Savjeti za rješavanje ASP.NET web forms dijela projektnog zadatka

1. Kako naptraviti CAPTCHA zaštitu na formi?

Ima određeni broj Nugeta s CAPTCHA zaštitom, a jedan od njih je i re-CAPTCHA. Nuget je ovdje: https://github.com/tanveery/recaptcha-net Upute su ovdje: https://www.nuget.org/packages/RecaptchaNet/

> Na reCAPTCHA portalu odaberite "v3 Admin Console". Kada registrirate site, za reCAPTCHA type odaberite "reCAPTCHA v2", "I'm not a robot"

Primjer kontrolera:

```
public class ApartmentController : Controller
{
    // GET: Apartment
    public ActionResult Index()
    {
        return View();
    }
    [HttpPost]
    public ActionResult Index(ContactReservationModel model)
    {
        var recaptchaHelper = this.GetRecaptchaVerificationHelper();
        if (String.IsNullOrEmpty(recaptchaHelper.Response))
        {
            ModelState.AddModelError(
                "",
                "Captcha answer cannot be empty.");
            return View(model);
        }

        var recaptchaResult = recaptchaHelper.VerifyRecaptchaResponse();
        if (!recaptchaResult.Success)
        {
            ModelState.AddModelError(
                "",
                "Incorrect captcha answer.");
            return View(model);
        }
        if (ModelState.IsValid)
        {
            return RedirectToAction("Index", "Home");
        }
        return View();
```

```
    }
}
```

Primjer viewa:

```
@model Javno.Models.ContactReservationModel
@using Recaptcha.Web.Mvc

<h2>Kontaktirajte nas za rezervaciju</h2>

@using (Html.BeginForm("Index", "Apartment", "POST"))
{
    <div class="form-group">
        @Html.LabelFor(x => x.FirstName)
        @Html.TextBoxFor(x => x.FirstName, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(x => x.LastName)
        @Html.TextBoxFor(x => x.LastName, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(x => x.Email)
        @Html.TextBoxFor(x => x.Email, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(x => x.PhoneMobile)
        @Html.TextBoxFor(x => x.PhoneMobile, new { @class = "form-control" })
    </div>
    <div class="form-group">
        @Html.LabelFor(x => x.NumberOfAdults)
        @Html.TextBoxFor(x => x.NumberOfAdults,
          new { @class = "form-control", @type = "number" })
    </div>
    <div class="form-group">
        @Html.LabelFor(x => x.NumberOfChildren)
        @Html.TextBoxFor(x => x.NumberOfChildren,
          new { @class = "form-control", @type = "number" })
    </div>
    <div class="form-group">
        <button type="submit" class="btn btn-primary">Submit</button>
    </div>
    @Html.RecaptchaWidget()
    if (!ViewData.ModelState.IsValid)
    {
    <div class="alert alert-danger">
        <span class="glyphicon glyphicon-exclamation-sign" aria-hidden="true"></span>
        @Html.ValidationSummary()
```

```
    </div>
    }
}
```

2. Kako dodati ocjene u obliku zvjezdica?

Jedan način jest downloadati Nuget ili JS lib za to. Drugi način je napisati sam skriptu koja će prikazati zvjezdice.

Rješenje za drugi način je ovdje opisano.

2.1. Model

```
public class StarRatingModel
{
    public string Question { get; set; }
    public int Rating { get; set; }
    public int MaxRating { get; set; } = 5;
}
```

2.2. Akcija

```
public ActionResult StarRating()
{
    var model =
        new StarRatingModel {
            Question = "Ocijenite apartman",
            MaxRating = 10
        };
    return View(model);
}
```

