

# *Documetantie pentru Personality Type Application*

**TEMA: APLICAȚIE WEB PENTRU DETERMINAREA TIPULUI DE  
PERSONALITATE**

**APLICATIE WEB REALIZATA IN CADRUL PRACTICII DE VARA PENTALOG - 2017**

MEMBRII ECHIPEI:

- MIRUNA DAIAN
- ANDREEA BOLONY
- IUGA PAULA

MENTORII ECHIPEI:

- GIURESCU ANDREI
- MARIAN FILIPOVICI

## CUPRINS

<b>ARGUMENT.....</b>	<b>2</b>
<b>APLICAȚIE WEB VERSUS SITE WEB .....</b>	<b>4</b>
<b>PREZENTARE GENERALĂ : VISUAL STUDIO ȘI MVC .....</b>	<b>5</b>
MODEL.....	6
VIEW.....	10
CONTROLLER.....	11
<b>MODUL DE LUCRU: DE LA MODEL LA BAZA DE DATE.....</b>	<b>12</b>
<b>STRUCTURA PAGINILOR .....</b>	<b>13</b>
<b>PRIMA PAGINĂ (PAGINA DE START A APLICAȚIEI): .....</b>	<b>14</b>
<b>CUM FUNCȚIONEAZĂ APLICAȚIA WEB?.....</b>	<b>18</b>
<b>CARUSELUL .....</b>	<b>22</b>
<b>RADIO BUTTONS ȘI CALCULUL REZULTATULUI FINAL .....</b>	<b>24</b>
<b>PROGRESS BAR-UL .....</b>	<b>27</b>
<b>BIBLIOGRAFIE.....</b>	<b>29</b>

## ARGUMENT

Pe lângă științele exacte, de pildă matematica și infomatica, **psihologia**, știința care, în viziunea mea, încearcă să dezlege nebănuitele taine ale creierului (acea mașinărie complexă care ne definește), va rămâne mereu știința care mă pasionează.

*„Psihologia este în cele din urmă mitologie,  
studiul poveștilor sufletului.”*

Din acest motiv, am ales ca tema aplicației mele web să fie tipurile de personalitate. Deși ființa umană este inimaginabil de complexă, totuși există anumite trăsături caracteristice care pot delimita indivizii în anumite „grupuri”, aceste trăsături definind tipul de personalitate.

Oamenii de știință spun că aproximativ jumătate din variațiile personalității umane sunt definite de factorii genetici. Cu toate acestea, personalitatea este construită din două trăsături diferite: caracterul(format pe baza experiențelor) și temperamentul(tendințe înnăscute). Cei mai mulți dintre noi suntem un amestec unic al

celor patru tipuri diferite de personalitate și toate trăsăturile acestora sunt prezente, într-o anumită măsură, în noi. Unele trăsături de personalitate apar însă predominant, iar acestea dictează personalitatea noastră de bază.

Aplicația web, inspirată din primul test de psihologie al lui **Wilhelm M. Wundt**, determină pe baza unui test grilă de 20 de întrebări, tipul de personalitate (*melancolic/sangvinic/coleric/flegmatic*) al utilizatorului în funcție de majoritatea răspunsurilor date.

## APLICAȚIE WEB VERSUS SITE WEB

Proiectul meu este, de fapt, o aplicație web. Deși majoritatea ar crede că este același lucru cu site-ul web, totuși există diferențe majore între acestea.

**Un site web** este o culegere de pagini web vizualizate cu ajutorul unui browser. Este un set static de pagini care oferă viewerului (persoanei care îl accesează/ manipulează ) detalii. Un site web sau un website **nu** permite ,însă, spectatorilor să realizeze o **comunicare eficientă** ,fiind statice.

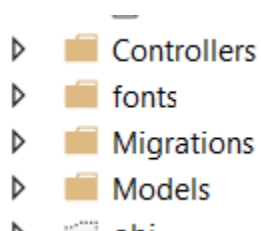
**O aplicație web** este **interactivă** (de exemplu Facebook), **dinamică** , bazându-se pe interacțiunea cu utilizatorul. Astfel, o aplicație web își schimbă mereu conținutul în funcție de comenzile date de utilizator, afișand numai informațiile dorite din baza de date și chiar informații preluate din alte surse.

## PREZENTARE GENERALĂ : VISUAL STUDIO ȘI MVC

Aplicația web este de tip **MVC** și este realizată în **Microsoft Visual Studio**.

**MVC** este prescurtarea pentru **Model-View-Controller** (în romana model-vizualizare-controlor). Acesta este un model arhitectural utilizat în ingineria software.

(În aplicația realizată există câte un folder (dosar) pentru fiecare parte a acestui sistem arhitectural.)



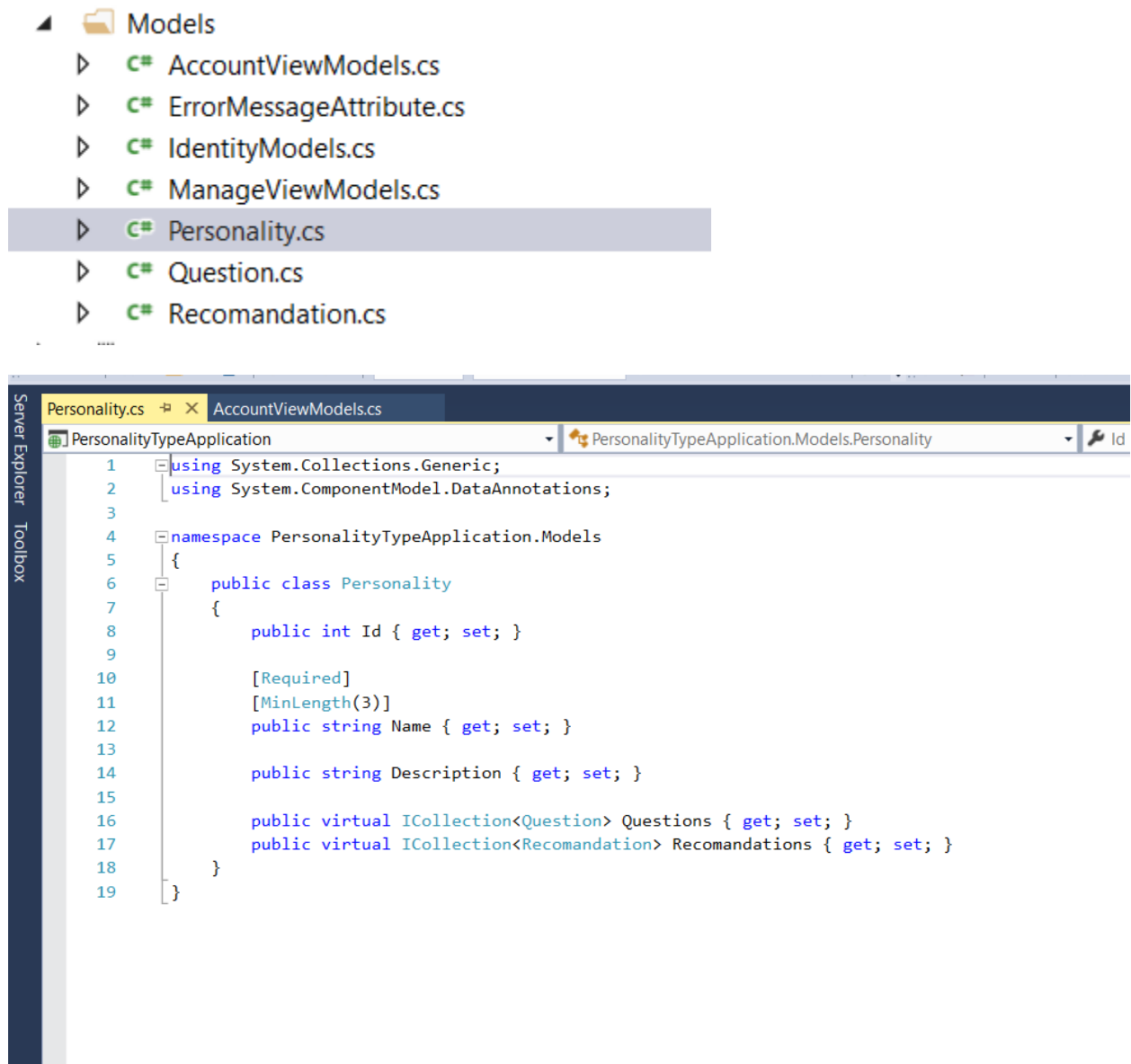
Sistemul arhitectural se bazează pe corespondență neputând să existe sau să funcționeze una dintre componente fără cealaltă:

