ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ
Τμήμα Πληροφορικής

Εργασία Μαθήματος «Εκπαιδευτικό Λογισμικό»
Τεχνικό Εγχειρίδιο

Όνομα φοιτητή – Αρ. Μητρώου - Email	Παπαδόπουλος Αθανάσιος - Π16108 - tesarena@gmail.com
	Πετριδίσογλου Παναγιώτης - Π16115 - panoleous@gmail.com
	Σμύρης Βασίλειος - Π16131 - billyssmiris@gmail.com
Ημερομηνία παράδοσης	1/7/2020

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

1 Εισαγωγή

Το παρόν έγγραφο αποτελεί το τεχνικό εγχειρίδιο για τις εφαρμογές που αναπτύχθηκαν κατά την υλοποίηση της εργασίας. Επιλέξαμε το θέμα "Πρόσωπα του 1821". Κάθε εφαρμογή αποτελείται από δύο κύρια κομμάτια, το κομμάτι του υλικού και το κομμάτι της αξιολόγησης. Το κομμάτι του υλικού περιλαμβάνει το εκπαιδευτικό υλικό της εφαρμογής, το οποίο είναι χωρισμένο σε θεματικές ενότητες, και το κομμάτι της αξιολόγησης περιλαμβάνει τεστ γνώσης πολλάπλής επιλογής πάνω στο εκπαιδευτικό υλικό. Υπάρχει ένα τεστ για κάθε ενότητα, καθώς κι ένα επαναληπτικό, το οποίο περιλαμβάνει τις ερωτήσεις όλων το κεφαλαίων. Και οι δύο εφαρμογές αναπτύχθηκαν με τη χρήση της γλώσσας C# σε περιβάλλον Visual Studio 2019 σε Windows 10. Η Desktop εφαρμογή αναπτύχθηκε με το WPF framework και η Web με το ASP.NET framework.

2 Γενικά

Σε αυτή την ενότητα παρουσιάζονται κάποια κοινά στοιχεία που χρησιμοποιούν οι δύο εφαρμογές.

2.1 Βάση δεδομένων χρηστών

Η εφαρμογές παρέχουν τη δυνατότητα σύνδεσης των χρηστών με όνομα χρήστη και κωδικό. Η αποθήκευση αυτών των δεδομένων γίνεται σε μια εξωτερική βάση δεδομένων PostgreSQL, στην οποία έχουν πρόσβαση και οι δύο εφαρμογές. Τα στοιχεία των χρηστών αποθηκεύονται στον πίνακα users. Για κάθε χρήστη αποθηκεύονται τέσσερα στοιχεία, ένας αριθμός ταυτοποίησης, το όνομα χρήστη, ο κωδικός του και ο τύπος χρήστη. Το τελευταίο στοιχείο παίρνει την τιμή true αν ο χρήστης είναι μαθητής και false αν είναι καθηγητής.

