PROG6212 Code V2

CMCS.Mvc

Program.cs

```
using CMCS.Mvc; using

CMCS.Mvc.Data; using

CMCS.Mvc.Services; using

Microsoft.EntityFrameworkCore;


var builder = WebApplication.CreateBuilder(args);


// Add services

builder.Services.AddControllersWithViews();

builder.Services.AddHttpContextAccessor();

builder.Services.AddDistributedMemoryCache();

builder.Services.AddSession(options =>

{
   options.IdleTimeout = TimeSpan.FromMinutes(30);
options.Cookie.HttpOnly = true;
options.Cookie.IsEssential = true;
});

// Add DbContext builder.Services.AddDbContext<CMCSDbContext>(options =>
options.UseSqlServer(builder.Configuration.GetConnectionString("CMCSDatabase
"))); // Register custom services builder.Services.AddScoped<SessionService>();

var app = builder.Build();

// Configure session middleware
app.UseSession();

// ... rest of middleware configuration

app.MapControllerRoute(     name: "default",
pattern:
"{controller=Login}/{action=Index}/{id?}");

app.Run();
```

Wwwroot/css/site.css

```
/* ===== MODERN CMCS DESIGN SYSTEM ===== */ :root { --primary: #4361ee; --
primarydark: #3a56d4; --secondary: #7209b7; --success: #4cc9f0; --danger:
#f72585; --warning:
#f8961e; --info: #4895ef; --dark: #2b2d42; --light: #f8f9fa; --gray: #6c757d; --border:
#e9ecef; --shadow: 0 4px 6px -1px rgba(0, 0, 0, 0.1), 0 2px 4px -1px rgba(0, 0, 0, 0.06); -
```

shadow-lg: 0 10px 15px -3px rgba(0, 0, 0, 0.1), 0 4px 6px -2px rgba(0, 0, 0, 0.05); --
radius:
12px; --transition: all 0.3s cubic-bezier(0.4, 0, 0.2, 1); }

- { margin: 0; padding: 0; box-sizing: border-box; }

body { font-family: 'Segoe UI', system-ui, -apple-system, sans-serif; background: lineargradient(135deg, #667eea 0%, #764ba2 100%); min-height: 100vh; color: var(--dark); lineheight: 1.6; }

/* ===== LAYOUT & CONTAINERS ===== */ .container { max-width: 1200px; margin: 0 auto; padding: 0 20px; }

/* ===== MODERN HEADER ===== */ .site-header { background: rgba(255, 255, 255, 0.95); backdrop-filter: blur(20px); border-bottom: 1px solid rgba(255, 255, 255, 0.2); box-shadow: var(--shadow); position: sticky; top: 0; z-index: 1000; }

```
.site-header .container {
display: flex;    justify-
content: space-between;
align-items: center;
padding: 1rem 20px;
}
```

.logo { font-size: 1.5rem; font-weight: 700; background: linear-gradient(135deg, var(-primary), var(--secondary)); -webkit-background-clip: text; -webkit-text-fill-color: transparent; background-clip: text; }

/* ===== ENHANCED NAVIGATION ===== */ .navbar { display: flex; align-items: center; gap:
1.5rem; }

```
.navbar a {
    text-decoration: none;
color: var(--dark);    font-
weight: 500;    padding:
0.5rem 1rem;    border-
radius: var(--radius);
transition: var(--
transition);    position:
relative;    overflow:
hidden;
}

    .navbar a::before
{        content: '';
position: absolute;
bottom: 0;
left: 0;        width:
0;        height: 2px;
background: var(--
primary);
transition: var(--
transition);
    }

    .navbar a:hover::before {
width: 100%;
    }

    .navbar a:hover {
color: var(--primary);
        background: rgba(67, 97, 238, 0.1);
    }
```

/* ===== MODERN LOGOUT BUTTON ===== */ .logout-form { display: inline; }

.btn-logout { background: linear-gradient(135deg, var(--danger), #b5179e); color: white; border: none; padding: 0.75rem 1.5rem; border-radius: 50px; cursor: pointer; font-weight: 600; transition: var(--transition); box-shadow: var(--shadow); display: flex; align-items: center; gap: 0.5rem; }

```
.btn-logout:hover {
transform: translateY(-2px);
box-shadow: var(--shadow-
lg);
    background: linear-gradient(135deg, #e63946, var(--danger));
}
```

/* ===== MAIN CONTENT AREA ===== */ main.container { padding: 2rem 20px; min-height: calc(100vh - 200px); }

/* ===== MODERN CARD COMPONENT ===== */ .card { background: rgba(255, 255, 255, 0.95); backdrop-filter: blur(20px); border-radius: var(--radius); box-shadow: var(--shadow); border: 1px solid rgba(255, 255, 255, 0.2); padding: 2rem; margin-bottom: 2rem; transition:
var(--transition); }

```
.card:hover {
    transform: translateY(-5px);
    box-shadow: var(--shadow-lg);
}
```

/* ===== ENHANCED TYPOGRAPHY ===== */ h1, h2, h3, h4, h5, h6 { color: var(--dark); margin-bottom: 1rem; font-weight: 700; }

h1 { font-size: 2.5rem; background: linear-gradient(135deg, var(--primary), var(-secondary)); -webkit-background-clip: text; -webkit-text-fill-color: transparent; background-clip: text; } h2 { font-size: 2rem; color: var(--dark); }

.hero-title { font-size: 3.5rem; text-align: center; margin-bottom: 1rem; }

.hero-subtitle { font-size: 1.25rem; text-align: center; color: var(--gray); margin-bottom:
3rem; }

/* ===== MODERN BUTTON SYSTEM ===== */ .btn-primary, .btn-secondary, .btn-danger { padding: 0.75rem 1.5rem; border: none; border-radius: 50px; font-weight: 600; textdecoration: none; display: inline-flex; align-items: center; gap: 0.5rem; transition: var(-transition); cursor: pointer; box-shadow: var(--shadow); }

.btn-primary { background: linear-gradient(135deg, var(--primary), var(--info)); color: white; }

```
.btn-primary:hover {
transform: translateY(-2px);
box-shadow: var(--shadow-
lg);    color: white;
}
```

.btn-secondary { background: rgba(108, 117, 125, 0.1); color: var(--gray); border: 2px solid transparent; }

```
.btn-secondary:hover {
background: var(--gray);
color: white;
}
```

```
.btn-danger { background: linear-gradient(135deg, var(--danger), #b5179e); color:
white; }
```

```
.btn-danger:hover {     transform: translateY(-2px);
box-shadow: var(--shadow-lg);     background: linear-
gradient(135deg, #e63946, var(--danger));     color: white;
}
```

```
/* ===== ENHANCED TABLE DESIGN ===== */ .styled-table { width: 100%;
background: white; border-radius: var(--radius); overflow: hidden; box-shadow: var(--
shadow); bordercollapse: collapse; }
```

```
.styled-table thead {
    background: linear-gradient(135deg, var(--primary), var(-
secondary));
}
```

```
.styled-table th {
color: white;     padding:
1rem;     text-align:
left;     font-weight:
600;     text-transform:
uppercase;     font-size:
0.875rem;
    letter-spacing: 0.5px;
}
```

```
.styled-table td {     padding: 1rem;
border-bottom: 1px solid var(--
border);     transition: var(--
transition);
}
```

```
.styled-table tbody tr:hover {
background: rgba(67, 97, 238, 0.05);
transform: scale(1.01);
}
```

```
/* ===== MODERN FORM STYLING ===== */ .form-grid { display: grid; gap: 1.5rem;
maxwidth: 600px; }
```

```
.form-control { width: 100%; padding: 1rem; border: 2px solid var(--border); border-
radius: var(--radius); font-size: 1rem; transition: var(--transition); background:
rgba(255, 255, 255,
0.8); }
```

```
.form-control:focus {     outline: none;
border-color: var(--primary);     box-shadow:
0 0 0 3px rgba(67, 97, 238, 0.1);
transform: translateY(-2px);
}
```

```
.form-control::placeholder {
    color: var(--gray);
}
```

```
/* ===== DASHBOARD ENHANCEMENTS ===== */ .dashboard { display: grid; gap:
2rem; }
```

```
.hero-section { text-align: center; padding: 4rem 2rem; background: rgba(255, 255,
255,
0.1); backdrop-filter: blur(20px); border-radius: var(--radius); border: 1px solid
rgba(255, 255, 255, 0.2); }
```

```
.hero-stats { display: grid; grid-template-columns: repeat(auto-fit, minmax(200px,
1fr)); gap: 1.5rem; margin-top: 3rem; }
```

```
.stat-card { background: rgba(255, 255, 255, 0.9); padding: 2rem; border-radius: var(-
radius); text-align: center; transition: var(--transition); border: 1px solid rgba(255, 255,
255,
0.2); }
```

```
.stat-card:hover {
transform: translateY(-
5px);
    box-shadow: var(--shadow-lg);
}
```

```
.stat-number { font-size: 2.5rem; font-weight: 700; background: linear-
gradient(135deg, var(--primary), var(--secondary)); -webkit-background-clip: text;
-webkit-text-fill-color: transparent; background-clip: text; }
```

```
.stat-label { color: var(--gray); font-size: 0.875rem; text-transform: uppercase;
letterspacing: 0.5px; }
```

```
/* ===== ACTION GRID ===== */ .action-grid { display: grid; grid-template-columns:
repeat(auto-fit, minmax(280px, 1fr)); gap: 1.5rem; margin-top: 2rem; }
```

```
.action-card { background: rgba(255, 255, 255, 0.9); padding: 2rem; border-radius:
var(-radius); text-decoration: none; color: var(--dark); transition: var(--transition);
border: 1px solid rgba(255, 255, 255, 0.2); position: relative; overflow: hidden; }
```

```
.action-card::before {     content: '';     position:
absolute;     top: 0;     left: -100%;     width: 100%;
height: 100%;     background: linear-gradient(90deg,
transparent, rgba(255, 255,
255, 0.4), transparent);
    transition: var(--transition);
}

.action-card:hover::before {
left: 100%;
}

.action-card:hover {
transform: translateY(-5px);
box-shadow: var(--shadow-
lg);    color: var(--dark);
}
```

```
.action-icon { font-size: 2rem; margin-bottom: 1rem; background: linear-
gradient(135deg, var(--primary), var(--secondary)); -webkit-background-clip: text; -
webkit-text-fill-color: transparent; background-clip: text; }
```

```
.action-arrow { position: absolute; top: 1.5rem; right: 1.5rem; font-size: 1.25rem;
transition: var(--transition); }

.action-card:hover .action-arrow { transform: translateX(5px); }

/* ===== STATUS BADGES ===== */ .status-badge { padding: 0.5rem 1rem; border-
radius:
50px; font-size: 0.875rem; font-weight: 600; text-transform: uppercase; letter-
spacing:
0.5px; }

.status-badge.pending {
background: rgba(248, 150, 30,
0.2);    color: #e85d04;
}

.status-badge.verified {
background: rgba(76, 201, 240,
0.2);    color: #0077b6;
}

.status-badge.approved {
background: rgba(72, 149, 239,
0.2);    color: var(--primary);
}

.status-badge.rejected {
background: rgba(247, 37, 133,
0.2);    color: var(--danger);
}


/* ===== BUTTON ROW ===== */ .button-row { display: flex; gap: 1rem; flex-wrap:
wrap; margin-top: 2rem; }

/* ===== DETAILS GRID ===== */ .details-grid { display: grid; grid-template-columns:
auto 1fr; gap: 1rem; align-items: start; }

.details-grid dt {
font-weight: 600;
    color: var(--gray);
```

```css
}

.details-grid dd {
margin: 0;
}


/* ===== FOOTER ===== */ .site-footer { background: var(--dark); color: white; text-
align:
center; padding: 2rem 0; margin-top: 4rem; }

/* ===== LOADING ANIMATION ===== */ .loading { text-align: center; padding: 2rem;
color: var(--gray); }

.loading::after {     content: '';
display: inline-block;     width:
20px;     height: 20px;     border:
2px solid var(--border);     border-
top: 2px solid var(--primary);
border-radius: 50%;     animation:
spin 1s linear infinite;     margin-
left: 1rem;
}


@keyframes spin { 0% { transform: rotate(0deg); }

100% {
    transform: rotate(360deg);
}


}

/* ===== RESPONSIVE DESIGN ===== */ @media (max-width: 768px) {
.siteheader .container { flex-direction: column; gap: 1rem; }

.navbar {
flex-wrap:
wrap;
    justify-content: center;
}
```

```
.hero-title {
font-size: 2.5rem;
}

.action-grid {
    grid-template-columns: 1fr;
}

.button-row {
    flex-direction: column;
}

.details-grid {
    grid-template-columns: 1fr;
}


}
```

/* ===== GLASS MORPHISM EFFECTS ===== */ .glass { background: rgba(255, 255, 255, 0.1); backdrop-filter: blur(20px); border: 1px solid rgba(255, 255, 255, 0.2); }

/* ===== FLOATING ACTION BUTTON ===== */ .fab { position: fixed; bottom: 2rem; right: 2rem; width: 60px; height: 60px; border-radius: 50%; background: linear-gradient(135deg, var(--primary), var(--secondary)); color: white; border: none; box-shadow: var(--shadow-lg); cursor: pointer; transition: var(--transition); display: flex; align-items: center; justifycontent: center; font-size: 1.5rem; z-index: 1000; }

```
.fab:hover {    transform:
scale(1.1) rotate(90deg);
}
```

/* ===== CUSTOM SCROLLBAR ===== */ ::-webkit-scrollbar { width: 8px; }

::-webkit-scrollbar-track { background: rgba(255, 255, 255, 0.1); }

::-webkit-scrollbar-thumb { background: linear-gradient(135deg, var(--primary), var(-secondary)); border-radius: 4px; }

```
::-webkit-scrollbar-thumb:hover {
    background: linear-gradient(135deg, var(--primary-dark),
var(-secondary));
}
```

Controllers (Folder)

AcademicManagerController.cs

```
using CMCS.Mvc.Models; using

CMCS.Mvc.Data; using

CMCS.Mvc.Services; using

Microsoft.AspNetCore.Mvc;

using

Microsoft.EntityFrameworkCore;


namespace CMCS.Mvc.Controllers

{

  public class AcademicManagerController : BaseController

  {

    private readonly CMCSDbContext _context;


    public AcademicManagerController(CMCSDbContext context, SessionService
sessionService)

      : base(sessionService)

    {

      _context = context;

    }
    // GET: /AcademicManager
public IActionResult Index()
    {
      var verifiedClaims = _context.Claims
.Include(c => c.User)
        .Where(c => c.Status == "Verified")
        .ToList();
      return View(verifiedClaims);
    }

    // POST: /AcademicManager/Approve/5
    [HttpPost]     public
IActionResult Approve(int id)
    {
      var claim =
_context.Claims.Find(id);       if
(claim == null) return NotFound();

      claim.Status = "Approved";
      _context.SaveChanges();

      return RedirectToAction(nameof(Index));
    }

    // POST: /AcademicManager/Reject/5
    [HttpPost]     public
IActionResult Reject(int id)
{
```

```
            var claim =
_context.Claims.Find(id);            if
(claim == null) return NotFound();

            claim.Status = "Rejected";
            _context.SaveChanges();

            return RedirectToAction(nameof(Index));
        }
    }
}
```

BaseController.cs

```
using
CMCS.Mvc.Services; using
Microsoft.AspNetCore.Mvc; using
Microsoft.AspNetCore.Mvc.Filters;

namespace CMCS.Mvc.Controllers
{
    public class BaseController : Controller
    {
        protected readonly SessionService _sessionService;

        public BaseController(SessionService sessionService)
        {
            _sessionService = sessionService;
        }
    public override void OnActionExecuting(ActionExecutingContext context)
    {
      if (!_sessionService.IsLoggedIn())
      {
        context.Result = RedirectToAction("Index", "Login");
return;
      }

      // Check role-based access          var currentRole =
_sessionService.GetCurrentUser()?.Role;          var
controllerName = context.Controller.GetType().Name;

      if (!HasAccess(currentRole, controllerName))
      {
        context.Result = RedirectToAction("AccessDenied", "Login");
return;
      }

      base.OnActionExecuting(context);
    }

    private bool HasAccess(string role, string controllerName)
    {
      return (role, controllerName) switch
      {
        ("HR", nameof(HRController)) => true,
        ("Lecturer", nameof(LecturerController)) => true,
        ("ProgrammeCoordinator", nameof(ProgrammeCoordinatorController)) =>
true,           ("AcademicManager", nameof(AcademicManagerController)) => true,
        _ => false
      };
    }
  }
}
```

ClaimsController.cs

```
using CMCS.Mvc.Models; using CMCS.Mvc.Data; using
Microsoft.AspNetCore.Hosting;
using Microsoft.AspNetCore.Mvc; using Microsoft.AspNetCore.Http; using
Microsoft.EntityFrameworkCore; using System.IO; using System.Linq;

namespace CMCS.Mvc.Controllers { public class ClaimsController : Controller {
private readonly CMCSDbContext _context; private readonly
IWebHostEnvironment _env;     public ClaimsController(CMCSDbContext
context, IWebHostEnvironment env)
    {
        _context = context;
        _env = env;
    }

    // GET: /Claims
    public IActionResult Index()
    {         var claims =
_context.Claims
            .Include(c => c.User)  // Changed from Lecturer to
User
            .ToList();
return View(claims);
    }

    // GET:/Claims/Details/5
    public IActionResult Details(int id)
    {        var claim =
_context.Claims
            .Include(c => c.User)  // Changed from Lecturer to
User
            .FirstOrDefault(c => c.Id ==
id);        if (claim == null) return
NotFound();        return View(claim);
    }

    // GET: /Claims/Create
public IActionResult
Create()
    {
        ViewBag.Users = _context.Users
            .Where(u => u.Role == "Lecturer" && u.IsActive)
            .OrderBy(u => u.Name)
            .ThenBy(u => u.Surname)
            .ToList();
return View();
    }

    // POST: /Claims/Create
    [HttpPost]
    [ValidateAntiForgeryToken]
     public IActionResult Create(int userId, string title, string
  description, int hours = 0, decimal rate = 0, IFormFile file =
                               null)
    {
try
        {
            // Validate required
fields            if (userId == 0)
            {
                ModelState.AddModelError("UserId", "Lecturer is
required.");
                ViewBag.Users = _context.Users.Where(u => u.Role
==
"Lecturer" && u.IsActive).ToList();
return View();
            }
            if (string.IsNullOrWhiteSpace(title))
            {
                ModelState.AddModelError("Title", "Title is
required.");
                ViewBag.Users = _context.Users.Where(u => u.Role
==
"Lecturer" && u.IsActive).ToList();
return View();
            }

            // Get the selected user to use their hourly rate if
rate is 0
            var selectedUser =
_context.Users.Find(userId);            if (selectedUser
!= null && rate == 0)
            {                rate =
selectedUser.HourlyRate;
            }

            // Validate file if provided
string filePath = null;
            if (file != null && file.Length > 0)
            {
                // File size limit: 10MB
                if (file.Length > 10 * 1024 * 1024)
                {
                    ModelState.AddModelError("file", "File size
must be less than 10MB.");
                    ViewBag.Users = _context.Users.Where(u =>
u.Role
== "Lecturer" && u.IsActive).ToList();
return View();
                }

                // Allowed file types
                var allowedExtensions = new[] { ".pdf", ".docx",
".xlsx", ".doc", ".xls" };
var fileExtension =
Path.GetExtension(file.FileName).ToLowerInvariant();
```

```
                if (!allowedExtensions.Contains(fileExtension))
                {
                    ModelState.AddModelError("file", "Only PDF,
Word
(.docx, .doc), and Excel (.xlsx, .xls) files are allowed.");
ViewBag.Users = _context.Users.Where(u => u.Role
== "Lecturer" && u.IsActive).ToList();
return View();
                }

                // Secure file storage                var
uploadsRoot = Path.Combine(_env.WebRootPath ?? "wwwroot",
"uploads");
                Directory.CreateDirectory(uploadsRoot);
                // Generate secure filename                var
safeFileName = Path.GetFileName(file.FileName);
var uniqueFileName =
$"{System.Guid.NewGuid()}_{safeFileName}";
                var relativePath = Path.Combine("uploads",
uniqueFileName);                var absolutePath =
Path.Combine(_env.WebRootPath ??
"wwwroot", relativePath);

                using (var stream = new
FileStream(absolutePath, FileMode.Create))                {
file.CopyTo(stream);
                }                 filePath =
relativePath.Replace(Path.DirectorySeparatorChar,
'/');
            }

            // Create claim
var claim = new Claim
            {
                UserId = userId,
                Title = title.Trim(),
                Description =
string.IsNullOrWhiteSpace(description) ?
"No description provided" : description.Trim(),
                Hours = hours,
                Rate = rate,
                FilePath = filePath,
                Status = "Pending",
                CreatedDate = DateTime.UtcNow,
                ModifiedDate = DateTime.UtcNow
            };

            _context.Claims.Add(claim);
_context.SaveChanges();

            return RedirectToAction(nameof(Index));
        }        catch
(System.Exception ex)
        {
            System.Diagnostics.Debug.WriteLine($"Error:
{ex.Message}");
            ModelState.AddModelError("", "Error submitting claim.
Please try again.");
            ViewBag.Users = _context.Users.Where(u => u.Role ==
"Lecturer" && u.IsActive).ToList();
return View();

}
}

    // GET: /Claims/Edit/5
    public IActionResult Edit(int id)
    {        var claim =
_context.Claims
            .Include(c => c.User)  // Changed from Lecturer to
User
            .FirstOrDefault(c => c.Id ==
id);        if (claim == null) return
NotFound();

        ViewBag.Users = _context.Users.Where(u => u.Role
== "Lecturer").ToList();  // Changed from Lecturers to
Users        return View(claim);
    }

    // POST:/Claims/Edit
    [HttpPost]
    [ValidateAntiForgeryToken]    public IActionResult Edit(int
id, int userId, string title, string description, int hours = 0,
decimal rate = 0, IFormFile file = null)  // Changed lecturerId
to userId
    {        var existingClaim =
_context.Claims.Find(id);        if
(existingClaim == null) return NotFound();

        try
        {
            // Validate required
fields            if (userId == 0)
            {
                ModelState.AddModelError("UserId", "User is
required.");  // Changed from LecturerId to UserId
                ViewBag.Users = _context.Users.Where(u => u.Role
==
"Lecturer").ToList();
return View(existingClaim);
            }            if
(string.IsNullOrWhiteSpace(title))
            {
                ModelState.AddModelError("Title", "Title is
required.");
                ViewBag.Users = _context.Users.Where(u => u.Role
==
"Lecturer").ToList();
return View(existingClaim);
            }

            // Validate file if provided
string filePath = existingClaim.FilePath;
if (file != null && file.Length > 0)
            {
                // File size limit: 10MB
                if (file.Length > 10 * 1024 * 1024)
                {
                    ModelState.AddModelError("file", "File size
must be less than 10MB.");
```

```
                    ViewBag.Users = _context.Users.Where(u =>
u.Role
== "Lecturer").ToList();
return View(existingClaim);
                }

                // Allowed file types
                var allowedExtensions = new[] { ".pdf", ".docx",
".xlsx", ".doc", ".xls" };
var fileExtension =
Path.GetExtension(file.FileName).ToLowerInvariant();
                if (!allowedExtensions.Contains(fileExtension))
                {
                    ModelState.AddModelError("file", "Only PDF,
Word (.docx, .doc), and Excel (.xlsx, .xls) files are allowed.");
                    ViewBag.Users = _context.Users.Where(u =>
u.Role
== "Lecturer").ToList();
return View(existingClaim);
                }

                // Delete old file if exists
if (!string.IsNullOrEmpty(existingClaim.FilePath))
                {
try
{
                        var oldFilePath =
Path.Combine(_env.WebRootPath ?? "wwwroot",
existingClaim.FilePath.Replace('/',
Path.DirectorySeparatorChar));                        if
(System.IO.File.Exists(oldFilePath))
{
                            System.IO.File.Delete(oldFilePath);

}
}
catch
                    {
                        // Ignore deletion errors
}
}

                // Secure file storage                var
uploadsRoot = Path.Combine(_env.WebRootPath ?? "wwwroot",
"uploads");
                Directory.CreateDirectory(uploadsRoot);

                // Generate secure filename
                var safeFileName =
Path.GetFileName(file.FileName);                var
uniqueFileName = $"{System.Guid.NewGuid()}_{safeFileName}";
                var relativePath = Path.Combine("uploads",
uniqueFileName);                var absolutePath =
Path.Combine(_env.WebRootPath ??
"wwwroot", relativePath);

                using (var stream = new
FileStream(absolutePath, FileMode.Create))                {
file.CopyTo(stream);
                }                filePath =
relativePath.Replace(Path.DirectorySeparatorChar,
'/');
            }

            // Update claim
            existingClaim.UserId = userId;  // Changed from
LecturerId to UserId
            existingClaim.Title = title.Trim();
existingClaim.Description =
string.IsNullOrWhiteSpace(description) ? "No description
provided" : description.Trim();
            existingClaim.Hours = hours;
existingClaim.Rate = rate;
            existingClaim.FilePath = filePath;

            _context.SaveChanges();

            return RedirectToAction(nameof(Index));
```

```
        }        catch
(System.Exception ex)
        {
            System.Diagnostics.Debug.WriteLine($"Error:
{ex.Message}");
            ModelState.AddModelError("", "Error updating claim.
Please try again.");
            ViewBag.Users = _context.Users.Where(u => u.Role ==
"Lecturer").ToList();            return View(existingClaim);

}
}

    // GET: /Claims/Delete/5
    public IActionResult Delete(int id)
    {        var claim =
_context.Claims
            .Include(c => c.User)  // Changed from Lecturer to
User
            .FirstOrDefault(c => c.Id ==
id);         if (claim == null) return
NotFound();        return View(claim);
    }

    // POST: /Claims/Delete/5
    [HttpPost, ActionName("Delete")]
[ValidateAntiForgeryToken]    public
IActionResult DeleteConfirmed(int id)
    {        var claim = _context.Claims.Find(id);        if
(claim != null && !string.IsNullOrEmpty(claim.FilePath))
        {            try            {                var
filePath = Path.Combine(_env.WebRootPath ??
"wwwroot",
claim.FilePath.Replace('/',
Path.DirectorySeparatorChar));
if (System.IO.File.Exists(filePath))
{
                    System.IO.File.Delete(filePath);

}
}
catch
            {
                // Ignore file system failures
            }
        }

        if (claim != null)
        {
            _context.Claims.Remove(claim);
            _context.SaveChanges();
        }

        return RedirectToAction(nameof(Index));
    }

    // GET: /Claims/Download/5
    public IActionResult Download(int id)
    {         var claim = _context.Claims.Find(id);         if
(claim == null || string.IsNullOrEmpty(claim.FilePath))
return NotFound();

        var absolutePath = Path.Combine(_env.WebRootPath ??
"wwwroot",             claim.FilePath.Replace('/',
Path.DirectorySeparatorChar));

        if (!System.IO.File.Exists(absolutePath)) return
NotFound();

        var fileName = Path.GetFileName(absolutePath);
        return PhysicalFile(absolutePath, "application/octet-
stream", fileName);
    }
}


}
```

