Controller

## HomeController.cshtml

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.Mvc;

using QuizApp.Models;

using System.IO;

namespace QuizApp.Controllers

{

public class HomeController : Controller

{

QuizAPPEntities1 db = new QuizAPPEntities1();

[HttpGet]

public ActionResult Admin\_Signup()

{

return View();

}

[HttpPost]

public ActionResult Admin\_Signup(Admin\_tb snup)

{

if (ModelState.IsValid == true)

{

db.Admin\_tb.Add(snup);

int a = db.SaveChanges();

if (a > 0)

{

ViewBag.InsertMessage = "<script>swal('Congradulations', 'Registered Successfully','success')</script>";

ModelState.Clear();

}

else

{

ViewBag.InsertMessage = "<script>alert('Registration Failed!!')</script>";

}

}

return View();

}

[HttpGet]

public ActionResult Admin\_login()

{

return View();

}

[HttpPost]

public ActionResult Admin\_login(Admin\_tb ad)

{

Admin\_tb admin = db.Admin\_tb.Where(x => x.ADMIN\_NAME == ad.ADMIN\_NAME && x.ADMIN\_PASSWORD == ad.ADMIN\_PASSWORD).SingleOrDefault();

if (admin != null)

{

Session["Admin\_id"] = admin.ADMIN\_ID.ToString();

Session["AdminName"] = admin.ADMIN\_NAME;

TempData.Keep();

return RedirectToAction("Dashboard");

}

else

{

ViewBag.msg = "Invalid UserName Or Password!";

}

return View();

}

[HttpGet]

public ActionResult Add\_Category()

{

if (Session["Admin\_id"]==null)

{

return RedirectToAction("Admin\_login");

}

//Session["ad\_id"] = 2;//remove it

int Admin\_id = Convert.ToInt32(Session["Admin\_id"]);

List<CATEGORY> cat = db.CATEGORies.Where(x=>x.CAT\_ADMIN\_ID == Admin\_id).OrderBy(x => x.CAT\_ID).ToList();

ViewData["list"] = cat;

return View();

}

[HttpPost]

public ActionResult Add\_Category(CATEGORY cat)

{

//----

List<CATEGORY> cat\_Li = db.CATEGORies.OrderBy(x => x.CAT\_ID).ToList();

ViewData["list"] = cat\_Li;

CATEGORY c = new CATEGORY();

Random r = new Random();

c.CAT\_NAME = cat.CAT\_NAME;

c.CAT\_ADMIN\_ID = Convert.ToInt32(Session["Admin\_id"].ToString());

c.cat\_encrypted\_string = Crypt.Encrypt(cat.CAT\_NAME.Trim() + r.Next().ToString(), true);

db.CATEGORies.Add(c);

Session["catid"] = c.CAT\_ID.ToString();

db.SaveChanges();

return RedirectToAction("Add\_Category");

}

[HttpGet]

public ActionResult AddQuestions()

{

if (Session["Admin\_id"] == null)

{

return RedirectToAction("Admin\_login");

}

int sid = Convert.ToInt32(Session["Admin\_id"]);

List<CATEGORY> cat\_li = db.CATEGORies.Where(x=>x.CAT\_ADMIN\_ID== sid).ToList();

ViewBag.list = new SelectList(cat\_li, "CAT\_ID", "CAT\_NAME");

List<QUESTION> que = db.QUESTIONS.ToList();

ViewData["que"] = que;

return View();

}

[HttpPost]

public ActionResult AddQuestions(QUESTION q)

{

int aid = Convert.ToInt32(Session["Admin\_id"]);

TempData["QUE\_ID"] = q.QUE\_ID.ToString();

TempData.Keep();

List<CATEGORY> cat\_li = db.CATEGORies.Where(x => x.CAT\_ADMIN\_ID == aid).ToList();

ViewBag.list = new SelectList(cat\_li, "CAT\_ID", "CAT\_NAME");

QUESTION qa = new QUESTION();

qa.QUE\_TEXT = q.QUE\_TEXT;

qa.OPT\_A = q.OPT\_A;

qa.OPT\_B = q.OPT\_B;

qa.OPT\_C= q.OPT\_C;

qa.OPT\_D = q.OPT\_D;

qa.CORRECT\_OPT = q.CORRECT\_OPT;

qa.QUE\_CAT\_ID = q.QUE\_CAT\_ID;

db.QUESTIONS.Add(qa);

db.SaveChanges();

ViewBag.msg = "Question Added Successfully";

ModelState.Clear();

return View();

}

[HttpGet]

public ActionResult StudentSignup()

{

return View();

}

[HttpPost]

public ActionResult StudentSignup(STUDENT\_TBL st)

{

string filename = Path.GetFileNameWithoutExtension(st.ImageFile.FileName);

string extension = Path.GetExtension(st.ImageFile.FileName);

HttpPostedFileBase postedFile = st.ImageFile;

int length = postedFile.ContentLength;

if (extension.ToLower() == ".jpg" || extension.ToLower() == ".jpeg" || extension.ToLower() == ".png")

{

if (length <= 1000000)