2.2 Κλάση πρόσβασης στη βάση δεδομένων

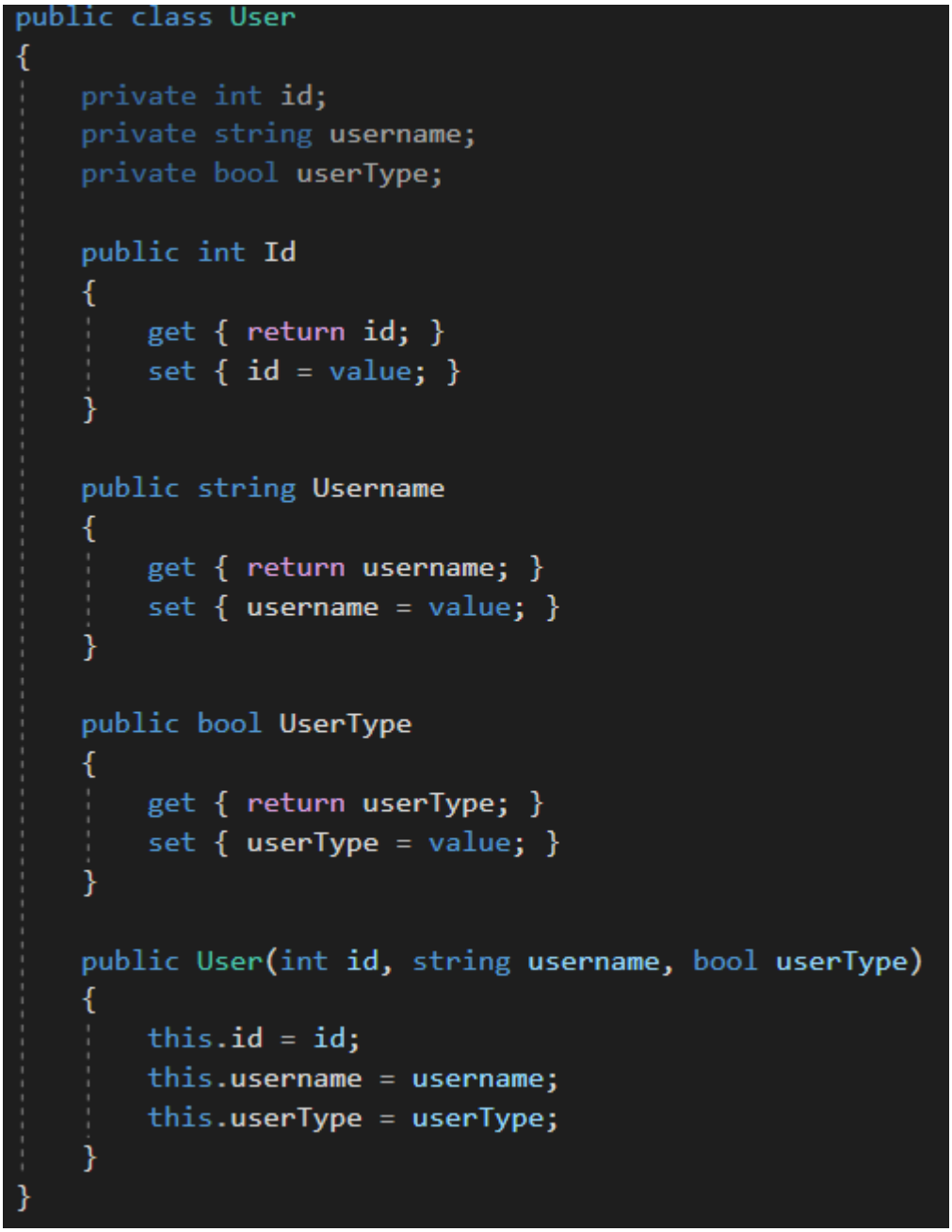
Για τη σύνδεση στη βάση και την ανάκτηση δεδομένων από αυτή, η εφαρμογές χρησιμοποιούν την κλάση DataAccess. Τα αντικείμενα της κλάσης αυτής συνδέονται στη βάση και περιέχουν συναρτήσεις που ανακτούν δεδομένα από αυτή.

```
private NpgsqlConnection con = new NpgsqlConnection("Host=localhost;Username=postgres;Password=19921992;Database=EkpaideftikoLogismiko");

public DataAccess()
{
    con.Open();
}

public User AuthenticateUser(string username, string password)
{
    var sql = "SELECT * FROM users WHERE username=@username;";
    var cmd = new NpgsqlCommand(sql, con);
    cmd.Parameters.Add("@username", NpgsqlTypes.NpgsqlDbType.Varchar);
    cmd.Parameters["@username"].Value = username;
    //cmd.Parameters.AddWithValue("@username", username);
    var dr = cmd.ExecuteReader();
    if (dr.Read())
    {
        if (password == (string)dr[2])
        {
            return new User((int)dr[0], (string)dr[1], (bool)dr[3]);
        }
    }
    return null;
}
```

Η συνάρτηση AuthenticateUser χρησιμοποιείται κατά τη σύνδεση των χρηστών για την αυθεντικοποίησή τους. Ανακτά τα στοιχεία του χρήστη από τη βάση με βάση το όνομα χρήστη που δόθηκε κατά τη σύνδεση,αν βρεβείο χρήστης, ελέγχεται ο κωδικός του κι αν είναι σωστός ο χρήστης συνδέεται. Η συνάρτηση επιστρέφει αντικείμενα της κλάσης User. Η κλάση αντιπροσωπεύει τους χρήστες στην εφαρμογή.



```
public class User
{
    private int id;
    private string username;
    private bool userType;

    public int Id
    {
        get { return id; }
        set { id = value; }
    }

    public string Username
    {
        get { return username; }
        set { username = value; }
    }

    public bool UserType
    {
        get { return userType; }
        set { userType = value; }
    }

    public User(int id, string username, bool userType)
    {
        this.id = id;
        this.username = username;
        this.userType = userType;
    }
}
```

Η κλάση User περιέχει πεδία για τον αριθμό ταυτοποίησης, το όνομα χρήστη και τον τύπο του χρήστη.

2.3 Αρχεία εφαρμογών

Το περιεχόμενο των εφαρμογών είναι αποθηκευμένο με μια συγκεκριμένη δομή αρχείων κειμένου η οποία ακολουθεί το προτόκολλο που ακολουθεί η κάθε εφαρμογή για να διαβάσει το περιεχόμενό της και να το παρουσιάσει στο χρήστη.