HomeController.cs

```
using Microsoft.AspNetCore.Mvc;

namespace CMCS.Mvc.Controllers
{
  public class HomeController : Controller
  {
    public IActionResult Index()
    {
      return View(); // This will show Views/Home/Index.cshtml
    }

    public IActionResult Error()
    {
      return View();
    }
  }
}
```

HRController.cs

```
using
CMCS.Mvc.Models; using
CMCS.Mvc.Data; using
CMCS.Mvc.Services; using
Microsoft.AspNetCore.Mvc;
using
Microsoft.EntityFrameworkCore;

namespace CMCS.Mvc.Controllers
```

```
{
    public class HRController : BaseController
    {
        private readonly CMCSDbContext _context;

        public HRController(CMCSDbContext context, SessionService
sessionService)            : base(sessionService)
        {
            _context = context;
        }

        // GET: /HR        public
IActionResult Index()
        {
            var users = _context.Users.ToList();
return View(users);
        }

        // GET: /HR/Create
public IActionResult
Create()
        {
            return View();
        }

        // POST: /HR/Create
        [HttpPost]        public
IActionResult Create(User user)
```

```
    {
      if (ModelState.IsValid)
      {
        _context.Users.Add(user);
_context.SaveChanges();          return
RedirectToAction(nameof(Index));
      }
      return View(user);
    }

    // GET: /HR/Edit/5
public IActionResult Edit(int
id)
    {
      var user =
_context.Users.Find(id);        if
(user == null) return NotFound();
return View(user);
    }

    // POST: /HR/Edit/5       [HttpPost]
public IActionResult Edit(int id, User
user)
    {
      if (id != user.Id) return NotFound();

      if (ModelState.IsValid)
      {
        _context.Update(user);
        _context.SaveChanges();
        return RedirectToAction(nameof(Index));
      }
      return View(user);
    }

    // GET: /HR/Reports
public IActionResult
Reports()
    {
      var claims = _context.Claims.ToList();
      ViewBag.TotalClaims = claims.Count;
      ViewBag.PendingClaims = claims.Count(c => c.Status == "Pending");
      ViewBag.VerifiedClaims = claims.Count(c => c.Status ==
"Verified");           ViewBag.ApprovedClaims = claims.Count(c =>
c.Status == "Approved");           return View();
    }

    // POST: /HR/GenerateInvoice
    [HttpPost]        public IActionResult GenerateInvoice(int userId, DateTime
startDate, DateTime endDate)
    {
      var user =
_context.Users.Find(userId);           var
claims = _context.Claims
.Where(c => c.UserId == userId &&
                  c.CreatedDate >= startDate &&
                  c.CreatedDate <= endDate &&
                  c.Status == "Approved")
        .ToList();

      // Simple invoice generation (in production, use a PDF
library)          var invoice = new
      {
        User = user.Name + " " + user.Surname,
        Period = $"{startDate:yyyy-MM-dd} to {endDate:yyyy-MM-dd}",
        TotalHours = claims.Sum(c => c.Hours),
        TotalAmount = claims.Sum(c => c.Amount),
        Claims = claims
      };

      return View("Invoice", invoice);
    }
}
}
```

LecturerController.cs

```
using CMCS.Mvc.Models; using CMCS.Mvc.Data; using CMCS.Mvc.Services; using
Microsoft.AspNetCore.Mvc; using Microsoft.EntityFrameworkCore; using
Microsoft.AspNetCore.Hosting; using Microsoft.AspNetCore.Http; using System.IO;
using System.Threading.Tasks; using System.Linq;

namespace CMCS.Mvc.Controllers { public class LecturerController : BaseController
{ private readonly CMCSDbContext _context; private readonly IWebHostEnvironment
_env;     public LecturerController(CMCSDbContext context,
SessionService sessionService, IWebHostEnvironment env)
            : base(sessionService)
        {
```

```
        _context = context;
        _env = env;
    }

    // GET: /Lecturer
    public IActionResult Index()
    {        var currentUser =
_sessionService.GetCurrentUser();        if (currentUser ==
null) return RedirectToAction("Index",
"Login");

        var claims = _context.Claims
            .Include(c => c.User)
            .Where(c => c.UserId == currentUser.Id)
            .OrderByDescending(c => c.CreatedDate)
            .ToList();

        return View(claims);
    }

    // GET: /Lecturer/Create
public IActionResult Create()
    {        var currentUser =
_sessionService.GetCurrentUser();        if (currentUser ==
null) return RedirectToAction("Index",
"Login");

        return View(new Claim { Rate = currentUser.HourlyRate });
    }

    // POST: /Lecturer/Create - SIMPLE AND WORKING
    [HttpPost]
    [ValidateAntiForgeryToken]
    public async Task<IActionResult> Create(string Title, string
Description, int Hours, IFormFile file)
    {        var currentUser =
_sessionService.GetCurrentUser();        if (currentUser ==
null) return RedirectToAction("Index", "Login");
```

```
        try
        {
            // Basic validation
if (string.IsNullOrWhiteSpace(Title))
            {
                ModelState.AddModelError("", "Title is
required.");                return View(new Claim { Title =
Title, Description =
Description, Hours = Hours, Rate = currentUser.HourlyRate });
            }

            if (Hours <= 0 || Hours > 180)
            {
                ModelState.AddModelError("", "Hours must be
between 1 and 180.");                return View(new Claim {
Title = Title, Description =
Description, Hours = Hours, Rate = currentUser.HourlyRate });
            }

            string filePath = null;

            // Handle file upload
if (file != null && file.Length > 0)
            {
                // File validation
if (file.Length > 10 * 1024 * 1024)
                {
                    ModelState.AddModelError("", "File size must
be less than 10MB.");                    return View(new Claim {
Title = Title, Description
= Description, Hours = Hours, Rate = currentUser.HourlyRate });
                }

                var allowedExtensions = new[] { ".pdf", ".docx",
".xlsx", ".doc", ".xls" };
var fileExtension =
Path.GetExtension(file.FileName).ToLowerInvariant();
```

```
                if
(!allowedExtensions.Contains(fileExtension))
{
                    ModelState.AddModelError("", "Only PDF, Word,
and
Excel files are allowed.");                    return View(new
Claim { Title = Title, Description
= Description, Hours = Hours, Rate = currentUser.HourlyRate });
                }

                // Save file                var uploadsRoot =
Path.Combine(_env.WebRootPath ??
"wwwroot", "uploads");                if
(!Directory.Exists(uploadsRoot))
                {
                    Directory.CreateDirectory(uploadsRoot);
                }

                var safeFileName =
Path.GetFileName(file.FileName);                var
uniqueFileName = $"{System.Guid.NewGuid()}_{safeFileName}";
                var relativePath = Path.Combine("uploads",
uniqueFileName);                var absolutePath =
Path.Combine(_env.WebRootPath ??
"wwwroot", relativePath);

                using (var stream = new
FileStream(absolutePath, FileMode.Create))                {
await file.CopyToAsync(stream);
                }                filePath =
relativePath.Replace(Path.DirectorySeparatorChar,
'/');
            }

            // Create and save the
claim            var claim = new Claim
            {
                UserId = currentUser.Id,
                Title = Title.Trim(),
                Description = string.IsNullOrWhiteSpace(Description)
? "No description provided" : Description.Trim(),
Hours = Hours,
                Rate = currentUser.HourlyRate,
                FilePath = filePath,
                Status = "Pending",
                CreatedDate = System.DateTime.UtcNow,
                ModifiedDate = System.DateTime.UtcNow
            };

            _context.Claims.Add(claim);
            var result = await _context.SaveChangesAsync();

            if (result > 0)
            {
                TempData["SuccessMessage"] = $"Claim
'{Title}' submitted successfully! It is now pending
review.";                return
RedirectToAction(nameof(Index));

}
else
            {
                ModelState.AddModelError("", "Failed to save
the claim. Please try again.");               return View(new
Claim { Title = Title, Description =
Description, Hours = Hours, Rate = currentUser.HourlyRate });
            }        }
catch (System.Exception ex)
        {
            // Log the error
            System.Diagnostics.Debug.WriteLine($"ERROR:
{ex.Message}");
            System.Diagnostics.Debug.WriteLine($"STACK TRACE:
{ex.StackTrace}");

            ModelState.AddModelError("", $"An error occurred:
{ex.Message}. Please try again.");
            return View(new Claim { Title = Title, Description =
Description, Hours = Hours, Rate = currentUser.HourlyRate
});        }
    }

    // GET: /Lecturer/Details/5
public IActionResult Details(int
id)
    {        var currentUser =
_sessionService.GetCurrentUser();        if (currentUser ==
null) return RedirectToAction("Index",
"Login");

        var claim = _context.Claims
            .Include(c => c.User)
            .FirstOrDefault(c => c.Id == id && c.UserId ==
currentUser.Id);

        if (claim == null) return NotFound();
return View(claim);
    }

    // DEBUG ACTION - Test database connection
public async Task<IActionResult> TestDatabase()
    {        try        {            var currentUser =
_sessionService.GetCurrentUser();            if (currentUser
== null) return Content("ERROR: No user logged in");

            // Test 1: Check if we can read users
var users = await _context.Users.Take(5).ToListAsync();
var usersCount = users.Count;

            // Test 2: Check if we can read claims
var claims = await _context.Claims.Take(5).ToListAsync();
var claimsCount = claims.Count;

            // Test 3: Create a test
claim            var testClaim = new
Claim
            {
                UserId = currentUser.Id,
                Title = "TEST CLAIM - " +
DateTime.Now.ToString("HH:mm:ss"),
                Description = "This is a test claim from the
debug function",
                Hours = 5,
                Rate = currentUser.HourlyRate,
Status = "Pending",
                CreatedDate = DateTime.UtcNow,
                ModifiedDate = DateTime.UtcNow
            };

            _context.Claims.Add(testClaim);
            var saveResult = await _context.SaveChangesAsync();

            return Content($"SUCCESS!\n" +
                         $"Users in database: {usersCount}\n" +
$"Claims in database: {claimsCount}\n" +
$"Test claim saved with result:
{saveResult}\n" +
                         $"New claim ID: {testClaim.Id}");
        }         catch
(Exception ex)
        {
            return Content($"DATABASE
ERROR:\n{ex.Message}\n\n{ex.StackTrace}");
        }
    }
}


}
```

LoginController.cs

using

CMCS.Mvc.Models;

using CMCS.Mvc.Data;

```
using
CMCS.Mvc.Services;
using
Microsoft.AspNetCore.
Mvc; using
Microsoft.EntityFrame
workCore;

namespace CMCS.Mvc.Controllers
{
    public class LoginController : Controller
    {
        private readonly CMCSDbContext _context;
private readonly SessionService
_sessionService;

        public LoginController(CMCSDbContext context, SessionService sessionService)
        {
            _context = context;
            _sessionService = sessionService;
        }

        // GET: /Login
public IActionResult
Index()
        {
            // If already logged in, redirect to appropriate
dashboard            if (_sessionService.IsLoggedIn())
```

```
        {
          return RedirectToDashboard();
        }
        return View();
      }

      // POST: /Login
      [HttpPost]       public IActionResult Login(string
username, string password)
      {
        // In production, use proper password
hashing          var user = _context.Users
          .FirstOrDefault(u => u.Username == username && u.Password == password
&& u.IsActive);

        if (user != null)
        {
          _sessionService.SetUserSession(user);
return RedirectToDashboard();
        }

        TempData["Error"] = "Invalid username or password";
return RedirectToAction("Index");
      }

      // POST: /Logout
      [HttpPost]
```

```
    [ValidateAntiForgeryToken]
public IActionResult Logout()
    {
      _sessionService.ClearSession();
      return RedirectToAction("Index", "Login");
    }

    private IActionResult RedirectToDashboard()
    {
      var role = _sessionService.GetCurrentUser()?.Role;
return role switch
      {
        "HR" => RedirectToAction("Index", "HR"),
        "Lecturer" => RedirectToAction("Index", "Lecturer"),
        "ProgrammeCoordinator" => RedirectToAction("Index",
"ProgrammeCoordinator"),
        "AcademicManager" => RedirectToAction("Index",
"AcademicManager"),          _ => RedirectToAction("Index")
      };
    }

    // GET: /Login/AccessDenied
public IActionResult
AccessDenied()
    {
      return View();
    }
  }
```

```
}
```

ProgrammeCoordinatorController.cs

```
using CMCS.Mvc.Models; using
CMCS.Mvc.Data;
using CMCS.Mvc.Services; using
Microsoft.AspNetCore.Mvc;
using
Microsoft.EntityFrameworkCore;

namespace CMCS.Mvc.Controllers
{
  public class ProgrammeCoordinatorController : BaseController
  {
    private readonly CMCSDbContext _context;

    public ProgrammeCoordinatorController(CMCSDbContext context,
SessionService sessionService)
      : base(sessionService)
    {
      _context = context;
    }

    // GET: /ProgrammeCoordinator
public IActionResult Index()
    {
      var pendingClaims = _context.Claims
        .Include(c => c.User)
```

```
            .Where(c => c.Status == "Pending")
            .ToList();
          return View(pendingClaims);
       }

       // POST: /ProgrammeCoordinator/Verify/5
       [HttpPost]        public
IActionResult Verify(int id)
       {
          var claim =
_context.Claims.Find(id);           if
(claim == null) return NotFound();

          claim.Status = "Verified";
_context.SaveChanges();

          return RedirectToAction(nameof(Index));
       }

}
}
```

Data (Folder)

CMCDbContext.cs

```
using Microsoft.EntityFrameworkCore;
using CMCS.Mvc.Models;

namespace CMCS.Mvc.Data
```

```
{
  public class CMCSDbContext : DbContext
  {
    public CMCSDbContext(DbContextOptions<CMCSDbContext> options) :
base(options)
    {
    }

    public DbSet<User> Users { get; set; }        public
DbSet<Claim> Claims { get; set; }        public
DbSet<ProofDocument> ProofDocuments { get; set; }

    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
      // Configure User entity
modelBuilder.Entity<User>(entity =>
      {
        entity.ToTable("Users");            entity.HasKey(u => u.Id);
entity.Property(u => u.Username).IsRequired().HasMaxLength(100);
entity.Property(u => u.Password).IsRequired().HasMaxLength(255);
entity.Property(u => u.Role).IsRequired().HasMaxLength(50);
entity.Property(u => u.Name).IsRequired().HasMaxLength(100);
entity.Property(u => u.Surname).IsRequired().HasMaxLength(100);
entity.Property(u => u.Email).IsRequired().HasMaxLength(255);
entity.Property(u => u.HourlyRate).HasColumnType("decimal(18,2)");
entity.Property(u => u.Department).HasMaxLength(100);
entity.Property(u => u.CreatedDate).HasDefaultValueSql("GETDATE()");
entity.Property(u => u.IsActive).HasDefaultValue(true);
```

```
        });

        // Configure Claim entity
modelBuilder.Entity<Claim>(entity =>
        {
          entity.ToTable("Claims");             entity.HasKey(c => c.Id);
entity.Property(c => c.Title).IsRequired().HasMaxLength(200);
entity.Property(c => c.Description).HasColumnType("NVARCHAR(MAX)");
entity.Property(c => c.Hours).IsRequired();             entity.Property(c =>
c.Rate).HasColumnType("decimal(18,2)").IsRequired();
entity.Property(c => c.FilePath).HasMaxLength(500);             entity.Property(c
=>
c.Status).IsRequired().HasMaxLength(50).HasDefaultValue("Pending");
entity.Property(c => c.CreatedDate).HasDefaultValueSql("GETDATE()");
entity.Property(c => c.ModifiedDate).HasDefaultValueSql("GETDATE()");

          // Relationship
entity.HasOne(c => c.User)
              .WithMany(u => u.Claims)
              .HasForeignKey(c => c.UserId)
              .OnDelete(DeleteBehavior.Restrict);
        });

        // Configure ProofDocument entity
modelBuilder.Entity<ProofDocument>(entity =>         {

          entity.ToTable("ProofDocuments");             entity.HasKey(p =>
p.Id);             entity.Property(p =>
p.FileName).IsRequired().HasMaxLength(255);             entity.Property(p
=> p.FilePath).IsRequired().HasMaxLength(500);
entity.Property(p => p.UploadedAt).HasDefaultValueSql("GETDATE()");
entity.Property(p => p.FileSize);

        // Relationship
entity.HasOne(p => p.Claim)
           .WithMany()
           .HasForeignKey(p => p.ClaimId)
           .OnDelete(DeleteBehavior.Cascade);
      });

      base.OnModelCreating(modelBuilder);
    }

}
}
```

Models

Claim.cs using

```
System.ComponentModel.DataAnnotations; using
System.ComponentModel.DataAnnotations.Schema;

namespace CMCS.Mvc.Models
{
  public class Claim
  {
    public int Id { get; set; }
    [Required]      public
int UserId { get; set; }

    [ForeignKey("UserId")]
public User User { get; set;
}

    [Required]      [StringLength(200)]
public string Title { get; set; } =
string.Empty;

    public string Description { get; set; }

    [Required]
    [Range(1, 180, ErrorMessage = "Hours must be between 1 and 180")]
public int Hours { get; set; }

    [Required]
    [Column(TypeName = "decimal(18,2)")]
public decimal Rate { get; set; }

    [StringLength(500)]
    public string FilePath { get; set; }

    [Required]      [StringLength(50)]
public string Status { get; set; } =
"Pending";
```

```
        public DateTime CreatedDate { get; set; } = DateTime.UtcNow;

        public DateTime ModifiedDate { get; set; } = DateTime.UtcNow;

        [NotMapped]       public decimal
Amount => Hours * Rate;

        [NotMapped]
        public string LecturerName => User?.Name + " " + User?.Surname;
    }
}
```

ClaimStatus.cs

```
namespace
CMCS.Mvc.Models
{
    public enum ClaimStatus
    {
        Draft = 0,
        Submitted = 1,
        Verified = 2,
        SentToManager = 3,
        Approved = 4,
        Rejected = 5,
        Settled = 6
    }
}
```

InMemoryStore.cs

```
using CMCS.Mvc.Models;
using
System.Collections.Generic;
using System.Linq;

namespace CMCS.Mvc
{
  /// <summary>
  /// Central in-memory store for Claims and Lecturers.
  /// Used instead of a database — data resets when the app restarts.
  /// </summary>
public class
InMemoryStore
  {
    // === CLAIMS ===      private readonly
List<Claim> _claims = new();      private int
_nextClaimId = 1;      private readonly
object _claimsLock = new();

    public List<Claim> GetAllClaims()
    {
      lock (_claimsLock)
      {
        return _claims.Select(CloneClaim).ToList();
      }
    }
    public Claim? GetClaim(int id)
    {
      lock (_claimsLock)
      {
        var claim = _claims.FirstOrDefault(c => c.Id == id);
return claim != null ? CloneClaim(claim) : null;
      }
    }

    public Claim AddClaim(Claim claim)
    {
      lock (_claimsLock)
      {
        var copy =
CloneClaim(claim);
copy.Id = _nextClaimId++;
_claims.Add(copy);
return CloneClaim(copy);
      }
    }

    public bool UpdateClaim(Claim claim)
    {
      lock (_claimsLock)
      {
        var existing = _claims.FirstOrDefault(c => c.Id ==
claim.Id);          if (existing == null) return false;
```

```
        existing.Title = claim.Title;
existing.Description = claim.Description;
existing.Hours = claim.Hours;
existing.Rate = claim.Rate;
existing.FilePath = claim.FilePath;
existing.Status = claim.Status;
return true;
      }
    }

    public bool DeleteClaim(int id)
    {
      lock (_claimsLock)
      {
        var c = _claims.FirstOrDefault(x => x.Id
== id);        if (c == null) return false;

        _claims.Remove(c);
return true;
      }
    }

    private static Claim CloneClaim(Claim c) => new()
    {
      Id = c.Id,
      Title = c.Title,
      Description = c.Description,
      Hours = c.Hours,
      Rate = c.Rate,

      FilePath = c.FilePath,

      Status = c.Status

    };


    // === LECTURERS ===

    private readonly List<Lecturer> _lecturers =

new();        private int _nextLecturerId = 1;

private readonly object _lecturerLock = new();


    public List<Lecturer> GetAllLecturers()

    {

      lock (_lecturerLock)

      {

        return _lecturers.Select(CloneLecturer).ToList();

      }

    }


    public Lecturer? GetLecturer(int id)

    {

      lock (_lecturerLock)

      {

        var lecturer = _lecturers.FirstOrDefault(l => l.Id == id);

return lecturer != null ? CloneLecturer(lecturer) : null;

      }

    }


    public Lecturer AddLecturer(Lecturer lecturer)
    {
      lock (_lecturerLock)
      {
        var copy =
CloneLecturer(lecturer);
copy.Id = _nextLecturerId++;
_lecturers.Add(copy);            return
CloneLecturer(copy);
      }
    }

    public bool UpdateLecturer(Lecturer lecturer)
    {
      lock (_lecturerLock)
      {
        var existing = _lecturers.FirstOrDefault(l => l.Id ==
lecturer.Id);            if (existing == null) return false;

        existing.Name = lecturer.Name;
existing.Email = lecturer.Email;
existing.Department = lecturer.Department;
existing.Phone = lecturer.Phone;
return true;
      }
    }

    public bool DeleteLecturer(int id)
    {
      lock (_lecturerLock)
      {
        var existing = _lecturers.FirstOrDefault(l => l.Id ==
id);            if (existing == null) return false;

        _lecturers.Remove(existing);
return true;
      }
    }

    private static Lecturer CloneLecturer(Lecturer l) => new()
    {
      Id = l.Id,
      Name = l.Name ?? string.Empty,
      Email = l.Email ?? string.Empty,
      Department = l.Department ?? string.Empty,
      Phone = l.Phone ?? string.Empty
    };

    // Add to InMemoryStore class
public List<Claim>
GetPendingClaims()
    {
      lock (_claimsLock)
      {
        return _claims.Where(c => c.Status == "Pending")
              .Select(CloneClaim)
              .ToList();
```

```
            }
        }

        public bool UpdateClaimStatus(int claimId, string newStatus)
        {
            lock (_claimsLock)
            {
                var claim = _claims.FirstOrDefault(c => c.Id ==
claimId);               if (claim == null)
                {
                    System.Diagnostics.Debug.WriteLine($"Claim {claimId} not found for status
update");
                    return false;
                }

                claim.Status = newStatus;
                System.Diagnostics.Debug.WriteLine($"Claim {claimId} status updated to:
{newStatus}");
                return true;
            }
        }
    }
}
```

Lecture.cs

```
using
System.ComponentModel.DataAnnotations;

namespace CMCS.Mvc.Models
{
  public class Lecturer
  {
    [Key]     public
int Id { get; set; }

    [Required]     [StringLength(100)]
public string Name { get; set; } =
string.Empty;

    [Required]
    [StringLength(100)]
    [EmailAddress]
    public string Email { get; set; } = string.Empty;

    [Required]     [StringLength(100)]     public
string Department { get; set; } = string.Empty;

    [Required]     [StringLength(20)]
public string Phone { get; set; } =
string.Empty;

    public DateTime CreatedDate { get; set; } = DateTime.UtcNow;
public bool IsActive { get; set; } = true;

    // Navigation property
    public ICollection<Claim> Claims { get; set; } = new List<Claim>();
  }
}
```

