**POE PART2 : Claims Management System**

**Name:** Mongezi Zulu

**Student ID:** ST10296317

**Module:** Prog6212

**Introduction**

This document contains the final submission for the Claims Management System project. The project is developed using ASP.NET Core, Entity Framework Core, and PostgreSQL, aiming to facilitate the submission and management of claims by lecturers.

**Technologies Used**

* ASP.NET Core
* Entity Framework Core
* PostgreSQL
* HTML/CSS for Frontend
* xUnit for Unit Testing

**Database Configuration**

**appsettings.json**

{

"ConnectionStrings": {

"DefaultConnection": "Host=localhost;Port=5432;Database=cmcs\_db;Username=postgres;Password=mageba2003"

},

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft": "Warning",

"Microsoft.Hosting.Lifetime": "Information"

}

},

"AllowedHosts": "\*"

}

**Models**

**Lecturer.cs**

public class Lecturer

{

public int LecturerID { get; set; }

public string Name { get; set; }

public string Email { get; set; }

}

**Claim.cs**

public class Claim

{

public int ClaimID { get; set; }

public int LecturerID { get; set; }

public decimal HoursWorked { get; set; }

public decimal HourlyRate { get; set; }

public string Status { get; set; }

public virtual Lecturer Lecturer { get; set; }

}

**SupportingDocument.cs**

public class SupportingDocument

{

public int DocumentID { get; set; }

public int ClaimID { get; set; }

public string FilePath { get; set; }

public virtual Claim Claim { get; set; }

}

**ApplicationDbContext**

using Microsoft.EntityFrameworkCore;

public class ApplicationDbContext : DbContext

{

public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options)

: base(options)

{

}

public DbSet<Lecturer> Lecturers { get; set; }

public DbSet<Claim> Claims { get; set; }

public DbSet<SupportingDocument> SupportingDocuments { get; set; }

protected override void OnModelCreating(ModelBuilder modelBuilder)

{

modelBuilder.Entity<Claim>()

.HasOne(c => c.Lecturer)

.WithMany()

.HasForeignKey(c => c.LecturerID);

modelBuilder.Entity<SupportingDocument>()

.HasOne(sd => sd.Claim)

.WithMany()

.HasForeignKey(sd => sd.ClaimID);

}

}

**ClaimsController**

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System.IO;

using System.Threading.Tasks;

public class ClaimsController : Controller

{

private readonly ApplicationDbContext \_context;

public ClaimsController(ApplicationDbContext context)

{

\_context = context;

}

[HttpGet]

public IActionResult Create()

{

return View();

}

[HttpPost]

public async Task<IActionResult> Create(Claim claim, IFormFile[] documents)

{

if (ModelState.IsValid)

{

\_context.Claims.Add(claim);

await \_context.SaveChangesAsync();

if (documents != null)

{

foreach (var document in documents)

{

if (document.Length > 0)

{

var filePath = Path.Combine("uploads", document.FileName);

if (!Directory.Exists("uploads"))

{

Directory.CreateDirectory("uploads");

}

using (var stream = new FileStream(filePath, FileMode.Create))

{

await document.CopyToAsync(stream);

}

var supportingDocument = new SupportingDocument

{

ClaimID = claim.ClaimID,

FilePath = filePath

};

\_context.SupportingDocuments.Add(supportingDocument);

}

}

await \_context.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

return View(claim);

}

[HttpGet]

public IActionResult Index()

{

var claims = \_context.Claims.ToList();

return View(claims);

}

[HttpPost]

public async Task<IActionResult> Approve(int id)

{

var claim = await \_context.Claims.FindAsync(id);

if (claim != null)

{

claim.Status = "Approved";

await \_context.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

[HttpPost]

public async Task<IActionResult> Reject(int id)

{

var claim = await \_context.Claims.FindAsync(id);

if (claim != null)

{

claim.Status = "Rejected";

await \_context.SaveChangesAsync();

}

return RedirectToAction(nameof(Index));

}

}

**Views**

**Create.cshtml**

@model Claim

<h2>Submit a Claim</h2>

<form asp-action="Create" method="post" enctype="multipart/form-data">

<div>

<label>Hours Worked:</label>

<input asp-for="HoursWorked" />

</div>

<div>

<label>Hourly Rate:</label>

<input asp-for="HourlyRate" />

</div>

<div>

<label>Supporting Documents:</label>

<input type="file" name="documents" multiple />

</div>

<div>

<button type="submit">Submit</button>

</div>

</form>

**Index.cshtml**

@model IEnumerable<Claim>

<h2>Claims</h2>

<table>

<thead>

<tr>

<th>Claim ID</th>

<th>Hours Worked</th>

<th>Hourly Rate</th>

<th>Status</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

@foreach (var claim in Model)

{

<tr>

<td>@claim.ClaimID</td>

<td>@claim.HoursWorked</td>

<td>@claim.HourlyRate</td>

<td>@claim.Status</td>

<td>

<form asp-action="Approve" asp-route-id="@claim.ClaimID" method="post">

<button type="submit">Approve</button>

</form>

<form asp-action="Reject" asp-route-id="@claim.ClaimID" method="post">

<button type="submit">Reject</button>

</form>

</td>

</tr>

}

</tbody>

</table>

**Startup Configuration**

using Microsoft.AspNetCore.Builder;

using Microsoft.AspNetCore.Hosting;

using Microsoft.Extensions.DependencyInjection;

using Microsoft.Extensions.Hosting;

using Microsoft.EntityFrameworkCore;

public class Startup

{

public void ConfigureServices(IServiceCollection services)

{

services.AddDbContext<ApplicationDbContext>(options =>

options.UseNpgsql(Configuration.GetConnectionString("DefaultConnection")));

services.AddControllersWithViews();

}

public void Configure(IApplicationBuilder app, IWebHostEnvironment env)

{

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

else

{

app.UseExceptionHandler("/Home/Error");

app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

pattern: "{controller=Claims}/{action=Index}/{id?}");

});

}

}

**Conclusion**

The Claims Management System provides an effective way for lecturers to submit and manage their claims. The application incorporates modern web development practices and frameworks to ensure a robust and user-friendly experience. Future improvements can include enhanced reporting features and user authentication mechanisms.

Top of Form

Bottom of Form