{

filename = filename + extension;

st.STUDENT\_IMAGE = "~/Content/Studentimg/" + filename;

filename = Path.Combine(Server.MapPath("~/Content/Studentimg/"),filename);

st.ImageFile.SaveAs(filename);

db.STUDENT\_TBL.Add(st);

int a= db.SaveChanges();

if (a>0)

{

TempData["CreateMsg"] = "<script>alert('Registered Successfully.');</script>";

ModelState.Clear();

return RedirectToAction("Student\_login","Home");

}

else

{

TempData["CreateMsg"] = "<script>alert('Registration Failed...');</script>";

}

}

else

{

TempData["SizeMsg"] = "<script>alert('Image Size should be less than 1 MB');</script>";

}

}

else

{

TempData["ExtensionMsg"] = "<script>alert('Format Not Supported');</script>";

}

return View();

}

public ActionResult Student\_login()

{

return View();

}

[HttpPost]

public ActionResult Student\_login(STUDENT\_TBL st)

{

STUDENT\_TBL student = db.STUDENT\_TBL.Where(x => x.STUDENT\_NAME == st.STUDENT\_NAME && x.STUDENT\_PASSWORD == st.STUDENT\_PASSWORD).SingleOrDefault();

if (student != null)

{

Session["studentId"] = student.STUDENT\_ID.ToString();

return RedirectToAction("ExamDashboard");

}

else

{

ViewBag.msg = "<script>alert('Invalid Name and Password..');</script>";

}

return View();

}

public ActionResult Dashboard()

{

if (Session["Admin\_id"] == null)

{

return RedirectToAction("Admin\_login");

}

return View();

}

public ActionResult ExamDashboard()

{

if (Session["studentId"] == null)

{

return RedirectToAction("Student\_login");

}

return View();

}

[HttpPost]

public ActionResult ExamDashboard(string room)

{

List<CATEGORY> list = db.CATEGORies.ToList();

foreach (var item in list)

{

if (item.cat\_encrypted\_string==room)

{

List<QUESTION> li = db.QUESTIONS.Where(x => x.QUE\_CAT\_ID == item.CAT\_ID).ToList();

Queue<QUESTION> queue = new Queue<QUESTION>();

foreach (QUESTION a in li)

{

queue.Enqueue(a);

}

TempData["questions"]=queue;

TempData["Score"] = 0;

//TempData["examid"] = item.CAT\_ID;

TempData.Keep();

return RedirectToAction("StartQuiz");

}

else

{

ViewBag.error = "<script>alert('No Room Found. Please Enter Correct Room Name....');</script>";

}

}

return View();

}

public ActionResult StartQuiz()

{

if (Session["studentId"]==null)

{

return RedirectToAction("Student\_login");

}

QUESTION q = null;

if (TempData["questions"]!=null)

{

Queue<QUESTION> qlist = (Queue<QUESTION>)TempData["questions"];

if (qlist.Count>0)

{

q = qlist.Peek();

qlist.Dequeue();

TempData["questions"] = qlist;

TempData.Keep();

}

else

{

return RedirectToAction("EndExam");

}

}

else

{

return RedirectToAction("ExamDashboard");

}

return View(q);

}

[HttpPost]

public ActionResult StartQuiz(QUESTION q)

{

string correctans = null;

if (q.OPT\_A!=null)

{

correctans = "A";

}

else if(q.OPT\_B!=null)

{

correctans = "B";

}

else if (q.OPT\_C != null)

{

correctans = "C";

}

else if (q.OPT\_D != null)

{

correctans = "D";

}

if (correctans.Equals(q.CORRECT\_OPT))

{

TempData["Score"] = Convert.ToInt32(TempData["Score"]) + 1;

}

TempData.Keep();

return RedirectToAction("StartQuiz");

}

public ActionResult EndExam()

{

return View();

}

public ActionResult AdminLogout()

{

Session.Abandon();

return RedirectToAction("Index", "Home");

}

public ActionResult Index()

{

return View();

}

public ActionResult About()

{

ViewBag.Message = "Your application description page.";

return View();

}

public ActionResult Contact()

{

ViewBag.Message = "Your contact page.";

return View();

}

}

}

Views

## Admit\_Signup.cshtml

@model QuizApp.Models.Admin\_tb

@{

ViewBag.Title = "Admin\_Signup";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<script src="https://unpkg.com/sweetalert/dist/sweetalert.min.js"></script>

<h2>Admin Sign Up</h2>

@Html.Raw(ViewBag.InsertMessage)

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.ADMIN\_NAME, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ADMIN\_NAME, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ADMIN\_NAME, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.Email, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.Email, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.Email, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ADMIN\_PASSWORD, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ADMIN\_PASSWORD, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ADMIN\_PASSWORD, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ConfirmPassword, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ConfirmPassword, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ConfirmPassword, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Sign Up" class="btn btn-success" />

</div>

</div>

</div>

}

<div>

<a href="@Url.Action("Admin\_login","Home")" class="btn btn-primary">If you have Registered already Please LOGIN</a>