ProofDocument.cs

```
using System.ComponentModel.DataAnnotations;

using System.Security.Claims;


namespace CMCS.Mvc.Models

{

  public class ProofDocument

  {

      public int Id { get; set; }

     public string FileName { get; set; } = "";        public string

FilePath { get; set; } = "";        public DateTime UploadedAt {

get; set; } = DateTime.UtcNow;


     // Navigation

     public int? ClaimId { get; set; }

public Claim? Claim { get; set; }

  }

}
```

User.cs

```
using

System.ComponentModel.DataAnnotations; using

System.ComponentModel.DataAnnotations.Schema;


namespace CMCS.Mvc.Models

{

  public class User

  {

     public int Id { get; set; }
```

```
    [Required]        [StringLength(100)]
public string Username { get; set; } =
string.Empty;

    [Required]
    [StringLength(255)] public string Password {
    get; set; } = string.Empty;        [Required]
    [StringLength(50)]        public string Role {
    get; set; } = string.Empty;

    [Required]        [StringLength(100)]
public string Name { get; set; } =
string.Empty;

    [Required]        [StringLength(100)]
public string Surname { get; set; } =
string.Empty;

    [Required]        [EmailAddress]
public string Email { get; set; } =
string.Empty;

    [Column(TypeName = "decimal(18,2)")]
public decimal HourlyRate { get; set; }

    [StringLength(100)]        public string
Department { get; set; } = string.Empty;
```

```
        public DateTime CreatedDate { get; set; } = DateTime.UtcNow;

        public bool IsActive { get; set; } = true;

        // Navigation property        public List<Claim> Claims
{ get; set; } = new List<Claim>();

}
}
```

Services (Folder)

SessonService.cs

```
using
CMCS.Mvc.Models;

namespace CMCS.Mvc.Services
{
    public class SessionService
    {
        private readonly IHttpContextAccessor _httpContextAccessor;

        public SessionService(IHttpContextAccessor httpContextAccessor)
        {
            _httpContextAccessor = httpContextAccessor;
        }

        public void SetUserSession(User user)
    {
      _httpContextAccessor.HttpContext.Session.SetString("UserId",
user.Id.ToString());
      _httpContextAccessor.HttpContext.Session.SetString("Username",
user.Username);
      _httpContextAccessor.HttpContext.Session.SetString("Role", user.Role);
_httpContextAccessor.HttpContext.Session.SetString("Name", user.Name);
      _httpContextAccessor.HttpContext.Session.SetString("Surname",
user.Surname);
      _httpContextAccessor.HttpContext.Session.SetString("Email", user.Email);
      _httpContextAccessor.HttpContext.Session.SetString("HourlyRate",
user.HourlyRate.ToString());
    }

    public User GetCurrentUser()
    {
      var userId =
_httpContextAccessor.HttpContext.Session.GetString("UserId");       if
(string.IsNullOrEmpty(userId)) return null;

      return new User
      {
        Id = int.Parse(userId),
        Username =
_httpContextAccessor.HttpContext.Session.GetString("Username"),
        Role = _httpContextAccessor.HttpContext.Session.GetString("Role"),
        Name = _httpContextAccessor.HttpContext.Session.GetString("Name"),
        Surname =
_httpContextAccessor.HttpContext.Session.GetString("Surname"),         Email =
_httpContextAccessor.HttpContext.Session.GetString("Email"),
            HourlyRate =
decimal.Parse(_httpContextAccessor.HttpContext.Session.GetString("HourlyRate")
?? "0")
         };
      }

      public bool IsLoggedIn()
      {
return
!string.IsNullOrEmpty(_httpContextAccessor.HttpContext.Session.GetString("UserI
d"));
      }

      public bool HasRole(string role)
      {
         return _httpContextAccessor.HttpContext.Session.GetString("Role") == role;
      }

      public void ClearSession()
      {
         _httpContextAccessor.HttpContext.Session.Clear();
      }

}
}
```

Views (Folder)

AcademicManger (Folder)

Index.chtml

```
@model List<CMCS.Mvc.Models.Claim>

@{

  Layout = "_Layout";

  ViewData["Title"] = "Verified Claims Approval";

}


<div class="dashboard-container">

  <!-- Header Section -->

  <div class="dashboard-header">

    <div class="header-content">

      <h1 class="page-title">Verified Claims Approval</h1>

      <p class="page-subtitle">Final approval for verified claims</p>

    </div>

    <div class="header-stats">

      <div class="stat-badge verified">

        <i class="fas fa-check-circle"></i>

        <span>@Model.Count Verified</span>

      </div>

      <div class="stat-badge amount">

        <i class="fas fa-money-bill-wave"></i>

        <span>@Model.Sum(c => c.Amount).ToString("C")</span>

      </div>

    </div>

  </div>
```

```
<!-- Approval Overview -->
<div class="approval-overview">
    <div class="overview-card">
        <div class="overview-icon">
            <i class="fas fa-file-invoice-dollar"></i>
        </div>
        <div class="overview-content">
            <h3>@Model.Count</h3>
            <p>Claims Ready for Approval</p>
            <div class="overview-trend">
                <i class="fas fa-arrow-up"></i>
                <span>Ready to process</span>
            </div>
        </div>
    </div>

    <div class="overview-card">
        <div class="overview-icon">
            <i class="fas fa-clock"></i>
        </div>
        <div class="overview-content">
            <h3>@Model.Average(c => (DateTime.Now - c.CreatedDate).TotalDays).ToString("F1")</h3>
            <p>Average Days in Review</p>
            <div class="overview-trend trend-good">
                <i class="fas fa-check"></i>
                <span>Efficient</span>
            </div>
        </div>
    </div>

    <div class="overview-card">
        <div class="overview-icon">
            <i class="fas fa-users"></i>
        </div>
        <div class="overview-content">
            <h3>@Model.Select(c => c.User.Department).Distinct().Count()</h3>
            <p>Departments</p>
            <div class="overview-trend">
                <i class="fas fa-building"></i>
                <span>Multiple</span>
            </div>
        </div>
    </div>
</div>

<!-- Bulk Actions -->
<div class="bulk-actions-card">
    <div class="bulk-header">
        <h3><i class="fas fa-layer-group"></i> Bulk Actions</h3>
        <p>Approve multiple claims at once</p>
    </div>
    <div class="bulk-actions">
        <button class="btn btn-success btn-lg" onclick="selectAllClaims()">
            <i class="fas fa-check-double"></i>
        Select All Claims
      </button>
      <button class="btn btn-primary btn-lg" onclick="approveSelected()" disabled id="approveSelectedBtn">
        <i class="fas fa-check-circle"></i>
        Approve Selected (<span id="selectedCount">0</span>)
      </button>
      <button class="btn btn-danger btn-lg" onclick="rejectSelected()" disabled id="rejectSelectedBtn">
        <i class="fas fa-times-circle"></i>
        Reject Selected (<span id="rejectCount">0</span>)
      </button>
    </div>
  </div>

  <!-- Claims Grid -->
  <div class="claims-grid">
    @foreach (var claim in Model.OrderByDescending(c => c.CreatedDate))
    {
      <div class="claim-card" data-claim-id="@claim.Id">
        <div class="claim-card-header">
          <div class="claim-checkbox">
            <input type="checkbox" class="claim-select" onchange="updateSelection()">
          </div>
          <div class="claim-meta">
            <div class="lecturer-info">
              <div class="avatar-small">
                @claim.User.Name[0]@claim.User.Surname[0]
              </div>
              <div>
                <h4>@claim.User.Name @claim.User.Surname</h4>
                <span class="department">@claim.User.Department</span>
              </div>
            </div>
            <div class="claim-amount-large">
              <span class="amount">@claim.Amount.ToString("C")</span>
              <span class="rate">@claim.Rate.ToString("C")/hr × @claim.Hours
hrs</span>
            </div>
          </div>
        </div>

        <div class="claim-card-body">
          <h3 class="claim-title">@claim.Title</h3>
          @if (!string.IsNullOrEmpty(claim.Description) && claim.Description != "No
description provided")
          {
            <p class="claim-description">@claim.Description</p>
          }

          <div class="verification-details">
            <div class="verification-step completed">
              <i class="fas fa-check-circle"></i>
              <span>Submitted by Lecturer</span>
<small>@claim.CreatedDate.ToString("MMM dd")</small>
            </div>
            <div class="verification-step completed">
```

```
              <i class="fas fa-check-circle"></i>
              <span>Verified by Coordinator</span>
              <small>@claim.CreatedDate.AddDays(1).ToString("MMM dd")</small>
            </div>
            <div class="verification-step current">
              <i class="fas fa-user-tie"></i>
              <span>Awaiting Your Approval</span>
              <small>Now</small>
            </div>
          </div>

          <div class="claim-documents">
            @if (!string.IsNullOrEmpty(claim.FilePath))
            {
              <a href="@Url.Action("Download", "Claims", new { id = claim.Id })" class="document-link" target="_blank">
                <i class="fas fa-file-pdf"></i>
                View Supporting Document
              </a>

}
else
            {
              <span class="no-document">
                <i class="fas fa-file-excel"></i>
                No supporting document
              </span>
            }
          </div>
        </div>

        <div class="claim-card-footer">
          <div class="claim-actions">
            <form method="post" action="@Url.Action("Approve", new { id = claim.Id
})" class="action-form">
              @Html.AntiForgeryToken()
              <button type="submit" class="btn btn-success btn-sm">
                <i class="fas fa-check"></i>
                Approve
              </button>
            </form>
            <form method="post" action="@Url.Action("Reject", new { id = claim.Id })"
class="action-form">
              @Html.AntiForgeryToken()
              <button type="submit" class="btn btn-danger btn-sm" onclick="return
confirm('Are you sure you want to reject this claim?')">
                <i class="fas fa-times"></i>
                Reject
              </button>
            </form>
            <a href="@Url.Action("Details", "Claims", new { id = claim.Id })" class="btn
btnsecondary btn-sm">
              <i class="fas fa-eye"></i>
              Details
            </a>
          </div>
          <div class="claim-date">
```

```
                <i class="fas fa-calendar"></i>
                Verified on @claim.CreatedDate.ToString("MMM dd, yyyy")
              </div>
            </div>
          </div>
        }
      </div>

      @if (!Model.Any())
      {
        <div class="empty-state">
          <div class="empty-icon">
            <i class="fas fa-check-double"></i>
          </div>
          <h3>All Claims Processed</h3>
          <p>There are no verified claims awaiting approval at this time.</p>
          <div class="empty-actions">
            <a href="/Claims" class="btn btn-primary">
              <i class="fas fa-list"></i> View All Claims
            </a>
            <a href="/Home" class="btn btn-secondary">
              <i class="fas fa-chart-bar"></i> View Reports
            </a>
          </div>
        </div>
      }

      <!-- Department Summary -->
```

```
@if (Model.Any())
{
    <div class="department-summary">
        <h3><i class="fas fa-chart-bar"></i> Department Summary</h3>
        <div class="summary-grid">
            @{
                var departments = Model.GroupBy(c => c.User.Department)
                .Select(g => new
                {
                    Department = g.Key,
                    Count = g.Count(),
                    Amount = g.Sum(c => c.Amount),
                    Percentage = (g.Count() * 100) / Model.Count
                })
                .OrderByDescending(d => d.Count);
            }
            @foreach (var dept in departments)
            {
                <div class="summary-item">
                    <div class="dept-header">
                        <span class="dept-name">@dept.Department</span>
                        <span class="dept-count">@dept.Count claims</span>
                    </div>
                    <div class="dept-amount">@dept.Amount.ToString("C")</div>
                    <div class="progress-container">
                        <div class="progress-bar">
                            <div class="progress" style="width: @dept.Percentage%;"></div>
                        </div>
                    <span class="progress-text">@dept.Percentage%</span>
                  </div>
                </div>
            }
          </div>
        </div>
    }
</div>

<style>
    .dashboard-
container {
max-width:
1400px;
margin: 0 auto;
padding: 2rem;
    }

    .dashboard-header {
        display: flex;        justify-
content: space-between;
align-items: center;
margin-bottom: 2rem;
gap: 2rem;
    }
```

```
  .page-title {     font-size: 2.5rem;     font-weight:
800;     background: linear-gradient(135deg,
#4cc9f0, #4895ef);
    -webkit-background-clip:
text;     -webkit-text-fill-color:
transparent;     background-
clip: text;     margin: 0;
  }

  .page-subtitle {
color: #6c757d;
font-size: 1.1rem;
margin: 0.5rem 0 0
0;
  }

  .header-stats {
    display:
    flex; gap:
    1rem;
  }

  .stat-badge {
display: inline-flex;
align-items: center;
gap: 0.5rem;
padding: 0.75rem
1.5rem;     border-
```

```
radius: 50px;        font-
weight: 600;        font-
size: 0.9rem;
    }

        .stat-badge.verified {
background: rgba(76, 201, 240, 0.1);
color: #4cc9f0;           border: 1px solid
rgba(76, 201, 240, 0.2);
        }

        .stat-badge.amount {
background: rgba(46, 204, 113, 0.1);
color: #27ae60;           border: 1px solid
rgba(46, 204, 113, 0.2);
        }

    .approval-overview {
display: grid;
        grid-template-columns: repeat(auto-fit,
minmax(250px, 1fr));        gap: 1.5rem;        margin-bottom:
2rem;
    }

    .overview-card {        background:
rgba(255, 255, 255, 0.95);
backdrop-filter: blur(20px);        border-
radius: 16px;        padding: 2rem;
```

```
border: 1px solid rgba(255, 255, 255,
0.2);        box-shadow: 0 8px 32px
rgba(0, 0, 0, 0.1);        display: flex;
align-items: center;        gap: 1.5rem;
transition: all 0.3s ease;
   }

      .overview-card:hover {           transform:
translateY(-5px);           box-shadow: 0 12px
40px rgba(0, 0, 0, 0.15);
      }

   .overview-icon {
      width: 70px;
      height: 70px;
      border-radius:
      16px;
      background: linear-gradient(135deg, #4cc9f0,
#4895ef);        display: flex;        align-items: center;
justify-content: center;        color: white;        font-size:
1.75rem;        flex-shrink: 0;
   }

   .overview-content {
      flex: 1;
   }
```

```
    .overview-
content h3 {
font-size: 2rem;
font-weight: 800;
color: #2b2d42;
margin: 0 0 0.5rem 0;
line-height: 1;
    }

    .overview-content p {
color: #6c757d;
margin: 0 0 0.5rem 0;
font-size: 0.9rem;
    }

  .overview-trend {     display:
inline-flex;     align-items:
center;     gap: 0.25rem;
padding: 0.25rem 0.75rem;
border-radius: 20px;
background: rgba(76, 201, 240,
0.1);     color: #4cc9f0;     font-
size: 0.8rem;     font-weight: 600;
  }

    .overview-trend.trend-good {
background: rgba(46, 204, 113, 0.1);
color: #27ae60;
```

```
      }

    .bulk-actions-card { background:
      rgba(255, 255, 255, 0.95);
      backdrop-filter: blur(20px); border-
      radius: 16px; border: 1px solid
      rgba(255, 255, 255, 0.2); box-
      shadow: 0 8px 32px rgba(0, 0, 0,
      0.1); margin-bottom: 2rem;
      overflow: hidden;
    }

    .bulk-header {        padding: 1.5rem;
  border-bottom: 1px solid rgba(0, 0, 0,
  0.1);
    }

      .bulk-header h3 {
  margin: 0 0 0.5rem
  0;           display: flex;
  align-items: center;
  gap: 0.5rem;
  color: #2b2d42;
      }

      .bulk-header
  p {          margin:
  0;          color:
```

```
#6c757d;
font-size: 0.9rem;
      }

   .bulk-actions {
padding: 1.5rem;
display: flex; gap:
1rem; align-items:
center;
      flex-wrap: wrap;
   }

   .btn-lg {
padding: 1rem
2rem;       font-
size: 1rem;
display: inline-flex;
align-items:
center;       gap:
0.5rem;
   }

   .claims-grid {
display: grid;
      grid-template-columns: repeat(auto-fill,
minmax(400px, 1fr));       gap: 1.5rem;       margin-bottom:
2rem;
   }
```

```
  .claim-card { background: rgba(255,
    255, 255, 0.95); backdrop-filter:
    blur(20px); border-radius: 16px;
    border: 1px solid rgba(255, 255,
    255, 0.2); box-shadow: 0 8px 32px
    rgba(0, 0, 0, 0.1); overflow: hidden;
    transition: all 0.3s ease;
  }

    .claim-card:hover {          transform:
translateY(-5px);          box-shadow: 0 12px
40px rgba(0, 0, 0, 0.15);          border-color:
rgba(76, 201, 240, 0.3);
    }

  .claim-card-header {       padding:
1.5rem;       border-bottom: 1px solid
rgba(0, 0, 0, 0.1);       display: flex;
align-items: flex-start;       gap: 1rem;
  }

  .claim-checkbox {
margin-top: 0.5rem;


}
```

```
    .claim-checkbox
      input[type="checkbox"] { width:
      18px; height: 18px;

      border-radius: 4px;
      border: 2px solid #dee2e6;
cursor: pointer;
    }

  .claim-meta {
    flex: 1;       display: flex;
justify-content: space-
between;       align-items:
flex-start;       gap: 1rem;
  }

  .lecturer-info {
display: flex;
align-items:
center;       gap:
1rem;
  }

  .avatar-
small {
width:
40px;
height:
40px;
      border-radius: 50%; background: linear-
      gradient(135deg, #4cc9f0, #4895ef); display: flex;
      align-items: center;
      justify-content: center;
      color: white;
     font-weight: 600;
font-size: 0.9rem;
flex-shrink: 0;
   }

   .lecturer-info h4 {
margin: 0 0 0.25rem
0;       color:
#2b2d42;       font-
size: 1rem;
   }

   .department {
color: #6c757d;
font-size:
0.8rem;
   }

   .claim-amount-large {
text-align: right;
   }
```

```
      .claim-amount-large .amount {
display: block;
         font-size: 1.25rem; font-
         weight: 700; color:
         #27ae60;

      }
       .claim-amount-large
                            .rate {
         display:
block;
color: #6c757d;
font-size: 0.8rem;
      }

   .claim-card-body {
padding: 1.5rem;
   }

   .claim-title {
margin: 0 0 1rem
0;      color:
#2b2d42;
font-size:
1.1rem;      font-
weight: 600;
line-height: 1.4;
   }
   .claim-
     description {
     margin: 0 0
     1.5rem 0;
     color:
     #6c757d; line-
     height: 1.5;
     font-size:
     0.9rem;
   }

   .verification-details
{      display: flex;
flex-direction:
column;      gap:
1rem;      margin-
bottom: 1.5rem;
   }

   .verification-
step {
display: flex;
align-items:
center;      gap:
1rem;
padding:
```

```
0.75rem;
border-radius:
8px;       font-
size: 0.9rem;
  }

    .verification-step.completed {
background: rgba(46, 204, 113, 0.1);
color: #27ae60;
    }

    .verification-step.current {
      background: rgba(76, 201,
      240, 0.1); color: #4cc9f0;
     border: 1px solid rgba(76, 201, 240,
                                     0.2);
    }
    .verification-
step i {        font-
size: 1rem;
width: 20px;
text-align: center;
    }

    .verification-step span {
      flex: 1;
      font-weight: 500;
    }
    .verification-step
small {          color:
inherit;          opacity:
0.7;
    }

  .claim-documents {
margin-bottom: 1rem;
  }

  .document-link, .no-
    document { display:
    inline-flex; align-items:
    center;

    gap: 0.5rem;
    padding: 0.5rem
    1rem; border-
    radius: 8px; font-
    size: 0.9rem;
    text-decoration: none;
  }

  .document-link {       background:
rgba(67, 97, 238, 0.1);       color:
#4361ee;       border: 1px solid
rgba(67, 97, 238, 0.2);
  }
```

```
    .document-
link:hover {
background: #4361ee;
color: white;
    }

  .no-document {       background:
rgba(108, 117, 125, 0.1);       color:
#6c757d;       border: 1px solid
rgba(108, 117, 125, 0.2);
  }

  .claim-card-footer {
    padding: 1.5rem;
    border-top: 1px solid
    rgba(0, 0, 0, 0.1);
    display: flex; justify-
    content: space-
    between; align-
    items: center;
    gap: 1rem;
  }

  .claim-actions {
display: flex;
gap: 0.5rem;
```

```
align-items:
center;
  }

    .action-
    form {
  margin: 0;
  }

  .btn {      display:
inline-flex;      align-
items: center;
gap: 0.5rem;
padding: 0.5rem
1rem;      border:
none;      border-
radius: 8px; font-
weight: 600; text-
decoration: none;
cursor: pointer;
transition: all 0.3s
ease; font-size:
0.8rem;
  }

  .btn-sm {
padding: 0.5rem
```

```
1rem;       font-size:
0.8rem;
  }

  .btn-success {       background: linear-
gradient(135deg, #27ae60, #2ecc71);       color: white;
  }

    .btn-success:hover {          transform:
translateY(-2px);          box-shadow: 0 4px
12px rgba(39, 174, 96, 0.3);
    }

  .btn-danger {       background: linear-
gradient(135deg, #e74c3c, #f72585);       color: white;


}
    .btn-danger:hover {
      transform: translateY(-2px);          box-
shadow: 0 4px 12px rgba(231, 76, 60, 0.3);
    }

  .btn-secondary {
    background: rgba(108, 117, 125,
0.1);       color: #6c757d;       border:
1px solid rgba(108, 117, 125, 0.2);
  }
```

```
      .btn-secondary:hover {
background: #6c757d;
color: white;
transform: translateY(-2px);
      }

   .btn-primary {       background: linear-
gradient(135deg, #4361ee, #3a56d4);       color: white;
   }

      .btn-primary:hover {          transform:
translateY(-2px);          box-shadow: 0 4px
12px rgba(67, 97, 238, 0.3);
      }

   .claim-date {
color: #6c757d;
font-size: 0.8rem;
display: flex;
align-items:
center;       gap:
0.5rem;
   }

   .department-summary {
background: rgba(255, 255, 255, 0.95);
backdrop-filter: blur(20px);       border-
radius: 16px;       border: 1px solid
rgba(255, 255, 255, 0.2);       box-
shadow: 0 8px 32px rgba(0, 0, 0, 0.1);
padding: 2rem;
  }

    .department-
summary h3 {
margin: 0 0 1.5rem 0;
display: flex;          align-
items: center;          gap:
0.5rem;          color:
#2b2d42;
    }

  .summary-grid {
    display: flex;
flex-direction:
column;       gap:
1rem;
  }

  .summary-item {
    display: flex;       flex-direction:
column;       gap: 0.5rem;
padding: 1rem;       background:
rgba(76, 201, 240, 0.05);
border-radius: 8px;
  }
  .dept-header {      display:
flex;      justify-content:
space-between;      align-
items: center;
  }

  .dept-name {
font-weight:
600;      color:
#2b2d42;
  }

  .dept-count {
    color: #6c757d;
    font-size: 0.9rem;
  }

  .dept-amount
{      font-
weight: 700;
color: #27ae60;
font-size:
1.1rem;
  }

  .progress-
container {
```

```
display: flex;
align-items:
center;       gap:
1rem;
  }

  .progress-bar {
    flex: 1;       height: 6px;
background: rgba(0, 0, 0,
0.1);       border-radius: 3px;
overflow: hidden;
  }

  .progress {
    height:
    100%;
    backgrou
    nd: linear-
    gradient(1
    35deg,
    #4cc9f0,
    #4895ef);
    border-
    radius:
    3px;
    transition: width 0.3s ease;
  }
  .progress-text
{     color:
#6c757d;
font-size:
0.8rem;
font-weight:
600;     min-
width: 40px;
text-align: right;
  }

  .empty-state {     text-align: center;
padding: 4rem 2rem;     background:
rgba(255, 255, 255, 0.95);
backdrop-filter: blur(20px);     border-
radius: 16px;     border: 1px solid
rgba(255, 255, 255, 0.2);     box-
shadow: 0 8px 32px rgba(0, 0, 0, 0.1);
  }

  .empty-
icon {
font-size:
4rem; color:
#4cc9f0;
margin-
bottom:
1rem;
```

```
  }

  .empty-state h3 {
color: #2b2d42;
margin-bottom:
1rem;
  }

  .empty-state p {
color: #6c757d;
margin-bottom:
2rem;
  }

  .empty-actions {
display: flex;
gap: 1rem;
justify-content:
center;       flex-
wrap: wrap;
  }

  @@media (max-width:
768px) {       .dashboard-
container {          padding:
1rem;
}
    .dashboard-header {          flex-direction: column;          text-align: center;
    }

    .header-stats {
justify-content: center;
    }

    .approval-overview {          grid-
template-columns: 1fr;
    }

    .overview-card {
flex-direction: column;
text-align: center;
    }

    .bulk-actions {
flex-direction: column;
align-items: stretch;
    }

    .claims-grid {          grid-
template-columns: 1fr;
    }
```

```
        .claim-card-footer {
flex-direction: column;
align-items: stretch;
        }

        .claim-actions {
justify-content: center;
        }

        .empty-actions {
flex-direction: column;
        }
    }
</style>

<script>    function updateSelection() {        const selectedClaims =
document.querySelectorAll('.claim-select:checked');        const
selectedCount = selectedClaims.length;

        document.getElementById('selectedCount').textContent = selectedCount;
document.getElementById('rejectCount').textContent = selectedCount;

        const approveBtn = document.getElementById('approveSelectedBtn');
const rejectBtn = document.getElementById('rejectSelectedBtn');

        if (selectedCount > 0) {
approveBtn.disabled = false;
rejectBtn.disabled = false;
```

```
      } else {
        approveBtn.disabled = true;
rejectBtn.disabled = true;
      }
    }

    function selectAllClaims() {       const checkboxes =
document.querySelectorAll('.claim-select');
checkboxes.forEach(checkbox => {          checkbox.checked =
true;
      });
      updateSelection();
    }

    function approveSelected() {       const selectedClaims =
document.querySelectorAll('.claim-select:checked');       if
(selectedClaims.length === 0) return;

      if (confirm(`Are you sure you want to approve ${selectedClaims.length}
claims?`)) {
        // Here you would typically submit bulk approval
selectedClaims.forEach(checkbox => {             const claimId =
checkbox.closest('.claim-card').dataset.claimId;
console.log(`Approving claim ${claimId}`);
        });
        alert(`${selectedClaims.length} claims approved successfully!`);
```

```
            // Reset selection            document.querySelectorAll('.claim-
select').forEach(cb => cb.checked = false);            updateSelection();
        }
    }

    function rejectSelected() {        const selectedClaims =
document.querySelectorAll('.claim-select:checked');        if
(selectedClaims.length === 0) return;

        if (confirm(`Are you sure you want to reject ${selectedClaims.length} claims?`)) {
            // Here you would typically submit bulk rejection
selectedClaims.forEach(checkbox => {                const claimId =
checkbox.closest('.claim-card').dataset.claimId;
console.log(`Rejecting claim ${claimId}`);
            });
            alert(`${selectedClaims.length} claims rejected.`);

            // Reset selection
            document.querySelectorAll('.claim-select').forEach(cb => cb.checked = false);
updateSelection();
        }
    }
</script>
```

