέσει να δει ο χρήστης πως λειτουργούν οι εξισώσεις. Σε αυτήν την εφαρμογή χρησιμοποιήθηκαν τεχνολογίες οι οποίες καταγράφονται στην αρχή, καθώς και η αρχιτεκτονική σχεδίαση και στην συνέχεια γίνεται παρουσίαση της εφαρμογής με την βοήθεια επεξηγηματικών εικόνων.

ABSTRACT

In this master's thesis a web application was developed for educational purposes. The user is trained both through theory and through competitions in order to enrich his knowledge in the equations. Each user after registering has the opportunity to take competitions from three different levels and see his score. It is also possible to see the scores of all players in general. As well, an educational calculator is given so that the user can see how the equations work. In this application technologies were used which are recorded at the beginning, as well as the architectural design and then the application is presented with the help of explanatory pictures.

ΠΕΡΙΕΧΟΜΕΝΑ

Κεφάλαιο 1 Εισαγωγή

1.1. Στόχος της μεταπτυχιακής διατριβής

Στην παρούσα μεταπτυχιακή διατριβή παρουσιάζεται η υλοποίηση μιας εφαρμογής για εκπαιδευτικούς σκοπούς. Λόγω του ότι η πληροφορική και η τεχνολογίες έχουν εισβάλει τα τελευταία χρόνια ραγδαία στην ζωή μας κυρίως με θετικό αντίκτυπο όλοι οι τομείς και οι κλάδοι προσπαθούν να προσαρτίσουν στον τομέα τους και αυτό το κομμάτι της νέας εξέλιξης. Στον τομέα της εκπαίδευσης, τα θετικά αντίκτυπα είναι πολλά καθώς η εκπαίδευση γίνεται πιο διαδραστική, αυξάνεται το ενδιαφέρον των μαθητών και γίνεται πιο άμεσο το υλικό στους μαθητές. Χρησιμοποιώντας και τις τελευταίες τεχνολογίες προγραμματιστικά καταφέραμε τα εξής:

- Δημιουργία μίας ιστοσελίδας με καινούριες τεχνολογίες με τις οποίες η εφαρμογή γίνεται πιο επεκτάσιμη, συντηρίσιμη και ασφαλείς.
- Για τους χρήστες θεωρία αλλά και πολλά διαγωνίσματα πολλαπλών επιπέδων βαθμού δυσκολίας για να εκπαιδευτούν στο κομμάτι των εξισώσεων,
- Ολοκλήρωση της εφαρμογής βάση βαθμολογιών. Ο κάθε χρήστης για να μπορέσει να πάει στο επόμενο επίπεδο διαγωνισμάτων πρέπει να έχει επιτύχει το προηγούμενο επίπεδο,
- Εμφάνιση των αποδόσεων για όλους τους χρήστες,
- Χρήση υπολογιστή για εξισώσεις,
- Χρήση της αρχιτεκτονικής Model View Controller, μέσω της οποίας χωρίσαμε το πρόγραμμα σε τρία επίπεδα και έτσι επιτύχαμε το inversion of control (IoC),μέσω του οποίου επιτυνχάνουμε να αντιστρέψουμε την ροή ελέγχου ενός συστήματος σε σύγκριση με τον διαδραστικό προγραμματισμό.

Εν κατά κλέιδι, η εφαρμογή αυτή περίχει μελέτη, αξιολόγηση, επίπεδα δυσκολίας και ολοκλήρωση εκμάθησης, χρησιμοποιώντας τις τελευταίες προγραμματιστικές τεχνολογίες οι οποίες κάνουν τον κώδικα πιο επεκτάσιμο και συντηρίσιμο.

Κεφάλαιο 2 Τεχνολογίες

2.1. Visual Studio 2019 και .NET Framework

Για την ανάπτυξη της εφαρμογής χρησιμοποιήθηκε το Microsoft Visual Studio 2019.Το Visual Studio είναι ένα ολοκληρωμένο περιβάλλον ανάπτυξης (IDE) από τη Microsoft. Χρησιμοποιείται για την ανάπτυξη προγραμμάτων ηλεκτρονικών υπολογιστών, καθώς και ιστοσελίδων, εφαρμογών ιστού, υπηρεσιών ιστού και εφαρμογών για κινητά. Το Visual Studio χρησιμοποιεί πλατφόρμες ανάπτυξης λογισμικού της Microsoft όπως τα Windows API, τα Windows Forms.

http://www.dga.gr/web/publications/files/csharp.pdf

2.2. Το .NET Framework

Το .NET Framework είναι ένα πλασίο λογισμικού για την πλατφόρμα των Windows. Αποτελείται απο μία μεγάλη βιβλιοθήκη κλάσεων και υποστηρίζει πολλές γλώσσες προγραμματισμού. Τα προγράμματα που γράφονται σε .NET framework εκτελούνται σε ένα περιβάλλον εκτέλεσης γνωστό ως Common Language Runtime (CLR), ειδικό λογισμικό για την εκτέλεση του προγράμματος και την συνεργασία του με το λειτουργικό σύστημα. Βασικές λειτουργίες, όπως οι γραφικές διεπαφές GUI, η επικοινωνία με βάσεις δεδομένων, η κρυπτογραφία παρέχονται από το Application Programming Interface (API) του .NET. Βασικά θετικά του .NET Framework είναι:

Α) Υποστήριξη πολλαπλών γλωσσών: Μέσω της διαγλωσσικής κληρονομικότητας πλέον η συνένωση κώδικα σε διαφορετικές γλώσσες είναι έυκολη υπόθεση,

Β) Ανάπτυξη βασισμένη σε components: Διαφορετικές λειτουργίες μίας εφαρμογής μπορούν να διαμοιραστούν σε διαφορετικά σημεία του κώδικα.

Γ) Μεταφερσιμότητα: Η εικονική μηχανή CLR μεταγλωτίζει το πρόγραμμα σε μία ενδιάμεση γλώσσα CIL και περιέχεται σε ένα assembly, το οποίο είναι ένα .exe ή ένα dll. Και με αυτόν τον τρόπο γίνεται ανεξάρτητο από την πλατφόρμα.

Δ) Αυτόματη διαχείριση μνήμης: αυτόματη διαχείριση της μνήμης μέσω του Garbage Collector και έτσι αποδεσμεύεται ο προγραμματιστής από την διαχείριση της μνήμης. Ο Garbage Collector ερευνά με ασφάλεια πότε μπορεί να ελευθερωθεί μνήμη απο κάτι που δεν χρειάζεται άλλο το πρόγραμμα. Η μνήμη δεσμεύεται με την αρχικοποίηση αντικειμένων στον σωρό (heap). Όσο υπάρχει αναφορά στο αντικείμενο θεωρείται ότι χρησιμοποιείται, αλλιώς είναι διαθέσιμο για καταστροφή. Ο garbage collector τρεχει σε ενα ξεχωριστό thread ανά τακτά χρονικά διαστήματα.

Ε) Ασφάλεια: Το .NET έχει ενσωματωμένο έναν μηχανισμό ασφάλειας με το όνομα Code Access Security (CAS) που απομονώνει τον κώδικα ανάλογα με την προέλευση του.

2.3. ASP.NET

Το ASP.NET είναι ένα web framework της Microsoft και χρησιμοποιείται για διαδικτυακό προγραμματισμό για την δημιουργία δυναμικών ιστοσελιδών στο διαδίκτυο.

Το ASP σημαίνει Active Server Pages και υποστηρίζει τα εξής μοντέλα:

- ASP.NET Web Forms, με την χρήση των οποίων χτίζεις ένα δυναμικό website με την χρήση του drag and drop,
- ASP.NET MVC, ένα «design pattern” για την επίτευξη του “separation of concerns”,
- ASP.NET Web Pages, με την χρήση των οποίων χτίζεις ένα δυναμικό website,
- ASP.NET API, με την χρήση των οποίων χτίζεις http services στα οποία μπορούν να έχουν πρόσβαση οποιοσδήποτε client, συμπεριλαμβανομένου των browsers και των κινητών συσκευών,

- ASP.NET Core, για να χτίζεις cross-platform εφαρμογές, που παίζουν στα εξής λειτουργικά Windows, Linux, macOS and Docker.

Στη παρούσα διπλωματική διατριβή έχει επιλεχθεί το μοντέλο ανάπτυξης ASP.NET MVC

2.4. Γλώσσα C#

Η C# είναι αντικειμενοστραφής γλώσσα προγραμματισμού και αποτελεί μέρος του .NET framework. Αναπτύχθηκε από την Microsoft κατά το 2000, από μία ομάδα κάτω από την ηγεσία του Andres Hejleberg και στηρίζεται στην γλώσσα C. Σαν αντικειμενοστραφής γλώσσα, υποστηρίζει την έννοια της ενθυλάκωσης, της κληρονομικότητας και του πολυμορφισμού. Επιπρόσθετα, η C# υποστηρίζει τον προγραμματισμό βασισμένο σε components (component-based programming) ο οποίος επιτρέπει τον προσδιορισμό αυτόνομων μονάδων λειτουργικότητας (components) που είναι απομονωμένα. Για να διασφαλιστεί ότι τα προγράμματα και οι βιβλιοθήκες C # μπορούν να εξελιχθούν με την πάροδο του χρόνου χωρίς προβλήματα, δόθηκε μεγάλη έμφαση στις εκδόσεις της C #. Πολλές γλώσσες προγραμματισμού δεν δίνουν μεγάλη προσοχή σε αυτό το ζήτημα και, ως εκ τούτου, τα προγράμματα γραμμένα σε αυτές τις γλώσσες σπάνε πιο συχνά από ό, τι είναι απαραίτητο όταν εισάγονται νεότερες εκδόσεις από βιβλιοθήκες που εξαρτώνται από προηγούμενες εκδόσεις βιβλιοθηκών. Μία μικρή αναδρομή στις εκδόσεις της C# είναι οι ακόλουθες:

Α) C# 2.0 : προστέθηκε η υποστήριξη των generics, των anonymous μεθόδων για την διαχείριση των events, των iterator blocks για τα αντικείμενα που συμπεριφέρονται ως collection τα οποία πρέπει να υλοποιούν το IEnumerable, τα partial types

Β) C# 3.0 : εισαγωγή τοπικών μεταβλητών με χρήση της λέξης var (εδώ ο τύπος της μεταβλητής ορίζεται κατά την εκτέλεση και όχι κατά την μεταγλώτισση), εισαγωγή της αυτόματης δημιουργίας των properties λειτουργία με την οποία μειώνουν την πιθανότητα λαθών, εισαγωγή των partial μεθόδων, έτσι δίνεται η δυνατότητα να δηλωθεί η signature μιας μεθόδου και μετά ο προγραμματιστής να επιλέξει την υλοποίηση της, εισαγωγή των lambda expressions δηλαδή μιας anonymous function που μπορεί να περιέχει expressions και statements.