Utilizatorul, prin comenzile sale trimite cererile către controller, acesta accesează modelurile. Este aleasă data (informația) corespunzătoare care este transmisă înapoi controllerului. Acesta accesează, în final, view-rile( în care apare conținutul modelului/ modul în care acesta va apărea pe ecran), răspunsul fiind trimis înapoi utilizatorului pe ecran.

### • Modelul

Este, practic, informația folosită de aplicație. Acesta conține modul în care este repartizată informația( sub formă de clase) și nu cum arată aceasta.

**Clasele** (classes) sunt o formă mai dezvoltată de structuri, pe lângă tipurile de date acestea pot conține și funcții.



În exemplul de mai sus este modelul pt. personalitate, toate cele 4 personalități fiind stucturate astfel:

fiecare dintre aceste personalități are câte un ID de tip număr întreg ( câmp care este obligatoriu, fiind validat cu **required** și are obligatoriu lungimea minimă de 3 caractere, un nume de tip șir de caractere și o descriere de tip șir de caractere, pe lângă acestea fiecărui tip de personalitate îi este atribuită o descriere și un anumit set de întrebări, care au la rândul lor un model).

**Informatiile apar structurate în baza de date la fel cum apar în model.** La fel apar și legaturile, de pildă între o personalitate (de exemplu sangvinic) și întrebările corespunzătoare ei (5)) se stabilește **o relație de tipul unu-la-mai-mulți (one-to-many)** (unei personalități îi corespund mai multe tipuri de întrebări).

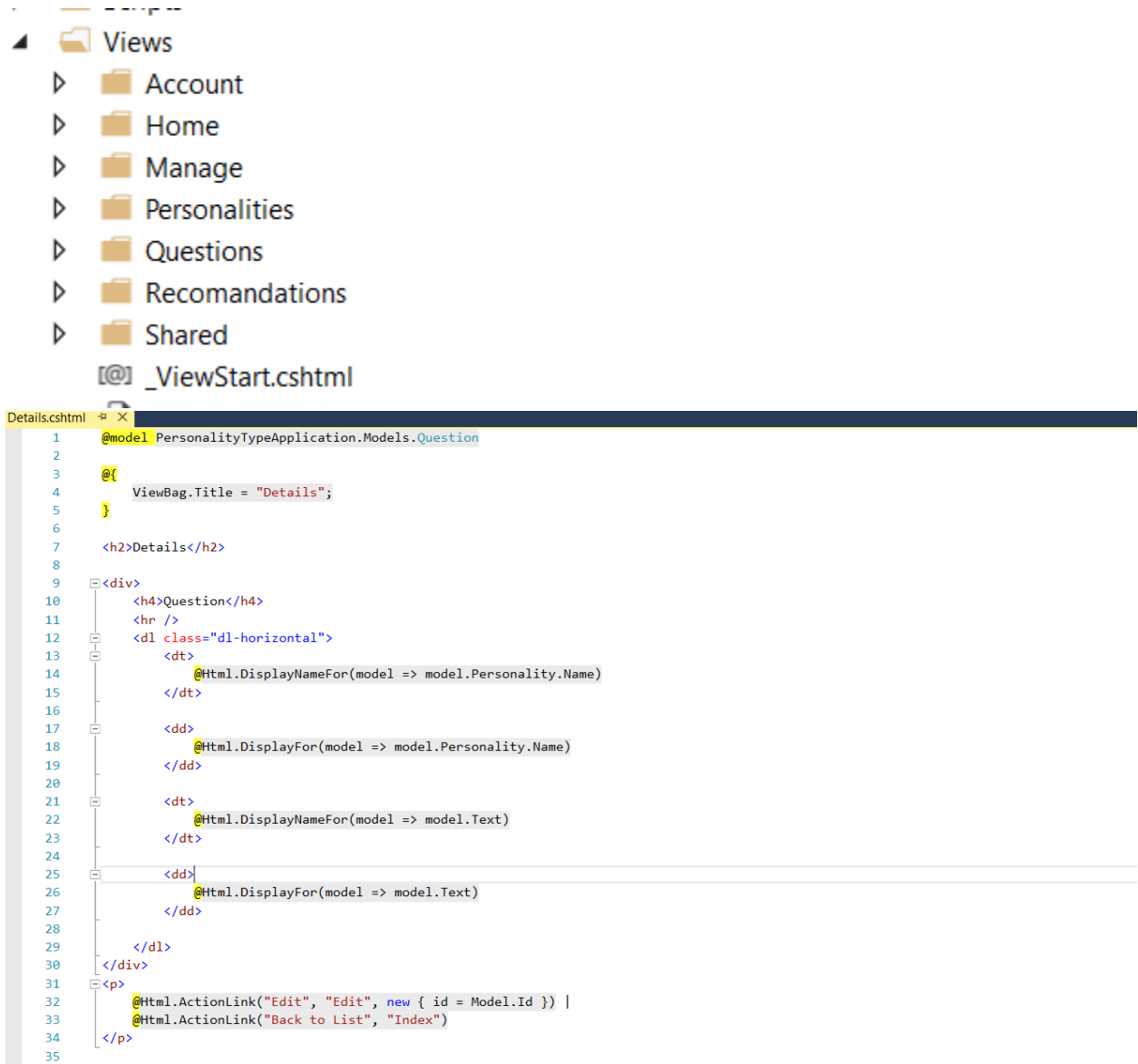
Id	Name	Description
2	Choleric	Someone with pure choleric temperament is usual...
3	Phlegmatic	The Phlegmatic goes through life doing as little a...
4	Melancholic	People with melancholic personality type love tra...
8	Sanguine	People with sanguine personality type tend to be ...
*	NULL	NULL



Id	Text	PersonalityId
11	I make plan...	2
12	I like to finis...	2
13	I work until ...	2
14	I feel my an...	2
15	I choose my...	2
16	I bet a lot O...	2
17	I get angry ...	2
18	I am not em...	2
19	I am good a...	2
20	I always sup...	2
21	New interac...	3
22	The relation...	3
23	Restoring th...	3
24	You are seri...	3
25	You regular...	3
26	The details ...	3
27	You usually ...	3
28	Thoughts d...	3
29	You have jus...	3
30	You are a lo...	3
31	I am easily i...	4
32	I panic easily...	4

- **View-ul**

Este **interfața grafică** ce interacționează cu utilizatorul în final. Este modul în care arată informația.