Review.cshtml

```
@model CMCS.Mvc.Models.Claim
@{
    Layout = "_Layout";
    ViewData["Title"] = "Review Claim";
}

<h2>Review Claim</h2>

<section class="card">
  <div class="claim-details">
    <h3>@Model.Title</h3>

    <dl class="details-grid">
      <dt>Lecturer Name:</dt>
      <dd><strong>@Model.LecturerName</strong></dd>

      <dt>Description:</dt>
      <dd>@Model.Description</dd>

      <dt>Hours Worked:</dt>
      <dd>@Model.Hours</dd>

      <dt>Hourly Rate:</dt>
      <dd>@Model.Rate.ToString("C")</dd>

      <dt>Total Amount:</dt>
      <dd><strong>@Model.Amount.ToString("C")</strong></dd>

      <dt>Current Status:</dt>
      <dd><strong>@Model.Status</strong></dd>

      @if (!string.IsNullOrEmpty(Model.FilePath))
```

```
            {
                <dt>Supporting Document:</dt>
                <dd>
                    <a href="@Url.Action("Download", "Claims", new { id = Model.Id })" class="btnsecondary" target="_blank">
                        View Document
                    </a>
                </dd>
            }
        </dl>
    </div>

    <div class="button-row" style="margin-top: 2rem; border-top: 1px solid #eee; paddingtop: 1rem;">
        <form method="post" action="@Url.Action("Approve", "AcademicManager", new { id = Model.Id })" style="display: inline;">
            @Html.AntiForgeryToken()
            <button type="submit" class="btn-primary" onclick="return confirm('Are you sure you want to APPROVE this claim?')">
                Approve Claim
            </button>
        </form>

        <form method="post" action="@Url.Action("Reject", "AcademicManager", new { id = Model.Id })" style="display: inline;">
            @Html.AntiForgeryToken()
            <button type="submit" class="btn-danger" onclick="return confirm('Are you sure you want to REJECT this claim?')">
                Reject Claim
            </button>
```

```
      </form>


      <a href="@Url.Action("Index")" class="btn-secondary">Back to Pending Claims</a>
   </div>
</section>
```

Claims (Folder)

Create.cshtml

@model CMCS.Mvc.Models.Claim @{ Layout = "_Layout"; ViewData["Title"] = "Submit New Claim"; }

Submit New Claim

Create a new claim for your work hours

Back to My Claims

```
<!-- Main Form Card -->
<div class="card form-card">
    <div class="card-header">
        <h3><i class="fas fa-file-medical"></i> Claim Details</h3>
        <div class="form-progress">
            <div class="progress-step active">
                <div class="step-number">1</div>
                <span>Basic Info</span>
</div>
            <div class="progress-step">
                <div class="step-number">2</div>
                <span>Hours & Amount</span>
            </div>
            <div class="progress-step">
                <div class="step-number">3</div>
                <span>Documentation</span>
            </div>
        </div>
</div>
```

```
    <div class="card-body">
        <form method="post" enctype="multipart/form-data"
class="modern-form">
            @Html.AntiForgeryToken()

            <!-- Success/Error Messages -->
            @if (TempData["Success"] != null)
            {
                <div class="alert alert-success">
                    <i class="fas fa-check-circle"></i>
@TempData["Success"]

</div>
}

            @if (!ViewData.ModelState.IsValid)
            {
                <div class="alert alert-danger">
                    <i class="fas fa-exclamation-triangle"></i>
                    <strong>Please fix the following
errors:</strong>
                    <div class="validation-errors">
                        @Html.ValidationSummary()
                    </div>
                </div>
            }

            <!-- Basic Information Section -->
            <div class="form-section active">
                <div class="section-header">
                    <h4><i class="fas fa-info-circle"></i> Basic
Information</h4>
                    <p>Provide the essential details about your
claim</p>
                </div>

                <div class="form-group">
                    <label asp-for="Title" class="form-label">
```

```
                        <i class="fas fa-heading"></i> Claim
Title *
                    </label>
                    <input asp-for="Title" class="form-control
moderninput"                           placeholder="e.g., August
Research Hours,
Course Development, Student Supervision" />
<span asp-validation-for="Title"
class="validation-message"></span>
                    <div class="form-hint">Be specific about the
work performed</div>                </div>

                <div class="form-group">
                    <label asp-for="Description" class="form-
label">
                        <i class="fas fa-align-left"></i>
Description
                    </label>
                    <textarea asp-for="Description"
class="formcontrol modern-textarea"
rows="4" placeholder="Provide detailed information about the
work completed, objectives achieved, and any relevant
context..."></textarea>
                    <span asp-validation-for="Description"
class="validation-message"></span>
                    <div class="form-hint">Optional but
recommended for better clarity</div>
                </div>
            </div>

            <!-- Hours & Amount Section -->
            <div class="form-section">
                <div class="section-header">
                    <h4><i class="fas fa-calculator"></i> Hours &
Compensation</h4>
                    <p>Enter your worked hours and
review the calculated amount</p>                </div>

                <div class="form-row">
                    <div class="form-group">
```

```
                        <label asp-for="Hours" class="form-
label">
                            <i class="fas fa-clock"></i> Hours
Worked
*
                        </label>
                        <input asp-for="Hours" type="number"
class="form-control modern-input"
min="1" max="180" value="@(Model.Hours
> 0 ? Model.Hours : "")"
placeholder="Enter hours" />
<span asp-validation-for="Hours" class="validation-
message"></span>
                        <div class="form-hint">Maximum 180 hours
per month</div>
                    </div>

                    <div class="form-group">
                        <label class="form-label">
                            <i class="fas fa-dollar-sign"></i>
Hourly Rate
                        </label>
                        <div class="rate-display">
                            <input type="text" class="form-
control modern-input"
                                   value="@Model.Rate.ToString("C")"
readonly />
                            <div class="rate-badge">Set by
HR</div>
                        </div>
                        <div class="form-hint">Your predefined
hourly rate</div>
                    </div>

                    <div class="form-group">
                        <label class="form-label">
                            <i class="fas fa-calculator"></i>
Total Amount
                        </label>
                        <div class="amount-display">
```

```
                            <input type="text" class="form-
control modern-input"
id="calculatedAmount" readonly
placeholder="Will calculate automatically" />
                            <div class="amount-badge"
id="amountBadge">Calculated</div>
                        </div>
                        <div class="form-hint">Hours × Hourly
Rate</div>
                    </div>
</div>

                <div class="amount-preview">
                    <div class="preview-card">
                        <div class="preview-icon">
                            <i class="fas fa-receipt"></i>
                        </div>
                        <div class="preview-content">
                            <h5>Amount Preview</h5>
                            <div class="preview-
details">                               <span
class="previewlabel">Hours:</span>
                                <span class="preview-value"
id="previewHours">0</span>
                            </div>
                            <div class="preview-
details">                               <span
class="previewlabel">Rate:</span>
                                <span
class="previewvalue">@Model.Rate.ToString("C")/hr</span>
                            </div>
                            <div class="preview-details total">
                                <span
class="previewlabel">Total:</span>
                                <span class="preview-
value" id="previewTotal">R 0.00</span>
</div>
                        </div>
                    </div>
                </div>
</div>

            <!-- Documentation Section -->
            <div class="form-section">
                <div class="section-header">
                    <h4><i class="fas fa-paperclip"></i>
Supporting Documentation</h4>
                    <p>Upload relevant documents to support your
claim
(optional)</p>
</div>

                <div class="form-group">
                    <label for="file" class="form-label">
                        <i class="fas fa-upload"></i>
Upload Supporting Document                    </label>

                    <div class="file-upload-area"
id="fileUploadArea">
                        <div class="upload-placeholder">
                            <i class="fas fa-cloud-upload-
alt"></i>                         <h5>Drop files here or click
to browse</h5>
                            <p>Supports: PDF, Word (.docx, .doc),
Excel (.xlsx, .xls)</p>
                            <p class="file-size">Maximum file
size:
10MB</p>
                        </div>
                        <input type="file" id="file"
name="file"                              class="file-
input" accept=".pdf,.docx,.xlsx,.doc,.xls" />
                    </div>

                    <div class="file-preview" id="filePreview">
                        <!-- File preview will appear here -->
                    </div>
                    <div class="upload-tips">
                        <h6><i class="fas fa-lightbulb"></i>
Upload Tips:</h6>
                        <ul>
                            <li>Timesheets, signed agreements, or
work reports are recommended</li>
                            <li>Ensure documents clearly show
hours worked and work description</li>
                            <li>Combine multiple pages into a
single PDF when possible</li>
                        </ul>
                    </div>
                </div>
</div>

            <!-- Form Actions -->
            <div class="form-actions">
                <div class="action-buttons">
                    <button type="button" class="btn btn-
secondary btn-prev" id="prevBtn" disabled>
                        <i class="fas fa-arrow-left"></i>
Previous
                    </button>

                    <button type="button" class="btn btn-primary
btnnext" id="nextBtn">
                        Next <i class="fas fa-arrow-right"></i>
                    </button>

                    <button type="submit" class="btn btn-success
btnsubmit" id="submitBtn">
                        <i class="fas fa-paper-plane"></i> Submit
Claim
                    </button>
</div>

                <div class="form-note">
                    <i class="fas fa-info-circle"></i>
                    Your claim will be reviewed by the Programme
Coordinator before approval
```

```
                </div>
            </div>
        </form>
    </div>
</div>

<!-- Quick Help Card -->
<div class="card help-card">
    <div class="card-header">
        <h4><i class="fas fa-question-circle"></i> Need
Help?</h4>    </div>
    <div class="card-body">
        <div class="help-items">
            <div class="help-item">
                <i class="fas fa-clock"></i>
                <div>
                    <strong>Hour Limits</strong>
                    <p>Maximum 180 hours per month across all
claims</p>
                </div>
            </div>
            <div class="help-item">
                <i class="fas fa-file-invoice"></i>
                <div>
                    <strong>Documentation</strong>
                    <p>Supporting documents increase approval
chances</p>
                </div>
            </div>
            <div class="help-item">
                <i class="fas fa-history"></i>
                <div>
                    <strong>Processing Time</strong>
                    <p>Claims typically processed within 3-5
business days</p>
                </div>
            </div>
        </div>
    </div>
</div>

@section Scripts {
```

Delete.cshtml

```
@model CMCS.Mvc.Models.Claim
@{
    Layout = "_Layout";
    ViewData["Title"] = "Delete Claim";
}

<h2>Delete Claim</h2>

<section class="card">
    <h3>Are you sure you want to delete this claim?</h3>

    <dl class="details-grid">
        <dt>Lecturer Name:</dt>
        <dd>@Model.LecturerName</dd>

        <dt>Title:</dt>
        <dd>@Model.Title</dd>

        <dt>Description:</dt>
        <dd>@(string.IsNullOrEmpty(Model.Description) ? "No description provided" : Model.Description)</dd>

        <dt>Hours Worked:</dt>
        <dd>@Model.Hours</dd>

        <dt>Hourly Rate:</dt>
        <dd>@Model.Rate.ToString("C")</dd>

        <dt>Total Amount:</dt>
        <dd><strong>@Model.Amount.ToString("C")</strong></dd>

        <dt>Status:</dt>
        <dd>@Model.Status</dd>
    </dl>

    <form asp-action="Delete" method="post">
        @Html.AntiForgeryToken()
        <input type="hidden" name="id" value="@Model.Id" />
        <div class="button-row">
          <button type="submit" class="btn-danger">Delete Claim</button>
          <a href="@Url.Action("Index")" class="btn-secondary">Cancel</a>
        </div>
    </form>
</section>
```

Detail.cshtml

```
@model CMCS.Mvc.Models.Claim
@{
  Layout = "_Layout";
  ViewData["Title"] = "Claim Details";
}
```

```
<h2>Claim Details</h2>


<section class="card">
  <dl class="details-grid">
    <dt>ID:</dt>
    <dd>@Model.Id</dd>


    <dt>Lecturer Name:</dt>
    <dd>@Model.LecturerName</dd>


    <dt>Title:</dt>
    <dd>@Model.Title</dd>


    <dt>Description:</dt>
    <dd>@(string.IsNullOrEmpty(Model.Description) ? "No description provided" :
Model.Description)</dd>


    <dt>Hours Worked:</dt>
    <dd>@Model.Hours</dd>


    <dt>Hourly Rate:</dt>
    <dd>@Model.Rate.ToString("C")</dd>


    <dt>Total Amount:</dt>
    <dd><strong>@Model.Amount.ToString("C")</strong></dd>


    <dt>Status:</dt>
    <dd>
```

```
      <span style="padding: 4px 8px; border-radius: 4px;
background-color: @(Model.Status == "Pending" ? "#fff3cd" :
                            Model.Status == "Approved" ? "#d1edff" :
                            Model.Status == "Rejected" ? "#f8d7da" : "#e2e3e5");
color: @(Model.Status == "Pending" ? "#856404" :
                      Model.Status == "Approved" ? "#004085" :
                      Model.Status == "Rejected" ? "#721c24" : "#383d41");">
        @Model.Status
      </span>
    </dd>

    @if (!string.IsNullOrEmpty(Model.FilePath))
    {
      <dt>Supporting Document:</dt>
      <dd>
        <a href="@Url.Action("Download", new { id = Model.Id })" class="btn-primary" target="_blank">
          Download Document
        </a>
      </dd>
    }
  </dl>

  <div class="button-row">
    <a href="@Url.Action("Edit", new { id = Model.Id })" class="btn-primary">Edit</a>
    <a href="@Url.Action("Index")" class="btn-secondary">Back to List</a>
  </div>
</section>
```

Edit.cshtml

```
@model CMCS.Mvc.Models.Claim

@{
    Layout = "_Layout";
    ViewData["Title"] = "Edit Claim";
}

<h2>Edit Claim</h2>

<section class="card">
    <form method="post" enctype="multipart/form-data" class="form-grid"> @Html.AntiForgeryToken()
        <input type="hidden" name="id" value="@Model.Id" />

        @if (!ViewData.ModelState.IsValid)
        {
            <div class="text-danger">
                @Html.ValidationSummary()
            </div>
        }

        <div>
            <label for="UserId">Lecturer *</label>
            <select id="UserId" name="userId" class="form-control" required>
                <option value="">Select Lecturer</option>
                @foreach (var user in ViewBag.Users)
                {
```

```
          <option value="@user.Id" selected="@(user.Id ==
Model.UserId)">@user.Name
@user.Surname</option>

        }

      </select>

    </div>



    <div>

      <label for="Title">Claim Title *</label>

      <input type="text" id="Title" name="title" class="form-control"
value="@Model.Title" required />

    </div>



    <div>

      <label for="Description">Description (Optional)</label>

      <textarea id="Description" name="description"
class="formcontrol">@Model.Description</textarea>

    </div>



    <div>

      <label for="Hours">Hours Worked (Optional)</label>

      <input type="number" id="Hours" name="hours" class="form-control" min="0"
value="@Model.Hours" />

    </div>



    <div>

      <label for="Rate">Hourly Rate (Optional)</label>

      <input type="number" id="Rate" name="rate" class="form-control" min="0"
step="0.01" value="@Model.Rate" />

    </div>
    <div>
      <label for="file">Supporting Document (Optional)</label>
      <input type="file" id="file" name="file" class="form-control"
accept=".pdf,.docx,.xlsx,.doc,.xls" />
      <small class="text-muted">
        Allowed file types: PDF, Word (.docx, .doc), Excel (.xlsx, .xls). Maximum file
size:
10MB.
      </small>
      <div id="fileNameDisplay" style="margin-top: 5px; font-style: italic;"></div>

      @if (!string.IsNullOrEmpty(Model.FilePath))
      {
        <small>Current file: <a href="@Url.Action("Download", new { id = Model.Id })"
target="_blank">View Document</a></small>
      }
    </div>

    <div class="button-row">
      <button type="submit" class="btn-primary">Save Changes</button>
      <a href="@Url.Action("Index")" class="btn-secondary">Cancel</a>
    </div>
  </form>
</section>

<script>
document.getElementById('file').addEventListener('change',
function(e) {     var fileNameDisplay =
document.getElementById('fileNameDisplay');       if (this.files.length
> 0) {          fileNameDisplay.textContent = 'Selected file: ' +
this.files[0].name;
      } else {
         fileNameDisplay.textContent = '';
      }
   });
</script>
```

Index.cshtml

@model List<CMCS.Mvc.Models.Claim> @{ Layout = "_Layout"; ViewData["Title"] = "All Claims"; }

All Claims

Comprehensive view of all submitted claims across the system

Submit New Claim
@if (Model.Any()) {

Total Claims @Model.Count
Pending @Model.Count(c => c.Status == "Pending")
Verified @Model.Count(c => c.Status == "Verified")
Approved @Model.Count(c => c.Status == "Approved")
Rejected @Model.Count(c => c.Status == "Rejected")

```
   <div class="table-container">
        <table class="styled-table">
            <thead>
                <tr>
                    <th>Lecturer</th>
                    <th>Department</th>
                    <th>Title</th>
                    <th>Hours</th>
                    <th>Amount</th>
                    <th>Status</th>
                    <th>Submitted</th>
                    <th>Actions</th>
                </tr>
            </thead>
            <tbody>
                @foreach (var claim in Model.OrderByDescending(c
=> c.CreatedDate))
                {
                    <tr>
                        <td>
                            <div class="lecturer-info">
                                <strong>@claim.User.Name
@claim.User.Surname</strong>
                                <small>@claim.User.Email</small>
                            </div>
                        </td>
                        <td>@claim.User.Department</td>
                        <td>
                            <div class="claim-title">
                                <strong>@claim.Title</strong>
                                @if
(!string.IsNullOrEmpty(claim.Description) && claim.Description !=
"No description provided")
                                {
                                    <small
class="textmuted">@claim.Description</small>
}
                        </div>
                    </td>
                    <td>@claim.Hours</td>
                    <td class="amount-cell">

<strong>@claim.Amount.ToString("C")</strong>

<small>@claim.Rate.ToString("C")/hr</small>
                    </td>
                    <td>
                        <span class="status-badge
@claim.Status.ToLower()">
                            <i class="fas
                                        @(claim.Status ==
"Pending" ?
"fa-clock" :
                     claim.Status == "Verified" ? "fa-check-circle" :
                        claim.Status == "Approved" ? "fa-thumbs-up" :
        claim.Status == "Rejected" ? "fa-times-circle" : "fa-file")">
                        </i>
                        @claim.Status
                    </span>
                </td>
                <td>@claim.CreatedDate.ToString("MMM dd,
yyyy")</td>
                <td>
                    <div class="action-buttons">
                        <a href="@Url.Action("Details", new { id
= claim.Id })" class="btn-action btn-info" title="View Details">
                            <i class="fas fa-eye"></i>
                        </a>
                        <a href="@Url.Action("Edit", new { id =
claim.Id })" class="btn-action btn-edit" title="Edit Claim">
                            <i class="fas fa-edit"></i>
                        </a>
                        <a href="@Url.Action("Delete", new { id =
claim.Id })" class="btn-action btn-delete" title="Delete Claim">
                            <i class="fas fa-trash"></i>
                        </a>
                        @if
(!string.IsNullOrEmpty(claim.FilePath))
                                {
                                    <a
href="@Url.Action("Download", new { id = claim.Id })" class="btn-
action btn-download" title="Download Document" target="_blank">
                                        <i class="fas
fadownload"></i>
                                    </a>
                                }
                            </div>
                        </td>
                    </tr>
                }
            </tbody>
        </table>
    </div>
}
else
{
    <div class="empty-state">
        <div class="empty-icon">
            <i class="fas fa-file-invoice-dollar"></i>
        </div>
        <h3>No Claims Found</h3>
        <p>There are no claims in the system yet. Be the first to submit a claim.</p>
        <a href="@Url.Action("Create")" class="btn-primary">
            <i class="fas fa-plus-circle"></i> Submit First Claim
        </a>
    </div>
}
```

Home (Folder)

Claims.cshml

```
@{
  Layout = "_Layout";
}
<h2>Claims</h2>

<section class="card">
  <div class="button-row">
    <button type="button" class="btn-primary">Add Claim</button>
  </div>
  <table class="styled-table">
    <thead>
```

```
      <tr>
        <th>Lecturer</th>
        <th>Month</th>
        <th>Amount</th>
        <th>Status</th>
        <th>Actions</th>
      </tr>
    </thead>
    <tbody>
      <tr>
        <td>Jane Doe</td>
        <td>August 2025</td>
        <td>R 4,500.00</td>
        <td>Submitted</td>
        <td><button class="btn-secondary">Verify</button> <button class="btnprimary">Send to Manager</button></td>
      </tr>
      <tr>
        <td>John Smith</td>
        <td>August 2025</td>
        <td>R 3,200.00</td>
        <td>Verified</td>
        <td><button class="btn-primary">Send to Manager</button></td>
      </tr>
    </tbody>
  </table>
</section>
```

Index.cshtml

```
@{
    Layout = "_Layout";
    ViewData["Title"] = "CMCS Dashboard";
}

<div class="dashboard">
    <!-- Hero Section -->
    <section class="hero-section">
        <div class="hero-content">
            <h1 class="hero-title">Contract Monthly Claim System</h1>
            <p class="hero-subtitle">Streamlining claim management for academic institutions</p>
            <div class="hero-stats">
                <div class="stat-card">
                    <div class="stat-number" id="totalClaims">0</div>
                    <div class="stat-label">Total Claims</div>
                </div>
                <div class="stat-card">
                    <div class="stat-number" id="pendingClaims">0</div>
                    <div class="stat-label">Pending Review</div>
                </div>
                <div class="stat-card">
                    <div class="stat-number" id="approvedClaims">0</div>
                    <div class="stat-label">Approved</div>
                </div>
            </div>
        </div>
    </section>
```

```
<!-- Quick Actions Section -->
<section class="quick-actions">
  <h2>Quick Actions</h2>
  <div class="action-grid">
    <a href="/Claims/Create" class="action-card">
      <div class="action-icon">??</div>
      <h3>Submit New Claim</h3>
      <p>Submit a monthly claim with supporting documents</p>
      <div class="action-arrow">?</div>
    </a>

    <a href="/Lecturers/Create" class="action-card">
      <div class="action-icon">?????</div>
      <h3>Add New Lecturer</h3>
      <p>Register a new lecturer in the system</p>
      <div class="action-arrow">?</div>
    </a>

    <a href="/Claims" class="action-card">
      <div class="action-icon">??</div>
      <h3>View All Claims</h3>
      <p>Browse and manage all submitted claims</p>
      <div class="action-arrow">?</div>
    </a>

    <a href="/AcademicManager" class="action-card">
      <div class="action-icon">?</div>
```

```
            <h3>Review Claims</h3>
            <p>Approve or reject pending claims</p>
            <div class="action-arrow">?</div>
          </a>
        </div>
      </section>

      <!-- Recent Activity Section -->
      <section class="recent-activity">
        <h2>Recent Activity</h2>
        <div class="activity-list" id="recentClaims">
          <div class="loading">Loading recent claims...</div>
        </div>
      </section>

      <!-- System Info Section -->
      <section class="system-info">
        <div class="info-grid">
          <div class="info-card">
            <h3>?? System Status</h3>
            <p>All systems operational</p>
            <span class="status-badge online">Online</span>
          </div>
          <div class="info-card">
            <h3>?? Response Time</h3>
            <p>Less than 2 seconds</p>
            <span class="status-badge excellent">Excellent</span>          </div>
          <div class="info-card">
            <h3>?? Security</h3>
            <p>Fully secured platform</p>
            <span class="status-badge secure">Secure</span>
          </div>
        </div>
      </section>
    </div>

    @section Scripts {
      <script>
    document.addEventListener('DOMContentLoaded',
    function() {          setTimeout(() => {
            document.getElementById('totalClaims').textContent = '24';
    document.getElementById('pendingClaims').textContent = '8';
    document.getElementById('approvedClaims').textContent = '16';

            const recentClaims = [
              { lecturer: 'Dr. Sarah Johnson', title: 'August Research Hours', status:
    'Pending', date: '2 hours ago' },
              { lecturer: 'Prof. Michael Chen', title: 'Course Development', status:
    'Approved', date: '1 day ago' },
              { lecturer: 'Dr. Emily Davis', title: 'Student Supervision', status: 'Rejected',
    date: '2 days ago' },
              { lecturer: 'Prof. Robert Wilson', title: 'Conference Attendance', status:
    'Approved', date: '3 days ago' }
            ];

            const activityList = document.getElementById('recentClaims');
    activityList.innerHTML = recentClaims.map(claim => `
              <div class="activity-item">
                <div class="activity-content">
                  <strong>${claim.lecturer}</strong> - ${claim.title}
                  <span class="activity-date">${claim.date}</span>
                </div>
                <span class="status-badge ${claim.status.toLowerCase()}">${claim.status}</span>
              </div>
            ` ).join('');
          }, 1000);
        });
      </script>
    }
```