Γ) C# 4.0 : εισαγωγή του τύπου dynamic που επιτρέπει την παράκαμψη του ελέγχου τύπου κατά τη μεταγλώττιση, εισαγωγή των named arguments, το πέρασμα παραμέτρων χωρίς να παίζει ρόλο η σειρά.

Με τη βοήθεια της C#, μπορούμε να αναπτύξουμε διαφορετικούς τύπους ασφαλών εφαρμογών

- Εφαρμογές παραθύρων (windows application)
- Εφαρμογές ιστού (windows application)
- Υπηρεσίες διαδικτύου (web services)
- Εφαρμογές βάσεων δεδομένων(Database application)

2.5. Σύνταξη Razor

Το MVC χρησιμοποιεί μηχανή προβολής Razor, ώστε να μπορούμε να γράψουμε επίσης κώδικα πλευράς διακομιστή(Server) σε HTML. Το Razor είναι μια σύνταξη που επιτρέπει να ενσωματωθεί κώδικας βασισμένο σε διακομιστές (Visual Basic και C #) σε ιστοσελίδες. Ο κώδικας που βασίζεται στον διακομιστή μπορεί να δημιουργήσει δυναμικό περιεχόμενο ιστού ενώ μια ιστοσελίδα διατίθεται στο πρόγραμμα περιήγησης. Όταν καλείται μια ιστοσελίδα, ο διακομιστής εκτελεί τον κώδικα που βασίζεται σε διακομιστή μέσα στη σελίδα πριν επιστρέψει τη σελίδα στο πρόγραμμα περιήγησης. Με την εκτέλεση του διακομιστή, ο κώδικας μπορεί να εκτελεί πολύπλοκες εργασίες, όπως πρόσβαση σε βάσεις δεδομένων.

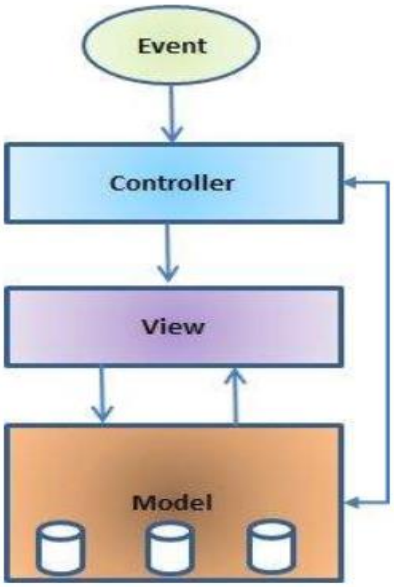
Inversion of Control (IoC) είναι ένα “design principle”, με το οποίο μπορούμε να αντιστρέψουμε διαφορετικά είδη από controls για να πετύχουμε την χαλαρή σύζευξη. Όταν λέμε controls, αναφερόμαστε σε οποιαδήποτε επιπρόσθετη ευθύνη μιας κλάσης. Οι κλάσεις δεν πρέπει να είναι στενά συνδεδεμένες, γιατί με οποιαδήποτε αλλαγή σε μία απο τις δύο επηρεάζει και την άλλη. Κάτι το οποίο επιτυγχάνεται μέσω της δημιουργίας αντικειμένων μέσω του “Factory pattern”.

Dependency Inversion Principle είναι ένα ακόμα «design principle», βάση του οποίου «high-level modules» δεν πρέπει να εξαρτώνται από «low-level modules», αλλά και τα δύο πρέπει να βασίζονται στο abstraction.

2.6. ASP.NET MVC (Model – View – Controller) ARCHITECTURE

Model – View – Controller είναι ένα design pattern για να προγραμματιστεί ένα σύστημα διεπαφής ανθρώπου – υπολογιστή, με το οποίο χωρίζουμε το πρόγραμμα σε τρία επίπεδα model, view, controller και με αυτό επιτυγχάνουμε το inversion of control (IoC), όπως αναφέραμε και πριν. Κάθε ένα από αυτά τα επίπεδα είναι κατασκευασμένα για να χειρίζονται συγκεκριμένες πτυχές ανάπτυξης μιας εφαρμογής. Το MVC είναι ένα από τα πιο συχνά και σύγχρονα χρησιμοποιούμενα πρότυπα ανάπτυξης ιστοσελίδων.

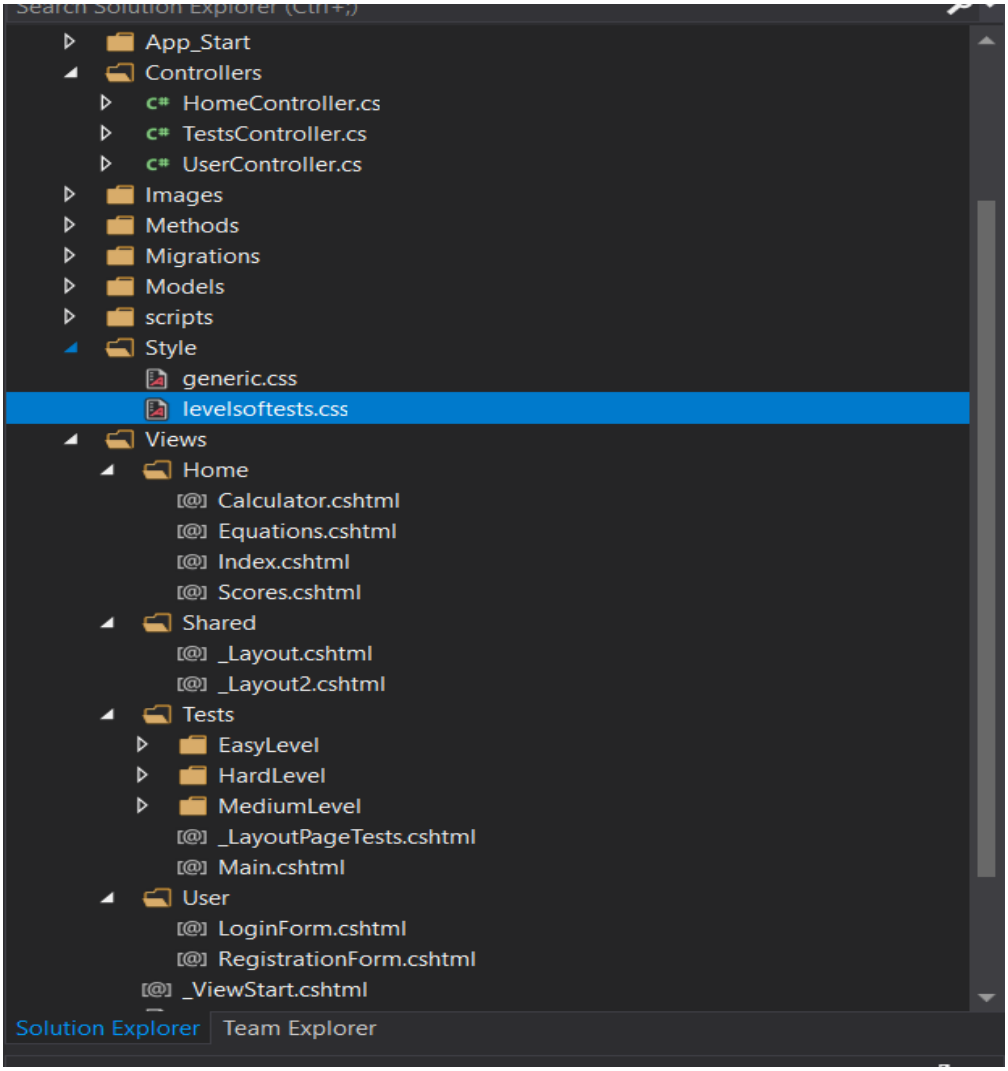
- Model: Το Μοντέλο είναι οι κλάσεις του προγράμματος οι οποίες έχουν ένα προς ένα αντιστοιχία με την βάση δεδομένων. Μία οντότητα της βάσης αναπαρίστατε σαν μία κλάση μέσα στο πρόγραμμα.
- View: Το View αποτελεί την διεπαφή του χρήστη με την εφαρμογή. Στο ASP.NET MVC είναι η HTML, CSS, RAZOR και μέσω αυτής επικοινωνεί το μοντέλο με τον controller.
- Controller: Ο Controller διαχειρίζεται τα αιτήματα του χρήστη. Ο χρήστης χρησιμοποιεί το view και μέσω αυτού στέλνει ένα HTTP request, το οποίο θα διαχειριστή από τον controller. Ο Controller στέλνει το αίτημα στην κατάλληλη διαδικασία που πρέπει να ακολουθηθεί και επιστρέφει στο σωστό view την σωστή απάντηση.