2.3. View

```
@model Javno.Models.StarRatingModel

@{
    ViewBag.Title = "StarRating";
}

<div class="form-group" id="rating-wrapper">
    <div>
        <div>@Model.Question</div>
        @Html.HiddenFor(x => x.Rating)
        <span>
```

```
            <span class="display-rating-number">@Model.Rating</span>
            <small> / @Model.MaxRating</small>
        </span>
    </div>
    @for (int i = 1; i <= Model.MaxRating; i++)
    {
        <button type="button" class="btn btn-default" data-stars="@i">
            <span class="fa fa-star"></span>
        </button>
    }
</div>

@section Scripts {
    <link
      rel="stylesheet"
      href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.1.1/css/all.min.css"
      integrity="sha512-KfkfwYDsLkIlwQp6LFnl8zNdLGxu9YAA1QvwINks4PhcElQSvqcyVLLD9aMhXd13uQj
      crossorigin="anonymous"
      referrerpolicy="no-referrer" />
    <script>
        $(function () {
            $("#rating-wrapper button").on('click', (function (e) {
                // Get data-stars value from the button
                var selectedValue = $(this).data("stars");

                // Set value to hidden and display
                $("#Rating").val(selectedValue);
                $(".display-rating-number").text(selectedValue);

                // Set star style
                $("#rating-wrapper").find(`button[data-stars]`).each(function () {
                    $this = $(this);
                    if ($this.data("stars") <= selectedValue) {
                        $this.removeClass('btn-default');
                        $this.addClass('btn-warning');
                    } else {
                        $this.removeClass('btn-warning');
                        $this.addClass('btn-default');
                    }
                })
            }));
        });
    </script>
}
```

3. Koncept iza prikaza liste apartmana i pojedinačnih apartmana

3.1. Lista apartmana

Za referencu vidi sliku u projektnom zadatku.

3.1.1. *SearchModel - glavni model pretraživanja*

Jednostavni filteri

- int FilterRooms
- int FilterAdults
- int FilterChildren

DropDown filteri

- int FilterCity
- List CityList - lista gradova koji se mogu napuniti u viewu koristeći npr.

  ```
  @Html.DropDownListFor(x => x.FilterCity,
      new SelectList(Model.CityList, "Id", "Name"))
  ```

- int Order
- List OrderList - lista smjerova sortiranja koji se mogu napuniti u viewu koristeći npr.

  ```
  @Html.DropDownListFor(x => x.Order,
      new SelectList(Model.OrderList, "Id", "Name"))
  ```

Lista pronađenih rezultata

- List SearchResult

3.1.2. *SearchResultModel - detalj pojedinog rezultata*

Prikazani atributi

- int Id
- string Name
- int? StarRating - prikazan kao zvjezdice
- string CityName
- int? BeachDistance
- int? TotalRooms
- int? MaxAdults
- int? MaxChildren
- decimal Price
- string RepresentativePicturePath - prikazan kao slika

3.1.3. *Search view - prikaz forme za pretraživanje*

Prvi dio viewa (filteri) možemo prikazati kao elemente forme, jer će biti ažurirani i poslani na server kod klika gumba.

Skica

```
// Napomena: ovdje se može koristiti varijanta s GET i POST,
// ovisi kako se izvede kontroler
@using (Html.BeginForm("Search", "Apartment", "GET"))
{
    <div class="form-group">
        @Html.LabelFor(x => x.FilterRooms)
        @Html.TextBoxFor(x => x.FilterRooms,
          new { @class = "form-control", @type = "number" })
    </div>
    ...
    <div class="form-group">
        <button type="submit" class="btn btn-primary">Go!</button>
    </div>
}
```

> Napomena: GET metoda je ovdje korištena za formu radi toga jer nema puno smisla slati POST zahtjev kod pretraživanja. Za POST se očekuje da mijenja stanje na serveru, a za GET da to ne radi. U slučaju prvog pretraživanja šalje se GET bez parametara. U slučaju naknadnih pretraživanja šalje se GET s parametrima. Tehnički, to znači da trebamo koristiti jednu te istu akciju kontrolera za prvo i svako sljedeće pretraživanje.