The screenshot shows the Visual Studio IDE. On the left, the 'Views' folder is expanded, showing subfolders: Account, Home, Manage, Personalities, Questions, Recomendations, and Shared. Below these is the file '\_ViewStart.cshtml'. The main editor window displays the 'Details.cshtml' file. The code is as follows:

```

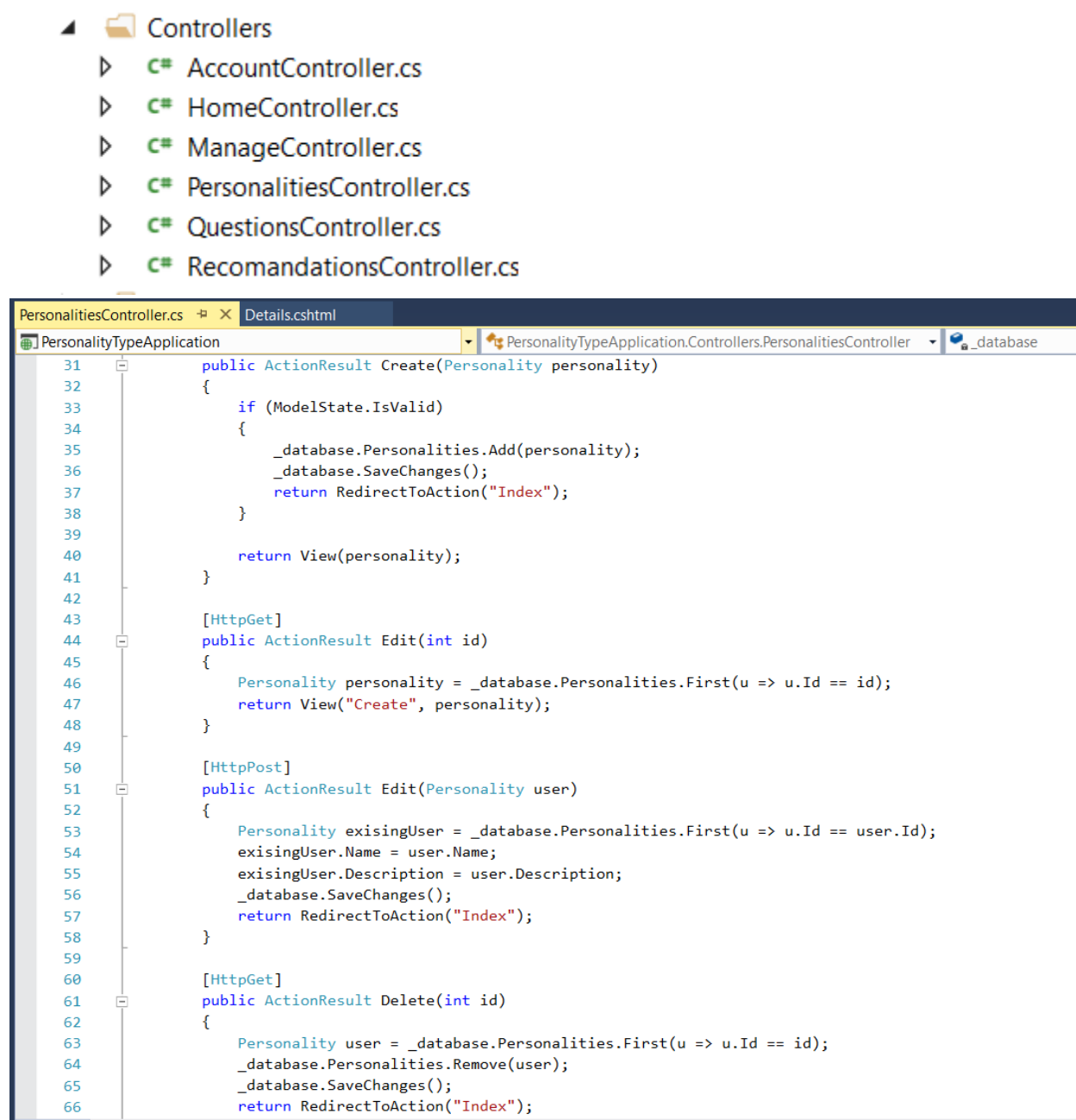
1  @model PersonalityTypeApplication.Models.Question
2
3  @{
4      ViewBag.Title = "Details";
5  }
6
7  <h2>Details</h2>
8
9  <div>
10     <h4>Question</h4>
11     <hr />
12     <dl class="dl-horizontal">
13         <dt>
14             @Html.DisplayNameFor(model => model.Personality.Name)
15         </dt>
16         <dd>
17             @Html.DisplayFor(model => model.Personality.Name)
18         </dd>
19         <dt>
20             @Html.DisplayNameFor(model => model.Text)
21         </dt>
22         <dd>
23             @Html.DisplayFor(model => model.Text)
24         </dd>
25     </dl>
26 </div>
27
28 <p>
29     @Html.ActionLink("Edit", "Edit", new { id = Model.Id }) |
30     @Html.ActionLink("Back to List", "Index")
31 </p>
32
33
34
35

```

(mai sus se află view-ul pt. detaliile unei întrebări(cum se afișează textul unei întrebări))

## • Controllerul

Face legătura dintre **model** și **view**, fiind practic „creierul” aplicației și **interpretează acțiunile utilizatorului**, cum ar fi click-urile pe anumite butoane.



The screenshot shows the Visual Studio IDE. At the top, a file explorer displays the 'Controllers' folder containing several files: AccountController.cs, HomeController.cs, ManageController.cs, PersonalitiesController.cs, QuestionsController.cs, and RecomendationsController.cs. Below this, the 'PersonalitiesController.cs' file is open in the editor. The code defines four actions: 'Create', 'Edit' (GET), 'Edit' (POST), and 'Delete'. The 'Create' action adds a new personality to the database and redirects to the 'Index' page. The 'Edit' (GET) action retrieves a personality by ID and returns the 'Create' view. The 'Edit' (POST) action updates an existing personality and redirects to the 'Index' page. The 'Delete' action removes a personality by ID and redirects to the 'Index' page. The code uses a database context named '\_database'.

```

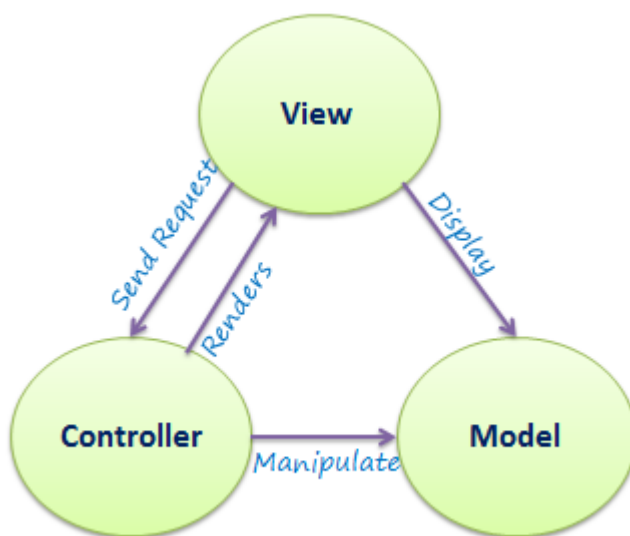
31 public ActionResult Create(Personality personality)
32 {
33     if (ModelState.IsValid)
34     {
35         _database.Personalities.Add(personality);
36         _database.SaveChanges();
37         return RedirectToAction("Index");
38     }
39
40     return View(personality);
41 }
42
43 [HttpGet]
44 public ActionResult Edit(int id)
45 {
46     Personality personality = _database.Personalities.First(u => u.Id == id);
47     return View("Create", personality);
48 }
49
50 [HttpPost]
51 public ActionResult Edit(Personality user)
52 {
53     Personality existingUser = _database.Personalities.First(u => u.Id == user.Id);
54     existingUser.Name = user.Name;
55     existingUser.Description = user.Description;
56     _database.SaveChanges();
57     return RedirectToAction("Index");
58 }
59
60 [HttpGet]
61 public ActionResult Delete(int id)
62 {
63     Personality user = _database.Personalities.First(u => u.Id == id);
64     _database.Personalities.Remove(user);
65     _database.SaveChanges();
66     return RedirectToAction("Index");

```

În controllerul „Personalities” avem mai multe funcții-acțiuni (de pildă pt. fiecare personalitate prin intermediul interfeței, utilizând contul de administrator, putem să edităm, să ștergem sau să adăugăm o descriere, schimbările făcându-se și în baza de date).