2.6.1. Πλεονεκτήματα

- Ο διαχωρισμός του προγράμματος. Αυτό σημαίνει ότι μπορούμε να χωρίσουμε σε τρία μέρη της εφαρμογής όπως Model, View, Controller,
- Ο προγραμματισμός γίνεται πιο γρήγορος,
- Εύκολο για πολλαπλούς developers να συνεργαστούν και να δουλέψουν ταυτόχρονα,
- Πιο εύκολο να κάνεις update στην εφαρμογή,
- Πιο εύκολο στο debug καθώς έχουμε πολλαπλά επίπεδα καθένα για την δικιά του λειτουργία.

2.7. Εφαρμογή



1. **Controllers**: διαχειρίζονται τα requests από τον client,
   - **HomeController**, τα requests για τις ακόλουθες αιτήσεις /Home/Equations, /Home/Calculator, /Home/Scores,
   - **TestsController**, τα requests για τα tests δηλαδή για τα /Tests/TestEasyLevel, /Tests/TestMediumLevel, /Tests/TestHardLevel,
   - **UserController**, τα requests για το sign in ή sign up των χρηστών /User/LoginForm, /User/RegistrationForm.
2. **App_Data:** έχει την βάση δεδομένων,
3. **Models:** η κάθε οντότητα της βάσης αναπαρίστατε σαν μια κλάση στο πρόγραμμα και μέσω του Entity Framework γίνεται αντιστοίχιση.
4. **Scripts**: Κώδικας σε Javascript, έτσι ώστε μια function να μπορεί να χρησιμοποιηθεί ταυτόχρονα από πολλά view χωρίς να επαναλαμβάνεται κομμάτι του κώδικα,
5. **Style**: Κώδικας σε css για το UI της εφαρμογής,
6. **Views**: Το UI της εφαρμογής, ότι βλέπει ο χρήστης.

2.7.1. MVC Controller

Ο Controller χειρίζεται τις αιτήσεις του Client, τραβάει από το μοντέλο δεδομένα και τα στέλνει στην κατάλληλη προβολή, στο κατάλληλο View.

```
namespace Statistics.Controllers
{
    public class HomeController : Controller
    {
           public ActionResult LoginForm(UserLogin login)
           {

               using (MyDatabase dc = new MyDatabase())
               {
                       var v = dc.User.Where(a => a.Email == login.Email
                            && a.Password == login.Password).FirstOrDefault();
                       if(v != null) {
                            Session["UserId"] = v.UserId;
                            Session["FirstName"] = v.FirstName;
                            Session["Email"] = v.Email;
                            Session["LastName"] = v.LastName;

                       return RedirectToAction("Index", "Home");
                   }
               }
               ModelState.AddModelError("", "Invalid  credentials");
               return View(login);
              }
         }
```

2.7.2. MVC Model

Τα μοντέλα στο MVC αποτελούν κλάσεις οι οποίες αναπαριστούν μία οντότητα από την βάση με ένα προς ένα αντιστοίχιση της οντότητας με την κλάση.

Κλάση User και Level_1 όπου συνδέονται με μοναδικό τρόπο χρησιμοποιώντας τον constructor User. Επίσης έχουμε τα instant fields κάθε κλάσης με τις ιδιότητες get-set για κάθε ένα αντικείμενο

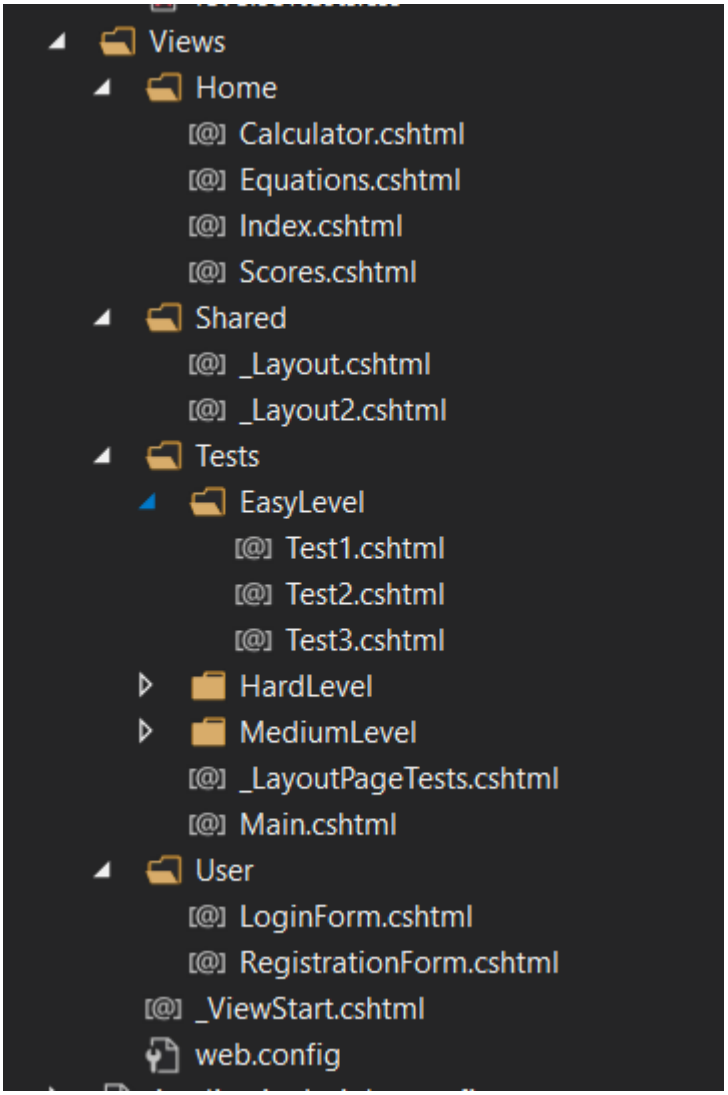
```
public partial class User
    {
        public User()
        {
            this.Level_1 = new HashSet<Level_1>();
            this.Level_2 = new HashSet<Level_2>();
            this.Level_3 = new HashSet<Level_3>();
        }

        public int UserId { get; set; }
        public string FirstName { get; set; }
        public string LastName { get; set; }
        public string Email { get; set; }
        public System.DateTime DateOfBirth { get; set; }
        public string Password { get; set; }
        public virtual ICollection<Level_1> Level_1 { get; set; }
        public virtual ICollection<Level_2> Level_2 { get; set; }
        public virtual ICollection<Level_3> Level_3 { get; set; }
      }
   }

public partial class Level_1
    {
        public int Id { get; set; }
        public int UserId { get; set; }
        public int Score { get; set; }
        public bool Completed { get; set; }
        public System.DateTime Date { get; set; }

        public virtual User User { get; set; }
    }
   }
```

2.7.3. MVC View



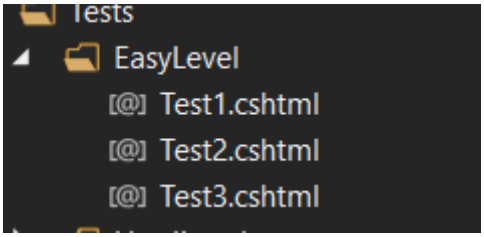
Ξεχωριστά View για κάθε σελίδα, αλλά εάν κάποια κομμάτια θέλουμε να επαναλαμβάνονται όπως το header, footer τότε τα βάζουμε σε ξεχωριστό view και τα κάνουμε inject σε άλλα view τα οποία θέλουμε να έχουν αυτό το κομμάτι προβολής.

```
Calculator.cshtml
1   @model Statistics.Models.User
2   @{
3       ViewBag.Title = "Statistics";
4       Layout = "~/Views/Shared/_Layout2.cshtml";
5   }
6
7
8   <title>Statistics Calculator</title>
9
```

Έτσι, ενώ είμαστε στο View Calculator.cshml κάνουμε και inject το _Layout2.cshtml το οποίο περιέχει το header.

Πρακτικά αυτό σημαίνει ότι δεν χρειάζεται να ξαναγράψουμε κώδικα σε σελίδες που έχουνε κοινά τμήματα UI (User Interface).

Τα Tests έχουν δικό τους view το καθένα, διότι θέλουμε κάθε test να έχει τις δικές του ερωτήσεις. Επίσης, για κάθε επίπεδο υπάρχουν διαφορετικά test, έτσι ώστε ο χρήστης εαν ξαναπαίξει να του εμφανίζεται κάθε φορά και διαφορετικό test.