Τα αρχεία περιεχομένου βρίσκονται στο φάκελο "Resources" στην εφαρμογή Desktop και στο φάκελο "Content" στην εφαρμογή Web. Για κάθε κεφάλαιο του εκπαιδευτικού υλικού, υπάρχουν 2 αρχεία τα οποία ακολοθούν το πρότυπο ονομασίας "unitx", όπου x ο αριθμός του κεφαλαίου. Το ένα αρχείο βρίσκεται στο φάκελο Images/Backgrounds. Είναι μια εικόνα οπισθόφυλλο που είναι θεματικά παρεμφερείς με το περιεχόμενο του κεφαλαίου στο οποίο αντιστοιχεί. Το δεύτερο αρχείο βρίσκεται στο φάκελο Data/unitdata. Περιέχει τις πληροφορίες του κεφαλαίου. Στην πρώτη γραμμή βρίσκεται ο τίτλος του κεφαλαίου σε φυσική γλώσσα. Στις υπόλοιπες γραμμές βρίσκονται τα κωδικά ονόματα των προσώπων που περιέχει το κεφάλαιο. Με το κωδικό όνομα ονομάζονται όλα τα αρχεία που έχουν να κάνουν με το κάθε πρόσωπο. Για κάθε πρόσωπο υπάρχουν 3 αρχεία. Στο φάκελο Images/People υπάρχει μια φωτογραφία με το πορτρέτο κάθε προσώπου. Στο φάκελο Data/peopleinfo, υπάρχει ένα αρχείο txt με κάποιες πληροφορίες για το πρόσωπο. Στην πρώτη γραμμή βρίσκεται το όνομά του ολογράφως. Στη δεύτερη η ημερομηνία γέννησής του. Στην τρίτη ο τόπος γέννησής του. Στην τέταρτη η ημερομηνία θανάτου του. Στην πέμπτη, ο τόπος θανάτου του. Ακόμα, υπάρχει ένα αρχείο στο φάκελο Data/peopledesc που περιέχει ένα κείμενο που περιγράφει τη ζωή του προσώπου. Στη Desktop εφαρμογή, αυτά τα αρχεία είναι τύπου xaml και στη Web είναι τύπου html. Χρησιμοποιούν τη γλώσσα μορφοποίησης που χρησιμοποιεί η τεχνολογία της κάθε εφαρμογής ώστε να μπορεί η εφαρμογή να δείχνει εύκολα όμορφα κι εμπλουτισμένα με πολυμέσα κείμενα.

Για τα τεστ αξιολόγησης, υπάρχει ένα αρχείο για κάθε κεφάλαιο κι ένα επαναληπτικό τεστ στο φάκελο Data/quizes. Τα αρχεία των τεστ των κεφαλαίων ονομάζονται "unitx" και του επαναληπτικού "revision". Στην πρώτη γραμμή βρίσκεται ο αριθμός των ερωτήσεων που περιέχει το τεστ. Για τις επόμενες ακολουθεί το παρακάτω. Για κάθε πέντε γραμμές, στην πρώτη βρίσκεται το κείμενο μιας ερώτησης και στις επόμενες τέσσερις οι απαντήσεις της ερώτησης. Κάθε γραμμή απάντησης αποτελείται από το κείμενο της απάντησης, το σύμβολο "|" και τη λέξη true ή false, που δείχνει αν η ερώτηση είναι σωστή η λάθος. Ακολουθώντας αυτή τη δομή αρχείων και τα πρότυπα ονομασίας, κάποιος μπορεί να τροποποιήσει υπαρκτά η να φτιάξει νέα κεφάλαια σχετικά εύκολα και γρήγορα και με ελάχιστες γνώσεις προγραμματισμού.

3 Εφαρμογή Desktop

Σε αυτή την ενότητα αναλύεται η Desktop εφαρμογή.

3.1 Σύνδεση

Κατά την εκκίνηση της εφαρμογής, εμφανίζεται στο χρήστη ένα παράθυρο με μία φόρμα σύνδεσης. Η φόρμα περιέχει ένα πεδίο για το όνομά του, τον κωδικό του κι ένα κουμπί καταχώρησης.

Όταν ο χρήστης πατήσει το κουμπί τα στοιχεία του ελέγχονται για την ορθότητά τους.

```
private void loginButton_Click(object sender, RoutedEventArgs e)
{
    if(!(String.IsNullOrEmpty(username.Text) || String.IsNullOrEmpty(password.Password)))
    {
        var d = new DataAccess();
        User user = d.AuthenticateUser(username.Text, password.Password);
        if (user != null)
        {
            new MainWindow(user).Show();
            this.Close();
        }
        else
        {
            username.Clear();
            password.Clear();
        }
    }
}
```

Τα στοιχεία του χρήστη ανακτούνται κι ελέγχονται με ένα αντικείμενο DataAccess. Αν τα στοιχεία είναι σωστά, τότε εμφανίζεται το κύριο μενού.