LecturerSubmit.cshtml

```
@{
  Layout = "_Layout";
}
<h2>Lecturer - Submit Claim</h2>

<section class="card">
  <form class="form-grid">
    <label>
      Lecturer Name
      <input placeholder="Full name" />
    </label>
    <label>
      Month
      <input placeholder="e.g. August 2025" />
```

```
        </label>
        <label>
          Hours Worked
          <input placeholder="Number of hours" />
        </label>
        <label>
          Hourly Rate
          <input placeholder="Rate per hour" />
        </label>
        <label>
          Claim Amount
          <input placeholder="Calculated amount" readonly />
        </label>
        <div class="button-row">
          <button type="button" class="btn-secondary">Attach Supporting
Docs</button>          <button type="button" class="btn-primary">Submit
Claim</button>
        </div>
    </form>
</section>
```

ManagerApprove.cshtml

```
@{
    Layout = "_Layout";
}
<h2>Academic Manager - Approve Claims</h2>

<section class="card">
```

```
  <div class="claim">
    <h4>Jane Doe — August 2025</h4>
    <p><strong>Amount:</strong> R 4,500.00</p>
    <p><strong>Supporting Docs:</strong> Attached</p>
    <div class="button-row">
      <button class="btn-primary">Approve</button>
      <button class="btn-danger">Reject</button>
    </div>
  </div>
</section>
```

StatusTracking.cshtml

```
@{
  Layout = "_Layout";
}
<h2>Claim Status Tracking</h2>

<section class="card">
  <ol class="status-list">
    <li>Submitted — <strong>Aug 5, 2025</strong></li>
    <li>Verified — <strong>Aug 7, 2025</strong></li>
    <li>Approved — <strong>Aug 9, 2025</strong></li>
    <li>Settled — <strong>Aug 12, 2025</strong></li>
  </ol>
</section>
```

UploadProof.cshtml

```
@{
    Layout = "_Layout";
}
<h2>Upload Supporting Documents</h2>

<section class="card">
    <form enctype="multipart/form-data" class="form-grid">
        <label>
            Select file
            <input type="file" />
        </label>
        <div class="button-row">
            <button type="button" class="btn-primary">Attach</button>
        </div>
    </form>
</section>
```

HR (Folder)

Create.cshtml

```
@model List<CMCS.Mvc.Models.User>
@{
    Layout = "_Layout";
    ViewData["Title"] = "HR - User Management";
}

<div class="page-header">
    <div class="header-content">
        <h1>User Management</h1>
        <p>Manage all system users and their roles</p>
```

```
  </div>
  <div class="header-actions">
    <a href="@Url.Action("Create")" class="btn-primary">
      <i class="fas fa-user-plus"></i> Add New User
    </a>
    <a href="@Url.Action("Reports")" class="btn-secondary">
      <i class="fas fa-chart-bar"></i> Generate Reports
    </a>
  </div>
</div>

<section class="card">
  @if (Model.Any())
  {
    <div class="users-summary">
      <div class="summary-item">
        <span class="summary-label">Total Users</span>
        <span class="summary-value">@Model.Count</span>
      </div>
      <div class="summary-item">
        <span class="summary-label">Active Users</span>
        <span class="summary-value status-approved">@Model.Count(u =>
u.IsActive)</span>
      </div>
      <div class="summary-item">
        <span class="summary-label">Lecturers</span>
        <span class="summary-value" style="color: #4361ee;">@Model.Count(u =>
u.Role == "Lecturer")</span>
      </div>
      <div class="summary-item">
        <span class="summary-label">HR Users</span>
        <span class="summary-value" style="color: #7209b7;">@Model.Count(u =>
u.Role == "HR")</span>
      </div>
    </div>

    <div class="table-container">
      <table class="styled-table">
        <thead>
          <tr>
            <th>User</th>
            <th>Role</th>
            <th>Department</th>
            <th>Hourly Rate</th>
            <th>Email</th>
            <th>Status</th>
            <th>Actions</th>
          </tr>
        </thead>
        <tbody>
          @foreach (var user in Model.OrderBy(u => u.Role).ThenBy(u => u.Name))
          {
            <tr>
              <td>
                <div class="user-info">
                  <strong>@user.Name @user.Surname</strong>
                  <small>@user.Username</small>
                </div>
              </td>
              <td>
                <span class="role-badge @user.Role.ToLower()">
                  <i class="fas
                        @(user.Role == "HR" ? "fa-users" :
user.Role == "Lecturer" ? "fa-chalkboard-teacher" :
user.Role == "ProgrammeCoordinator" ? "fa-tasks" :
                                        user.Role == "AcademicManager" ? "fa-user-
tie" :
"fa-user")">
              </i>
              @user.Role
            </span>
          </td>
          <td>@user.Department</td>
          <td class="amount-cell">
            @if (user.Role == "Lecturer")
                {
                  <strong>@user.HourlyRate.ToString("C")</strong>
<small>per hour</small>

}
else
                {
                  <span class="text-muted">-</span>
                }
              </td>
```

```
                            <td>
                                <div class="email-cell">
                                    <i class="fas fa-envelope"></i>
                                    @user.Email
                                </div>
                            </td>
                            <td>
                                <span class="status-badge @(user.IsActive ? "active" : "inactive")">
                                    <i class="fas @(user.IsActive ? "fa-check-circle" : "fa-times-circle")"></i>
                                    @(user.IsActive ? "Active" : "Inactive")
                                </span>
                            </td>
                            <td>
                                <div class="action-buttons">
                                    <a href="@Url.Action("Edit", new { id = user.Id })" class="btn-action btnedit" title="Edit User">
                                        <i class="fas fa-edit"></i>
                                    </a>
                                    @if (user.Role != "HR")
                                    {
                                        <a href="@Url.Action("Delete", new { id = user.Id })" class="btn-action btn-delete" title="Delete User">
                                            <i class="fas fa-trash"></i>
                                        </a>
                                    }
                                </div>
                            </td>
                        </tr>
```

```
          }
        </tbody>
      </table>
    </div>
  }
  else
  {
    <div class="empty-state">
      <div class="empty-icon">
        <i class="fas fa-users"></i>
      </div>
      <h3>No Users Found</h3>
      <p>There are no users in the system yet. Create the first user to get started.</p>
      <a href="@Url.Action("Create")" class="btn-primary">
        <i class="fas fa-user-plus"></i> Create First User
      </a>
    </div>
  }
</section>

<style>
  .header-
actions {
display: flex;
gap: 1rem;
align-items:
center;
  }
  .users-summary {
    display: grid;
    grid-template-columns: repeat(auto-fit,
minmax(150px, 1fr));      gap: 1rem;      margin-bottom:
2rem;      padding: 1.5rem;      background: rgba(102, 126,
234, 0.05);
    border-radius: 12px;
    border: 1px solid rgba(102, 126, 234, 0.1);
  }

  .user-info {
min-width: 150px;
  }

    .user-info strong {
display: block;
margin-bottom: 0.25rem;
    }

    .user-info small
{        color:
#6c757d;
font-size: 0.875rem;
    }
  .role-badge {
display: inline-flex;
align-items: center;
gap: 0.5rem;
padding: 0.5rem
1rem;      border-
radius: 50px;
font-size: 0.875rem;
font-weight: 600;
text-transform:
uppercase;
letter-spacing:
0.5px;
  }

    .role-badge.hr {       background:
rgba(114, 9, 183, 0.2);       color:
#7209b7;       border: 1px solid
rgba(114, 9, 183, 0.3);
    }

    .role-badge.lecturer {
background: rgba(67, 97, 238, 0.2);
color: #4361ee;       border: 1px solid
rgba(67, 97, 238, 0.3);
    }
```

```
      .role-badge.programmecoordinator
{          background: rgba(248, 150, 30,
0.2);          color: #f8961e;          border:
1px solid rgba(248, 150, 30, 0.3);
      }

      .role-badge.academicmanager {
background: rgba(76, 201, 240, 0.2);
color: #4cc9f0;          border: 1px solid
rgba(76, 201, 240, 0.3); }


   .email-cell {
display: flex;
align-items:
center;       gap:
0.5rem;
color: #6c757d;
   }

   .status-badge.active {
background: rgba(46, 204, 113, 0.2);
color: #27ae60;       border: 1px solid
rgba(46, 204, 113, 0.3);
   }

   .status-badge.inactive {
background: rgba(108, 117, 125, 0.2);
color: #6c757d;        border: 1px solid
rgba(108, 117, 125, 0.3);
  }

  @@media (max-width: 768px) {
      .header-actions {
flex-direction: column;
align-items: stretch;
width: 100%;
      }

          .header-
actions a {
width: 100%;
text-align: center;
          }

      .users-summary {          grid-
template-columns: repeat(2, 1fr);
      }
  }
</style>
```

Edit.cshtml

```
@model CMCS.Mvc.Models.User
@{
  Layout = "_Layout";
  ViewData["Title"] = "Edit User";
}

<h2>Edit User</h2>

<section class="card">
  <form method="post" class="form-grid">
    @Html.AntiForgeryToken()
    <input type="hidden" asp-for="Id" /> <div>
      <label for="Username">Username *</label>
      <input type="text" id="Username" name="Username" class="form-control" value="@Model.Username" required />
    </div>

    <div>
      <label for="Password">Password *</label>
      <input type="password" id="Password" name="Password" class="form-control" value="@Model.Password" required />
    </div>

    <div>
      <label for="Role">Role *</label>
      <select id="Role" name="Role" class="form-control" required>
        <option value="HR" selected="@(Model.Role == "HR")">HR</option>
        <option value="Lecturer" selected="@(Model.Role == "Lecturer")">Lecturer</option>
        <option value="ProgrammeCoordinator" selected="@(Model.Role == "ProgrammeCoordinator")">Programme Coordinator</option>
        <option value="AcademicManager" selected="@(Model.Role == "AcademicManager")">Academic Manager</option>
      </select>
    </div>

    <div>
      <label for="Name">Name *</label>
      <input type="text" id="Name" name="Name" class="form-control" value="@Model.Name" required />
    </div>

    <div>
      <label for="Surname">Surname *</label>
      <input type="text" id="Surname" name="Surname" class="form-control" value="@Model.Surname" required />
    </div>

    <div>
      <label for="Email">Email *</label>
      <input type="email" id="Email" name="Email" class="form-control" value="@Model.Email" required />
    </div>

    <div>
      <label for="HourlyRate">Hourly Rate</label>
      <input type="number" id="HourlyRate" name="HourlyRate" class="form-control" min="0" step="0.01" value="@Model.HourlyRate" />
    </div>

    <div>
      <label for="Department">Department</label>
      <input type="text" id="Department" name="Department" class="form-control" value="@Model.Department" />
    </div>

    <div>
      <label for="IsActive">Active Status</label>
      <select id="IsActive" name="IsActive" class="form-control">
        <option value="true" selected="@Model.IsActive">Active</option>
        <option value="false" selected="@(!Model.IsActive)">Inactive</option>
      </select>
    </div>

    <div class="button-row">
      <button type="submit" class="btn-primary">Save Changes</button>
      <a href="@Url.Action("Index")" class="btn-secondary">Cancel</a>
    </div>
  </form>
</section>
```

Index.cshtml

```
@model List<CMCS.Mvc.Models.User>
@{
  Layout = "_Layout";
  ViewData["Title"] = "HR Dashboard";
}

<div class="dashboard-container">
  <!-- Header Section -->
  <div class="dashboard-header">
```

```
    <div class="header-content">
        <h1 class="page-title">HR Dashboard</h1>
        <p class="page-subtitle">Manage users and system analytics</p>
    </div>
    <div class="header-actions">
        <a href="@Url.Action("Create")" class="btn btn-primary btn-icon">
            <i class="fas fa-user-plus"></i>
            Add User
        </a>
        <a href="@Url.Action("Reports")" class="btn btn-secondary btn-icon">
            <i class="fas fa-chart-bar"></i>
            Reports
        </a>
    </div>
</div>

<!-- Stats Overview -->
<div class="stats-grid">
    <div class="stat-card">
        <div class="stat-icon">
            <i class="fas fa-users"></i>
        </div>
        <div class="stat-content">
            <h3 class="stat-number">@Model.Count</h3>
            <p class="stat-label">Total Users</p>
        </div>
        <div class="stat-trend">
            <i class="fas fa-arrow-up"></i>
            <span>12%</span>
          </div>
        </div>

        <div class="stat-card">
          <div class="stat-icon">
            <i class="fas fa-chalkboard-teacher"></i>
          </div>
          <div class="stat-content">
            <h3 class="stat-number">@Model.Count(u => u.Role ==
"Lecturer")</h3>                <p class="stat-label">Lecturers</p>
          </div>
          <div class="stat-trend">
            <i class="fas fa-arrow-up"></i>
            <span>8%</span>
          </div>
        </div>

        <div class="stat-card">
          <div class="stat-icon">
            <i class="fas fa-user-check"></i>
          </div>
          <div class="stat-content">
            <h3 class="stat-number">@Model.Count(u => u.IsActive)</h3>
            <p class="stat-label">Active Users</p>
          </div>
          <div class="stat-trend trend-up">
            <i class="fas fa-arrow-up"></i>
            <span>5%</span>
        </div>
    </div>

    <div class="stat-card">
        <div class="stat-icon">
            <i class="fas fa-clock"></i>
        </div>
        <div class="stat-content">
            <h3 class="stat-number">@Model.Count(u => !u.IsActive)</h3>
            <p class="stat-label">Inactive</p>
        </div>
        <div class="stat-trend trend-down">
            <i class="fas fa-arrow-down"></i>
            <span>3%</span>
        </div>
    </div>
</div>

<!-- Role Distribution -->
<div class="card-grid">
    <div class="card">
        <div class="card-header">
            <h3><i class="fas fa-pie-chart"></i> Role Distribution</h3>
        </div>
        <div class="card-body">
            <div class="role-distribution">
                @{
```

```
            var roles = Model.GroupBy(u => u.Role).Select(g => new { Role = g.Key, Count = g.Count() });
          }
          @foreach (var role in roles)
          {
            <div class="role-item">
              <div class="role-info">
                <span class="role-badge @role.Role.ToLower()">
                  <i class="fas @GetRoleIcon(role.Role)"></i>
                  @role.Role
                </span>
                <span class="role-count">@role.Count</span>
              </div>
              <div class="role-bar">
                <div class="role-progress" style="width: @((role.Count * 100) / Model.Count)%; background: @GetRoleColor(role.Role);"></div>
              </div>
            </div>
          }
        </div>
      </div>
    </div>

    <!-- Quick Actions -->
    <div class="card">
      <div class="card-header">
        <h3><i class="fas fa-bolt"></i> Quick Actions</h3>
      </div>
```

```
<div class="card-body">
    <div class="quick-actions">
        <a href="@Url.Action("Create")" class="quick-action">
            <div class="action-icon">
                <i class="fas fa-user-plus"></i>
            </div>
            <span>Add User</span>
        </a>
        <a href="@Url.Action("Reports")" class="quick-action">
            <div class="action-icon">
                <i class="fas fa-file-invoice"></i>
            </div>
            <span>Generate Reports</span>
        </a>
        <a href="/Claims" class="quick-action">
            <div class="action-icon">
                <i class="fas fa-list"></i>
            </div>
            <span>View All Claims</span>
        </a>
        <a href="#" class="quick-action">
            <div class="action-icon">
                <i class="fas fa-download"></i>
            </div>
            <span>Export Data</span>
        </a>
    </div>
</div>
```

```
    </div>
</div>

<!-- Users Table -->
<div class="card">
    <div class="card-header">
        <h3><i class="fas fa-list"></i> User Management</h3>
        <div class="card-actions">
            <input type="text" class="search-input" placeholder="Search users...">
            <button class="btn btn-sm btn-secondary">
                <i class="fas fa-filter"></i> Filter
            </button>
        </div>
    </div>
    <div class="card-body">
        @if (Model.Any())
        {
            <div class="table-responsive">
                <table class="modern-table">
                    <thead>
                        <tr>
                            <th>User</th>
                            <th>Role</th>
                            <th>Department</th>
                            <th>Hourly Rate</th>
                            <th>Status</th>
                            <th>Actions</th>
                        </tr>
            </thead>
            <tbody>
              @foreach (var user in Model.OrderBy(u => u.Role).ThenBy(u =>
u.Name))
              {
                <tr>
                  <td>
                    <div class="user-avatar">
                      <div class="avatar">
                        @user.Name[0]@user.Surname[0]
                      </div>
                      <div class="user-details">
                        <strong>@user.Name @user.Surname</strong>
                        <small>@user.Email</small>
                      </div>
                    </div>
                  </td>
                  <td>
                    <span class="role-tag @user.Role.ToLower()">
                      <i class="fas @GetRoleIcon(user.Role)"></i>
                      @user.Role
                    </span>
                  </td>
                  <td>@user.Department</td>
                  <td>
                    @if (user.Role == "Lecturer")
                    {
                      <span
class="rate">@user.HourlyRate.ToString("C")<small>/hr</small></span>


}
else
                    {
                      <span class="text-muted">-</span>
                    }
                  </td>
                  <td>
                    <span class="status @(user.IsActive ? "active" : "inactive")">
                      <i class="fas fa-circle"></i>
                      @(user.IsActive ? "Active" : "Inactive")
                    </span>
                  </td>
                  <td>
                    <div class="action-buttons">
                      <a href="@Url.Action("Edit", new { id = user.Id })" class="btn-
action btn-edit" title="Edit">
                        <i class="fas fa-edit"></i>
                      </a>
                      @if (user.Role != "HR")
                      {
                        <a href="@Url.Action("Delete", new { id = user.Id })"
class="btnaction btn-delete" title="Delete">
                          <i class="fas fa-trash"></i>
                        </a>
                      }
                    </div>
```

```
                        </td>
                      </tr>
                    }
                  </tbody>
                </table>
              </div>

}
else
          {
              <div class="empty-state">
                <div class="empty-icon">
                  <i class="fas fa-users"></i>
                </div>
                <h3>No Users Found</h3>
                <p>Get started by creating the first user in the system.</p>
                <a href="@Url.Action("Create")" class="btn btn-primary">
                  <i class="fas fa-user-plus"></i> Create First User
                </a>
              </div>
          }
        </div>
    </div>
</div>

@functions {    private string
GetRoleIcon(string role)
    {
        return role switch
        {
            "HR" => "fa-users",
            "Lecturer" => "fa-chalkboard-teacher",
            "ProgrammeCoordinator" => "fa-tasks",
            "AcademicManager" => "fa-user-tie",
            _ => "fa-user"
        };
    }

    private string GetRoleColor(string role)
    {
        return role switch
        {
            "HR" => "#7209b7",
            "Lecturer" => "#4361ee",
            "ProgrammeCoordinator" => "#f8961e",
            "AcademicManager" => "#4cc9f0",
            _ => "#6c757d"
        };
    }
}

<style>
.dashboard-
container {
max-width:
1400px;
```

```
margin: 0 auto;

padding: 2rem;

  }


  .dashboard-header {

    display: flex;     justify-

content: space-between;

align-items: flex-start;

margin-bottom: 2rem;

gap: 2rem;

  }

  .page-title {     font-size: 2.5rem;     font-weight: 800;

background: linear-gradient(135deg, #667eea 0%, #764ba2

100%);

    -webkit-background-clip:

text;     -webkit-text-fill-color:

transparent;     background-

clip: text;     margin: 0;

  }


  .page-subtitle {

color: #6c757d;

font-size: 1.1rem;

margin: 0.5rem 0 0

0;

  }
```

```
   .header-
actions {
display: flex;
gap: 1rem;
align-items:
center;
   }

   .btn-icon {
display: inline-
flex;       align-
items: center;
gap: 0.5rem;
text-decoration:
none;
   }

   .stats-grid {
display: grid;
      grid-template-columns: repeat(auto-fit,
minmax(280px, 1fr));       gap: 1.5rem;       margin-bottom:
2rem;
   }

   .stat-card {       background:
rgba(255, 255, 255, 0.95);
backdrop-filter: blur(20px);       border-
radius: 16px;       padding: 1.5rem;
```

```
border: 1px solid rgba(255, 255, 255,
0.2);        box-shadow: 0 8px 32px
rgba(0, 0, 0, 0.1);        display: flex;
align-items: center;        gap: 1rem;
transition: all 0.3s ease;
   }

      .stat-card:hover {            transform:
translateY(-5px);            box-shadow: 0 12px
40px rgba(0, 0, 0, 0.15); }


   .stat-icon {        width: 60px;        height: 60px;
border-radius: 12px;        background: linear-
gradient(135deg, #667eea, #764ba2);        display: flex;
align-items: center;        justify-content: center;
color: white;        font-size: 1.5rem;
   }

   .stat-content {
      flex: 1;
   }

   .stat-number
{        font-size:
2rem;        font-
weight: 800;
color:
#2b2d42;
```

```
margin: 0;
line-height: 1;
  }

  .stat-label {
color: #6c757d;
margin: 0.25rem 0 0
0;      font-size:
0.9rem;
  }

  .stat-trend {      display: flex;
align-items: center;      gap:
0.25rem;      padding: 0.25rem
0.5rem;      border-radius: 20px;
background: rgba(46, 204, 113,
0.1);      color: #27ae60;      font-
size: 0.8rem;      font-weight: 600;
  }

    .stat-trend.trend-down {
background: rgba(231, 76, 60, 0.1);
color: #e74c3c;
    }

  .card-grid {      display:
grid;      grid-template-
```

```
columns: 1fr 1fr; gap:
1.5rem;
      margin-bottom: 2rem;
   }

   .card {
      background: rgba(255, 255, 255,
0.95);       backdrop-filter: blur(20px);
border-radius: 16px;       border: 1px
solid rgba(255, 255, 255, 0.2);       box-
shadow: 0 8px 32px rgba(0, 0, 0, 0.1);
overflow: hidden;
   }

   .card-header {       padding: 1.5rem;
border-bottom: 1px solid rgba(0, 0, 0,
0.1);       display: flex;       justify-
content: space-between;       align-
items: center;
   }

      .card-
header h3 {
margin: 0;
display: flex;
          align-items:
          center; gap:
          0.5rem;
      color: #2b2d42;
    }

  .card-body {
padding: 1.5rem;
  }

  .role-distribution {
display: flex;
flex-direction:
column;     gap:
1rem;
  }

  .role-item {
display: flex;
flex-direction:
column;     gap:
0.5rem;
  }

  .role-info {     display: flex;
justify-content: space-
between;     align-items:
center;
  }

  .role-badge {
    display: inline-flex;
    align-items: center;
gap: 0.5rem;
padding: 0.5rem 1rem;
border-radius: 20px;
font-size: 0.8rem;
font-weight: 600;
text-transform:
uppercase;      letter-
spacing: 0.5px;
  }

    .role-badge.hr {         background:
rgba(114, 9, 183, 0.1);         color:
#7209b7;         border: 1px solid
rgba(114, 9, 183, 0.2);
    }

    .role-badge.lecturer {
background: rgba(67, 97, 238, 0.1);
color: #4361ee;         border: 1px solid
rgba(67, 97, 238, 0.2);
    }

    .role-
      badge.programmecoordinator
      { background: rgba(248, 150,
      30, 0.1); color: #f8961e;
```

```
          border: 1px solid rgba(248, 150, 30, 0.2);
        }

        .role-badge.academicmanager {
background: rgba(76, 201, 240, 0.1);
color: #4cc9f0;           border: 1px solid
rgba(76, 201, 240, 0.2);
        }

    .role-count {
font-weight:
600;        color:
#2b2d42;
    }

    .role-bar {        height: 6px;
background: rgba(0, 0, 0,
0.1);        border-radius: 3px;
overflow: hidden;
    }

    .role-progress {
height: 100%;
border-radius: 3px;
transition: width 0.3s
ease;
    }
```

```
  .quick-actions {
display: grid;     grid-
template-columns: 1fr 1fr;
gap: 1rem;
  }

  .quick-action {     display: flex;
flex-direction: column;     align-
items: center;     gap: 0.75rem;
padding: 1.5rem;     background:
rgba(102, 126, 234, 0.05);
border-radius: 12px;     text-
decoration: none;     color:
#2b2d42;     transition: all 0.3s
ease;     border: 1px solid
transparent;
  }

    .quick-action:hover {
background: rgba(102, 126, 234, 0.1);
border-color: rgba(102, 126, 234, 0.2);
transform: translateY(-2px); color:
#4361ee;

    }

  .action-icon {     width: 50px;     height: 50px;
border-radius: 12px;     background: linear-
gradient(135deg, #667eea, #764ba2);     display: flex;
```

```
align-items: center;       justify-content: center;
color: white;       font-size: 1.25rem;
   }

   .quick-action
span {       font-
weight: 600;
font-size:
0.9rem;
text-align:
center;
   }

   .table-responsive {
overflow-x: auto;
   }

   .modern-table {
width: 100%;
border-collapse:
collapse;
   }

      .modern-table th {          background: linear-
gradient(135deg, #667eea, #764ba2);          color: white;
padding: 1rem;          text-align: left;          font-weight:
```

```
600;          text-transform: uppercase;          font-size:
0.8rem;          letter-spacing: 0.5px;
      }

      .modern-table td {          padding:
1rem;          border-bottom: 1px solid
rgba(0, 0, 0, 0.1);
      }

      .modern-table tbody tr {
transition: all 0.3s ease;
      }

         .modern-table tbody tr:hover {
background: rgba(102, 126, 234, 0.05);
            transform: translateX(5px);
         }

   .user-avatar {
display: flex;
align-items:
center;       gap:
1rem;
   }

   .avatar {       width: 40px;       height: 40px;       border-
radius: 50%;       background: linear-gradient(135deg,
#667eea, #764ba2);       display: flex;       align-items:
```

```
center;     justify-content: center;     color: white;
font-weight: 600;     font-size: 0.9rem;
  }

  .user-details strong {
display: block;
margin-bottom:
0.25rem;
  }

  .user-details
small {
color: #6c757d;
font-size:
0.8rem;
  }

  .role-tag {     display:
inline-flex;     align-
items: center;     gap:
0.5rem;     padding:
0.5rem 1rem;     border-
radius: 20px;     font-
size: 0.8rem;     font-
weight: 600;     text-
transform: uppercase;
letter-spacing: 0.5px;
  }
```

```
    .role-tag.hr {          background:
rgba(114, 9, 183, 0.1);          color:
#7209b7;          border: 1px solid
rgba(114, 9, 183, 0.2);
    }

    .role-tag.lecturer {
background: rgba(67, 97, 238, 0.1);
color: #4361ee;
      border: 1px solid rgba(67, 97, 238, 0.2);
    }

    .role-tag.programmecoordinator {
background: rgba(248, 150, 30, 0.1);
color: #f8961e;          border: 1px solid
rgba(248, 150, 30, 0.2);
    }

    .role-tag.academicmanager {
background: rgba(76, 201, 240, 0.1);
color: #4cc9f0;          border: 1px solid
rgba(76, 201, 240, 0.2);
    }

  .rate {
font-weight:
```

```
600;       color:
#2b2d42;
  }

    .rate small {
font-size: 0.8rem;
color: #6c757d;
    }

  .status {
    display: inline-flex;
    align-items:
center;       gap:
0.5rem;       padding:
0.5rem 1rem;
border-radius: 20px;
font-size: 0.8rem;
font-weight: 600;
  }

    .status.active {          background:
rgba(46, 204, 113, 0.1);          color:
#27ae60;          border: 1px solid
rgba(46, 204, 113, 0.2);
    }

    .status.inactive {          background:
rgba(108, 117, 125, 0.1);          color:
```

```
#6c757d;           border: 1px solid
rgba(108, 117, 125, 0.2);
      }

      .status i {
font-size: 0.6rem;
      }

   .action-buttons {
      display:
flex;       gap:
0.5rem;
   }

   .btn-action {
width: 35px;
height: 35px;
border-radius: 8px;
display: inline-flex;
align-items: center;
justify-content:
center;       text-
decoration: none;
transition: all 0.3s
ease;       border:
none;       cursor:
pointer;
   }
   .btn-edit {       background:
rgba(243, 156, 18, 0.1);       color:
#f39c12;
   }

      .btn-edit:hover {
background: #f39c12;
color: white;
transform: scale(1.1);
}

   .btn-delete {       background:
rgba(231, 76, 60, 0.1);       color:
#e74c3c;
   }

      .btn-delete:hover {
background: #e74c3c;
color: white;
transform: scale(1.1);
      }

   .empty-state {
text-align: center;
padding: 3rem
2rem;
```

```
   }

   .empty-icon {
font-size: 4rem;
color: #dee2e6;
margin-bottom:
1rem;
   }

   .empty-state h3 {
color: #6c757d;
margin-bottom:
1rem;

   }

   .empty-state p {
color: #6c757d;
margin-bottom:
2rem;
   }

   .search-input {       padding:
0.5rem 1rem;       border: 1px
solid rgba(0, 0, 0, 0.1);
border-radius: 8px;       font-
size: 0.9rem;
   }
```

```
  .card-actions {
display: flex;
gap: 1rem;
align-items:
center;
  }

  .btn-sm {
padding: 0.5rem
1rem;     font-size:
0.8rem;
  }
  @@media (max-width:
768px) {     .dashboard-
container {       padding:
1rem;
    }

    .dashboard-header {
flex-direction: column;
text-align: center;
    }

    .page-title {
font-size: 2rem;
    }
```

```
        .stats-grid {            grid-
template-columns: 1fr;
        }

        .card-grid {            grid-
template-columns: 1fr;
        }

        .quick-actions {            grid-
template-columns: 1fr;
        }
         .header-actions {
            width: 100%;
justify-content: center;
        }

        .card-actions {
flex-direction: column;
width: 100%;
        }

        .search-input {
width: 100%;
        }
    }
</style>
```