Η επιλογή αυτή γίνεται στον Controller, δείτε το ακόλουθο:

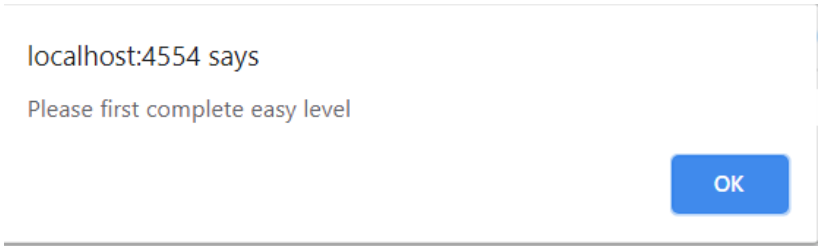
```
public ActionResult TestEasyLevel()
 {
        Random rand = new Random();
        int EasyLevel = rand.Next(1, 4);
        return View("~/Views/Tests/EasyLevel/Test" + EasyLevel + ".cshtml");
 }
```

Επίσης, γίνεται έλεγχος αν ο χρήστης έχει ολοκληρώσει τα προηγούμενα Levels για να προχωρήσει στα επόμενα:

```
public ActionResult TestMediumLevel()
      {
          Random rand = new Random();
          int level2 = rand.Next(1, 3);
          var userId = Session["UserId"].ToString();
                      var completed = user.EasyLevel.Where(a =>
          a.UserId.ToString() == userId && a.Completed ==
          true).FirstOrDefault();
          if (completed == null)
```

```
{
TempData["msg"] = "<script>alert('Please first complete easy
level);</script>";
    return RedirectToAction("Main", "Tests");
}
return View("~/Views/Tests/MediumLevel/Test" + MediumLevel +
".cshtml");
}
```

Στην περίπτωση που δεν τα έχει ολοκληρώσει του εμφανίζεται μήνυμα για να ολοκληρώσει το εύκολο επίπεδο πρώτα, μέσω javascript.



2.7.4. MVC Δρομολόγηση(Routing)

Οι Controllers χρησιμοποιούν το routing του mvc για να γίνει το «match» του URL από το εισερχόμενο request και γινεται «mapping» με το αντίστοιχο action. Στο routing περιγράφεται πως το url path αντιστοιχεί στο action:

Στο RouteConfig.cs ορίζεται η δρομολόγηση:

```
namespace Statistics
{
    public class RouteConfig
    {
        public static void RegisterRoutes(RouteCollection routes)
        {
            routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

            routes.MapRoute(
                name: "Default",
                url: "{controller}/{action}/{id}",
                defaults: new { controller = "Home", action = "Index", id =
                UrlParameter.Optional }
            );


        }
    }
}
```

2.8. Front-end development

Χρησιμοποιούμαι HTML5 Hypertext Markup Language, CSS3 Cascading Style Sheets και Javascript.

2.8.1. HTML

Η HTML5 είναι η τελευταία και πιο εξελιγμένη έκδοση του HTML. Η HTML δεν είναι γλώσσα προγραμματισμού αλλά μια markup language και χρησιμοποιείται για την δημιουργία ιστοσελίδων. Τα έγγραφα HTML είναι αρχεία τύπου κειμένου, που μπορεί να παραχθούν χρησιμοποιώντας οποιοδήποτε επεξεργαστή, όπως κειμενογράφους, πολυπλοκότερες εφαρμογές IDE.

Επομένως, η HTML παρέχει τις ακόλουθες δυνατότητες:

- Δημοσίευση εγγράφων με επικεφαλίδες, κείμενο, πίνακες, λίστες, φωτογραφίες,
- Ανάκτηση πληροφορίας μέσω υπερσυνδέσμων (hyperlinks),
- Σχεδιασμό φορμών για συλλογή πληροφορίας από τους χρήστες,
- Συνεργασία με γλώσσες σεναρίων, επικοινωνία και αποστολή στοιχείων σε απομακρυσμένους εξυπηρετητές π.χ. παραγγελία προιόντων.
- Προσθήκη προγραμμάτων ήχου και βίντεο.
- Κάθε δήλωση ενός τύπου κάποιου στοιχείου περιλαμβάνει 3 μέρη:

    - ✓ Μία αρχική ετικέτα
    - ✓ Το περιεχόμενο
    - ✓ Μία τελική ετικέτα

Ως εκ τούτου, η HTML είναι μια γλώσσα σήμανσης που χρησιμοποιείται για τη δημιουργία ελκυστικών ιστοσελίδων με τη βοήθεια της μορφοποίησης και η οποία φαίνεται σε μια ωραία μορφή σε ένα web browser. Ένα έγγραφο HTML αποτελείται από πολλές ετικέτες HTML και κάθε ετικέτα HTML περιέχει διαφορετικό περιεχόμενο.

<!DOCTYPE..> αν και δεν αποτελεί ετικέτα της html αποτελεί το πρώτο στοιχείο στο έγγραφο και ορίζει στον φυλλομετρητή την έκδοση της html που θα χρησιμοποιηθεί.

<meta charset = “UTF-8”> για την κωδικοποίση του εγγράφου.

ATTRIBUTES: τα elements μπορεί να περιέχουν attributes γιατί μεσω αυτών δηλώνονται χαρακτηριστικά σε ένα element, π.χ. .. . Το attribute id δίνει ένα αναγνωριστικό για ένα στοιχείο το οποίο είναι μοναδικό σε ολόκληρο το έγγραφο. Το χρησιμοποιεί το CSS για να αλλάξει τον τρόπο με το οποίο εμφανίζεται. Το attribute class δεν είναι μοναδικό αλλά μπορεί να χρησιμοποιηθεί σε πολλά attributes και να χρησιμοποιηθεί πάλι απο το css και να δώσει τα ίδια εικαστικά σε όλα αυτά τα attributes. Για να δώσω εικαστικά σε ένα attribute μπορώ να πάω μέσα στο ίδιο το html και με την ιδιότητα style να δώσω το στυλ που θέλω, αν και δεν θεωρείται καλή τακτική.

```
<div class="header">
  <div class="headercentral">
    <div class="A">
      <div id='container'>
        <img src="../images/tree.png">
      </div>
    </div><div class="B">
          <div class="wrapper-B1B2B3">
            <div class="wrapper-B1B2">
              <div class="B1">
```

```
            <p class="message-offer">Mathematics for management
<span style="color:pink;">Equations</span> !</p>
          </div><div class="B2">
            <div class="B2-left"><p class="orario">Mon - Fri : 09am to
06pm </p></div><div class="B2-right"> +1 800 789 0000</div>
          </div>
        </div><div class="wrapperB3-B4">
          <div class="B3">
            <ul>
              <li class="kathgoria">
                <a href="~/Home/Equations" class="katA">Equations</a>
              </li>
              <li class="kathgoria">
                <a href="~/Home/Calculator" class="katB">Calculator</a>
              </li>
              @if (Session["UserId"] != null)
              {
                <li class="kathgoria">
                  <a href="~/Tests/Main" class="katC">Tests</a>
                </li>
                <li class="kathgoria"><a href="@Url.Action("Scores",
"Home")" class="katD">Scores</a></li>
              }
              else
              {
                <li><a href="" onclick="alert('Please Sign up/Sign in to
view Tests section')" class="katC">Tests</a></li>
                <li><a href="" onclick="alert('Please Sign up/Sign in to
view Top Learners section')" class="katD">Scores</a></li>
              }

            </ul>
          </div><div class="B4">
            <ul class="group max">
              @if (Session["UserId"] == null)
              {
                <li class="kathgoria dropdown">
                  <a href="#" class="katD dropdown-toggle" data-
toggle="dropdown">Sign<b class="caret"></b></a>
                  <ul class="dropdown-menu">
                    <li><a href="~/User/LoginForm">Sign in</a></li>
                    <li><a href="~/User/RegistrationForm">Sign
up</a></li>
                  </ul>
```

```
                    </li>
                }
                else
                {
                    <li class="kathgoria"><a href="@Url.Action("Logout", "User")" class="katE">Logout</a></li>
                }
            </ul>
            <div class="headerlinkspopup" onclick="popup()">
                <img src="~/Images/header.png" />
            </div>
            <div class="popup popup-active">
                <div class="popup-inner">
                    <div class="header-links">
                        <ul>
                            @if (Session["UserId"] == null)
                            {
                                <li class="kathgoria dropdown">
                                    <a href="#" class="katD dropdown-toggle" data-toggle="dropdown">Sign<b class="caret"></b></a>
                                    <ul class="dropdown-menu">
                                        <li><a href="~/User/LoginForm">Sign in</a></li>
                                        <li><a href="~/User/RegistrationForm">Sign up</a></li>
                                    </ul>
                                </li>
                            }
                            else
                            {
                                <li class="kathgoria"><a href="@Url.Action("Logout", "User")" class="katE">Logout</a></li>
                            }
                        </ul><div class="popup-x" onclick="popupclose()"><img src="/images/mobileX.png" alt="Mobile X"></div>
                    </div>
                </div>
            </div>
        </div>
    </div>
</div>
    </div>
```

```
</div>
```

2.8.2. Cascading Style Sheets

- Αναπτύχθηκε μαζί με την HTML 4 και απο τη μη κερδοσκοπική διεθνή εταιρική συνεργασία του παγκόσμιου ιστού.
- Ορίζουν την εμφάνιση των στοιχείων ενός εγγράφου HTML,
- Μειώνει την πολυπλοκότητα στην ανάπτυξη και τη δημιουργία των σελίδων HTML,
- Διαχωρίζουν το περιεχόμενο της σελίδας από τον τρόπο εμφάνισης του,
- Συμβάλλει στην ομοιόμορφη παρουσίαση όλων των σελίδων που ανήκουν σε ένα δικτυακό τόπο.