</div>

# Admin\_Login.cshtml

@model QuizApp.Models.Admin\_tb

@{

ViewBag.Title = "Admin\_login";

}

<script src="~/Scripts/jquery-3.4.1.min.js"></script>

<script src="~/Scripts/jquery.validate.min.js"></script>

<script>

$(document).ready(function () {

$("#left-div").animate({ width: '30%' }, 900);

$("#teacherimg").animate({ height: '450', width: '320', }, 900);

$("#login\_card").animate({ width: '70%', height: '450px' }, 900);

});

</script>

<style>

#teacherimg{

margin-top:20px;

}

.invalid-error {

font-family: 'Lucida Sans', 'Lucida Sans Regular', 'Lucida Grande', 'Lucida Sans Unicode', Geneva, Verdana, sans-serif;

}

</style>

<div class="col-lg-6 col-md-6 col-sm-6" id="left-div" style="width:50%; float:left;">

<img src="~/Content/img/Teacher.png" id="teacherimg" height="500" width="400" alt="Teacher image" />

</div>

<div class="card text-info mb-3 col-lg-6 col-md-6 col-sm-6" id="login\_card" style=" width: 5px; height:5px; float: right; background-color: #fffffc; margin-top: 3%; border: 1px solid #fffffc; border-radius: 10px; box-shadow: 4px 4px 15px -2px rgba(0,0,0,0.78) ";>

<div class="card-body">

<div>

<center>

<p class="invalid-error" style="color:red; margin-left:5px;">@ViewBag.msg</p>

</center>

<center>

<div>

<img src="~/Content/img/logincard.png" alt="login image" width="200" height="200" />

</div>

</center>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<center> <h4> Please Login your Account</h4></center>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.ADMIN\_NAME, htmlAttributes: new { @class = "control-label col-md-4" })

<div class="col-md-8">

@Html.EditorFor(model => model.ADMIN\_NAME, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ADMIN\_NAME, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ADMIN\_PASSWORD, htmlAttributes: new { @class = "control-label col-md-4" })

<div class="col-md-8">

@Html.EditorFor(model => model.ADMIN\_PASSWORD, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ADMIN\_PASSWORD, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-4 col-md-8">

<input type="submit" value="Login" class="btn btn-info" />

</div>

</div>

</div>

}

</div>

<br />

<a href="@Url.Action("Admin\_Signup","Home")" class="btn btn-primary" style="margin-top:2px;">Go to Registration</a>

</div>

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

# StudentSignup.cshtml

@model QuizApp.Models.STUDENT\_TBL

@{

ViewBag.Title = "StudentSignup";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>StudentSignup</h2>

@Html.Raw(TempData["CreateMsg"])

@Html.Raw(TempData["SizeMsg"])

@Html.Raw(TempData["ExtensionMsg"])

@using (Html.BeginForm("StudentSignup", "Home", FormMethod.Post, new { enctype = "multipart/form-data" }))

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.STUDENT\_NAME, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.STUDENT\_NAME, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.STUDENT\_NAME, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.StudentEmail, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.StudentEmail, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.StudentEmail, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.STUDENT\_IMAGE, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

<input type="file" name="ImageFile" class="form-control btn btn-primary" required />

@Html.ValidationMessageFor(model => model.STUDENT\_IMAGE, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.STUDENT\_PASSWORD, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.STUDENT\_PASSWORD, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.STUDENT\_PASSWORD, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.ConfirmPassword, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.ConfirmPassword, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.ConfirmPassword, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Signup" class="btn btn-info" />

</div>

</div>

</div>

}

<div>

@Html.ActionLink("If you have already Registered Please LOGIN", "Student\_login")

</div>

# Student\_Login.cshtml

@model QuizApp.Models.STUDENT\_TBL

@{

ViewBag.Title = "Student\_login";

}

<h2 style="color:deepskyblue;">Student Log In</h2>

@Html.Raw(ViewBag.msg);

<div>

@using (Html.BeginForm())

{

@Html.AntiForgeryToken()

<div class="form-horizontal">

<center> <h4> Please Login your Account</h4></center>

<hr />

@Html.ValidationSummary(true, "", new { @class = "text-danger" })

<div class="form-group">

@Html.LabelFor(model => model.STUDENT\_NAME, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.STUDENT\_NAME, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.STUDENT\_NAME, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

@Html.LabelFor(model => model.STUDENT\_PASSWORD, htmlAttributes: new { @class = "control-label col-md-2" })

<div class="col-md-10">

@Html.EditorFor(model => model.STUDENT\_PASSWORD, new { htmlAttributes = new { @class = "form-control" } })

@Html.ValidationMessageFor(model => model.STUDENT\_PASSWORD, "", new { @class = "text-danger" })

</div>

</div>

<div class="form-group">

<div class="col-md-offset-2 col-md-10">

<input type="submit" value="Login" class="btn btn-info" />

</div>

</div>

</div>

}

</div>

@section Scripts {

@Scripts.Render("~/bundles/jqueryval")

}

Models

# Admin Table

//------------------------------------------------------------------------------

// <auto-generated>

// This code was generated from a template.