Reports.cshtml

```
@{
    Layout = "_Layout";
    ViewData["Title"] = "HR - Reports";
}

<h2>Reports</h2>

<section class="card">
    <h3>Claims Summary</h3>
    <div class="report-grid">
        <div class="report-card">
            <h4>Total Claims</h4>
            <p class="report-number">@ViewBag.TotalClaims</p>
        </div>
        <div class="report-card">
            <h4>Pending</h4>
            <p class="report-number">@ViewBag.PendingClaims</p>
        </div>
        <div class="report-card">
            <h4>Verified</h4>
            <p class="report-number">@ViewBag.VerifiedClaims</p>
        </div>
        <div class="report-card">
            <h4>Approved</h4>
            <p class="report-number">@ViewBag.ApprovedClaims</p>
        </div>
    </div>

    <div class="button-row">
    <a href="@Url.Action("Index")" class="btn-secondary">Back to User
Management</a>
  </div>
</section>

<style>
  .report-grid {
    display: grid;
    grid-template-columns: repeat(auto-fit,
minmax(200px, 1fr));      gap: 1rem;      margin: 2rem 0;
  }

  .report-card {
background: #f8f9fa;
padding: 1.5rem;
border-radius: 8px;
text-align: center;
border: 1px solid
#dee2e6;
  }

  .report-number {
font-size: 2rem;
font-weight: bold;
color: #007bff;
margin: 0.5rem 0 0
0;
  }
```

```
</style>
```

Lecturers (Folder)

Create.cshtml

```
@model CMCS.Mvc.Models.Claim
@{
    Layout = "_Layout";
    ViewData["Title"] = "Submit New Claim";
}

<div class="container mt-4">
    <div class="row justify-content-center">
        <div class="col-md-8">
            <div class="card">
                <div class="card-header bg-primary text-white">
                    <h4 class="mb-0">
                        <i class="fas fa-file-medical"></i> Submit New Claim
                    </h4>
                </div>
                <div class="card-body">
                    @if (TempData["SuccessMessage"] != null)
                    {
                        <div class="alert alert-success alert-dismissible fade show" role="alert">
                            <i class="fas fa-check-circle"></i> @TempData["SuccessMessage"]
                            <button type="button" class="btn-close" data-bs-dismiss="alert"></button>
                        </div>
                    }
```

```
          <form method="post" enctype="multipart/form-data">
            @Html.AntiForgeryToken()

            @if (!ViewData.ModelState.IsValid)
            {
              <div class="alert alert-danger">
                <h5>Please fix the following errors:</h5>
                <ul class="mb-0">
                  @foreach (var error in ViewData.ModelState.Values.SelectMany(v => v.Errors))
                  {
                    <li>@error.ErrorMessage</li>
                  }
                </ul>
              </div>
            }

            <div class="mb-3">
              <label for="Title" class="form-label fw-bold">Claim Title *</label>
              <input type="text" class="form-control form-control-lg" id="Title" name="Title"
                 value="@Model.Title" placeholder="Enter claim title" required>
              <div class="form-text">Describe the work you performed</div>
            </div>

            <div class="mb-3">
              <label for="Description" class="form-label fw-bold">Description</label>
              <textarea class="form-control" id="Description" name="Description"
```

```
                    rows="3" placeholder="Optional details about your work">@Model.Description</textarea>
            </div>

            <div class="row">
              <div class="col-md-6">
                <div class="mb-3">
                  <label for="Hours" class="form-label fw-bold">Hours Worked *</label>
                  <input type="number" class="form-control" id="Hours" name="Hours"
                      value="@(Model.Hours > 0 ? Model.Hours : "")" min="1" max="180" placeholder="Enter hours" required>
                  <div class="form-text">Maximum 180 hours per month</div>
                </div>
              </div>
              <div class="col-md-6">
                <div class="mb-3">
                  <label class="form-label fw-bold">Hourly Rate</label>
                  <div class="input-group">
                    <input type="text" class="form-control" value="@Model.Rate.ToString("C")" readonly>
                    <span class="input-group-text bg-light">
                      <i class="fas fa-lock text-muted"></i>
                    </span>
                  </div>
                  <div class="form-text">Set by HR department</div>
                </div>
              </div>
            </div>

            <div class="mb-3">
              <label class="form-label fw-bold">Total Amount</label>
              <input type="text" class="form-control fw-bold text-success" id="calculatedAmount"
                  value="R 0.00" readonly style="font-size: 1.2rem;">
            </div>

            <div class="mb-4">
              <label for="file" class="form-label fw-bold">Supporting Document (Optional)</label>
              <input type="file" class="form-control" id="file" name="file"
                  accept=".pdf,.docx,.xlsx,.doc,.xls">
              <div class="form-text">
                Allowed: PDF, Word (.docx, .doc), Excel (.xlsx, .xls) - Max 10MB
              </div>
            </div>

            <div class="d-grid gap-2 d-md-flex justify-content-md-end">
              <a href="@Url.Action("Index")" class="btn btn-secondary me-md-2">
                <i class="fas fa-times"></i> Cancel
              </a>
              <button type="submit" class="btn btn-success btn-lg">
                <i class="fas fa-paper-plane"></i> Submit Claim
              </button>
            </div>
          </form>
        </div>
```

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

@section Scripts {
    <script>
        document.addEventListener('DOMContentLoaded', function() {
            const rate = @Model.Rate;
            const hoursInput = document.getElementById('Hours');
            const amountInput = document.getElementById('calculatedAmount');

            function calculateAmount() {
                const hours = parseInt(hoursInput.value) || 0;
                const amount = hours * rate;
                amountInput.value = 'R ' + amount.toFixed(2);
            }

            hoursInput.addEventListener('input', calculateAmount);

            // Initial calculation if hours are pre-filled
            if (hoursInput.value) {
                calculateAmount();
            }
        });
    </script>
}
```

Delete.cshtml

```
@model CMCS.Mvc.Models.Lecturer
@{
    Layout = "_Layout";
    ViewData["Title"] = "Delete Lecturer";
}

<h2>Delete Lecturer</h2>

<section class="card">
    <h3>Are you sure you want to delete this lecturer?</h3>

    <dl class="details-grid">
        <dt>Name:</dt>
        <dd>@Model.Name</dd>

        <dt>Email:</dt>
        <dd>@Model.Email</dd>

        <dt>Department:</dt>
        <dd>@Model.Department</dd>

        <dt>Phone:</dt>
        <dd>@Model.Phone</dd>
    </dl>

    <form asp-action="Delete" method="post">
        @Html.AntiForgeryToken()
        <input type="hidden" asp-for="Id"
/>        <div class="button-row">
<button type="submit" class="btn-danger">Delete</button>
            <a href="@Url.Action("Index")" class="btn-secondary">Cancel</a>
        </div>
    </form>
</section>
```

Detail.cshtml

```
@model CMCS.Mvc.Models.Claim
@{
    Layout = "_Layout";
    ViewData["Title"] = "Claim Details";
}

<h2>Claim Details</h2>

<section class="card">
    <dl class="details-grid">
        <dt>Title:</dt>
        <dd>@Model.Title</dd>

        <dt>Description:</dt>
        <dd>@(string.IsNullOrEmpty(Model.Description) ? "No description provided" : Model.Description)</dd>

        <dt>Hours Worked:</dt>
        <dd>@Model.Hours</dd>
        <dt>Hourly Rate:</dt>
        <dd>@Model.Rate.ToString("C")</dd>

        <dt>Total Amount:</dt>
        <dd><strong>@Model.Amount.ToString("C")</strong></dd>

        <dt>Status:</dt>
        <dd>
          <span style="padding: 4px 8px; border-radius: 4px;
background-color: @(Model.Status == "Pending" ? "#fff3cd" :
                         Model.Status == "Verified" ? "#d1edff" :
                         Model.Status == "Approved" ? "#d4edda" :
                         Model.Status == "Rejected" ? "#f8d7da" : "#e2e3e5");
color: @(Model.Status == "Pending" ? "#856404" :
                     Model.Status == "Verified" ? "#004085" :
                     Model.Status == "Approved" ? "#155724" :
                     Model.Status == "Rejected" ? "#721c24" : "#383d41");">
        @Model.Status
          </span>
        </dd>

        <dt>Submitted:</dt>
        <dd>@Model.CreatedDate.ToString("yyyy-MM-dd HH:mm")</dd>

        @if (!string.IsNullOrEmpty(Model.FilePath))
        {
          <dt>Supporting Document:</dt>
          <dd>
```

```
        <a href="@Url.Action("Download", "Claims", new { id = Model.Id })"
class="btnprimary" target="_blank">
          Download Document
        </a>
      </dd>
    }
  </dl>

  <div class="button-row">
    <a href="@Url.Action("Index")" class="btn-secondary">Back to My Claims</a>
  </div>
</section>
```

Edit.cshtml

```
@model CMCS.Mvc.Models.Lecturer
@{
  Layout = "_Layout";
  ViewData["Title"] = "Edit Lecturer";
}

<h2>Edit Lecturer</h2>

<section class="card">
  <form asp-action="Edit" method="post" class="form-grid">
    @Html.AntiForgeryToken()
    <input type="hidden" asp-for="Id" />

    <div asp-validation-summary="All" class="text-danger"></div>
```

```
<div>
   <label asp-for="Name" class="control-label">Name</label>
   <input asp-for="Name" class="form-control" />
   <span asp-validation-for="Name" class="text-danger"></span>
</div>

<div>
   <label asp-for="Email" class="control-label">Email</label>
   <input asp-for="Email" class="form-control" />
   <span asp-validation-for="Email" class="text-danger"></span>
</div>

<div>
   <label asp-for="Department" class="control-label">Department</label>
   <input asp-for="Department" class="form-control" />
   <span asp-validation-for="Department" class="text-danger"></span>
</div>

<div>
   <label asp-for="Phone" class="control-label">Phone</label>
   <input asp-for="Phone" class="form-control" />
   <span asp-validation-for="Phone" class="text-danger"></span>
</div>

<div class="button-row">
   <button type="submit" class="btn-primary">Save Changes</button>
   <a href="@Url.Action("Index")" class="btn-secondary">Cancel</a>
</div>
```

```
  </form>
</section>
```

Index.cshtml

```
@model List<CMCS.Mvc.Models.Claim>
@{
  Layout = "_Layout";
  ViewData["Title"] = "My Claims";
}

<div class="dashboard-container">
  <!-- Header Section -->
  <div class="dashboard-header">
    <div class="header-content">
      <h1 class="page-title">My Claims</h1>
      <p class="page-subtitle">Manage and track your submitted claims</p>
    </div>
    <div class="header-actions">
      <a href="@Url.Action("Create")" class="btn btn-primary btn-icon">
        <i class="fas fa-plus-circle"></i>
        Submit New Claim
      </a>
      <div class="user-welcome">
        <div class="avatar">