Παράδειγμα generic.css

```
/*-----------index-zone-1----------*/
.header {
   width: 100%;
   BACKGROUND-COLOR: #d3d3d3eb;
   POSITION: FIXED;
   Z-INDEX: 999;
   width: 105%;
   top: 0px;
}

#container {
   margin: auto;
   width: 100%;
   padding-top: 8px;
   padding-bottom: 8px;
}

    #container > img {
      width: 37%;
      height: 100%;
      transform: skew(35deg);
      object-fit: cover;
    }

.message-offer {
   font-size: 1em;
   font-weight: 400px;
   color: #fff;
   font-weight: 300;
```

```
    letter-spacing: 0.9px;
    font-family: "Gill Sans", sans-serif;
    text-align: left;
    padding-left: 90px;
    margin-bottom: 30px;
    margin-top: 7px;
}

.orario {
    margin-bottom: 30px;
    margin-top: 7px;
    font-size: 0.91em;
    text-align: center;
    position: relative;
}

    .orario:before {
        content: "";
        background-image: url(../images/icon-clock-black.png);
        background-size: 14px;
        background-repeat: no-repeat;
        object-fit: cover;
        width: 20%;
        height: 120%;
        position: absolute;
        top: 0%;
        right: 200px;
        text-align: center;
    }


 .B2-right {
    margin-bottom: 30px;
    margin-top: 7px;
    font-size: 0.91em;
    text-align: center;
    position: relative;
}
```

2.8.3. JavaScript

Είναι μία γλώσσα σεναρίων της οποίας ο κώδικας ενσωματώνεται σε αυτόν της HTML και τρέχει στην πλευρά του πελάτη. Η JavaScript δεν έχει σχέση με την Java. Μέσω της JavaScript:

- Παρακολουθεί τα γεγονότα που συμβαίνουν σε μία ιστοσελίδα,
- Διαχειρίζεται στοιχεία και αντικείμενα της σελίδας HTML,
- Χρησιμοποιείται για τον έλεγχο της ορθότητας των στοιχείων που εισάγει ο χρήστης σε μία φόρμα,
- Μπορεί να ελέγξει τον τύπο και την έκδοση του φυλλομετρητή του χρήστη,
- Μπορεί να χρησιμοποιηθεί για την δημιουργία cookies – αποθήκευση και ανάκτηση πληροφορίας από τον υπολογιστή του χρήστη,
- Μπορεί να χρησιμοποιηθεί για τη διαχείριση συνόδων (sessions),
- Σχεδιασμό και animation.

Ωστόσο δεν είναι μόνο ιστοσελίδες που χρησιμοποιούνε JavaScript. Υπάρχουνε πολλά προγράμματα από τη πλευρά του πελάτη (server programs) που τη χρησιμοποιούνε. Node.js είναι η πιο γνωστή. Επιπλέον κάποιες βάσεις δεδομένων όπως MongoDB, τη χρησιμοποιούνε ως γλώσσα προγραμματισμού.

Παράδειγμα για την εμφάνιση της μπάρας που μετράει τον χρόνο:

```
var time = 600;
var time = $('#progressBar').attr('value');

progress(time, time, $('#progressBar'));
// Σεταρισμα χρόνου της μπαρας
function progress(timeleft, timetotal, $element) {

    // Υπολογισμο μηκους μπάρας
    var progressBarWidth = timeleft * $element.width() / timetotal;

    // κινηση μπάρας και μειωση του χρόνου
    $element.find('div')
        .animate({ width: progressBarWidth }, 500)
        .html(Math.floor(timeleft / 60) + ":" + timeleft % 60);

    // καλουμε την ιδια συνάρτηση μεχρι να γίνει μήδεν ανα 1 δεθτερολεπτο
    if (timeleft > 0) {
        setTimeout(function () {
            progress(timeleft - 1, timetotal, $element);

        }, 1000);
    } else {
        //javascript dialog
        if (window.confirm("Τέλος Χρόνου!!!"))
            window.location = "score_table.php?page=4";
    }
};
```

2.8.4. jQuery βιβλιοθήκη

jQuery είναι μια βιβλιοθήκη της JavaScript και δημιουργήθηκε για να απλοποιεί της διάφορες εντολές της JavaScript.
Μερικά από τα πλεονεκτήματα

- Είναι cross-platform δηλαδή μπορεί να χρησιμοποιηθεί σε όλα τα λειτουργικά
- Απλοποιεί αρκετά κλήσεις από τη πλευρά του server με AJAX
- Έχει αρκετά επιπλέον events
- Υποστηρίζεται από τους περισσότερους browser

Για παράδειγμα, η change είναι function της jQuery και χρησιμοποιείται όταν έχει αλλάξει τιμή το element, η checked είναι function της jQuery και χρησιμοποιείται όταν γίνεται check το input, η closest επιστρέφει τον πρώτο πρόγονο του element από το DOM.

```
$('input[type=radio]').change(function () {
    if (this.checked) {
        $(this).closest('.question')
            .find('input[type=radio]').not(this)
            .prop('checked', false);
    }
});
```

Η επόμενη συνάρτηση μας δίνει σε τυχαία σειρά από κάθε διαγώνισμα τόσο τις ερωτήσεις όσο και τις απαντήσεις χρησιμοποιώντας συναρτήσεις της jQuery:

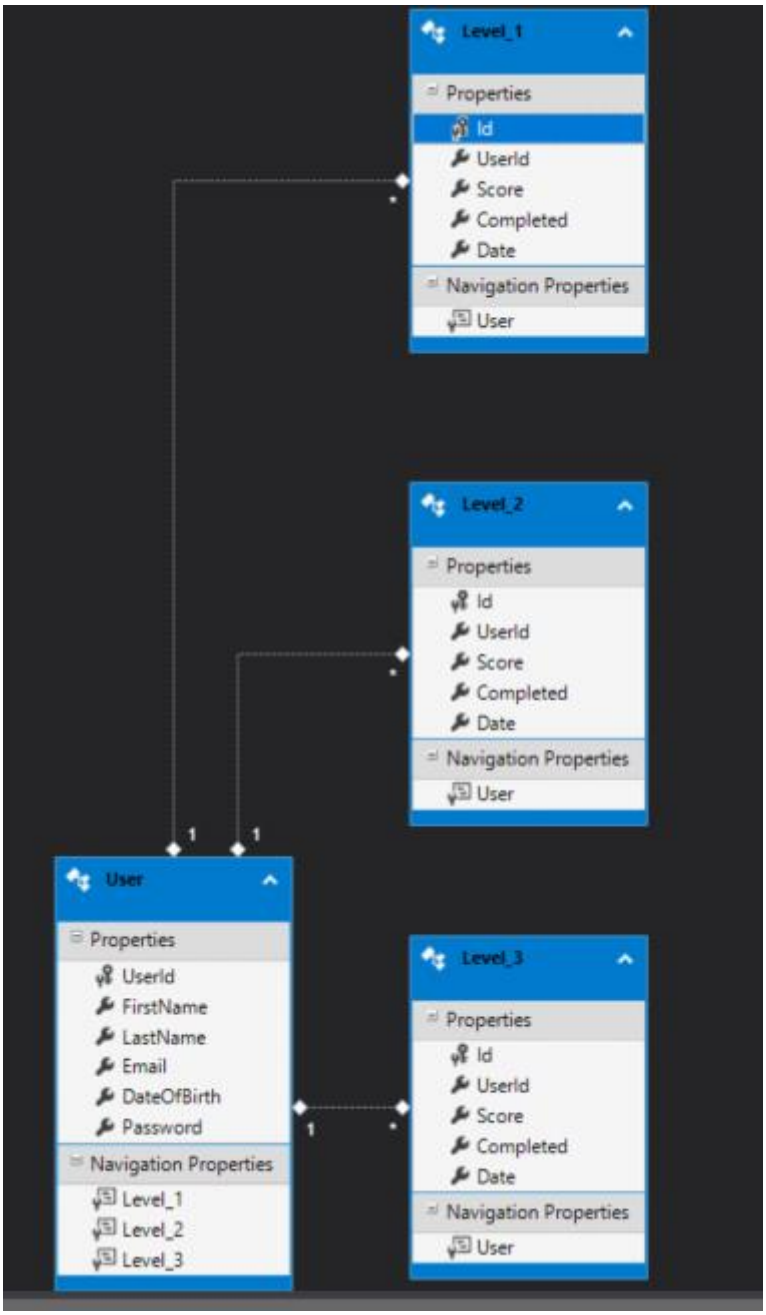
```
$(document).ready(function () {
    for (i = 1; i <= 5; i++) {
        var parent = $(".answers" + i);
        var divs = parent.children();
        while (divs.length) {
            parent.append(divs.splice(Math.floor(Math.random() * divs.length),
1)[0]);
        }
    }
    var parent = $(".questions");
    var divs = parent.children();
    while (divs.length) {
        parent.append(divs.splice(Math.floor(Math.random() * divs.length),
1)[0]);
    }
});
```

2.9. Βάση Δεδομένων

Στο παρακάτω σχήμα βλέπουμε το σχεδιασμό της Βάσης Δεδομένων. Ο κάθε χρήστης έχει ένα μοναδικό κλειδί (UserId).Επίσης όνομα, επίθετο, email, ημερομηνία γέννησης και κωδικό. Οπού ο πίνακας αυτό συνδέεται με τους υπολοίπους πίνακες με βάση το UserId . Στους πίνακες Level_1, Level_2, Level_3

Υπάρχουνε οι εγγραφές

- Id. Μοναδικό αναγνωριστικό (Πρωτεύον κλειδί)
- UserId. το κλειδί που αντιστοιχεί από το πίνακα User(Ξένο κλειδί)
- Score. Το αποτέλεσμα που έλαβε ο χρήστης από το τεστ
- Completed. Αν η βαθμολογία ήτανε πάνω από 50%
- Date. Η ημερομηνία που συμμετείχε στο τεστ