Drugi dio filtera (rezultati pretraživanja) možemo prikazati i izvan forme jer te podatke ne treba slati na server.

Skica

```
@for (int i = 0; i < Model.SearchResult.Count; i++)
{
    <div>@Model.SearchResult[i].Name</div>
    <div>@Model.SearchResult[i].StarRating</div>
    <div><img
      src="/Apartment/Picture?Path=@Model.SearchResult[i].RepresentativePicturePath" /></div
    <ul>
        <li>@Model.SearchResult[i].CityName</li>
        <li>@Model.SearchResult[i].BeachDistance</li>
        <li>@Model.SearchResult[i].TotalRooms</li>
        <li>@Model.SearchResult[i].MaxAdults</li>
        <li>@Model.SearchResult[i].MaxChildren</li>
    </ul>
    <div>@Model.SearchResult[i].Price</div>
```

```
    @Html.ActionLink("Više...", "Details",
        new { id = Model.SearchResult[i].Id },
        new { @class = "btn btn-primary" })
}
```

Primijetite oblik putanje na kojoj se očekuje slika: `/Apartment/Picture?Path=...`

3.1.4. *Apartment kontroler, Search akcija*

Rješenje 1 ("klasičan setup", koristi se često u web formama)

- `GET Search()` - poziva se na prvi load, bez parametara
- `POST Search(SearchModel model)` - poziva se na submit, vrijednosti forme završe kao parametri u modelu

Skica

```
public ActionResult Search()
{
    var model = new SearchModel();
    model.CityList = _cityRepository.GetCities();
    model.OrderList = _orderRepository.GetOrders();
    return View(model);
}

[HttpPost]
public ActionResult Search(SearchModel model)
{
    model.SearchResult =
      _apartmentRepository.Search(
          model.FilterRooms,
          model.FilterAdults,
          model.FilterChildren,
          model.FilterCity,
          model.Order);
    model.CityList = _cityRepository.GetCities();
    model.OrderList = _orderRepository.GetOrders();

    return View(model);
}
```

Rješenje 2 (čisti GET)

Želimo li pojednostavniti održavanje i izbjeći POST metodu tamo gdje je nepotrebna jer ne mijenja stanje resursa na serveru, možemo napraviti sljedeće:

- `GET Search(SearchModel model)` - poziva se i prvi puta i na submit svaki sljedeći puta

Skica

```
public ActionResult Search(SearchModel model)
{
    if (model == null)
    {
        model = new SearchModel();
    }
    else
    {
        model.SearchResult =
          _apartmentRepository.Search(
              model.FilterRooms,
              model.FilterAdults,
              model.FilterChildren,
              model.FilterCity,
              model.Order);
    }

    model.CityList = _cityRepository.GetCities();
    model.OrderList = _orderRepository.GetOrders();

    return View(model);
}
```

3.1.5. *Korišteni repozitoriji*

Primijetite da rješenja kontrolera podrazumijevaju postojanje repozitorija koji je u stanju dohvatiti filtrirane i sortirane podatke.

Npr.:

```
public List<SearchResultModel> Search(
    int? rooms,
    int? adults,
    int? children,
    int? destination,
    int? order)
{
    var commandParameters = new List<SqlParameter>();

    ...

    var ds = SqlHelper.ExecuteDataset(
        _connectionString,
        CommandType.StoredProcedure,
        "dbo.SearchApartments",
        commandParameters.ToArray());
```

```
    var resList = new List<SearchResultModel>();
    foreach (DataRow row in ds.Tables[0].Rows)
    {
        var ap = new Javno.Models.SearchResultModel();

        ...

        resList.Add(ap);
    }

    return resList;
}
```

3.2. Pojedini apartman

Za referencu vidi sliku u projektnom zadatku.