@Context.Session.GetString("Name")?[0]@Context.Session.GetString("Surname")?[0]
        </div>
        <div class="welcome-text">
          Welcome back, <strong>@Context.Session.GetString("Name")</strong>
```

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

<!-- Quick Stats -->
<div class="stats-grid">
    <div class="stat-card">
        <div class="stat-icon">
            <i class="fas fa-file-invoice-dollar"></i>
        </div>
        <div class="stat-content">
            <h3 class="stat-number">@Model.Count</h3>
            <p class="stat-label">Total Claims</p>
        </div>
        <div class="stat-trend">
            <i class="fas fa-history"></i>
            <span>All Time</span>
        </div>
    </div>

    <div class="stat-card">
        <div class="stat-icon">
            <i class="fas fa-clock"></i>
        </div>
        <div class="stat-content">
            <h3 class="stat-number">@Model.Count(c => c.Status == "Pending")</h3>
            <p class="stat-label">Pending</p>
            </div>
            <div class="stat-trend trend-warning">
                <i class="fas fa-clock"></i>
                <span>In Review</span>
            </div>
        </div>

        <div class="stat-card">
            <div class="stat-icon">
                <i class="fas fa-check-circle"></i>
            </div>
            <div class="stat-content">
                <h3 class="stat-number">@Model.Count(c => c.Status == "Approved")</h3>
                <p class="stat-label">Approved</p>
            </div>
            <div class="stat-trend trend-success">
                <i class="fas fa-check"></i>
                <span>Completed</span>
            </div>
        </div>

        <div class="stat-card">
            <div class="stat-icon">
                <i class="fas fa-money-bill-wave"></i>
            </div>
            <div class="stat-content">
                <h3 class="stat-number">@Model.Where(c => c.Status == "Approved").Sum(c
=> c.Amount).ToString("C")</h3>
```

```
          <p class="stat-label">Total Approved</p>
        </div>
        <div class="stat-trend trend-up">
          <i class="fas fa-arrow-up"></i>
          <span>Amount</span>
        </div>
    </div>
</div>

<!-- Recent Activity & Quick Actions -->
<div class="content-grid">
    <!-- Recent Claims -->
    <div class="card main-content">
        <div class="card-header">
          <h3><i class="fas fa-history"></i> Recent Claims</h3>
          <div class="card-actions">
            <a href="@Url.Action("Create")" class="btn btn-sm btn-primary">
              <i class="fas fa-plus"></i> New Claim
            </a>
          </div>
        </div>
        <div class="card-body">
          @if (Model.Any())
          {
            <div class="claims-table">
              <div class="table-header">
                <span>Claim Details</span>
                <span>Amount</span>
                <span>Status</span>
                <span>Date</span>
                <span>Actions</span>
              </div>
              <div class="table-body">
                @foreach (var claim in Model.OrderByDescending(c => c.CreatedDate).Take(5))
                {
                  <div class="table-row">
                    <div class="claim-details">
                      <strong class="claim-title">@claim.Title</strong>
                      @if (!string.IsNullOrEmpty(claim.Description) && claim.Description != "No description provided")
                      {
                        <p class="claim-desc">@claim.Description</p>
                      }
                      <div class="claim-meta">
                        <span class="hours">@claim.Hours hours</span>
                        <span class="rate">@claim.Rate.ToString("C")/hr</span>
                      </div>
                    </div>
                    <div class="claim-amount">
                      <strong>@claim.Amount.ToString("C")</strong>
                    </div>
                    <div class="claim-status">
                      <span class="status-badge @claim.Status.ToLower()">
                        <i class="fas @GetStatusIcon(claim.Status)"></i>
                        @claim.Status
                      </span>
```

```
</div>
<div class="claim-date">
    @claim.CreatedDate.ToString("MMM dd, yyyy")
</div>
<div class="claim-actions">
    <a href="@Url.Action("Details", new { id = claim.Id })" class="btn-action btn-info" title="View Details">
        <i class="fas fa-eye"></i>
    </a>
    @if (claim.Status == "Pending")
    {
        <a href="@Url.Action("Edit", new { id = claim.Id })" class="btn-action btn-edit" title="Edit Claim">
            <i class="fas fa-edit"></i>
        </a>
    }
    <a href="@Url.Action("Delete", new { id = claim.Id })" class="btn-action btn-delete" title="Delete Claim" onclick="return confirm('Are you sure you want to delete this claim?')">
        <i class="fas fa-trash"></i>
    </a>
    @if (!string.IsNullOrEmpty(claim.FilePath))
    {
        <a href="@Url.Action("Download", "Claims", new { id = claim.Id })" class="btn-action btn-download" title="Download Document" target="_blank">
            <i class="fas fa-download"></i>
        </a>
    }
</div>
```

```
                        </div>
                    }
                </div>
                @if (Model.Count > 5)
                {
                    <div class="table-footer">
                        <a href="@Url.Action("Index")" class="btn btn-text">
                            View All @Model.Count Claims <i class="fas fa-arrow-right"></i>
                        </a>
                    </div>
                }
            </div>
        }
        else
        {
            <div class="empty-state-small">
                <div class="empty-icon">
                    <i class="fas fa-file-invoice-dollar"></i>
                </div>
                <h4>No Claims Yet</h4>
                <p>Get started by submitting your first claim</p>
                <a href="@Url.Action("Create")" class="btn btn-primary">
                    <i class="fas fa-plus-circle"></i> Submit First Claim
                </a>
            </div>
        }
    </div>
</div>
```

```
<!-- Sidebar -->
<div class="sidebar">
  <!-- Quick Submit -->
  <div class="card">
    <div class="card-header">
      <h3><i class="fas fa-bolt"></i> Quick Submit</h3>
    </div>
    <div class="card-body">
      <div class="quick-submit">
        <a href="@Url.Action("Create")" class="quick-action large">
          <div class="action-icon">
            <i class="fas fa-plus"></i>
          </div>
          <div class="action-content">
            <span class="action-title">New Claim</span>
            <span class="action-desc">Submit hours worked</span>
          </div>
        </a>
      </div>
    </div>
  </div>

  <!-- Status Overview -->
  <div class="card">
    <div class="card-header">
      <h3><i class="fas fa-chart-pie"></i> Status Overview</h3>
    </div>
```

```
        <div class="card-body">
          <div class="status-overview">
            @{
              var statusGroups = Model.GroupBy(c => c.Status)
              .Select(g => new { Status = g.Key, Count = g.Count(), Amount = g.Sum(c =>
c.Amount) })
              .OrderBy(g => g.Status);
            }
            @foreach (var group in statusGroups)
            {
              <div class="status-item">
                <div class="status-info">
                  <span class="status-badge small @group.Status.ToLower()">
                    <i class="fas @GetStatusIcon(group.Status)"></i>
                    @group.Status
                  </span>
                  <span class="status-count">@group.Count</span>
                </div>
                <div class="status-amount">@group.Amount.ToString("C")</div>
              </div>
            }
          </div>
        </div>
      </div>

      <!-- Help Resources -->
      <div class="card">
        <div class="card-header">
```

```
            <h3><i class="fas fa-life-ring"></i> Need Help?</h3>
          </div>
          <div class="card-body">
            <div class="help-links">
              <a href="#" class="help-link">
                <i class="fas fa-book"></i>
                <span>Claim Guidelines</span>
              </a>
              <a href="#" class="help-link">
                <i class="fas fa-question-circle"></i>
                <span>FAQ</span>
              </a>
              <a href="#" class="help-link">
                <i class="fas fa-envelope"></i>
                <span>Contact Support</span>
              </a>
              <a href="#" class="help-link">
                <i class="fas fa-download"></i>
                <span>Download Templates</span>
              </a>
            </div>
          </div>
        </div>
      </div>
    </div>

    <!-- Monthly Summary -->
    <div class="card">
      <div class="card-header">
        <h3><i class="fas fa-chart-bar"></i> Monthly Summary</h3>
        <div class="period-selector">
          <select class="period-select">
            <option>Last 6 Months</option>
            <option>Last 3 Months</option>
            <option>Last Month</option>
            <option>This Month</option>
          </select>
        </div>
      </div>
      <div class="card-body">
        <div class="monthly-stats">
          @{
            var monthlyData = Model.GroupBy(c => new { c.CreatedDate.Year,
  c.CreatedDate.Month })
            .Select(g => new
            {
              Period = new DateTime(g.Key.Year, g.Key.Month, 1).ToString("MMM yyyy"),
              Claims = g.Count(),
              Amount = g.Sum(c => c.Amount),
              Approved = g.Count(c => c.Status == "Approved")
            })
            .OrderByDescending(g => g.Period)
            .Take(6);
          }
          @foreach (var month in monthlyData)
          {
            <div class="month-item">
              <div class="month-header">
                <span class="month-name">@month.Period</span>
                <span class="month-claims">@month.Claims claims</span>
              </div>
              <div class="month-amount">@month.Amount.ToString("C")</div>
              <div class="month-progress">
                <div class="progress-bar">
                  <div class="progress" style="width: @((month.Approved * 100) / Math.Max(month.Claims, 1))%;"></div>
                </div>
                <span class="progress-text">@month.Approved/@month.Claims approved</span>
              </div>
            </div>
          }
        </div>
      </div>
    </div>
  </div>

  @functions {
    private string GetStatusIcon(string status)
    {
      return status switch
      {
        "Pending" => "fa-clock",
        "Verified" => "fa-check-circle",
        "Approved" => "fa-thumbs-up",
            "Rejected" => "fa-times-circle",
            _ => "fa-file"
        };
    }
}

<style>
    .dashboard-container {
        max-width: 1400px;
        margin: 0 auto;
        padding: 2rem;
    }

    .dashboard-header {
        display: flex;
        justify-content: space-between;
        align-items: center;
        margin-bottom: 2rem;
        gap: 2rem;
    }

    .page-title {
        font-size: 2.5rem;
        font-weight: 800;
        background: linear-gradient(135deg, #4361ee, #3a56d4);
        -webkit-background-clip: text;
        -webkit-text-fill-color: transparent;
        background-clip: text;
```

```
  margin: 0;
}

.page-subtitle {
  color: #6c757d;
  font-size: 1.1rem;
  margin: 0.5rem 0 0 0;
}

.header-actions {
  display: flex;
  align-items: center;
  gap: 2rem;
}

.btn {
  display: inline-flex;
  align-items: center;
  gap: 0.5rem;
  padding: 0.75rem 1.5rem;
  border: none;
  border-radius: 8px;
  font-weight: 600;
  text-decoration: none;
  cursor: pointer;
  transition: all 0.3s ease;
}
```

```
.btn-primary {
  background: linear-gradient(135deg, #4361ee, #3a56d4);
  color: white;
}

  .btn-primary:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 12px rgba(67, 97, 238, 0.3);
  }

.btn-icon {
  display: inline-flex;
  align-items: center;
  gap: 0.5rem;
}

.user-welcome {
  display: flex;
  align-items: center;
  gap: 1rem;
  padding: 0.75rem 1rem;
  background: rgba(255, 255, 255, 0.1);
  border-radius: 12px;
  border: 1px solid rgba(255, 255, 255, 0.2);
}

.avatar {
  width: 40px;
  height: 40px;
  border-radius: 50%;
  background: linear-gradient(135deg, #4361ee, #3a56d4);
  display: flex;
  align-items: center;
  justify-content: center;
  color: white;
  font-weight: 600;
  font-size: 1rem;
}

.welcome-text {
  color: #2b2d42;
  font-size: 0.9rem;
}

.stats-grid {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
  gap: 1.5rem;
  margin-bottom: 2rem;
}

.stat-card {
  background: rgba(255, 255, 255, 0.95);
  backdrop-filter: blur(20px);
  border-radius: 16px;
  padding: 1.5rem;
  border: 1px solid rgba(255, 255, 255, 0.2);
  box-shadow: 0 8px 32px rgba(0, 0, 0, 0.1);
  display: flex;
  align-items: center;
  gap: 1rem;
  transition: all 0.3s ease;
}

  .stat-card:hover {
    transform: translateY(-5px);
    box-shadow: 0 12px 40px rgba(0, 0, 0, 0.15);
  }

.stat-icon {
  width: 60px;
  height: 60px;
  border-radius: 12px;
  background: linear-gradient(135deg, #4361ee, #3a56d4);
  display: flex;
  align-items: center;
  justify-content: center;
  color: white;
  font-size: 1.5rem;
}

.stat-content {
  flex: 1;
}
```

```
.stat-number {
  font-size: 2rem;
  font-weight: 800;
  color: #2b2d42;
  margin: 0;
  line-height: 1;
}

.stat-label {
  color: #6c757d;
  margin: 0.25rem 0 0 0;
  font-size: 0.9rem;
}

.stat-trend {
  display: flex;
  align-items: center;
  gap: 0.25rem;
  padding: 0.25rem 0.75rem;
  border-radius: 20px;
  background: rgba(67, 97, 238, 0.1);
  color: #4361ee;
  font-size: 0.8rem;
  font-weight: 600;
}

  .stat-trend.trend-success {
      background: rgba(46, 204, 113, 0.1);
      color: #27ae60;
  }

  .stat-trend.trend-warning {
      background: rgba(248, 150, 30, 0.1);
      color: #f8961e;
  }

  .stat-trend.trend-up {
      background: rgba(155, 89, 182, 0.1);
      color: #9b59b6;
  }

.content-grid {
  display: grid;
  grid-template-columns: 2fr 1fr;
  gap: 1.5rem;
  margin-bottom: 2rem;
}

.card {
  background: rgba(255, 255, 255, 0.95);
  backdrop-filter: blur(20px);
  border-radius: 16px;
  border: 1px solid rgba(255, 255, 255, 0.2);
  box-shadow: 0 8px 32px rgba(0, 0, 0, 0.1);
  overflow: hidden;
```

```
}

.card-header {
  padding: 1.5rem;
  border-bottom: 1px solid rgba(0, 0, 0, 0.1);
  display: flex;
  justify-content: space-between;
  align-items: center;
}

  .card-header h3 {
    margin: 0;
    display: flex;
    align-items: center;
    gap: 0.5rem;
    color: #2b2d42;
  }

.card-body {
  padding: 1.5rem;
}

.btn-sm {
  padding: 0.5rem 1rem;
  font-size: 0.8rem;
}

.claims-table {
  display: flex;
  flex-direction: column;
}

.table-header {
  display: grid;
  grid-template-columns: 2fr 1fr 1fr 1fr 1fr;
  gap: 1rem;
  padding: 1rem 0;
  border-bottom: 1px solid rgba(0, 0, 0, 0.1);
  font-weight: 600;
  color: #6c757d;
  font-size: 0.8rem;
  text-transform: uppercase;
  letter-spacing: 0.5px;
}

.table-body {
  display: flex;
  flex-direction: column;
}

.table-row {
  display: grid;
  grid-template-columns: 2fr 1fr 1fr 1fr 1fr;
  gap: 1rem;
  padding: 1.5rem 0;
  border-bottom: 1px solid rgba(0, 0, 0, 0.05);
  transition: all 0.3s ease;
}

  .table-row:hover {
    background: rgba(67, 97, 238, 0.05);
    transform: translateX(5px);
  }

.claim-details {
  display: flex;
  flex-direction: column;
  gap: 0.5rem;
}

.claim-title {
  font-weight: 600;
  color: #2b2d42;
  margin: 0;
}

.claim-desc {
  color: #6c757d;
  font-size: 0.9rem;
  margin: 0;
  line-height: 1.4;
}

.claim-meta {
  display: flex;
  gap: 1rem;
  font-size: 0.8rem;
}

.hours, .rate {
  color: #6c757d;
}

.claim-amount {
  font-weight: 700;
  color: #27ae60;
  display: flex;
  align-items: center;
}

.claim-status {
  display: flex;
  align-items: center;
}

.status-badge {
  display: inline-flex;
  align-items: center;
  gap: 0.5rem;
  padding: 0.5rem 1rem;
  border-radius: 20px;
  font-size: 0.8rem;
  font-weight: 600;
  text-transform: uppercase;
  letter-spacing: 0.5px;
}

  .status-badge.small {
    padding: 0.25rem 0.75rem;
    font-size: 0.7rem;
  }

  .status-badge.pending {
    background: rgba(248, 150, 30, 0.1);
    color: #f8961e;
    border: 1px solid rgba(248, 150, 30, 0.2);
  }

  .status-badge.verified {
    background: rgba(76, 201, 240, 0.1);
    color: #4cc9f0;
    border: 1px solid rgba(76, 201, 240, 0.2);
  }

  .status-badge.approved {
    background: rgba(46, 204, 113, 0.1);
    color: #27ae60;
    border: 1px solid rgba(46, 204, 113, 0.2);
  }
```

```
    .status-badge.rejected {
      background: rgba(231, 76, 60, 0.1);
      color: #e74c3c;
      border: 1px solid rgba(231, 76, 60, 0.2);
    }

  .claim-date {
    color: #6c757d;
    font-size: 0.9rem;
    display: flex;
    align-items: center;
  }

  .claim-actions {
    display: flex;
    gap: 0.5rem;
    align-items: center;
  }

  .btn-action {
    width: 35px;
    height: 35px;
    border-radius: 8px;
    display: inline-flex;
    align-items: center;
    justify-content: center;
    text-decoration: none;
    transition: all 0.3s ease;
    border: none;

    cursor: pointer;

  }


  .btn-info {

    background: rgba(52, 152, 219, 0.1);

    color: #3498db;

  }


    .btn-info:hover {

      background: #3498db;

      color: white;

      transform: scale(1.1);

    }


  .btn-edit {

    background: rgba(243, 156, 18, 0.1);

    color: #f39c12;

  }


    .btn-edit:hover {

      background: #f39c12;

      color: white;

      transform: scale(1.1);

    }


  .btn-delete {

    background: rgba(231, 76, 60, 0.1);
    color: #e74c3c;
  }

    .btn-delete:hover {
      background: #e74c3c;
      color: white;
      transform: scale(1.1);
    }

  .btn-download {
    background: rgba(46, 204, 113, 0.1);
    color: #2ecc71;
  }

    .btn-download:hover {
      background: #2ecc71;
      color: white;
      transform: scale(1.1);
    }

  .table-footer {
    padding: 1rem 0 0 0;
    text-align: center;
  }

  .btn-text {
    background: transparent;
    color: #4361ee;
    border: none;

    text-decoration: none;

    display: inline-flex;

    align-items: center;

    gap: 0.5rem;

  }


    .btn-text:hover {

      color: #3a56d4;

    }


  .empty-state-small {

    text-align: center;

    padding: 3rem 2rem;

  }


    .empty-state-small .empty-icon {

      font-size: 3rem;

      color: #dee2e6;

      margin-bottom: 1rem;

    }


    .empty-state-small h4 {

      color: #6c757d;

      margin-bottom: 0.5rem;

    }


    .empty-state-small p {
        color: #6c757d;
        margin-bottom: 1.5rem;
    }

.sidebar {
    display: flex;
    flex-direction: column;
    gap: 1.5rem;
}

.quick-submit {
    display: flex;
    flex-direction: column;
    gap: 1rem;
}

.quick-action.large {
    display: flex;
    align-items: center;
    gap: 1rem;
    padding: 1.5rem;
    background: rgba(67, 97, 238, 0.05);
    border-radius: 12px;
    text-decoration: none;
    color: #2b2d42;
    transition: all 0.3s ease;
    border: 2px dashed rgba(67, 97, 238, 0.2);
}
```

```
  .quick-action.large:hover {
    background: rgba(67, 97, 238, 0.1);
    border-color: rgba(67, 97, 238, 0.4);
    transform: translateY(-2px);
    color: #4361ee;
  }

  .quick-action.large .action-icon {
    width: 50px;
    height: 50px;
    border-radius: 12px;
    background: linear-gradient(135deg, #4361ee, #3a56d4);
    display: flex;
    align-items: center;
    justify-content: center;
    color: white;
    font-size: 1.25rem;
    flex-shrink: 0;
  }

.action-content {
  flex: 1;
}

.action-title {
  display: block;
  font-weight: 600;
  margin-bottom: 0.25rem;
}

.action-desc {
  display: block;
  color: #6c757d;
  font-size: 0.8rem;
}

.status-overview {
  display: flex;
  flex-direction: column;
  gap: 1rem;
}

.status-item {
  display: flex;
  justify-content: space-between;
  align-items: center;
  padding: 0.75rem;
  background: rgba(248, 249, 250, 0.8);
  border-radius: 8px;
}

.status-info {
  display: flex;
  align-items: center;
  gap: 1rem;
```

```
}

.status-count {
   font-weight: 600;
   color: #2b2d42;
}

.status-amount {
   font-weight: 700;
   color: #27ae60;
   font-size: 0.9rem;
}

.help-links {
   display: flex;
   flex-direction: column;
   gap: 0.75rem;
}

.help-link {
   display: flex;
   align-items: center;
   gap: 1rem;
   padding: 0.75rem;
   background: rgba(248, 249, 250, 0.8);
   border-radius: 8px;
   text-decoration: none;
   color: #2b2d42;
```

```
  transition: all 0.3s ease;
}

  .help-link:hover {
    background: rgba(67, 97, 238, 0.1);
    color: #4361ee;
    transform: translateX(5px);
  }

  .help-link i {
    width: 20px;
    text-align: center;
    color: #4361ee;
  }

.period-selector {
  display: flex;
  align-items: center;
  gap: 0.5rem;
}

.period-select {
  padding: 0.5rem 1rem;
  border: 1px solid rgba(0, 0, 0, 0.1);
  border-radius: 8px;
  font-size: 0.9rem;
  background: white;
}
```

```
.monthly-stats {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(200px, 1fr));
  gap: 1.5rem;
}

.month-item {
  display: flex;
  flex-direction: column;
  gap: 0.75rem;
  padding: 1.5rem;
  background: rgba(67, 97, 238, 0.05);
  border-radius: 12px;
  border: 1px solid rgba(67, 97, 238, 0.1);
}

.month-header {
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.month-name {
  font-weight: 600;
  color: #2b2d42;
}
```

```
.month-claims {
   color: #6c757d;
   font-size: 0.8rem;
}

.month-amount {
   font-weight: 700;
   color: #27ae60;
   font-size: 1.25rem;
}

.month-progress {
   display: flex;
   flex-direction: column;
   gap: 0.5rem;
}

.progress-bar {
   height: 6px;
   background: rgba(0, 0, 0, 0.1);
   border-radius: 3px;
   overflow: hidden;
}

.progress {
   height: 100%;
   background: linear-gradient(135deg, #4361ee, #3a56d4);
   border-radius: 3px;
```

```
  transition: width 0.3s ease;
}

.progress-text {
  color: #6c757d;
  font-size: 0.8rem;
}

@@media (max-width: 768px) {
  .dashboard-container {
    padding: 1rem;
  }

  .dashboard-header {
    flex-direction: column;
    text-align: center;
  }

  .header-actions {
    flex-direction: column;
    width: 100%;
  }

  .user-welcome {
    width: 100%;
    justify-content: center;
  }
```

```
    .stats-grid {
      grid-template-columns: 1fr;
    }

    .content-grid {
      grid-template-columns: 1fr;
    }

    .table-header {
      display: none;
    }

    .table-row {
      grid-template-columns: 1fr;
      gap: 1rem;
      padding: 1.5rem;
      border: 1px solid rgba(0, 0, 0, 0.1);
      border-radius: 12px;
      margin-bottom: 1rem;
    }

    .monthly-stats {
      grid-template-columns: 1fr;
    }
  }

</script >
```

Login (Folder)

AccessDenied.cshtml

```
@{
    Layout = null;
}

<!DOCTYPE html>
<html>
<head>
    <title>Access Denied</title>
</head>
<body>
    <h2>Access Denied</h2>
    <p>You do not have permission to access this page.</p>
    <a href="@Url.Action("Index", "Login")">Return to Login</a>
</body>
</html>
```

Index.cshtml

```
@{
    Layout = null;
}


<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
```

```
  <title>CMCS - Login</title>

  <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.4.0/css/all.min.css">

  <style>

    * {

      margin: 0;

      padding: 0;

      box-sizing: border-box;

    }



    body {

      font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif;

      min-height: 100vh;

      display: flex;

      align-items: center;

      justify-content: center;

      padding: 20px;

      position: relative;

      overflow: hidden;

      background: #111;

    }



    /* Amazing Background Effect */

    .background-container {

      --color-0: #fff;
      --color-1: #111;

      --color-2: #222;

      --color-3: #333;

      --color-4: #2e2e2e;

      --color-5: #d2b48c;

      --color-6: #b22222;

      --color-7: #871a1a;

      --color-8: #ff6347;

      --color-9: #ff3814;

      width: 100%;

      height: 100%;

      position: absolute;

      top: 0;

      left: 0;

      background-color: var(--color-1);

      background-image: linear-gradient( to top, var(--color-2) 5%, var(--color-1) 6%, var(--color-1) 7%, transparent 7% ), linear-gradient(to bottom, var(--color-1) 30%, transparent 80%), linear-gradient(to right, var(--color-2), var(--color-4) 5%, transparent 5%), linear-gradient( to right, transparent 6%, var(--color-2) 6%, var(--color-4) 9%, transparent 9% ), linear-gradient( to right, transparent 27%, var(--color-2) 27%, var(--color-4) 34%, transparent 34% ), linear-gradient( to right, transparent 51%, var(--color-2) 51%, var(--color-4) 57%, transparent 57% ), linear-gradient(to bottom, var(--color-1) 35%, transparent 35%), linear-gradient( to right, transparent 42%, var(--color-2) 42%, var(--color-4) 44%, transparent 44% ), linear-gradient( to right, transparent 45%, var(--color-2) 45%, var(--color-4) 47%, transparent 47% ), linear-gradient( to right, transparent 48%, var(--color-2) 48%, var(--color-4) 50%, transparent 50% ), linear-gradient( to right, transparent 87%, var(--color-2) 87%, var(--color-4) 91%, transparent 91% ), linear-gradient(to bottom, var(--color-1) 37.5%, transparent 37.5%), linear-gradient( to right, transparent 14%, var(--color-2) 14%, var(--color-4) 20%, transparent 20% ), linear-gradient(to bottom, var(--color-1) 40%, transparent 40%), linear-gradient( to right, transparent 10%, var(--color-2) 10%, var(--color-4) 13%, transparent 13% ), linear-
```

```
gradient( to right, transparent 21%, var(--color-2) 21%, #1a1a1a 25%, transparent 25% ),
linear-gradient( to right, transparent 58%, var(--color-2) 58%, var(--color-4) 64%,
transparent 64% ), linear-gradient( to right, transparent 92%, var(--color-2) 92%, var(--
color-4) 95%, transparent 95% ), linear-gradient(to bottom, var(--color-1) 48%,
transparent 48%), linear-gradient( to right, transparent 96%, var(--color-2) 96%,
#1a1a1a 99%, transparent 99% ), linear-gradient( to bottom, transparent 68.5%,
transparent 76%, var(--color-1) 76%, var(--color-1) 77.5%, transparent 77.5%,
transparent 86%, var(--color-1) 86%, var(--color-1) 87.5%, transparent 87.5% ), linear-
gradient( to right, transparent 35%, var(--color-2) 35%, var(--color-4) 41%, transparent
41% ), linear-gradient(to bottom, var(--color-1) 68%, transparent 68%), linear-gradient(
to right, transparent 78%, var(--color-3) 78%, var(--color-3) 80%, transparent 80%,
transparent 82%, var(--color-3) 82%, var(--color-3) 83%, transparent 83% ), linear-
gradient( to right, transparent 66%, var(--color-2) 66%, var(--color-4) 85%, transparent
85% );

      background-size: 300px 150px;

      background-position: center bottom;

    }


      .background-container:before {

        content: "";

        width: 100%;

        height: 100%;

        position: absolute;

        inset: 0;

        background-color: var(--color-1);

        background-image: linear-gradient( to top, var(--color-5) 5%, var(--color-1) 6%,
var(--color-1) 7%, transparent 7% ), linear-gradient(to bottom, var(--color-1) 30%,
transparent 30%), linear-gradient(to right, var(--color-6), var(--color-7) 5%, transparent
5%), linear-gradient( to right, transparent 6%, var(--color-8) 6%, var(--color-9) 9%,
transparent 9% ), linear-gradient( to right, transparent 27%, #556b2f 27%, #39481f 34%,
transparent 34% ), linear-gradient( to right, transparent 51%, #fa8072 51%, #f85441
57%, transparent 57% ), linear-gradient(to bottom, var(--color-1) 35%, transparent 35%),
linear-gradient( to right, transparent 42%, #008080 42%, #004d4d 44%, transparent
44% ), linear-gradient( to right, transparent 45%, #008080 45%, #004d4d 47%,
transparent 47% ), linear-gradient( to right, transparent 48%, #008080 48%, #004d4d
50%, transparent 50% ), linear-gradient( to right, transparent 87%, #789 87%, #4f5d6a
91%, transparent 91% ), linear-gradient(to bottom, var(--color-1) 37.5%, transparent
37.5%), linear-gradient( to right, transparent 14%, #bdb76b 14%, #989244 20%,
transparent 20% ), linear-gradient(to bottom, var(--color-1) 40%, transparent 40%),
linear-gradient( to right, transparent 10%, #808000 10%, #4d4d00 13%, transparent
13% ), linear-gradient( to right, transparent 21%, #8b4513 21%, #5e2f0d 25%,
transparent 25% ), linear-gradient( to right, transparent 58%, #8b4513 58%, #5e2f0d
64%, transparent 64% ), linear-gradient( to right, transparent 92%, #2f4f4f 92%, #1c2f2f
95%, transparent 95% ), linear-gradient(to bottom, var(--color-1) 48%, transparent 48%),
linear-gradient( to right, transparent 96%, #2f4f4f 96%, #1c2f2f 99%, transparent 99% ),
linear-gradient( to bottom, transparent 68.5%, transparent 76%, var(--color-1) 76%, var(-
-color-1) 77.5%, transparent 77.5%, transparent 86%, var(--color-1) 86%, var(--color-1)
87.5%, transparent 87.5% ), linear-gradient( to right, transparent 35%, #cd5c5c 35%,
#bc3a3a 41%, transparent 41% ), linear-gradient(to bottom, var(--color-1) 68%,
transparent 68%), linear-gradient( to right, transparent 78%, #bc8f8f 78%, #bc8f8f 80%,
transparent 80%, transparent 82%, #bc8f8f 82%, #bc8f8f 83%, transparent 83% ),
linear-gradient( to right, transparent 66%, #a52a2a 66%, #7c2020 85%, transparent 85%
);

          background-size: 300px 150px;

          background-position: center bottom;

          clip-path: circle(150px at center center);

          animation: flashlight 20s ease infinite;

        }



        .background-container:after {

          content: "";

          width: 25px;

          height: 10px;

          position: absolute;

          left: calc(50% + 59px);

          bottom: 100px;
```

```
            background-repeat: no-repeat;

            background-image: radial-gradient(circle, #fff 50%, transparent 50%), radial-gradient(circle, #fff 50%, transparent 50%);

            background-size: 10px 10px;

            background-position: left center, right center;

            animation: eyes 20s infinite;

          }


      @@keyframes flashlight {

          0% {

            clip-path: circle(150px at -25% 10%);

          }


          38% {

            clip-path: circle(150px at 60% 20%);

          }


          39% {

            opacity: 1;

            clip-path: circle(150px at 60% 86%);

          }


          40% {

            opacity: 0;

            clip-path: circle(150px at 60% 86%);
```

```
}

41% {
   opacity: 1;
   clip-path: circle(150px at 60% 86%);
}

42% {
   opacity: 0;
   clip-path: circle(150px at 60% 86%);
}

54% {
   opacity: 0;
   clip-path: circle(150px at 60% 86%);
}

55% {
   opacity: 1;
   clip-path: circle(150px at 60% 86%);
}

59% {
   opacity: 1;
```

```
  }

    clip-path: circle(150px at 60% 86%);

  }


  64% {

    clip-path: circle(150px at 45% 78%);

  }


  68% {

    clip-path: circle(150px at 85% 89%);

  }


  72% {

    clip-path: circle(150px at 60% 86%);

  }


  74% {

    clip-path: circle(150px at 60% 86%);

  }


  100% {

    clip-path: circle(150px at 150% 50%);

  }

}
```

```
    @@keyframes eyes {
      0%,
      38% {
        opacity: 0;
      }

      39%,
      41% {
        opacity: 1;
        transform: scaleY(1);
      }

      40% {
        transform: scaleY(0);
        filter: none;
        background-image: radial-gradient(circle, #fff 50%, transparent 50%), radial-gradient(circle, #fff 50%, transparent 50%);
      }

      41% {
        transform: scaleY(1);
        background-image: radial-gradient(circle, #ff0000 50%, transparent 50%), radial-gradient(circle, #ff0000 50%, transparent 50%);
        filter: drop-shadow(0 0 4px #ff8686);
      }
```

```
  42%,
  100% {
    opacity: 0;
  }
}

/* Login Container Styles */
.login-container {
  background: rgba(255, 255, 255, 0.95);
  backdrop-filter: blur(20px);
  border-radius: 20px;
  box-shadow: 0 20px 40px rgba(0, 0, 0, 0.3), 0 0 0 1px rgba(255, 255, 255, 0.2);
  padding: 40px;
  width: 100%;
  max-width: 440px;
  position: relative;
  z-index: 10;
  animation: slideUp 0.8s ease-out;
  border: 1px solid rgba(255, 255, 255, 0.3);
}

@@keyframes slideUp {
  from {
        opacity: 0;

        transform: translateY(30px);

    }


    to {

        opacity: 1;

        transform: translateY(0);

    }

}


.login-header {

    text-align: center;

    margin-bottom: 40px;

}


.logo {

    font-size: 2.5rem;

    font-weight: 700;

    background: linear-gradient(135deg, #667eea, #764ba2);

    -webkit-background-clip: text;

    -webkit-text-fill-color: transparent;

    background-clip: text;

    margin-bottom: 10px;

}
```

```
.login-subtitle {
    color: #6c757d;
    font-size: 1.1rem;
    font-weight: 400;
}

.alert {
    padding: 12px 16px;
    border-radius: 10px;
    margin-bottom: 20px;
    font-weight: 500;
    animation: shake 0.5s ease-in-out;
}

.alert-danger {
    background: rgba(220, 53, 69, 0.1);
    border: 1px solid rgba(220, 53, 69, 0.2);
    color: #dc3545;
}

@@keyframes shake {
    0%, 100% {
        transform: translateX(0);
```

```
  }

  25% {
    transform: translateX(-5px);
  }

  75% {
    transform: translateX(5px);
  }
}

.form-group {
  margin-bottom: 25px;
  position: relative;
}

  .form-group label {
    display: block;
    margin-bottom: 8px;
    font-weight: 600;
    color: #495057;
    font-size: 0.95rem;
  }
```

```
.form-control {
  width: 100%;
  padding: 14px 16px;
  border: 2px solid #e9ecef;
  border-radius: 12px;
  font-size: 1rem;
  transition: all 0.3s ease;
  background: rgba(255, 255, 255, 0.8);
}

  .form-control:focus {
    outline: none;
    border-color: #667eea;
    box-shadow: 0 0 0 3px rgba(102, 126, 234, 0.1);
    transform: translateY(-2px);
  }

  .form-control:hover {
    border-color: #adb5bd;
  }

.btn-login {
  width: 100%;
  background: linear-gradient(135deg, #667eea, #764ba2);
```

```
    color: white;

    border: none;

    padding: 16px;

    border-radius: 12px;

    font-size: 1.1rem;

    font-weight: 600;

    cursor: pointer;

    transition: all 0.3s ease;

    box-shadow: 0 4px 15px rgba(102, 126, 234, 0.3);

    position: relative;

    overflow: hidden;

  }


    .btn-login:hover {

      transform: translateY(-2px);

      box-shadow: 0 8px 25px rgba(102, 126, 234, 0.4);

    }


    .btn-login:active {

      transform: translateY(0);

    }


    .btn-login::before {

      content: '';
```

```
                position: absolute;
                top: 0;
                left: -100%;
                width: 100%;
                height: 100%;
                background: linear-gradient(90deg, transparent, rgba(255, 255, 255, 0.2), transparent);
                transition: left 0.5s;
            }

            .btn-login:hover::before {
                left: 100%;
            }

        .demo-accounts {
            margin-top: 30px;
            padding-top: 30px;
            border-top: 1px solid #e9ecef;
        }

            .demo-accounts h4 {
                color: #495057;
                margin-bottom: 15px;
                font-size: 1rem;
                text-align: center;
```

```
  }

.account-list {
  list-style: none;
  padding: 0;
}

  .account-list li {
    padding: 8px 12px;
    margin-bottom: 8px;
    background: rgba(102, 126, 234, 0.05);
    border-radius: 8px;
    font-size: 0.9rem;
    color: #495057;
    border-left: 3px solid #667eea;
    transition: all 0.3s ease;
  }

    .account-list li:hover {
      background: rgba(102, 126, 234, 0.1);
      transform: translateX(5px);
    }

  .account-list strong {
        color: #667eea;
    }

/* Responsive design */
@@media (max-width: 480px) {
    .login-container {
        padding: 30px 25px;
        margin: 20px;
    }

    .logo {
        font-size: 2rem;
    }

    .login-subtitle {
        font-size: 1rem;
    }
}

/* Loading animation for button */
.btn-loading {
    pointer-events: none;
    opacity: 0.8;
}
```

```
      .btn-loading::after {
        content: '';
        display: inline-block;
        width: 16px;
        height: 16px;
        border: 2px solid transparent;
        border-top: 2px solid #ffffff;
        border-radius: 50%;
        animation: spin 1s linear infinite;
        margin-left: 8px;
      }

    @@keyframes spin {
      0% {
        transform: rotate(0deg);
      }

      100% {
        transform: rotate(360deg);
      }
    }
  </style>
</head>
```

```
<body>
    <!-- Amazing Animated Background -->
    <div class="background-container"></div>

    <div class="login-container">
        <div class="login-header">
            <div class="logo">CMCS</div>
            <div class="login-subtitle">Contract Monthly Claim System</div>
        </div>

        @if (TempData["Error"] != null)
        {
            <div class="alert alert-danger">
                <i class="fas fa-exclamation-circle"></i> @TempData["Error"]
            </div>
        }

        <form method="post" action="@Url.Action("Login", "Login")" id="loginForm">
            <div class="form-group">
                <label for="username"><i class="fas fa-user"></i> Username</label>
                <input type="text" id="username" name="username" class="form-control" placeholder="Enter your username" required />
            </div>

            <div class="form-group">
```

```
          <label for="password"><i class="fas fa-lock"></i> Password</label>
          <input type="password" id="password" name="password" class="form-control" placeholder="Enter your password" required />
        </div>

        <button type="submit" class="btn-login" id="loginButton">
          <i class="fas fa-sign-in-alt"></i> Sign In
        </button>
      </form>

      <div class="demo-accounts">
        <h4><i class="fas fa-users"></i> Demo Accounts</h4>
        <ul class="account-list">
          <li><strong>HR:</strong> hr@cmcs.com / hr123</li>
          <li><strong>Lecturer:</strong> lecturer1@cmcs.com / lecturer123</li>
          <li><strong>Programme Coordinator:</strong> coordinator@cmcs.com / coordinator123</li>
          <li><strong>Academic Manager:</strong> manager@cmcs.com / manager123</li>
        </ul>
      </div>
    </div>

    <script>
      document.addEventListener('DOMContentLoaded', function() {
        const loginForm = document.getElementById('loginForm');
        const loginButton = document.getElementById('loginButton');

        loginForm.addEventListener('submit', function() {
          // Add loading state to button
          loginButton.classList.add('btn-loading');
          loginButton.innerHTML = '<i class="fas fa-spinner"></i> Signing In...';
        });

        // Add focus effects to form inputs
        const inputs = document.querySelectorAll('.form-control');
        inputs.forEach(input => {
          input.addEventListener('focus', function() {
            this.parentElement.style.transform = 'translateY(-2px)';
          });

          input.addEventListener('blur', function() {
            this.parentElement.style.transform = 'translateY(0)';
          });
        });
      });
    </script>
  </body>
  </html>
```