//

// Manual changes to this file may cause unexpected behavior in your application.

// Manual changes to this file will be overwritten if the code is regenerated.

// </auto-generated>

//------------------------------------------------------------------------------

using System.ComponentModel.DataAnnotations;

namespace QuizApp.Models

{

using System;

using System.Collections.Generic;

public partial class Admin\_tb

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public Admin\_tb()

{

this.CATEGORies = new HashSet<CATEGORY>();

}

public int ADMIN\_ID { get; set; }

[Required(ErrorMessage ="Admin Name is Required")]

[Display(Name ="Admin Name")]

public string ADMIN\_NAME { get; set; }

[Required(ErrorMessage = "Admin Password is Required")]

[Display(Name = "Admin Password")]

[DataType(DataType.Password)]

public string ADMIN\_PASSWORD { get; set; }

[Required(ErrorMessage = "Confirm Password is Required")]

[Display(Name = "Confirm Password")]

[DataType(DataType.Password)]

public string ConfirmPassword { get; set; }

[Required(ErrorMessage = "Admin Email is Required")]

[Display(Name = "Email")]

[DataType(DataType.EmailAddress, ErrorMessage ="Email is not Valid")]

[RegularExpression("^[a-zA-Z0-9\_\\.-]+@([a-zA-Z0-9-]+\\.)+[a-zA-Z]{2,6}$", ErrorMessage = "E-mail is not valid")]

public string Email { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<CATEGORY> CATEGORies { get; set; }

}

}

# Subject Category Table

//------------------------------------------------------------------------------

// <auto-generated>

// This code was generated from a template.

//

// Manual changes to this file may cause unexpected behavior in your application.

// Manual changes to this file will be overwritten if the code is regenerated.

// </auto-generated>

//------------------------------------------------------------------------------

using System.ComponentModel.DataAnnotations;

namespace QuizApp.Models

{

using System;

using System.Collections.Generic;

public partial class CATEGORY

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public CATEGORY()

{

this.QUESTIONS = new HashSet<QUESTION>();

}

public int CAT\_ID { get; set; }

[Required(ErrorMessage = "Admin Name is Required")]

[Display(Name = "Subject")]

public string CAT\_NAME { get; set; }

public Nullable<int> CAT\_ADMIN\_ID { get; set; }

public string cat\_encrypted\_string { get; set; }

public virtual Admin\_tb Admin\_tb { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

public virtual ICollection<QUESTION> QUESTIONS { get; set; }

}

}

# Questions Table

//------------------------------------------------------------------------------

// <auto-generated>

// This code was generated from a template.

//

// Manual changes to this file may cause unexpected behavior in your application.

// Manual changes to this file will be overwritten if the code is regenerated.

// </auto-generated>

//------------------------------------------------------------------------------

using System.ComponentModel.DataAnnotations;

namespace QuizApp.Models

{

using System;

using System.Collections.Generic;

public partial class QUESTION

{

public int QUE\_ID { get; set; }

[Required]

[Display(Name = "Question")]

public string QUE\_TEXT { get; set; }

[Required]

[Display(Name = "A")]

public string OPT\_A { get; set; }

[Required]

[Display(Name = "B")]

public string OPT\_B { get; set; }

[Required]

[Display(Name = "C")]

public string OPT\_C { get; set; }

[Required]

[Display(Name = "D")]

public string OPT\_D { get; set; }

[Required]

[Display(Name = "Correct Option")]

public string CORRECT\_OPT { get; set; }

[Required]

[Display(Name = "Subject Category")]

public Nullable<int> QUE\_CAT\_ID { get; set; }

public virtual CATEGORY CATEGORY { get; set; }

}

}

# Student Table

//------------------------------------------------------------------------------

// <auto-generated>

// This code was generated from a template.

//

// Manual changes to this file may cause unexpected behavior in your application.

// Manual changes to this file will be overwritten if the code is regenerated.

// </auto-generated>

//------------------------------------------------------------------------------

namespace QuizApp.Models

{

using System;

using System.Collections.Generic;

using System.Web;

using System.ComponentModel.DataAnnotations;

public partial class STUDENT\_TBL

{

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2214:DoNotCallOverridableMethodsInConstructors")]

public STUDENT\_TBL()

{

this.SET\_EXAM = new HashSet<SET\_EXAM>();

}

public int STUDENT\_ID { get; set; }

[Required(ErrorMessage = "Student Name is Required")]

[Display(Name = "Student Name")]

public string STUDENT\_NAME { get; set; }

[Required(ErrorMessage = "Student Password is Required")]

[Display(Name = "Student Password")]

[DataType(DataType.Password)]

public string STUDENT\_PASSWORD { get; set; }

[Required(ErrorMessage = "Student image is Required")]

[Display(Name = "Student Image")]

public string STUDENT\_IMAGE { get; set; }

[Required(ErrorMessage = "Student Email is Required")]

[Display(Name = "Email")]

[DataType(DataType.EmailAddress, ErrorMessage = "Email is not Valid")]