2.9.1. Βάση Δεδομένων Microsoft SQL Server

SQL Server είναι λογισμικό (Σχεσιακό Σύστημα Διαχείρισης Βάσεων Δεδομένων) που αναπτύχθηκε από τη Microsoft. Ονομάζεται επίσης MS SQL Server. Εφαρμόζεται από τις προδιαγραφές των σχεσιακών βάσεων δεδομένων που οι πίνακες συνδέονται με πρωτεύοντα (μοναδικά) και ξένα κλειδιά δηλαδή κλειδιά που αναφέρονται σε πρωτεύοντα. Οι σχεσιακές βάσεις δεδομένων αναφέρονται ως RDBMS (Relational Database Management System

Για την δημιουργία των πινάκων User, Level_1, Level_2, Level_3

```
CREATE TABLE [dbo].[User] (
    [UserId]      INT            IDENTITY (1, 1) NOT NULL,
    [FirstName]   NVARCHAR (50)  NOT NULL,
    [LastName]    NVARCHAR (50)  NOT NULL,
    [Email]       NVARCHAR (50)  NOT NULL,
    [DateOfBirth] DATE           NOT NULL,
    [Password]    NVARCHAR (MAX) NOT NULL,
    PRIMARY KEY CLUSTERED ([UserId] ASC)
   );
```

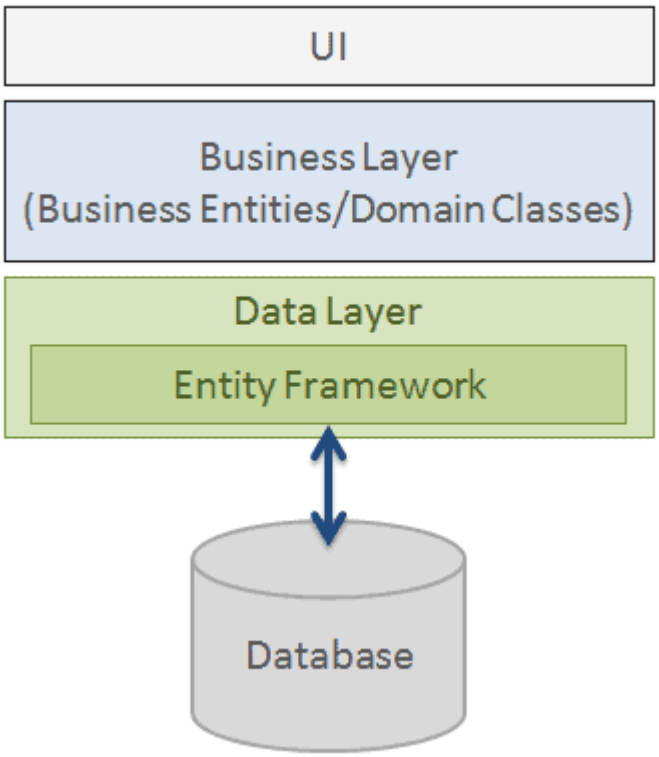
```
CREATE TABLE [dbo].[Level 1] (
    [Id]        INT  IDENTITY (1, 1) NOT NULL,
    [UserId]    INT  NOT NULL,
    [Score]     INT  DEFAULT ((0)) NOT NULL,
    [Completed] BIT  DEFAULT ((0)) NOT NULL,
    [Date]      DATE NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC),
    FOREIGN KEY ([UserId]) REFERENCES [dbo].[User] ([UserId]) ON DELETE CASCADE
ON UPDATE CASCADE
   );
```

```
CREATE TABLE [dbo].[Level 2] (
    [Id]        INT  IDENTITY (1, 1) NOT NULL,
    [UserId]    INT  NOT NULL,
    [Score]     INT  DEFAULT ((0)) NOT NULL,
    [Completed] BIT  DEFAULT ((0)) NOT NULL,
    [Date]      DATE NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC),
    FOREIGN KEY ([UserId]) REFERENCES [dbo].[User] ([UserId]) ON DELETE CASCADE
ON UPDATE CASCADE
   );
```

```
CREATE TABLE [dbo].[Level 3] (
    [Id]        INT  IDENTITY (1, 1) NOT NULL,
    [UserId]    INT  NOT NULL,
    [Score]     INT  DEFAULT ((0)) NOT NULL,
    [Completed] BIT  DEFAULT ((0)) NOT NULL,
    [Date]      DATE NOT NULL,
    PRIMARY KEY CLUSTERED ([Id] ASC),
    FOREIGN KEY ([UserId]) REFERENCES [dbo].[User] ([UserId]) ON DELETE CASCADE
ON UPDATE CASCADE
   );
```

2.9.2. Τεχνολογία Entity Framework

Παλαιότερα, χρησιμοποιούσαμε ADO.NET για να τραβήξουμε δεδομένα από την βάση, ανοίγαμε ένα connection, δημιουργούσαμε ένα DataSet για να τραβήξουμε ή να υποβάλλουμε δεδομένα στην βάση, μετατρέπαμε τα δεδομένα από το DataSet σε .NET objects. Για να αποφευχθεί όλη αυτή η κουραστική δουλειά, η Microsoft μας παρέχει ένα framework το «Entity Framework», ένα open-source ORM Framework για .NET applications.



Το Entity Framework είναι ανάμεσα στα business entities και στην βάση δεδομένων. Σώζει δεδομένα από την βάση στα properties των business entities, και τραβάει δεδομένα και τα μετατρέπει σε business enitities objects αυτόματα.

Χαρακτηριστικά:

- Cross-platform: Το entity framework μπορεί να λειτουργήσει σε Windows, Linux και Mac.
- Μοντελοποίηση: Το EF (Entity Framework) δημιουργεί ένα EDM (Μοντέλο δεδομένων οντοτήτων) που βασίζεται σε οντότητες POCO (Plain Old CLR Object) με ιδιότητες get / set διαφορετικών τύπων δεδομένων. Χρησιμοποιεί αυτό το μοντέλο όταν ερωτά ή αποθηκεύει δεδομένα οντότητας στην υποκείμενη βάση δεδομένων.
- Πρόσβαση: Το EF επιτρέπει τη χρήση ερωτημάτων LINQ (C # / VB.NET) για την ανάκτηση δεδομένων από την υποκείμενη βάση δεδομένων. Ο πάροχος βάσης δεδομένων θα μεταφράσει αυτά τα ερωτήματα LINQ στη γλώσσα ερωτήματος που αφορά τη συγκεκριμένη βάση δεδομένων (π.χ. SQL για μια σχεσιακή βάση δεδομένων). Το EF μας επιτρέπει επίσης να εκτελούμε ερωτήσεις SQL χωρίς να το κάνουμε απευθείας στη βάση δεδομένων.
- Αποθήκευση: Το EF εκτελεί εντολές INSERT, UPDATE και DELETE στη βάση δεδομένων με βάση τις αλλαγές που έγιναν στις οντότητες σας όταν καλείτε τη

μέθοδο SaveChanges (). Το EF παρέχει επίσης τη μέθοδο ασύγχρονης μεθόδου SaveChangesAsync ().

2.9.3. Language Integrated Query (LINQ)

Language Integrated Query είναι το όνομα για ένα σετ τεχνολογιών βασισμένο στην εξέλιξη των δυνατοτήτων των query απευθείας από την C#. Το Integrated Query Language (LINQ) είναι μια ισχυρή γλώσσα ερωτημάτων που ξεκίνησε με το .Net 3.5 & Visual Studio 2008. Παραδοσιακά, τα queries εκφράζονταν σαν strings χωρίς type checking στο compile time. Επομένως, έπρεπε να χρησιμοποιήσεις μία διαφορετική γλώσσα για ερωτήματα για κάθε πηγή δεδομένων: SQL databases, XML documents. Ενώ η LINQ είναι η πρώτη γλώσσα με κλάσεις και ερωτήματα.

Παραδείγματα χρήσης LINQ

- Αποθήκευση βαθμολογία χρήστη μετά την ολοκλήρωση ενός διαγωνίσματος

```
var author1 = new Statistics.Models.Level_1 { Score = 2 * count,
Completed = isCompleted, UserId = (int)Session["UserId"], Date =
DateTime.UtcNow.Date };
context.Level_1.Add(author1);
 context.SaveChanges();
```

dbo.User [Data] | MyModel.Context.cs | MyModel.Context.tt | User.cs | resetbuttons.js

Max Rows: 1000

UserId	FirstName	LastName	Email	DateOfBirth	Password
50	christina	Papa	testact@test.gr	7/3/2020	123456
NULL	NULL	NULL	NULL	NULL	NULL

Στο παραπάνω πίνακα στη πρώτη στήλη είναι το μοναδικό Id του χρήστη (πρωτεύον κλειδί), το UserId που αντιστοιχεί στο πίνακα με τους χρήστες (είναι ξένο κλειδί). Το Score που έλαβε ο χρήστης, Completed αν ολοκλήρωσε επιτυχώς το διαγώνισμα και τέλος Date η ημερομηνία που συμμετείχε στο διαγώνισμα.