3.2 Κύριο μενού

Στο κύριο μενού υπάρχουν τρία κουμπιά. Το πρώτο εμφανίζει το μενού επιλογής κεφαλαίων. Το δεύτερο το μενού επιλογής τεστ. Το τρίτο κλείνει την εφαρμογή. Το κύριο μενού και όλες οι άλλες οθόνες της εφαρμογής δεν είναι δικά του παράθυρα. Είναι controls τα οποία τίθενται ως περιεχόμενο σε ένα ContentControl που βρίσκεται σε ένα παράθυρο.

```
public partial class Menu : UserControl
{
    public Menu()
    {
        InitializeComponent();
    }

    private void unitsButton_Click(object sender, RoutedEventArgs e)
    {
        (VisualTreeHelper.GetParent(VisualTreeHelper.GetParent(this)) as ContentControl).Content = new UnitMenu();
    }

    private void exitButton_Click(object sender, RoutedEventArgs e)
    {
        Window.GetWindow(this).Close();
    }

    private void quizButton_Click(object sender, RoutedEventArgs e)
    {
        (VisualTreeHelper.GetParent(VisualTreeHelper.GetParent(this)) as ContentControl).Content = new QuizMenu();
    }
}
```

3.3 Μενού επιλογής κεφαλαίου

Από αυτό το μενού μπορεί να επιλέξει ο χρήστης ένα κεφάλαιο για να διαβάσει το περιεχόμενό του. Το κάθε κεφάλαιο αντιπροσωπεύεται από μια θεματική εικόνα και τον τίτλο του. Ο χρήστης μπορεί να πατήσει πάνω σε ένα από αυτά τα ζευγάρια εικόνας-τίτλου. Όταν ο χρήστης περνά τον κέρσορα πάνω από ένα κεφάλαιο, η εικόνα του αποκτά ένα πράσινο περίγραμμα.

Κάθε ένα στοιχείο του μενού έχει ένα όνομα, ανάλογα σε ποιο κεφάλαιο αντιστοιχεί. Για παράδειγμα το στοιχείο του δεύτερου κεφαλαίου έχει όνομα "unit2".

```
<StackPanel Orientation="Vertical" Grid.Row="1" Grid.Column="1">
    <Border Name="unit2" Style="{StaticResource ImageHoverBorder}" MouseUp="OpenUnit">
        <Image Source="/Resources/Images/Backgrounds/unit2.jpg"/>
    </Border>
    <TextBlock Text="Κεφάλαιο 2:" HorizontalAlignment="Center" VerticalAlignment="Center"/>
    <TextBlock Text="Έλληνες Επαναστάτες" HorizontalAlignment="Center" VerticalAlignment="Center"/>
</StackPanel>
```

Το όνομα αυτό δίνεται ως όρισμα κατά τη δημιουργία της οθόνης του κεφαλαίου για να προσπελαστούν τα αρχεία του κεφαλαίου, τα οποία έχουν το ίδιο όνομα.

```
private void OpenUnit(object sender, MouseButtonEventArgs e)
{
    (VisualTreeHelper.GetParent(VisualTreeHelper.GetParent(this)) as ContentControl).Content = new Unit((sender as Border).Name);
}
```

3.4 Κεφάλαιο

Στην οθόνη του κεφαλαίου παρουσιάζονται τα πορτρέτα των προσώπων για τα οποία περιέχει πληροφορίες το κεφάλαιο. Εμφανίζονται μέχρι τρία πορτρέτα σε κάθε γραμμή και συνολικά εμφανίζονται μέχρι έξι πορτρέτα.

```
public Unit(string unit)
{
    InitializeComponent();
    this.unit = unit;
    StreamResourceInfo info = Application.GetResourceStream(new Uri("/Resources/Data/unitdata/" + unit + ".txt", UriKind.Relative));
    StreamReader sr = new StreamReader(info.Stream);
    string line;
    sr.ReadLine();
    int i = 0;
    while ((line = sr.ReadLine()) != null && i < 6)
    {
        Image img = new Image();
        Border b = new Border();
        b.Name = line;
        b.AddHandler(Border.MouseUpEvent, new RoutedEventHandler(ShowDetails));
        b.Style = Resources["UnitPersonImageHover"] as Style;
        var y = (System.Windows.SystemParameters.PrimaryScreenWidth) / 6 - b.Width / 2 - b.BorderThickness.Left;
        b.Margin = new Thickness(y, b.Margin.Top, y, b.Margin.Bottom);
        img.Source = new BitmapImage(new Uri(BaseUriHelper.GetBaseUri(this), "Resources/Images/People/" + line + ".jpg"));
        img.Style = Resources["UnitPersonImage"] as Style;
        b.Child = img;
        wrapPanel.Children.Add(b);
        i++;
    }
    sr.Close();
    Background = new ImageBrush(new BitmapImage(new Uri(BaseUriHelper.GetBaseUri(this), "Resources/Images/Backgrounds/" + unit + ".jpg")));
}
```