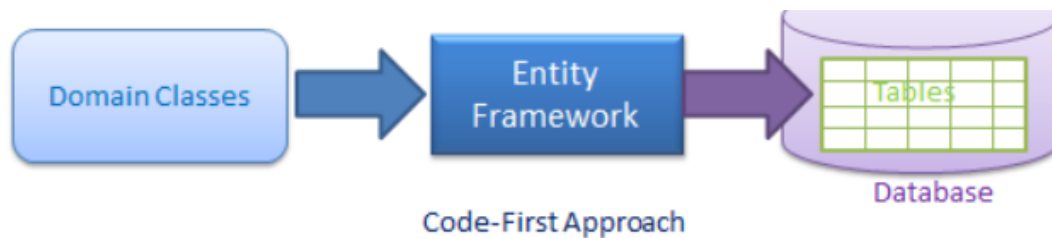
Sistemul architectural **MVC** se bazează pe corespondență neputând să existe sau să funcționeze una dintre componente fără cealaltă:

*Utilizatorul, prin comenzile sale trimite cererile către controller, acesta accesează **modelurile**. Este aleasă data (informația) corespunzătoare care este transmisă înapoi **controllerului**. Acesta accesează, în final, **view-rile** (în care apare conținutul modelului, modul în care acesta va apărea pe ecran), răspunsul fiind trimis înapoi utilizatorului pe ecran.*



## MODUL DE LUCRU DE LA MODEL LA BAZA DE DATE

Pentru realizarea aplicației web am lucrat **code-first**, utilizând **Entity-Framework**. Acest mod de lucru vizează faptul că întâi sunt create clasele, adică modelele, apoi acestea sunt configurate, în final creându-se automat bazele de date sub formă de tabel pe baza modelelor.



De exemplu pe baza modelului „Questions” s-a creat automat prin **migrare** tabelul aferent din baza de date .

```

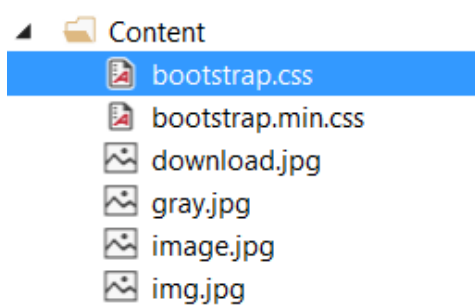
1 using System.ComponentModel;
2 using System.ComponentModel.DataAnnotations;
3
4 namespace PersonalityTypeApplication.Models
5 {
6     public class Question
7     {
8         public int Id { get; set; }
9
10        [Display(Name = "Question text")]
11        public string Text { get; set; }
12
13        [Display(Name = "Personality type")]
14        public int PersonalityId { get; set; }
15
16        [Display(Name = "Personality type")]
17        public virtual Personality Personality { get; set; }
18    }
19 }
    
```



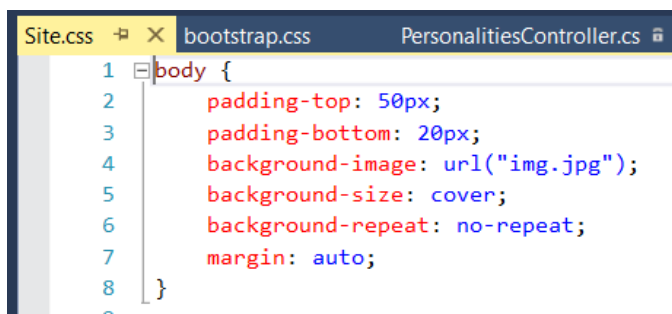
dbo.Questions [Data]			
Question.cs			
Max Rows: 1000			
	Id	Text	PersonalityId
	11	I make plan...	2
	12	I like to finis...	2
	13	I work until ...	2
	14	I feel my an...	2
	15	I choose my...	2
	16	I bet a lot o...	2
	17	I get angry ...	2
	18	I am not em...	2
	19	I am good a...	2
	20	I always sup...	2
	21	New interac...	3
	22	The relation...	3

## STRUCTURA PAGINILOR

Toate paginile au aceeași temă, designul acestora bazându-se pe o temă principală (culori, fonturi ,aranjare în pagină) de **Bootstrap**<sup>1</sup>, descărcată de pe <https://getbootstrap.com> și atașată proiectului in folderul *Content*:



Imaginea de fundal de pe fiecare pagină este atașată utilizând **CSS**:



Pe fiecare pagină apare în partea de sus **Navigation barul** :



**Navigation barul** este creat utilizând Html și CSS și conține 3 butoane(**action-linkuri**) organizate sub forma unei liste neordonate.

<sup>1</sup> Bootstrap este un framework CSS gratuit și open-source, orientat către dezvoltarea de web-uri responsabile. Acesta conține șabloane de design CSS bazate sau nu pe JavaScript pentru tipografie, formulare, butoane, navigație și alte componente ale interfeței.

Un site care utilizează Bootstrap foarte fi vizualizat la fel de bine pe orice dispozitiv.