[RegularExpression("^[a-zA-Z0-9\_\\.-]+@([a-zA-Z0-9-]+\\.)+[a-zA-Z]{2,6}$", ErrorMessage = "E-mail is not valid")]

public string StudentEmail { get; set; }

[Required(ErrorMessage = "Confirm Password is Required")]

[Display(Name = "Confirm Password")]

[DataType(DataType.Password)]

public string ConfirmPassword { get; set; }

public HttpPostedFileBase ImageFile { get; set; }

[System.Diagnostics.CodeAnalysis.SuppressMessage("Microsoft.Usage", "CA2227:CollectionPropertiesShouldBeReadOnly")]

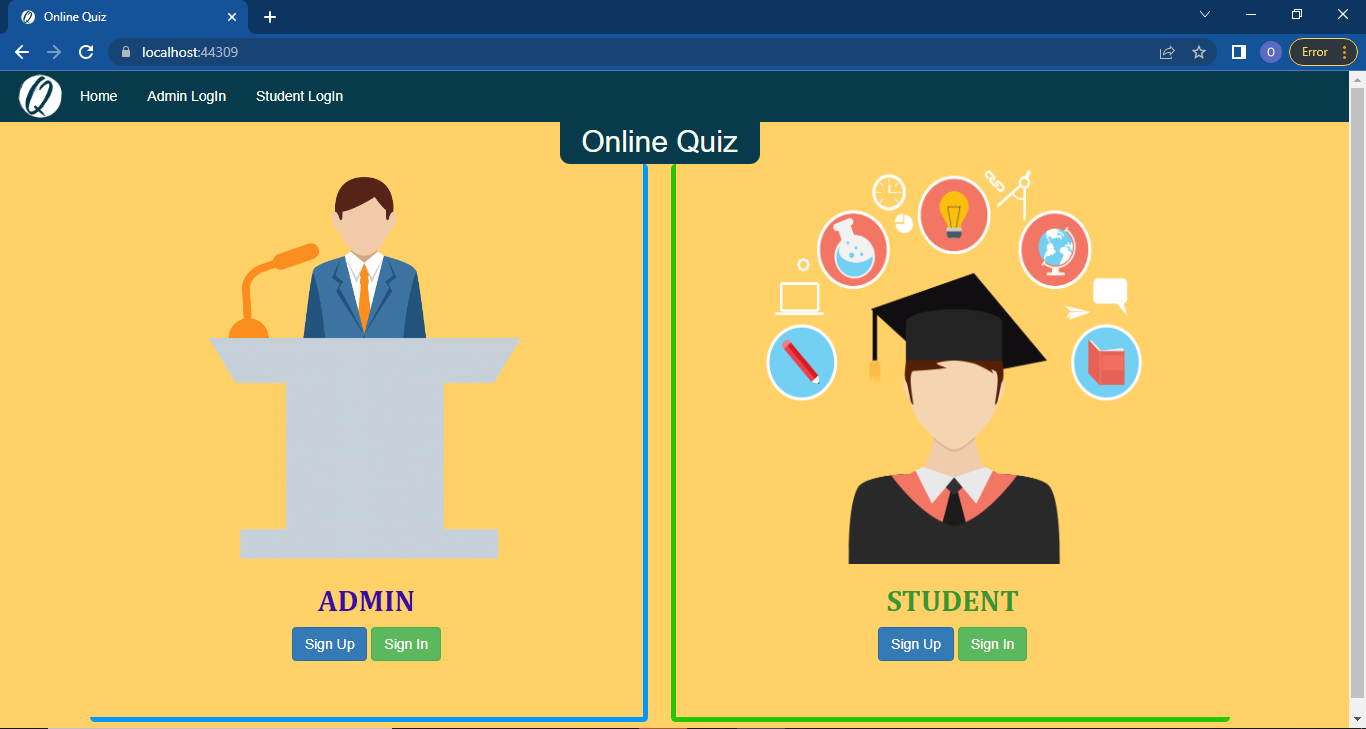
public virtual ICollection<SET\_EXAM> SET\_EXAM { get; set; }