Κατά την κατασκευή της οθόνης του κεφαλαίου, διαβάζεται το αρχείο με τις πληροφορίες του κεφαλαίου. Η πρώτη γραμμή με τον τίτλο παραλείπεται. Διαβάζονται τα κωδικά ονόματα των προσώπων του κεφαλαίου και ανακτούνται οι φωτογραφίες τους. Δημιουργούνται τα controls του μενού με τις εικόνες των προσώπων. Τα controls παίρνουν το όνομα του προσώπου στο οποίο αντιστοιχούν. Με το πάτημα τους, τα controls εκτελούν τη συνάρτηση Show Details.

```
private void ShowDetails(object sender, RoutedEventArgs e)
{
   string name = (sender as Border).Name;
    StreamResourceInfo info = Application.GetResourceStream(new Uri("/Resources/Data/peopledata/peopleinfo/" + name + ".txt", UriKind.Relative));
    StreamReader sr = new StreamReader(info.Stream);
    detailsPanel.name.Text = sr.ReadLine();
    detailsPanel.portrait.Source = new BitmapImage(new Uri(BaseUriHelper.GetBaseUri(this), "Resources/Images/People/" + name + ".jpg"));
    detailsPanel.birthDate.Text = sr.ReadLine();
    detailsPanel.birthPlace.Text = sr.ReadLine();
    detailsPanel.deathDate.Text = sr.ReadLine();
    detailsPanel.deathPlace.Text = sr.ReadLine();
    wrapPanel.Visibility = Visibility.Collapsed;
    backButton.Visibility = Visibility.Collapsed;
    detailsPanel.Visibility = Visibility.Visible;
    //new Uri(BaseUriHelper.GetBaseUri(this), "Resources/Data/peopledata/peopledesc/" + name + ".xaml")
    detailsPanel.flowDocumentScrollViewer.Document = Application.LoadComponent(new Uri("/Resources/Data/peopledata/peopledesc/" + name + ".xaml",
    sr.Close();
}
```

Η συνάρτηση διαβάζει το όνομα του controls και ανακτά τις πληροφορίες του προσώπου στο οποίο αντιστοιχεί το όνομα. Η πληροφορίες τοποθετούνται στα κατάλληλα controls ενός κρυμμένου πάνελ, και μετά αυτό το πάνελ εμφανίζεται.

3.5 Μενού επιλογής τεστ

Από αυτό το μενού ο χρήστης μπορεί να επιλέξει ένα τεστ. Υπάρχει ένα κουμπί για κάθε τεστ. Το κάθε κουμπί έχει ως όνομα το όνομα ενός τεστ. Πατώντας το, το όνομα περνιέται ως όρισμα στη στον constructor που φτιάχνει την οθόνη του τεστ.

```
<Button x:Name="unit2" Style="{StaticResource MenuButton}" Content="Κεφάλαιο 2" Grid.Column="1" Grid.Row="2" Click="startQuiz" />
```

```
public void startQuiz(object sender, RoutedEventArgs e)
{
    (VisualTreeHelper.GetParent(VisualTreeHelper.GetParent(this)) as ContentControl).Content = new Quiz((sender as Button).Name);
}
```

3.6 Τεστ

Για τη δημιουργία του τεστ, διαβάζεται το αντίστοιχο αρχείο με τις ερωτήσεις και τις απαντήσεις.

```
public partial class Quiz : UserControl
{
    private static int questionTime = 30;
    private string unit;
    private StreamReader sr;
    private DispatcherTimer timer;
    private int time;
    private bool[] answers = new bool[4];
    private double correctQuestions = 0;
    private double totalQuestions = 0;
    private int currentQuestions = 0;

    public Quiz(string unit)
    {
        InitializeComponent();
        this.DataContext = this;
        this.unit = unit;
        StreamResourceInfo info = Application.GetResourceStream(new Uri("/Resources/Data/quizes/" + unit + ".txt", UriKind.Relative));
        sr = new StreamReader(info.Stream);
        totalQuestions = double.Parse(sr.ReadLine());
        time = questionTime;
        questionTimer.Text = TimeString();
        timer = new DispatcherTimer();
        timer.Tick += new EventHandler(timer_Tick);
        timer.Interval = new TimeSpan(0, 0, 1);
        timer.Start();
        RenderQuestion();
    }
```

Αρχικοποιείται το χρονόμετρο της ερώτησης και εμφανίζεται η πρώτη ερώτηση με τη συνάρτηση RenderQuestion.

```
private bool RenderQuestion()
{
    string q = sr.ReadLine();
    if (string.IsNullOrEmpty(q))
    {
        timer.Stop();
        grade.Text = "Βαθμολογία: " + Math.Floor(correctQuestions / totalQuestions * 100) + "%";
        questionPanel.Visibility = Visibility.Collapsed;
        endPanel.Visibility = Visibility.Visible;
        questionTimer.Visibility = Visibility.Hidden;
        title.Text = "Τέλος";
        return false;
    }
    currentQuestions++;
    title.Text = "Ερώτηση " + currentQuestions + "/" + totalQuestions;
    question.Text = q;
    string[] a;
    a = sr.ReadLine().Split("|");
    answer1.Content = a[0];
    answers[0] = bool.Parse(a[1]);
    a = sr.ReadLine().Split("|");
    answer2.Content = a[0];
    answers[1] = bool.Parse(a[1]);
    a = sr.ReadLine().Split("|");
    answer3.Content = a[0];
    answers[2] = bool.Parse(a[1]);
    a = sr.ReadLine().Split("|");
    answer4.Content = a[0];
    answers[3] = bool.Parse(a[1]);
    ResetControls();
    ResetTimer();
    return true;
}
```