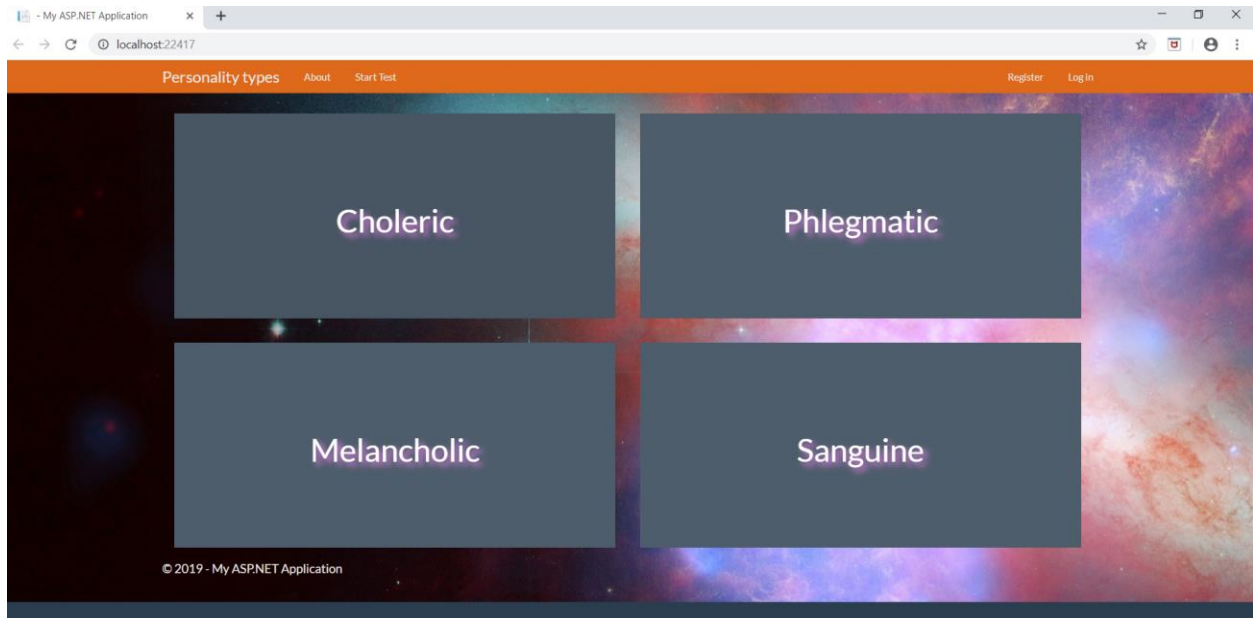
Acestea deschid câte o pagină nouă atunci când sunt apăsate în funcție de controllerul și view-ul corespunzătoare.

```

10 </head>
11 <body>
12     <div class="navbar navbar-inverse navbar-fixed-top">
13         <div class="container">
14             <div class="navbar-header">
15                 <button type="button" class="navbar-toggle" data-toggle="collapse" data-target=".navbar-collapse">
16                     <span class="icon-bar"></span>
17                     <span class="icon-bar"></span>
18                     <span class="icon-bar"></span>
19                 </button>
20                 <@Html.ActionLink("Personality types", "Index", "Home", new { area = "" }, new { @class = "navbar-brand" })>
21             </div>
22             <div class="navbar-collapse collapse">
23                 <ul class="nav navbar-nav">
24
25                     <li><@Html.ActionLink("About", "About", "Home")</li>
26                     <li><@Html.ActionLink("Start Test", "TestInstructions", "Questions")</li>
27
28                 </ul>
29
30                 <@if (User.Identity.Name.Equals("admin@yahoo.com"))>
31                 {
32                     <li><@Html.ActionLink("PERSONALITIES", "Index", "Personalities")</li>
33                     <li><@Html.ActionLink("Questions", "Index", "Questions")</li>
34                 }
35             </div>

```

## PRIMA PAGINĂ (PAGINA DE START A APLICAȚIEI):





Conține un tabel în care sunt introduse cele 4 tipuri de personalități:

```

Index.cshtml
7 <h2>Index</h2>
8
9 <p>
10     @Html.ActionLink("Create New", "Create")
11 </p>
12 <table class="table">
13 <tr>
14 <th>
15     @Html.DisplayNameFor(model => model.Name)
16 </th>
17 <th>
18     @Html.DisplayNameFor(model => model.Description)
19 </th>
20 </tr>
21 </tr>
22
23 @foreach (var item in Model) {
24 <tr>
25 <td>
26     @Html.DisplayFor(modelItem => item.Name)
27 </td>
28 <td>
29     @Html.DisplayFor(modelItem => item.Description)
30 </td>
31 </tr>

```

Astfel, pentru fiecare element de tip personalitate din baza de date se creeaza practic câte un buton în care este afișat numele tipului de personalitate(prin intermediul funcției **JavaScript** **DisplayFor(modelItem => Item name)** din structura repetitivă **foreach**, care afișează numele pt. fiecare personalitate)

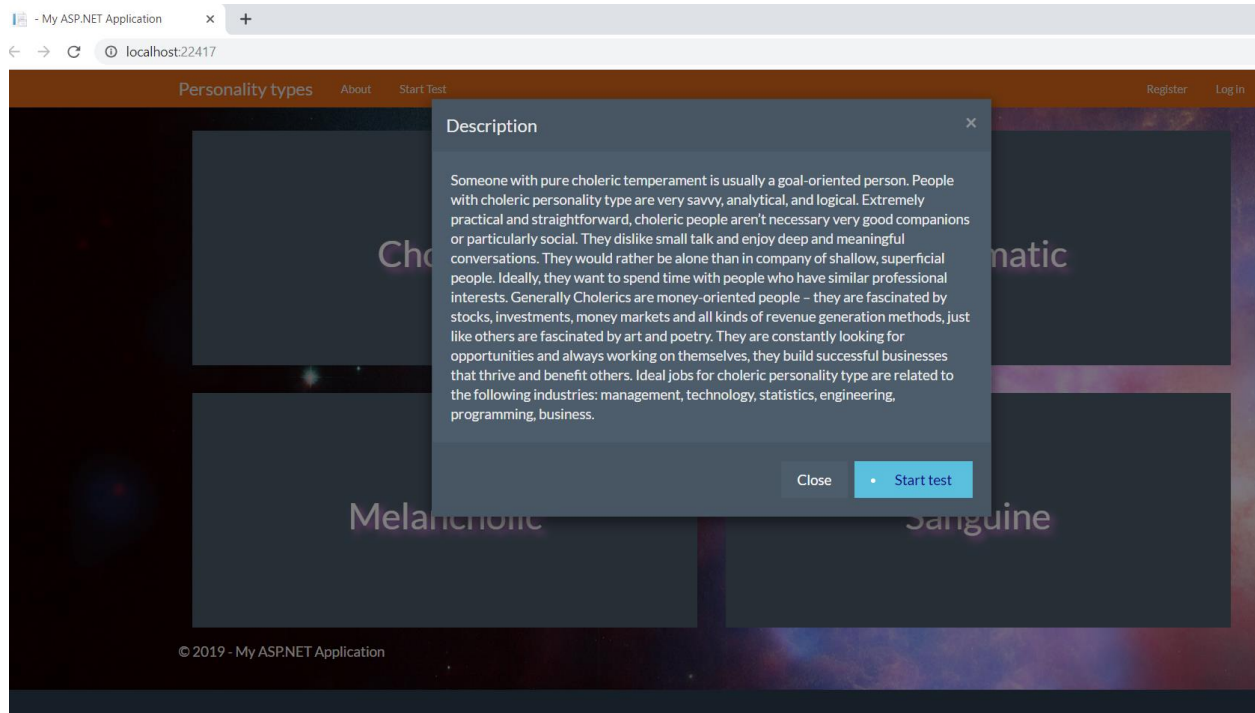
Dacă se apasă pe un tip de personalitate apare o fereastră în care se găsește numele descrierii ,un buton de close și unul pt. Start\_test(pt a începe testul de personalitate). Fereastra care se deschide este, de fapt, un **modal** creat în **html și JavaScript**, butoanele sunt, de fapt, **action linkuri** către noi pagini. Acestea trimit cereri către **controlerul Questions** pt. a încărca pagina cu instrucțiunile testului din **view-ul Starttest**.