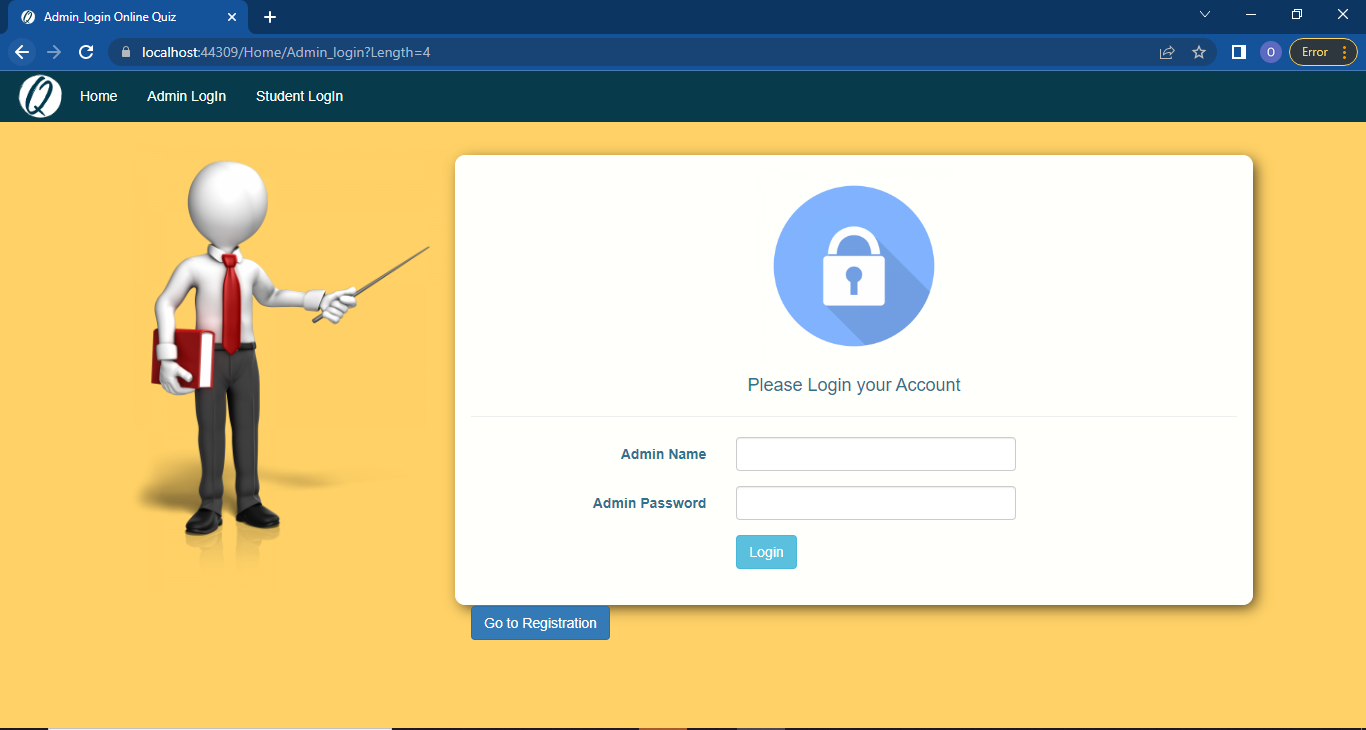
}

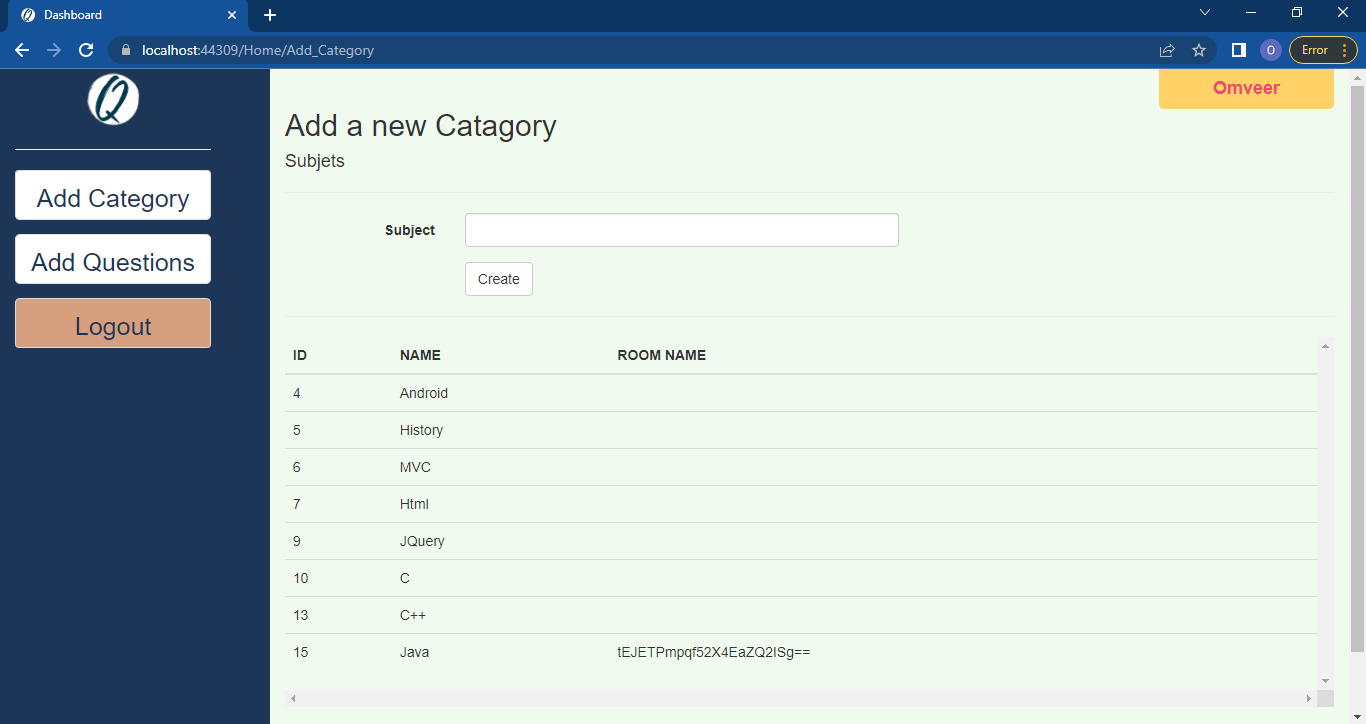
}