3.2.1. *ApartmentDetails - glavni model detaljnog prikaza*

Identifikator

- int Id

Atributi prikazani s lijeve strane, gore

- string Name
- ApartmentPicture RepresentativePicture
- string CityName
- int? BeachDistance
- int? TotalRooms
- int? MaxAdults
- int? MaxChildren
- string OwnerName
- decimal Price

Atributi prikazani s lijeve strane, po sredini

- List Tags

Atributi prikazani s desne strane

- int? StarRating
- string ContactFirstName
- string ContactLastName
- string ContactEmail
- string ContactPhoneMobile
- int ContactNumberOfAdults
- int ContactNumberOfChildren
- DateTime ContactDateFrom
- DateTime ContactDateTo

Prikazano dolje

- List Pictures

3.2.2. *ApartmentPicture - model za prikaz slike*

- int Id
- string Path
- string Name
- bool IsRepresentative

3.2.3. *Details view - prikaz pojedinog apartmana*

Prikaz se sastoji od nekoliko dijelova. Stranicu je potrebno formatirati tako da ti dijelovi budu prikazani tamo gdje je potrebno.

Skica kako je npr. moguće napraviti prikaz slika

```
<div class="container-fluid">
    <div class="row">
        @for (int i = 0; i < Model.Pictures.Count; i++)
        {
            <div class="col-md-3">
                <div class="panel panel-default">
                    <div class="panel-heading">
                      <h2>@Model.Pictures[i].Name</h2></div>
                    <div class="panel-body"><img
                      src="/Apartment/Picture?Path=@Model.Pictures[i].Path" Width="100%"></d
                    <div class="panel-footer">...detalji...</div>
                </div>
            </div>
        }
    </div>
</div>
```

Primijetite oblik putanje na kojoj se očekuje slika: `/Apartment/Picture?Path=...`

3.2.4. *Apartment kontroler, Search akcija*

Jedan od problema koje ovaj kontroler treba riješiti jest dohvaćanje retka koji treba pokazati, te vezanih listi (tagovi, slike). Ako dohvatimo sve te podatke odjednom, biti će vjerojatno grupirani na način na koji ne želimo. Postoji i rješenje kojim možemo pozivom jedne procedure dohvatiti više tablica odjednom (Multiple Active Resultsets - MARS).

Ovdje ćemo problem riješiti tako da pozovemo tri procedure za redom da bismo dohvatili navedene podatke:

- dohvat retka apartmana

- dohvat tagova
- dohvat slika

Skica

```
public ActionResult Details(int id)
{
    var model = _apartmentRepository.GetPublicApartment(id);
    model.Tags = _apartmentRepository.GetPublicApartmentTags(id);
    model.Pictures = _apartmentRepository.GetPublicApartmentPictures(id);
    return View(model);
}
```

3.2.5. *Korišteni repozitoriji*

Primijetite da rješenje u kontroleru podrazumijeva postojanje repozitorija koji je u stanju dohvatiti tri seta podataka za pojedini apartman: redak, tagovi, slike.

3.3. *Dohvat slika*

Search view i details view podrazumijevaju da se slika nalazi na određenom endpointu: `/Apartment/Picture?Path=...`.

Najjednostavnija implementacija takvog dohvata slike jest:

```
public ActionResult Picture(string path)
{
    var picturePath = Path.Combine("C:\neka\mapa\sa\slikama", path);
    string mimeType = MimeMapping.GetMimeMapping(path);
    return new FilePathResult(picturePath, mimeType);
}
```

Skica u našem slučaju

```
public ActionResult Picture(string path)
{
  if (path == null || string.IsNullOrEmpty(path))
      return Content(content: "File missing"); // Rješenje "nabrzaka", nije najbolje

  // Popravi putanju do slike, u bazi nije cijela putanja!
  var javnoRoot = Server.MapPath("~");
  var adminRoot = Path.Combine(javnoRoot, "../Admin/Content/Pictures");
  var picturePath = Path.Combine(adminRoot, path);

  string mimeType = MimeMapping.GetMimeMapping(picturePath);

  return new FilePathResult(picturePath, mimeType);
}
```