- Ερώτημα προς τη βάση αν ο συνδεδεμένος χρήστης έχει ολοκληρώσει ευτυχώς το Level 1

```
Statistics.Models.UserDatabaseEntities user = new
Statistics.Models.UserDatabaseEntities();
    var userId = Session["UserId"].ToString();
    var completedLevel1 = user.Level_1.Where(a=>
a.UserId.ToString() == userId && a.Completed ==
true).FirstOrDefault();
```

- Επίσης μπορούμε να χρησιμοποιήσουμε LINQ μέσα στο Controller για να ελέγξουμε παράδειγμα αν ένας χρήστης που κάνει login δίνει σωστά το email και το password. Αναζητούμε λοιπόν στη βάση αν το email (το οποίο είναι μοναδικό) και ο κωδικός μάς έδωσε ο χρήστης υπάρχει στη βάση

```
[ValidateAntiForgeryToken]
        public ActionResult LoginForm(UserLogin login)
        {

            using (UserDatabaseEntities dc = new UserDatabaseEntities())
            {
                var v = dc.User.Where(a => a.Email == login.Email && a.Password
            == login.Password).FirstOrDefault();
                if(v != null) {
                    Session["UserId"] = v.UserId;
                    Session["FirstName"] = v.FirstName;
                    Session["Email"] = v.Email;

                    return RedirectToAction("Index", "Home");
                }
            }
            ModelState.AddModelError("", "Invalid  credentials");
            return View(login); }
```

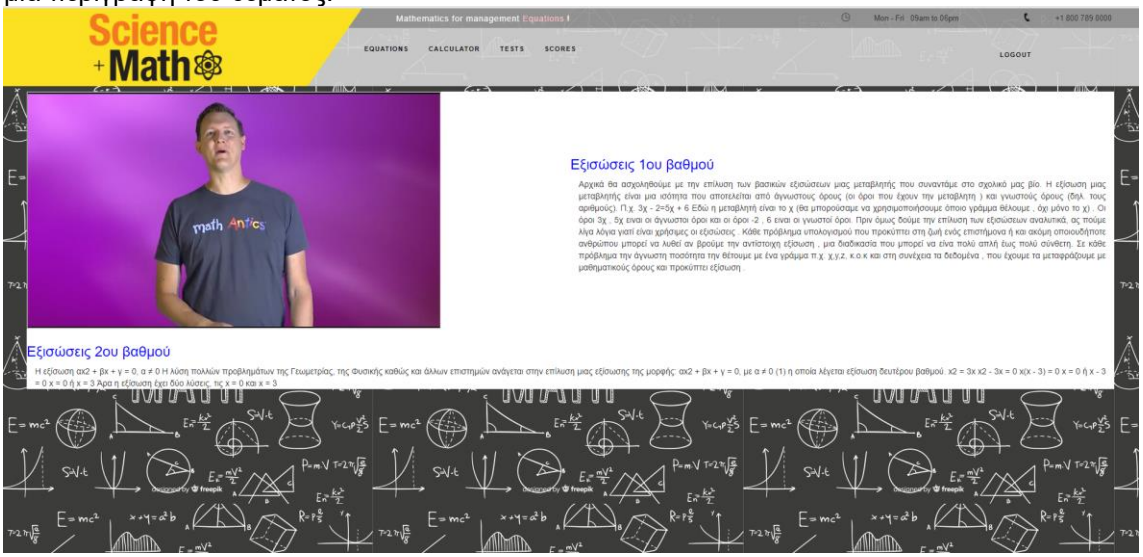
Κεφάλαιο 3 Παρουσίαση της Εφαρμογής

3.1. Αρχική σελίδα

Στην αρχική σελίδα ο χρήστης έχει το μενού επιλογών, μπορεί να πλοηγηθεί στις ενότητες:

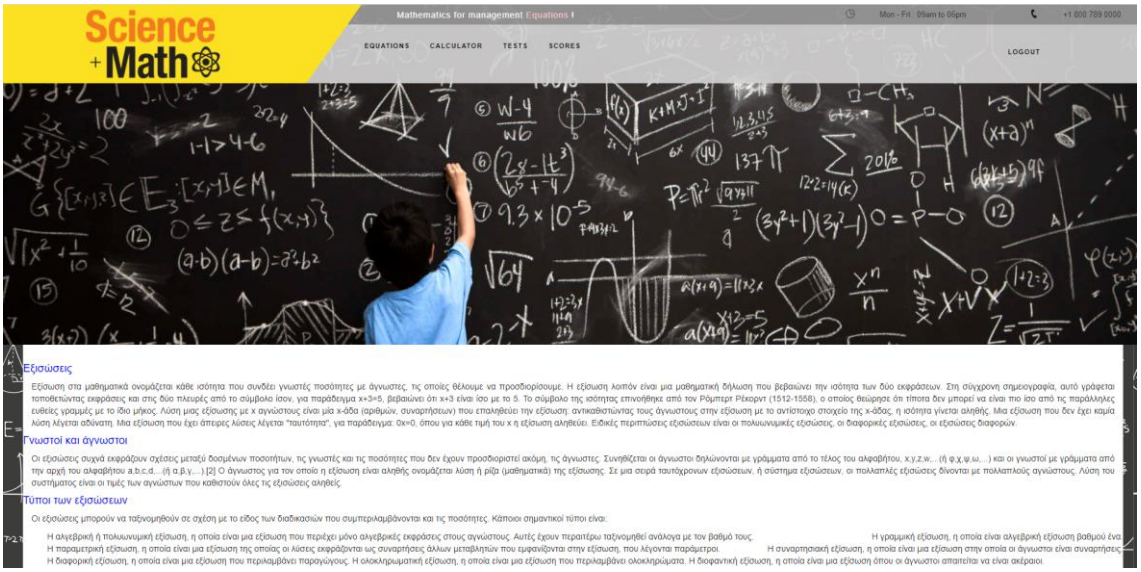
- Equations
- Calculator
- Tests
- Scores
- Sign / Logout

Μαζί με μία περιγραφή του θέματος.



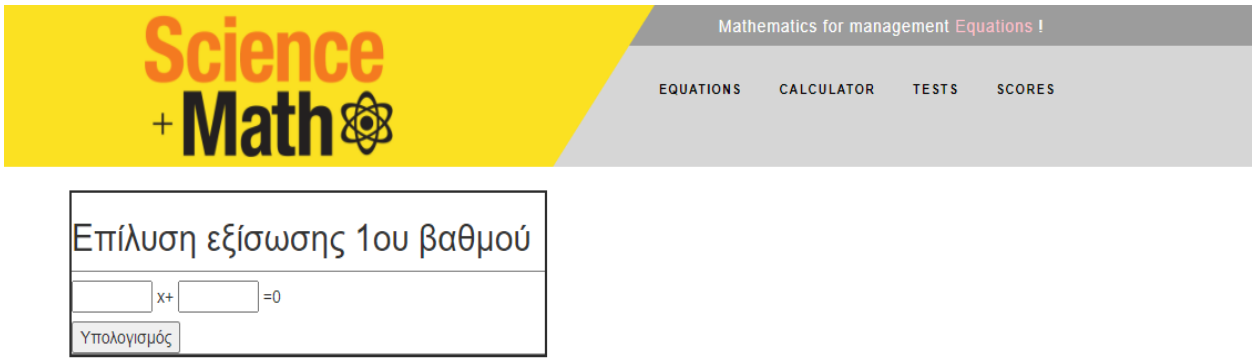
3.2. Equations

Όταν ζητηθεί η σελίδα equations όλοι οι χρήστες θα πλοηγηθούν σε αυτήν την σελίδα.

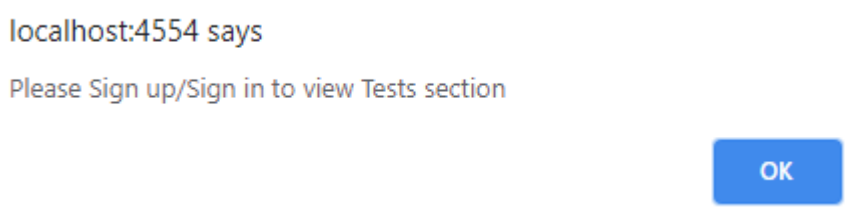


3.3. Calculator

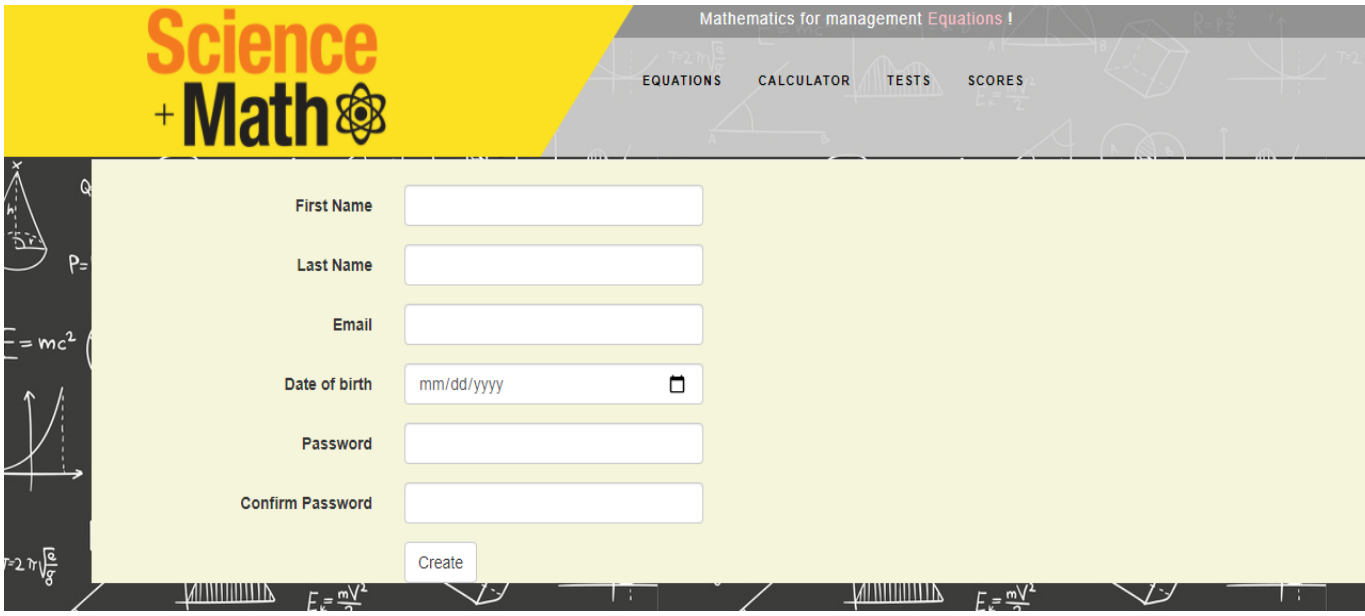
Όλοι οι χρήστες μπορούν να πλοηγηθούν στην σελίδα Calcuator.