## Documentatie pentru Personality Type Application

```
@foreach (var item in Model)
{
    <div class="col-xs-6">

        <button type="button" class="btn btn-default btn-lg personalityBtn" data-toggle="modal" data-target="#@item.Name">
            <h1>@item.Name</h1>
        </button>

        <!-- Modal -->
        <div class="modal fade" id="@item.Name" tabindex="-1" role="dialog" aria-labelledby="myModallabel">
            <div class="modal-dialog" role="document">
                <div class="modal-content">
                    <div class="modal-header">
                        <button type="button" class="close" data-dismiss="modal" aria-label="Close"><span aria-hidden="true">&times;</span></button>
                        <h4 class="modal-title" id="myModallabel">Description</h4>
                    </div>
                    <div class="modal-body">
                        @item.Description
                    </div>
                    <div class="modal-footer">
                        <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
                        <button type="button" class="btn btn-info">
                            <li>@Html.ActionLink("Start test", "TestInstructions", "Questions")</li>
                        </button>
                    </div>
                </div>
            </div>
        </div>
    }
}
```



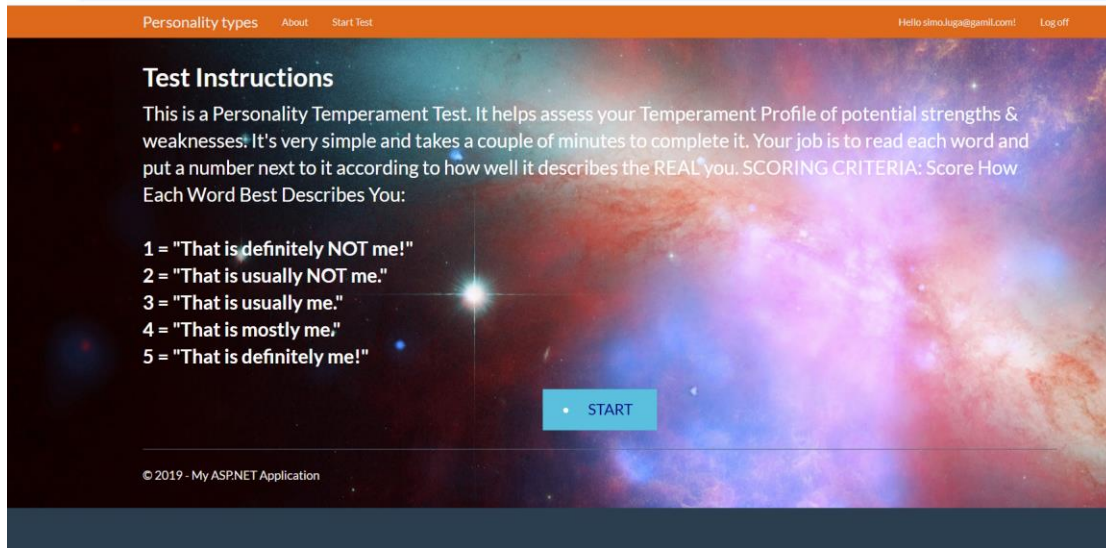
## CUM FUNCȚIONEAZĂ APLICAȚIA WEB?

Pentru a începe testul utilizatorul trebuie să apese **butonul start test** din bara de navigație sau din descrierea oricărei personalități. Mai mult, utilizatorul trebuie să fie **obligatoriu logat**. Din acest motiv, dacă nu a fost logat înainte să apese pe butonul strat\_test i se deschide automat pagina de „**REGISTER**”.

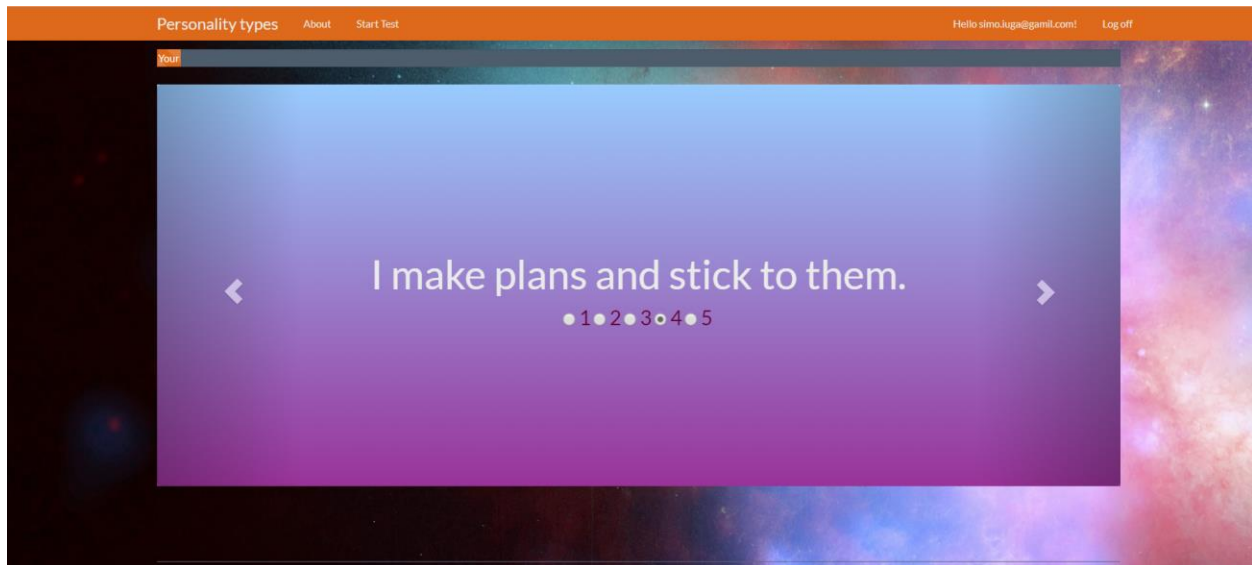
!Parola pe care o va introduce trebuie să repescte câteva validări (minim 6 caractere, să conțină cel puțin o cifră și cel puțin o literă mare), în cazul nerespectării acestora se afișează un mesaj de alertă, de pildă:

Dupa logare/înregistrare utilizatorul poate începe testul:

Apare pagina cu instrucțiunile testului:



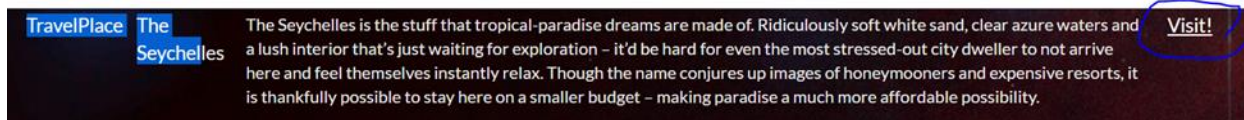
Apoi, utilizatorul trebuie să aleagă câte un raspuns pentru fiecare întrebare afișată:



La sfârșitul setului de întrebări apare **butonul „View results”**, care, în urma apăsării afișează pagina cu tipul de personalitate determinat (predominant) și recomandările portivite acestuia. **Cuvintele „visit!” sunt linkuri** către paginile Wikipedia care descriu locurile sau filmele/serialele recomandate pt. acel tip de personalitate.

Type	Name	Description	Link
Movie	Everything, Everything	Drama, Romance; A 17 year old girl named Madeline Whittier has a rare disease that causes her to have to stay indoors 24/7 with her filtered air. Her whole life is basically books, her mom, and Carla (her nurse). One day, a moving truck pulls in next door. There she sees Olly. Olly Bright is Maddy's new neighbor. They get to know each other through emails. The more they get to know each other, the more they fall in love. Olly starts to make Maddy realize that she isn't really living. This starts the adventures of Maddy's new life.	<a href="#">Visit!</a>
Movie	WALL-E	Animation, Adventure, Family, Sci-Fi; In a distant, but not so unrealistic, future where mankind has abandoned earth because it has become covered with trash from products sold by the powerful multi-national Buy N Large corporation, WALL-E, a garbage collecting robot has been left to clean up the mess. Mesmerized with trinkets of Earth's history and show tunes, WALL-E is alone on Earth except for a sprightly pet cockroach. One day, EVE, a sleek (and dangerous) reconnaissance robot, is sent to Earth to find proof that life is once again sustainable. WALL-E falls in love with EVE.	<a href="#">Visit!</a>
Movie	Passengers	Adventure, Drama, Romance; The spaceship, Starship Avalon, in its 120-year voyage to a distant colony planet known as the "Homestead Colony" and transporting 5,258 people has a malfunction in one of its sleep chambers. As a result one hibernation pod opens prematurely and the one person that awakes, Jim Preston (Chris Pratt) is stranded on the spaceship, still 90 years from his destination.	<a href="#">Visit!</a>
Series	Hawaii 5-0	Action, Crime, Drama; Steve McGarrett comes to Hawaii to avenge his father's death but when the governor offers his own task force, he accepts. He picks up team members on the way, Danny Williams, the head detective on his father's case, Chin Ho Kelly, a former HPD Detective who was fired for accused corruption and McGarrett's father's old patrol partner, Kono Kalakau, a Cadet at the Police Academy who's 1 week from graduating, and Captain Lou Grover, a former Chicago PD Head of SWAT.	<a href="#">Visit!</a>





## CARUSELUL

Toate întrebările au fost așezate într-un carusel, pe fiecare pagină a acestuia fiind afișată câte o întrebare și butoanele de tip „radio” care rețin răspunsul a cărui valoare urmează a fi calculată în vederea obținerii rezultatului.