Αρχικά ελέγχεται αν έχουν τελειώσει οι ερωτήσεις. Αν έχουν τελειώσει, εμφανίζεται η βαθμολογία του χρήστη και το τεστ κρύβεται. Αν όχι, διαβάζονται 5 γραμμές από το αρχείο του τεστ, μια ερώρηση και τέσσερις απαντήσεις και τα κείμενα τους τοποθετούνται στα κατάλληλα controls. Τα radio buttons του τεστ παίρνουν τις τιμές αληθείας των απαντήσεων.

Ο χρήστης έχει 30 δευτερόλεπτα να απαντήσει στην ερώτηση. Πρέπει να επιλέξει μια απάντηση για να ενεργοποιηθεί το κουμπί καταχώρησης της απάντησης. Αν δεν προλάβει να πατήσει το κουμπί πριν τη λήξη του χρόνου, η απάντηση θεωρείται λάθος, ακόμα κι αν έχει πατήσει τη σωστή απάντηση. Όπως φαίνεται στην παρακάτω εικόνα σε κάθε κύκλο του χρονόμετρου, ο χρόνος μειώνεται και ελέγχεται αν έχει τελειώσει και αν έχει εμφανίζεται η επόμενη ερώτηση.

```
public void timer_Tick(object sender, EventArgs e)
{
    time--;
    questionTimer.Text = TimeString();
    if (time == 0)
    {
        RenderQuestion();
    }
}
```

Όταν ο χρήστης πατήσει το κουμπί καταχώρησης, ελέγχεται αν η ερώτηση είναι σωστή και αν υπάρχει επόμενη, εμφανίζεται. Επίσης, το χρονόμετρο επανεκκινείται.

```
private void answerButton_Click(object sender, RoutedEventArgs e)
{
    timer.Stop();
    if (CheckAnswer()) {
        correctQuestions++;
    }

    if (RenderQuestion())
    {
        ResetTimer();
        timer.Start();
    }
}
```

4 Εφαρμογή Web

Σε αυτή την ενότητα αναλύεται η Web εφαρμογή.

4.1 Σύνδεση

Κατά την είσοδο στη σελίδα της εφρμογής ο χρήστης οδηγείται στην κεντρική σελίδα. Πάνω δεξιά, μπορεί να πατήσει το κουμπί σύνδεση για να εμφανιστεί μια φόρμα σύνδεσης. Ο χρήστης πρέπει να συπμηρώσει τα στοιχεία του και να πατήσει το κουμπί σύνδεση. Τη διαδικασία αναλαμβάνει η συνάρτηση Login του Login Controller.

```
public ActionResult Login()
{
    NameValueCollection formData = Request.Form;
    User user = new DataAccess().AuthenticateUser(formData["username"], formData["password"]);
    if (user != null)
    {
        HttpCookie cookie = new HttpCookie("username");
        cookie.Values.Add("username", formData["username"]);
        cookie.Expires = DateTime.Now.AddHours(12);
        Response.Cookies.Add(cookie);
    }
    //return View("../Home/Index");
    return RedirectToAction("Index", "Home");
}
```

Η συνάρτηση δέχεται την είσοδο του χρήστη, ανακτά τα στοιχεία του από τη βάση, ελέγχει τον κωδικό του και αν είναι όλα σωστα αποθηκεύει το όνομα του σε ένα cookie και τον μεταφέρει στην αρχική σελίδα.

Η συνάρτηση Logout στον ίδιο controller αναλαμβάνει την αποσύνδεσή του χρήστη, διάγράφοντας το cookie με το όνομα.

```
public ActionResult Logout()
{
    if (Request.Cookies["username"] != null)
    {
        Response.Cookies["username"].Expires = DateTime.Now.AddDays(-1);
    }
    return RedirectToAction("Index", "Home");
}
```

4.2 Κύριο σελίδα

Στην κύρια σελίδα, ο χρήστης μπορεί να πατήσει ένα από τα δύο εικονίδα για να μεταφερθεί είτε στη σελίδα επιλογής κεφαλαίου, είτε στη σελίδα επιλογής τεστ.

```
<div class="main-menu-action" action="Units">
    <span class="fa fa-book"></span>
    Υλικό
</div>
```

Το attribute "action" στο εικονίδιο δείχνει σε ποιά από τις δύο σελίδες ανακατευθύνει το χρήστη. Κατά το πάτημά του, εκτελείται το παρακάτω script.

```
$(".main-menu-action").click(function () {
    var action = this.getAttribute("action");
    window.location.href = "/" + action + "/" + action;
});
```

Η συνάρτηση διαβάζει το attribute "action" και ανακατευθύνει στο αντίστοιχο action του αντίστοιχου controller.