Όταν ο χρήστης πατήσει στο link για τα tests εάν δεν είναι εγγεγραμένος θα του εμφανιστεί αυτό το μήνυμα.

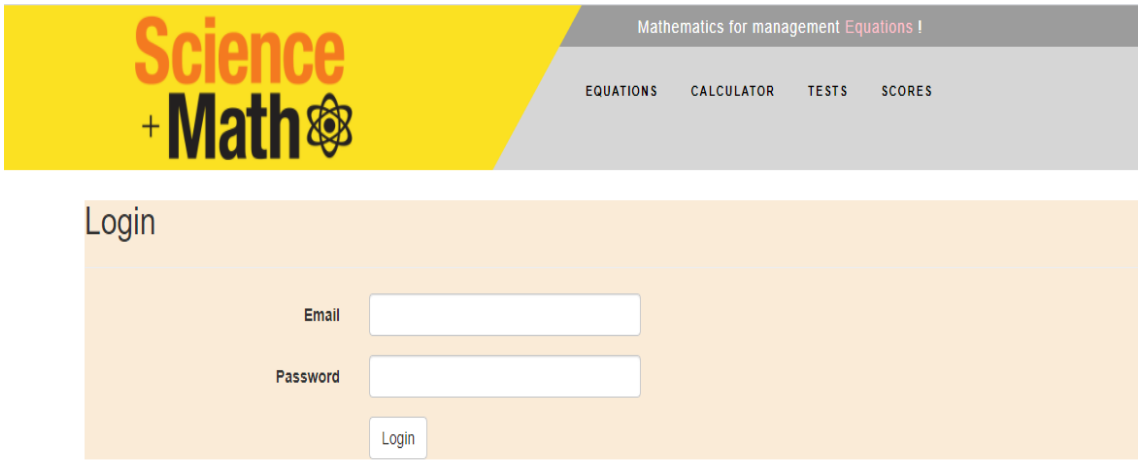


3.4. Sign in / Login



Στην ενότητα εγγραφής θα πρέπει να συμπληρωθούνε όνομα, επίθετο, ημερομηνία γέννησης καθώς και κωδικός χρήστη και επαλήθευση κωδικού. Ο κωδικός θα πρέπει να περιλαμβάνει τουλάχιστον 6 χαρακτήρες και επίσης το email να είναι έγκυρο. Επιπλέον το email να είναι μοναδικό καθώς ελέγχεται αν υπάρχει ίδιο email στη βάση δεδομένων.

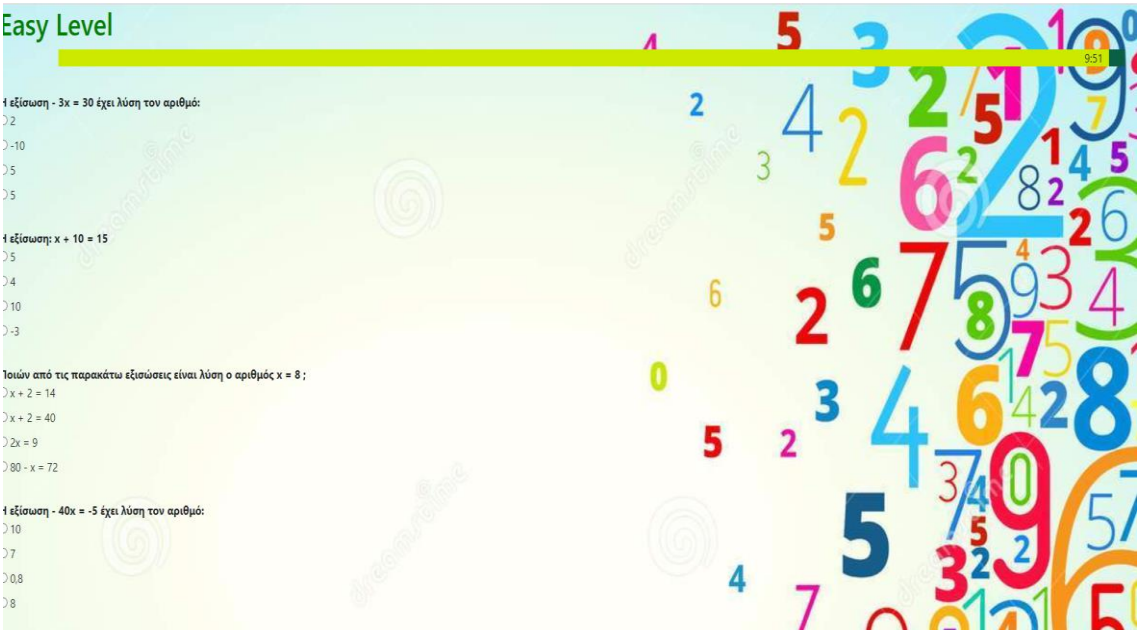
Σε περίπτωση που ο χρήστης είναι ήδη καταχωρημένος στην βάση δεδομένων τότε του εμφανίζεται η login φόρμα.



3.5. TESTS

Στην ενότητα των tests ο χρήστης μπορεί να δεί τα στοιχεία του, ποια επίπεδα έχει ολοκληρώσει και να επιλέξει σε ποιό επίπεδο θέλει να παίξει.

Τα επίπεδα είναι Easy level, Medium Level και Hard Level.

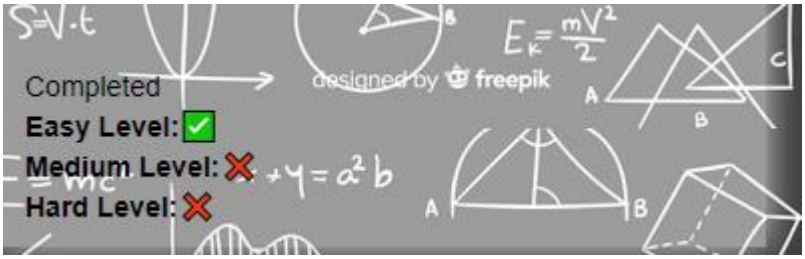


Κάθε επίπεδο έχει τις δικές του ερωτήσεις οι οποίες έρχονται ανακατεμένες και οι ερωτήσεις και οι απαντήσεις αλλά ακόμα και τα tests.

Κάθε επίπεδο έχει τον δικό του χρόνο. Στο εύκολο επίπεδο ο χρόνος είναι 10 λεπτά ενώ στον μέτριο επίπεδο είναι 6 λεπτά.

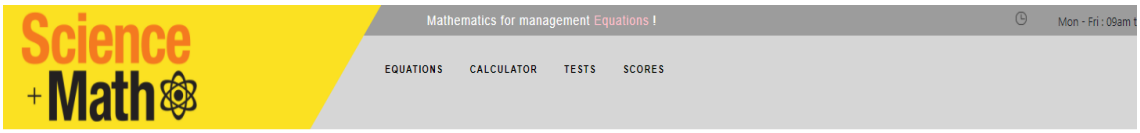
Μόλις τελειώσει το παιχνίδι ο χρήστης του εμφανίζονται σε πόσες απαντήσεις ήταν σωστός.

Και ποιά επίπεδα έχει ολοκληρώσει ο συγκεκριμένος χρήστης.



3.6. Scores

Εάν ο χρήστης θέλει να δει τα score γενικά όλων των χρηστών μπορεί να πάει στην καρτέλα με τα scores.



Προβολή Απόδοσης

Hard Level	Όνομα	Επίθετο	Βαθμολογία
Med Level	**Όνομα**	**Επίθετο**	**Βαθμολογία**
	christina	Papa	0
Easy Level	**Όνομα**	**Επίθετο**	**Βαθμολογία**
	christina	Papa	8

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1]https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/

[2] https://www.tutorialspoint.com/mvc_framework/index.htm

[3] https://en.wikipedia.org

[4] https://www.w3schools.com/asp/webpages_chart.asp

[4] https://stackoverflow.com/

[5] https://www.c-sharpcorner.com/

[6] http://www.entityframeworktutorial.net/querying-entity-graph-in-entity-framework.aspx

[7]https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/database-first-development/changing-the-database

[8] https://dotnet-helpers.com/mvc/retrieve-views-from-different-folders-in-mvc/

[9] http://ebooks.edu.gr/modules/ebook/show.php/DSGL-C100/493/3203,13010/

[10] https://www.mathsisfun.com/data/probability.html

[11] https://www.sololearn.com/

[12] https://codepen.io/bsngr/pen/frDqh

[13] http://www.dotnetawesome.com/2017/04/complete-login-registration-system-asp-mvc.html

[14] https://www.w3schools.com/asp/default.asp

[15] https://www.javatpoint.com/asp-net-tutorial

[16] http://www.tutorialsteacher.com/linq/linq-tutorials

[17] https://docs.microsoft.com/en-us/dotnet/csharp/linq/

[18] http://www.entityframeworktutorial.net/Querying-with-EDM.aspx

[19]https://www.aspsnippets.com/Articles/ASPNet-MVC-Database-Connection-Tutorial-with-example.aspx

[20]https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/creating-a-connection-string

[21] https://docs.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/index