Pt. carusel am folosit CSS, Bootstrap și Java Script.

```

.carousel-control {
  position: absolute;
  top: 0;
  left: 0;
  bottom: 0;
  width: 15%;
  opacity: 0.5;
  filter: alpha(opacity=50);
  font-size: 20px;
  color: #ffffff;
  text-align: center;
  text-shadow: 0 1px 2px rgba(0, 0, 0, 0.6);
  background-color: rgba(0, 0, 0, 0);
}

.carousel-control.left {
  background-image: -webkit-linear-gradient(left, rgba(0, 0, 0, 0.5) 0%, rgba(0, 0, 0, 0.0001) 100%);
  background-image: -o-linear-gradient(left, rgba(0, 0, 0, 0.5) 0%, rgba(0, 0, 0, 0.0001) 100%);
  background-image: -webkit-gradient(linear, left top, right top, from(rgba(0, 0, 0, 0.5)), to(rgba(0, 0, 0, 0.0001)));
  background-image: linear-gradient(to right, rgba(0, 0, 0, 0.5) 0%, rgba(0, 0, 0, 0.0001) 100%);
  background-repeat: repeat-x;
  filter: progid:DXImageTransform.Microsoft.gradient(startColorstr='#80000000', endColorstr='#00000000', GradientType=1);
}

52 .carousel {
53   width: 1140px;
54   height: 475px;
55   background: linear-gradient(to bottom, #99ccff 0%, #993399 100%);
56 }
57
58 .carousel{
59   padding-top: 190px;
60   padding-bottom: 50px;
61 }
62

```

**-CSS/BOOTSTRAP**

```

var old = $.fn.carousel

$.fn.carousel = function (option) {
  return this.each(function () {
    var $this = $(this)
    var data = $this.data('bs.carousel')
    var options = $.extend({}, Carousel.DEFAULTS, $this.data(), typeof option == 'object' && option)
    var action = typeof option == 'string' ? option : options.slide

    if (!data) $this.data('bs.carousel', (data = new Carousel(this, options)))
    if (typeof option == 'number') data.to(option)
    else if (action) data[action]()
    else if (options.interval) data.pause().cycle()
  })
}

$.fn.carousel.Constructor = Carousel

;(function (document) {
  $(document).on('click.bs.carousel.data-api', '[data-slide], [data-slide-to]', function (e) {
    var $this = $(this), href
    var $target = $($this.attr('data-target') || (href = $this.attr('href')) && href.replace(/.*(?=#[^\s]+$)/, '')) //strip for ie7
    var options = $.extend({}, $target.data(), $this.data())
    var slideIndex = $this.attr('data-slide-to')
    if (slideIndex) options.interval = false

    $target.carousel(options)

    if (slideIndex = $this.attr('data-slide-to')) {
      $target.data('bs.carousel').to(slideIndex)
    }

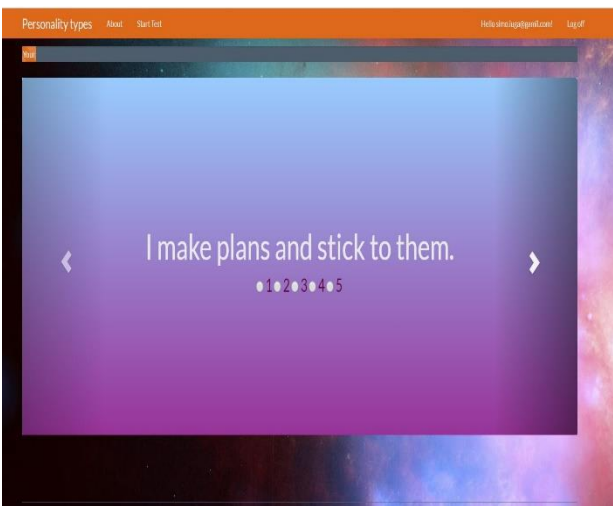
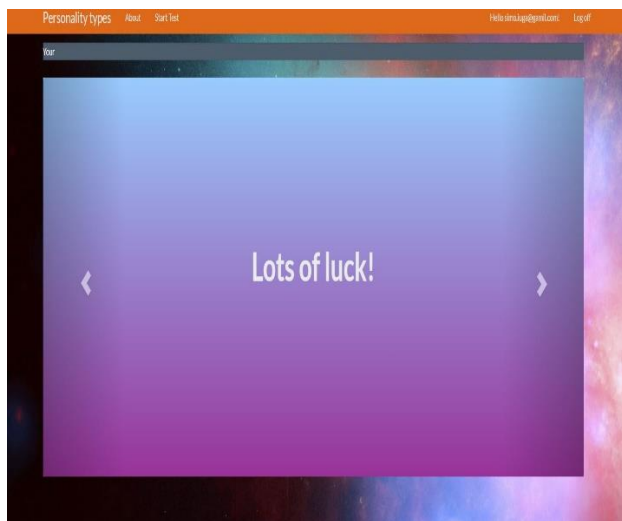
    e.preventDefault()
  })

  $(window).on('load', function () {
    $('[data-ride="carousel"]').each(function () {
      var $carousel = $(this)
      $carousel.carousel($carousel.data())
    })
  })
})(document)

```

Această funcție  
în JavaScript  
construiește  
carouselul.

această funcție în  
JavaScript actualizează  
fiecare pagină a  
carouselului





## RADIO BUTTONS ȘI CALCULUL REZULTATULUI FINAL

Butoanele de pe fiecare pagină a caruselului sunt, de fapt, **radio buttons**, care înregistrează fiecare răspuns pt. a-l contoriza la final, în vederea obținerii rezultatului (tipul de personalitate predominant).

```

</div>
@{ int i = 1;}
@foreach (var item in Model)
{
    <div class="item" align="center">
        <font size="35">
            @Html.DisplayFor(modelItem => item.Text)
        </font>

        <form action="" id="checked_radio_form_@i">
            <font size="5" color="#660033">
                <input type="radio" name="number" value="1"> 1
                <input type="radio" name="number" value="2"> 2
                <input type="radio" name="number" value="3"> 3
                <input type="radio" name="number" value="4"> 4
                <input type="radio" name="number" value="5"> 5
            </font>
        </form>
    </div>
    i++;
}
...