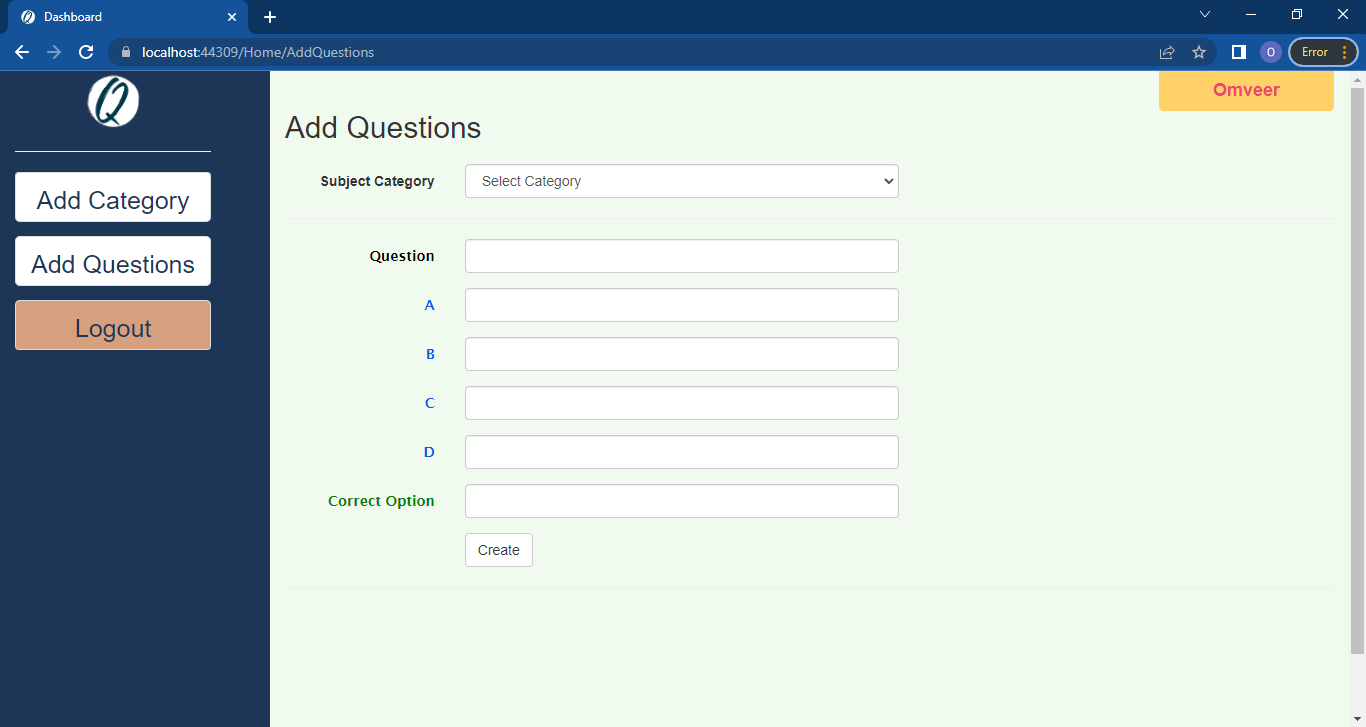
Output ScreenShots

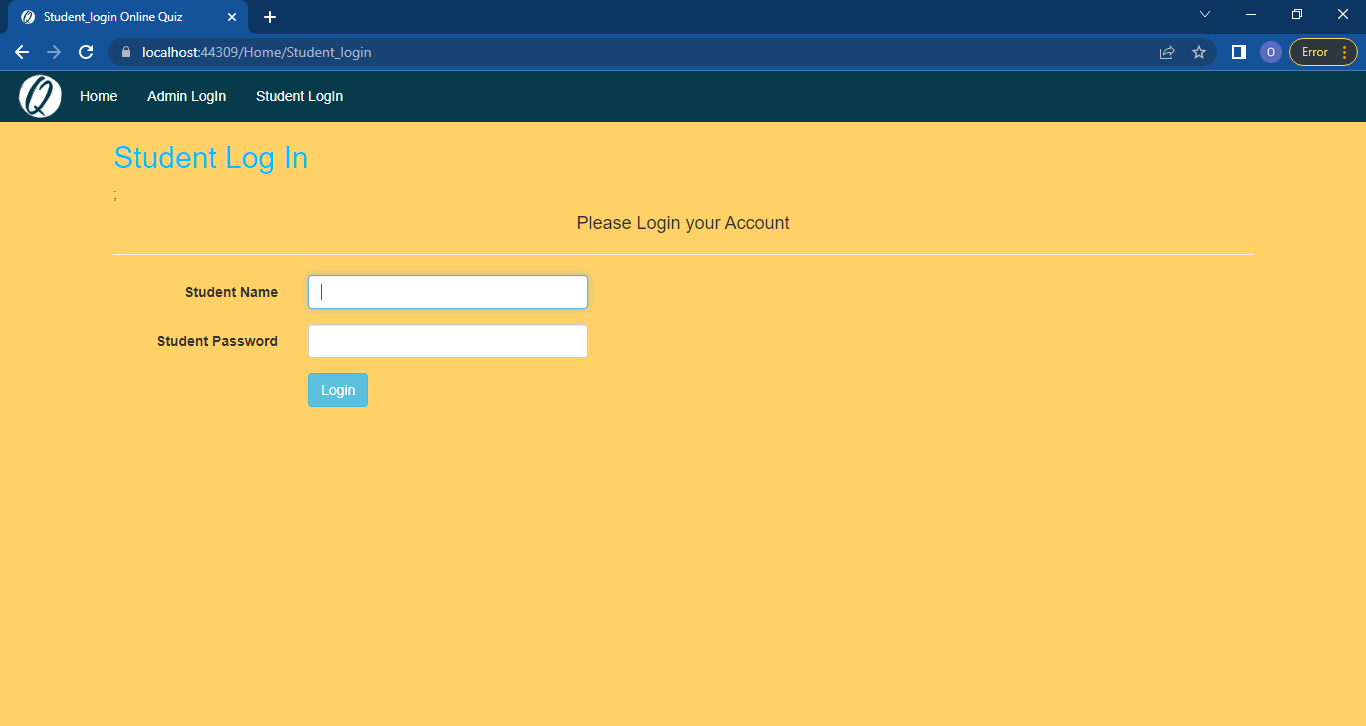