4.3 Μενού επιλογής κεφαλαίου

Από αυτό το μενού μπορεί να επιλέξει ο χρήστης ένα κεφάλαιο για να διαβάσει το περιεχόμενό του. Το κάθε κεφάλαιο αντιπροσωπεύεται από μια θεματική εικόνα και τον τίτλο του. Ο χρήστης μπορεί να πατήσει πάνω σε ένα από αυτά τα ζευγάρια εικόνας-τίτλου. Όταν ο χρήστης περνά τον κέρσορα πάνω από ένα κεφάλαιο, η εικόνα γίνεται ελαφρώς διάφανη. Το μενού χτίζεται με τον παρακάτω τρόπο.

```
public ActionResult Units()
{
    string[] units = Directory.GetFiles(HostingEnvironment.MapPath("~/Content/Data/unitdata"), "unit*.txt");
    string[] titles = new string[units.Length];
    StreamReader sr = null;
    for(int i = 0; i < titles.Length; i++)
    {
        sr = new StreamReader(units[i], Encoding.Default);
        titles[i] = sr.ReadLine();
        sr.Close();
    }

    units = units.Select(Path.GetFileName).ToArray();
    for (int i = 0; i < units.Length; i++)
    {
        units[i] = units[i].Split('.')[0];
    }
    ViewBag.units = units;
    ViewBag.titles = titles;
    return View();
}
```

Για κάθε αρχείο με μορφή unit* στο φάκελο Data/unitdata διαβάζεται η πρώτη γραμμή, ο τίτλος του κεφαλαίου σε φυσική γλώσσα, και κρατείται το όνομα του αρχείου ως εσωτερικό όνομα του κεφαλαίου. Δημιουργούνται 2 πίνακες με τα στοιχεία αυτά για όλα τα κεφάλαια.

```
@{ int i = 0; }
@foreach(string unit in ViewBag.units)
{
    i++;
    <div class="col-xs-6 text-center">
        <div id="@(unit)" class="unit">
            <img src="~/Content/Images/Backgrounds/@(unit).jpg" height="300" />
            <div>Κεφάλαιο @(i): @(ViewBag.titles[i - 1])</div>
        </div>
    </div>
}
```

Με τις προηγούμενες πληροφορίες, δημιουργείται στο μενού μία επιλογή για κάθε κεφάλαιο.

4.4 Κεφάλαιο

Στην οθόνη του κεφαλαίου παρουσιάζονται τα πορτρέτα των προσώπων για τα οποία περιέχει πληροφορίες το κεφάλαιο. Εμφανίζονται μέχρι τρία πορτρέτα σε κάθε γραμμή.

```
public ActionResult Unit(string unit)
{
    StreamReader sr = new StreamReader(HostingEnvironment.MapPath("~/Content/Data/unitdata/" + unit + ".txt"), Encoding.Default);
    ViewBag.unitTitle = sr.ReadLine();
    List<string> people = new List<string>();
    while(!sr.EndOfStream)
    {
        string temp = sr.ReadLine();
        people.Add(temp);
    }
    sr.Close();
    ViewBag.unit = unit;
    ViewBag.people = people;
    ViewBag.num = unit[unit.Length - 1];
    return View();
}
```

Κατά την κατασκευή της οθόνης του κεφαλαίου, διαβάζεται το αρχείο με τις πληροφορίες του κεφαλαίου. Διαβάζονται τα κωδικά ονόματα των προσώπων του κεφαλαίου και δημιουργείται πίνακας με αυτά.

```
<div class="container">
    @foreach(string person in ViewBag.people)
    {
        <div class="col-xs-4 text-center">
            <div id="@(person)" class="person">
                <img src="~/Content/Images/People/@(person).jpg" height="300"/>
            </div>
        </div>
    }
</div>
```

Δημιουργούνται τα controls με τα πρόσωπα με τις κατάλλες φωτογραφίες, χρησιμοποιώντας τον πίνακα με τα ονόματα. Πατώντας ένα control, εμφανίζονται οι πληροφορίες του προσώπου. Οι πληροφορίες ανακτούνται από το server στο παρασκήνιο με ένα AJAX call.

```
$(".person").click(function () {
    var id = this.id;
    $.ajax({
        url: '/Units/Info',
        data: {
            person: id,
            format: 'json'
        },
        error: function () {

        },
        dataType: 'json',
        success: function (data) {
            $('#name').html(data.name);
            $('#img').attr('src', "/Content/Images/People/" + id + ".jpg");
            $('#birthdate').html(data.birthdate);
            $('#birthplace').html(data.birthplace);
            $('#deathdate').html(data.deathdate);
            $('#deathplace').html(data.deathplace);
            $('#desc').html(data.desc);
        },
        type: 'POST'
    });
    $("#detailsModal").modal('toggle');
});
```

Αφού ανακτηθούν, οι πληροφορίες καταγράφονται στο πάνελ πληροφοριών. Οι πληροφορίες λαμβάνονται ως JSON αρχείο.