```

Utilizând rezultatele date de radio buttons în controllerul „**RecomandationsController**”, cu ajutorul unei funcții calculăm care este tipul predominant de personalitate astfel:

În vectorul **personalitiesCount** fiecărui indice îi corespunde câte un tip de personalitate. Inițializăm vektorul cu 0. Apoi în funcție de tipul întrebării și de personalitatea corespunzătoare (conform relației pe baza ID-ului din baza de date) **questions[i].PersonalityId** , adunăm la personalitatea respectivă valoarea butonului radio (acestea sunt de tip string și sunt salvate în vectorul values, iar pt. a le aduna la vectorul

**personalitiesCount** se convertesc la int prin intermediul funcției **Convert.ToInt32** )

```
public JsonResult GetPredominantPersonality(List<String> values)
{
    List<int> personalitiesCount = new List<int>();
    for (int i = 0; i<4; i++)
    {
        personalitiesCount.Add(0);
    }

    var questions = db.Questions.ToList();
    for (int i = 0; i<questions.Count - 1; i++)
    {
        if (questions[i].PersonalityId == 8)
        {
            try { personalitiesCount[0] += Convert.ToInt32(values[i]); }
            catch { }
        }

        if (questions[i].PersonalityId == 2)
        {
            try { personalitiesCount[1] += Convert.ToInt32(values[i]); }
            catch { }
        }

        if (questions[i].PersonalityId == 3)
        {
            try { personalitiesCount[2] += Convert.ToInt32(values[i]); }
            catch { }
        }

        if (questions[i].PersonalityId == 4)
```

Apoi în variabila **indexMax** calculăm valoarea indicelui corespunzător personalității cu scor maxim.

```
int indexMax
= !personalitiesCount.Any() ? -1 :
personalitiesCount
.Select((value, index) => new { Value = value, Index = index })
.Aggregate((a, b) => (a.Value > b.Value) ? a : b)
.Index;
```

În funcție de rezultatul returnat de funcție, încărcăm recomandările pt. acel tip de personalitate cu ajutorul funcției următoare care preia informațiile din baza de date(**db.Recomandations.Where, db.Personalities.Where**):

```
return Json(new
{
    redirectUrl = Url.Action("RecomandationsBasedOnPersonalityType", "Recomandations", new { personalityId = String.Format(predominantPersonalityId.ToString()) },
    isRedirect = true
});
}

public ActionResult RecomendationsBasedOnPersonalityType(string personalityId)
{
    var personalityIdInt = Convert.ToInt32(personalityId);
    var recomandations = db.Recomandations.Where(r => r.PersonalityId == personalityIdInt).ToList();

    var personalityName = db.Personalities.Where(p => p.Id == personalityIdInt).FirstOrDefault().Name;
    ViewBag.PredominantPersonalityType = personalityName;
    return View(recomandations);
}
```

## PROGRESS BAR-UL

Pe parcursul testului utilizatorul poate observa **progress-barul** ,din partea de sus a paginii web. Aceasta se umple pe măsură ce utilizatorul răspunde la câte o întrebare.



Funcția care determină colorarea progress\_barului este în **JavaScript**. În variabila **progressBarStatus** dacă se apasă în carusel săgeata din stânga, adică **butonul „left”**, iar **progressBarStatus** este mai mare decât 0 (s-a parcurs cel puțin o întrebare) atunci **progressBarStatus** scade cu 2.5%.

```

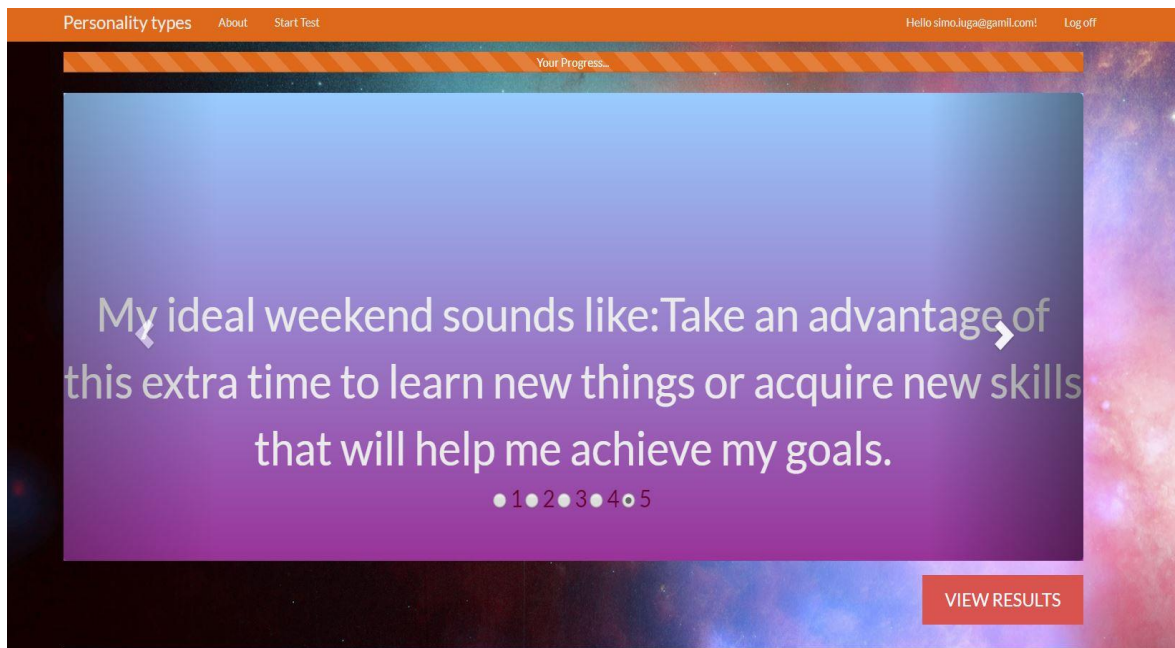
<code>
$( ".left" ).click(function () {
    if (progressBarStatus > 0) {
        progressBarStatus -= 2.5;
    }
    $( ".progress-bar" ).css({ width: progressBarStatus + '%' });

    if (progressBarStatus == 0) {
        return false;
    }
})
</code>

```







Dacă se apasă **butonul „right”** din carusel, atunci ,**progressBarStatus** dacă este diferit de 100, crește cu 2.5%. Dacă ajunge la 100 atunci se face vizibil **butonul Go\_to\_recommendations (View results)**.

```
$( ".right" ).click(function () {  
    if (progressBarStatus != 100) {  
        progressBarStatus += 2.5;  
        //progressBarStatus += 50;  
    }  
    $( ".progress-bar" ).css({ width: progressBarStatus + '%' });  
  
    if (progressBarStatus == 100) {  
        $( ".GoToRecomandations" ).css("visibility", "visible");  
  
        var tempList = new Array();  
        for (var i = 1; i <= 40; i++) {  
            tempList.push( $( 'input[name=number]:checked', '#checked_radio_form_' + i ).val() );  
        }  
  
        allElements = tempList;  
  
        return false;  
    }  
})
```



## BIBLIOGRAFIE

Informațiile despre personalitate au fost preluate și traduse din următoarele surse:

-  <https://en.wikipedia.org/wiki/Wilhelm/M. Wundt>
-  <http://www.consultanta-psihologica.com/istoricul-testelor-psihologice/>
-  <https://ro.wikipedia.org/wiki/Psihologie>
-  <https://suntfericita.manager.ro/temperamentul-caracterul-si-cele-4-tipuri-de-personalitate-sangvinic-flegmatic-coleric-si-melancolic-1412.htm>
-  [Deepak Chopra- „Supercreierul”](#)
-  [Deepak Chopra- ”Cele șapte legi spirituale ale succesului”](#)

*„Cine nu își oferă posibilitatea de a experimenta din când în când tăcerea nu face altceva decât să își stimuleze la infinit dialogul interior mental.”*

*(Deepak Chopra)*