# Screenshots

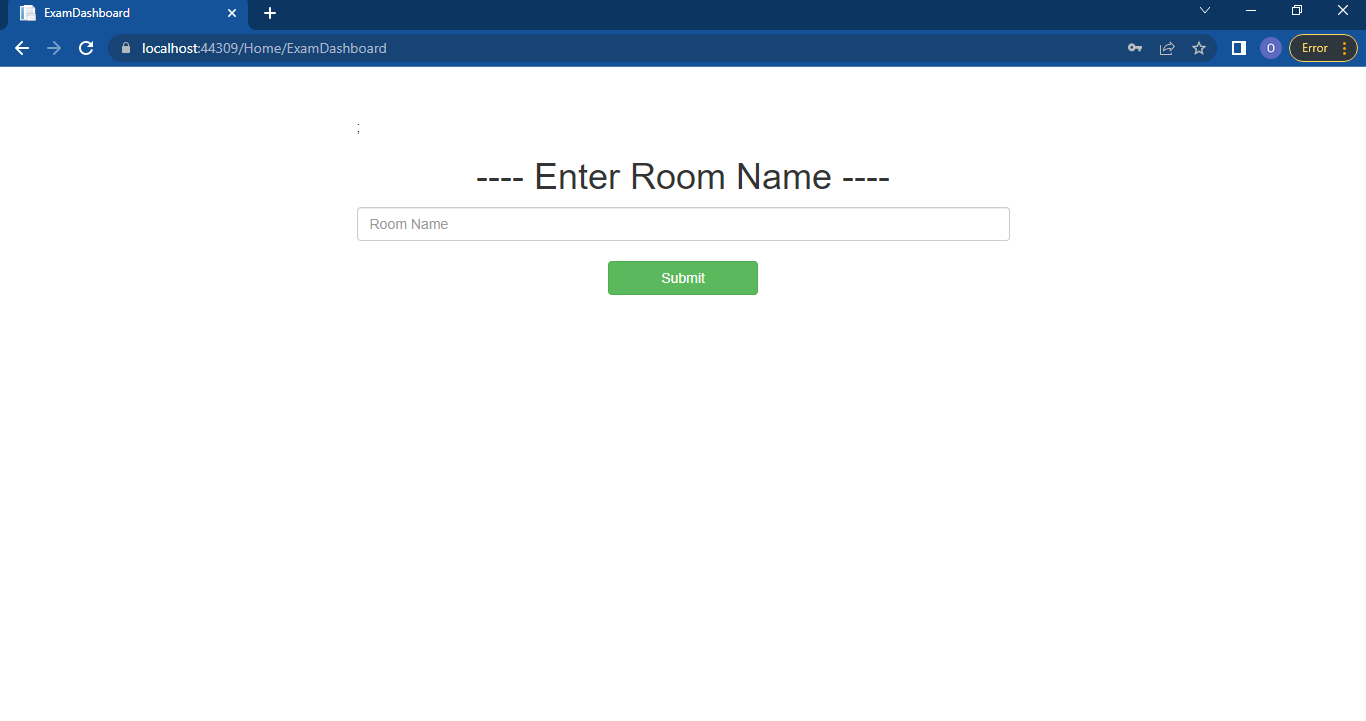












Project Folders

# Folders