Από τη μεριά του server, μαζεύονται οι πληροφορίες από τα σχετικά αρχεία και πακετάρονται σε ένα JSON αρχείο.

```
public ActionResult Info(string person)
{
    StreamReader sr = new StreamReader(HostingEnvironment.MapPath("~/Content/Data/peopledata/peopleinfo/" + person + ".txt"), Encoding.Default);
    string Name = sr.ReadLine();
    string Birthdate = sr.ReadLine();
    string Birthplace = sr.ReadLine();
    string Deathdate = sr.ReadLine();
    string Deathplace = sr.ReadLine();
    sr = new StreamReader(HostingEnvironment.MapPath("~/Content/Data/peopledata/peopledesc/" + person + ".html"), Encoding.Default);
    string Desc = sr.ReadToEnd();
    return Json(new { name = Name, birthdate = Birthdate, birthplace = Birthplace, deathdate = Deathdate, deathplace = Deathplace, desc = Desc });
}
```

4.5 Μενού επιλογής τεστ

Από αυτό το μενού ο χρήστης μπορεί να επιλέξει ένα τεστ. Υπάρχει ένας σύνδεσμος για κάθε τεστ. Οι σύνδεσμοι δημιουργούνται με παρόμοιο τρόπο με τα controls στο μενού επιλογής κεφαλαίου.

```
@{
    int i = 0;
}
@foreach (string unit in ViewBag.units)
{
    i++;
    <a href="/Quizes/Quiz/@(unit)" class="quiz-link">Κεφάλαιο @(i): @(ViewBag.titles[i - 1])</a><br/>
}
<a href="/Quizes/Quiz/revision" class="quiz-link">Επανάληψη</a>
```

4.6 Τεστ

Για τη δημιουργία του τεστ, διαβάζεται το αντίστοιχο αρχείο με τις ερωτήσεις και τις απαντήσεις.

```
public ActionResult Quiz(string unit)
{
    List<Question> questions = new List<Question>();
    List<Answer> answers = null;
    StreamReader sr;
    if (unit != "revision")
    {
        sr = new StreamReader(HostingEnvironment.MapPath("~/Content/Data/unitdata/" + unit + ".txt"), Encoding.Default);
        ViewBag.unitTitle = sr.ReadLine();
    }
    else
    {
        ViewBag.unitTitle = "Επανάληψη";
    }
    sr = new StreamReader(HostingEnvironment.MapPath("~/Content/Data/quizes/" + unit + ".txt"), Encoding.Default);
    ViewBag.totalQuestions = sr.ReadLine();
    while (!sr.EndOfStream)
    {
        var text = sr.ReadLine();
        answers = new List<Answer>();
        for(int i = 0; i < 4; i++)
        {
            var answer = sr.ReadLine().Split('|');
            answers.Add(new Answer(answer[0], bool.Parse(answer[1])));
        }
        questions.Add(new Question(text, answers));
    }
    sr.Close();
    ViewBag.unit = unit;
    ViewBag.questions = questions;
    ViewBag.num = unit[unit.Length - 1];
    return View();
}
```

Αρχικοποιείται το χρονόμετρο της ερώτησης και εμφανίζεται η πρώτη ερώτηση με τη συνάρτηση RenderQuestion.

```
function RenderQuestion() {
    i++;
    if (i == totalQuestions) {
        clearInterval(timer);
        $("#questionForm").hide();
        $("#score").html('Επιτυχία: ' + parseInt(correctAnswers / totalQuestions * 100) + '%');
        $("#end").css('visibility', 'visible');
    }
    else {
        $("#questionCounter").html("Ερώτηση " + (i + 1) + "/" + totalQuestions);
        $("#questionText").html(questions[i].Text);
        var answers = questions[i].Answers;
        $("#answer0Label").html(answers[0].Text);
        $("#answer1Label").html(answers[1].Text);
        $("#answer2Label").html(answers[2].Text);
        $("#answer3Label").html(answers[3].Text);
    }
}
```

Ο χρήστης έχει 30 δευτερόλεπτα να απαντήσει στην ερώτηση. Πρέπει να επιλέξει μια απάντηση και να πατήσει το κουμπί καταχώρησης της απάντησης. Αν δεν προλάβει να πατήσει το κουμπί πριν τη λήξη του χρόνου, η απάντηση θεωρείται λάθος, ακόμα κι αν έχει πατήσει τη

σωστή απάντηση. Όπως φαίνεται στην παρακάτω εικόνα σε κάθε κύκλο του χρονόμετρου, ο χρόνος μειώνεται και ελέγχεται αν έχει τελειώσει και αν έχει εμφανίζεται η επόμενη ερώτηση.