ProgrammeCoordinator (Folder)

Index.cshtml

```
@model List<CMCS.Mvc.Models.Claim>

@{

  Layout = "_Layout";

  ViewData["Title"] = "Pending Claims Review";

}


<div class="dashboard-container">

  <!-- Header Section -->

  <div class="dashboard-header">

    <div class="header-content">

      <h1 class="page-title">Pending Claims Review</h1>

      <p class="page-subtitle">Verify and approve claims from lecturers</p>

    </div>

    <div class="header-stats">

      <div class="stat-badge pending">

        <i class="fas fa-clock"></i>

        <span>@Model.Count(c => c.Status == "Pending") Pending</span>

      </div>

      <div class="stat-badge total">

        <i class="fas fa-file-invoice"></i>

        <span>@Model.Count Total Claims</span>

      </div>

    </div>

  </div>


  <!-- Quick Stats -->
  <div class="stats-grid">
    <div class="stat-card">
      <div class="stat-icon">
        <i class="fas fa-hourglass-half"></i>
      </div>
      <div class="stat-content">
        <h3 class="stat-number">@Model.Count</h3>
        <p class="stat-label">Awaiting Review</p>
      </div>
      <div class="stat-badge urgent">
        <i class="fas fa-exclamation-circle"></i>
        Priority
      </div>
    </div>

    <div class="stat-card">
      <div class="stat-icon">
        <i class="fas fa-calendar-week"></i>
      </div>
      <div class="stat-content">
        <h3 class="stat-number">@Model.Count(c => c.CreatedDate >= DateTime.Now.AddDays(-7))</h3>
        <p class="stat-label">This Week</p>
      </div>
      <div class="stat-trend trend-up">
        <i class="fas fa-arrow-up"></i>
        <span>New</span>
      </div>
```

```
</div>

<div class="stat-card">
  <div class="stat-icon">
    <i class="fas fa-money-bill-wave"></i>
  </div>
  <div class="stat-content">
    <h3 class="stat-number">@Model.Sum(c => c.Amount).ToString("C")</h3>
    <p class="stat-label">Total Value</p>
  </div>
  <div class="stat-trend">
    <i class="fas fa-chart-line"></i>
    <span>Amount</span>
  </div>
</div>

<div class="stat-card">
  <div class="stat-icon">
    <i class="fas fa-users"></i>
  </div>
  <div class="stat-content">
    <h3 class="stat-number">@Model.Select(c => c.UserId).Distinct().Count()</h3>
    <p class="stat-label">Lecturers</p>
  </div>
  <div class="stat-trend">
    <i class="fas fa-user-friends"></i>
    <span>Active</span>
  </div>
  </div>
</div>

<!-- Claims List -->
<div class="card">
  <div class="card-header">
    <h3><i class="fas fa-tasks"></i> Claims Requiring Verification</h3>
    <div class="card-actions">
      <div class="filter-group">
        <select class="filter-select">
          <option>All Departments</option>
          <option>Computer Science</option>
          <option>Mathematics</option>
          <option>Physics</option>
        </select>
        <input type="text" class="search-input" placeholder="Search claims...">
      </div>
    </div>
  </div>
  <div class="card-body">
    @if (Model.Any())
    {
      <div class="claims-list">
        @foreach (var claim in Model.OrderByDescending(c => c.CreatedDate))
        {
          <div class="claim-item">
            <div class="claim-header">
              <div class="claim-meta">
    <div class="lecturer-avatar">
      <div class="avatar">
        @claim.User.Name[0]@claim.User.Surname[0]
      </div>
      <div class="lecturer-info">
        <h4>@claim.User.Name @claim.User.Surname</h4>
        <p>@claim.User.Department • @claim.User.Email</p>
      </div>
    </div>
    <div class="claim-amount">
      <span class="amount">@claim.Amount.ToString("C")</span>
      <span class="hours">@claim.Hours hours</span>
    </div>
  </div>
  <div class="claim-status">
    <span class="status-badge pending">
      <i class="fas fa-clock"></i>
      Awaiting Verification
    </span>
    <span class="claim-date">
      <i class="fas fa-calendar"></i>
      @claim.CreatedDate.ToString("MMM dd, yyyy")
    </span>
  </div>
</div>

<div class="claim-content">
  <h3 class="claim-title">@claim.Title</h3>
```

```
    @if (!string.IsNullOrEmpty(claim.Description) && claim.Description != "No description provided")
    {
        <p class="claim-description">@claim.Description</p>
    }

    <div class="claim-details">
        <div class="detail-item">
            <i class="fas fa-clock"></i>
            <span>@claim.Hours hours</span>
        </div>
        <div class="detail-item">
            <i class="fas fa-dollar-sign"></i>
            <span>@claim.Rate.ToString("C")/hour</span>
        </div>
        <div class="detail-item">
            <i class="fas fa-calculator"></i>
            <span>Total: @claim.Amount.ToString("C")</span>
        </div>
        @if (!string.IsNullOrEmpty(claim.FilePath))
        {
            <div class="detail-item">
                <i class="fas fa-paperclip"></i>
                <span>Document Attached</span>
            </div>
        }
    </div>
</div>
```

```
              <div class="claim-actions">
                <form method="post" action="@Url.Action("Verify", new { id = claim.Id })" class="action-form">
                  @Html.AntiForgeryToken()
                  <button type="submit" class="btn btn-success btn-icon">
                    <i class="fas fa-check-circle"></i>
                    Verify Claim
                  </button>
                </form>
                <a href="@Url.Action("Details", "Claims", new { id = claim.Id })" class="btn btn-secondary btn-icon">
                  <i class="fas fa-eye"></i>
                  View Details
                </a>
                @if (!string.IsNullOrEmpty(claim.FilePath))
                {
                  <a href="@Url.Action("Download", "Claims", new { id = claim.Id })" class="btn btn-outline btn-icon" target="_blank">
                    <i class="fas fa-download"></i>
                    Document
                  </a>
                }
                <button class="btn btn-text btn-icon" onclick="showRejectionModal(@claim.Id)">
                  <i class="fas fa-flag"></i>
                  Flag Issue
                </button>
              </div>
```

```
            </div>
          }
        </div>
      }
      else
      {
        <div class="empty-state">
          <div class="empty-icon">
            <i class="fas fa-check-circle"></i>
          </div>
          <h3>All Caught Up!</h3>
          <p>There are no pending claims requiring verification at this time.</p>
          <div class="empty-actions">
            <a href="/Claims" class="btn btn-primary">
              <i class="fas fa-list"></i> View All Claims
            </a>
            <a href="/Home" class="btn btn-secondary">
              <i class="fas fa-home"></i> Return to Dashboard
            </a>
          </div>
        </div>
      }
  </div>
</div>

<!-- Department Overview -->
<div class="card-grid">
  <div class="card">
```

```
        <div class="card-header">
          <h3><i class="fas fa-chart-pie"></i> Department Distribution</h3>
        </div>
        <div class="card-body">
          <div class="department-stats">
            @{
              var departments = Model.GroupBy(c => c.User.Department)
              .Select(g => new { Department = g.Key, Count = g.Count(), Amount = g.Sum(c => c.Amount) })
              .OrderByDescending(d => d.Count);
            }
            @foreach (var dept in departments)
            {
              <div class="department-item">
                <div class="dept-info">
                  <span class="dept-name">@dept.Department</span>
                  <span class="dept-count">@dept.Count claims</span>
                </div>
                <div class="dept-amount">@dept.Amount.ToString("C")</div>
                <div class="progress-bar">
                  <div class="progress" style="width: @((dept.Count * 100) / Model.Count)%;"></div>
                </div>
              </div>
            }
          </div>
        </div>
      </div>
```

```html
<div class="card">
  <div class="card-header">
    <h3><i class="fas fa-bolt"></i> Quick Actions</h3>
  </div>
  <div class="card-body">
    <div class="quick-actions-vertical">
      <a href="/Claims/Create" class="quick-action">
        <div class="action-icon">
          <i class="fas fa-plus-circle"></i>
        </div>
        <div class="action-content">
          <span class="action-title">Create New Claim</span>
          <span class="action-desc">Submit a claim on behalf of lecturer</span>
        </div>
      </a>
      <a href="/Claims" class="quick-action">
        <div class="action-icon">
          <i class="fas fa-list"></i>
        </div>
        <div class="action-content">
          <span class="action-title">View All Claims</span>
          <span class="action-desc">Browse complete claims history</span>
        </div>
      </a>
      <a href="#" class="quick-action">
        <div class="action-icon">
          <i class="fas fa-download"></i>
        </div>
```

```
                <div class="action-content">
                    <span class="action-title">Export Report</span>
                    <span class="action-desc">Download verification report</span>
                </div>
            </a>
            <a href="#" class="quick-action">
                <div class="action-icon">
                    <i class="fas fa-cog"></i>
                </div>
                <div class="action-content">
                    <span class="action-title">Settings</span>
                    <span class="action-desc">Manage verification preferences</span>
                </div>
            </a>
        </div>
      </div>
    </div>
  </div>
</div>

<!-- Rejection Modal -->
<div id="rejectionModal" class="modal">
  <div class="modal-content">
    <div class="modal-header">
      <h3><i class="fas fa-flag"></i> Flag Claim Issue</h3>
      <button class="modal-close" onclick="closeRejectionModal()">&times;</button>
    </div>
    <div class="modal-body">
```

```
      <p>Please provide details about the issue with this claim:</p>
      <textarea id="rejectionReason" placeholder="Enter reason for flagging this claim..." rows="4"></textarea>
    </div>
    <div class="modal-actions">
      <button class="btn btn-secondary" onclick="closeRejectionModal()">Cancel</button>
      <button class="btn btn-danger" onclick="submitRejection()">Flag Claim</button>
    </div>
  </div>
</div>

<style>
  .dashboard-container {
    max-width: 1200px;
    margin: 0 auto;
    padding: 2rem;
  }

  .dashboard-header {
    display: flex;
    justify-content: space-between;
    align-items: center;
    margin-bottom: 2rem;
    gap: 2rem;
  }

  .page-title {
```

```
  font-size: 2.5rem;
  font-weight: 800;
  background: linear-gradient(135deg, #f8961e, #f9c74f);
  -webkit-background-clip: text;
  -webkit-text-fill-color: transparent;
  background-clip: text;
  margin: 0;
}

.page-subtitle {
  color: #6c757d;
  font-size: 1.1rem;
  margin: 0.5rem 0 0 0;
}

.header-stats {
  display: flex;
  gap: 1rem;
}

.stat-badge {
  display: inline-flex;
  align-items: center;
  gap: 0.5rem;
  padding: 0.75rem 1.5rem;
  border-radius: 50px;
  font-weight: 600;
  font-size: 0.9rem;
```

```
}

  .stat-badge.pending {
    background: rgba(248, 150, 30, 0.1);
    color: #f8961e;
    border: 1px solid rgba(248, 150, 30, 0.2);
  }

  .stat-badge.total {
    background: rgba(67, 97, 238, 0.1);
    color: #4361ee;
    border: 1px solid rgba(67, 97, 238, 0.2);
  }

.stats-grid {
  display: grid;
  grid-template-columns: repeat(auto-fit, minmax(250px, 1fr));
  gap: 1.5rem;
  margin-bottom: 2rem;
}

.stat-card {
  background: rgba(255, 255, 255, 0.95);
  backdrop-filter: blur(20px);
  border-radius: 16px;
  padding: 1.5rem;
  border: 1px solid rgba(255, 255, 255, 0.2);
  box-shadow: 0 8px 32px rgba(0, 0, 0, 0.1);
  display: flex;
  align-items: center;
  gap: 1rem;
  transition: all 0.3s ease;
  position: relative;
}

  .stat-card:hover {
    transform: translateY(-5px);
    box-shadow: 0 12px 40px rgba(0, 0, 0, 0.15);
  }

.stat-icon {
  width: 60px;
  height: 60px;
  border-radius: 12px;
  background: linear-gradient(135deg, #f8961e, #f9c74f);
  display: flex;
  align-items: center;
  justify-content: center;
  color: white;
  font-size: 1.5rem;
}

.stat-content {
  flex: 1;
}
```

```
.stat-number {
  font-size: 2rem;
  font-weight: 800;
  color: #2b2d42;
  margin: 0;
  line-height: 1;
}

.stat-label {
  color: #6c757d;
  margin: 0.25rem 0 0 0;
  font-size: 0.9rem;
}

.stat-badge.urgent {
  position: absolute;
  top: -8px;
  right: -8px;
  background: rgba(231, 76, 60, 0.1);
  color: #e74c3c;
  border: 1px solid rgba(231, 76, 60, 0.2);
  padding: 0.25rem 0.75rem;
  font-size: 0.7rem;
  border-radius: 20px;
}

.stat-trend {
  display: flex;
  align-items: center;
  gap: 0.25rem;
  padding: 0.25rem 0.75rem;
  border-radius: 20px;
  background: rgba(46, 204, 113, 0.1);
  color: #27ae60;
  font-size: 0.8rem;
  font-weight: 600;
}

  .stat-trend.trend-up {
    background: rgba(52, 152, 219, 0.1);
    color: #3498db;
  }

.card {
  background: rgba(255, 255, 255, 0.95);
  backdrop-filter: blur(20px);
  border-radius: 16px;
  border: 1px solid rgba(255, 255, 255, 0.2);
  box-shadow: 0 8px 32px rgba(0, 0, 0, 0.1);
  overflow: hidden;
  margin-bottom: 2rem;
}

.card-header {
  padding: 1.5rem;
  border-bottom: 1px solid rgba(0, 0, 0, 0.1);
  display: flex;
  justify-content: space-between;
  align-items: center;
}

  .card-header h3 {
    margin: 0;
    display: flex;
    align-items: center;
    gap: 0.5rem;
    color: #2b2d42;
  }

.card-body {
  padding: 1.5rem;
}

.filter-group {
  display: flex;
  gap: 1rem;
  align-items: center;
}

.filter-select, .search-input {
  padding: 0.5rem 1rem;
  border: 1px solid rgba(0, 0, 0, 0.1);
  border-radius: 8px;
  font-size: 0.9rem;
  background: white;
}

.claims-list {
  display: flex;
  flex-direction: column;
  gap: 1.5rem;
}

.claim-item {
  background: rgba(255, 255, 255, 0.8);
  border-radius: 12px;
  border: 1px solid rgba(0, 0, 0, 0.1);
  overflow: hidden;
  transition: all 0.3s ease;
}

  .claim-item:hover {
    transform: translateY(-2px);
    box-shadow: 0 8px 25px rgba(0, 0, 0, 0.1);
    border-color: rgba(248, 150, 30, 0.3);
  }

.claim-header {
  padding: 1.5rem;
  border-bottom: 1px solid rgba(0, 0, 0, 0.1);
  display: flex;
  justify-content: space-between;
  align-items: flex-start;
  gap: 1rem;
}

.claim-meta {
  display: flex;
  align-items: flex-start;
  gap: 1rem;
  flex: 1;
}

.lecturer-avatar {
  display: flex;
  align-items: center;
  gap: 1rem;
}

.avatar {
  width: 50px;
  height: 50px;
  border-radius: 50%;
  background: linear-gradient(135deg, #f8961e, #f9c74f);
  display: flex;
  align-items: center;
  justify-content: center;
  color: white;
  font-weight: 600;
  font-size: 1rem;
  flex-shrink: 0;
}

.lecturer-info h4 {
  margin: 0 0 0.25rem 0;
  color: #2b2d42;
  font-size: 1.1rem;
}

.lecturer-info p {
  margin: 0;
  color: #6c757d;
  font-size: 0.9rem;
}

.claim-amount {
  text-align: right;
  margin-left: auto;
}

.amount {
  display: block;
  font-size: 1.5rem;
  font-weight: 700;
  color: #27ae60;
}

.hours {
  display: block;
  color: #6c757d;
  font-size: 0.9rem;
}

.claim-status {
  display: flex;
  flex-direction: column;
  align-items: flex-end;
  gap: 0.5rem;
}

.status-badge {
  display: inline-flex;
  align-items: center;
  gap: 0.5rem;
  padding: 0.5rem 1rem;
  border-radius: 20px;
  font-size: 0.8rem;
  font-weight: 600;
  text-transform: uppercase;
  letter-spacing: 0.5px;
}

  .status-badge.pending {
    background: rgba(248, 150, 30, 0.1);
    color: #f8961e;
    border: 1px solid rgba(248, 150, 30, 0.2);
  }

.claim-date {
  color: #6c757d;
  font-size: 0.8rem;
  display: flex;
  align-items: center;
  gap: 0.25rem;
}

.claim-content {
  padding: 1.5rem;
  border-bottom: 1px solid rgba(0, 0, 0, 0.1);
}

.claim-title {
  margin: 0 0 1rem 0;
  color: #2b2d42;
  font-size: 1.2rem;
  font-weight: 600;
}

.claim-description {
  margin: 0 0 1rem 0;
  color: #6c757d;
  line-height: 1.5;
}
```

```
.claim-details {
  display: flex;
  gap: 2rem;
  flex-wrap: wrap;
}

.detail-item {
  display: flex;
  align-items: center;
  gap: 0.5rem;
  color: #6c757d;
  font-size: 0.9rem;
}

  .detail-item i {
    color: #f8961e;
    width: 16px;
  }

.claim-actions {
  padding: 1.5rem;
  display: flex;
  gap: 1rem;
  align-items: center;
  flex-wrap: wrap;
}

.action-form {
  margin: 0;
}

.btn {
  display: inline-flex;
  align-items: center;
  gap: 0.5rem;
  padding: 0.75rem 1.5rem;
  border: none;
  border-radius: 8px;
  font-weight: 600;
  text-decoration: none;
  cursor: pointer;
  transition: all 0.3s ease;
  font-size: 0.9rem;
}

.btn-success {
  background: linear-gradient(135deg, #27ae60, #2ecc71);
  color: white;
}

  .btn-success:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 12px rgba(39, 174, 96, 0.3);
  }

.btn-secondary {
  background: rgba(108, 117, 125, 0.1);
  color: #6c757d;
  border: 1px solid rgba(108, 117, 125, 0.2);
}

  .btn-secondary:hover {
    background: #6c757d;
    color: white;
    transform: translateY(-2px);
  }

.btn-outline {
  background: transparent;
  color: #4361ee;
  border: 1px solid #4361ee;
}

  .btn-outline:hover {
    background: #4361ee;
    color: white;
    transform: translateY(-2px);
  }

.btn-text {
  background: transparent;
  color: #6c757d;
  border: none;
}
```

```
  .btn-text:hover {
    color: #e74c3c;
    background: rgba(231, 76, 60, 0.1);
  }

.btn-icon {
  display: inline-flex;
  align-items: center;
  gap: 0.5rem;
}

.card-grid {
  display: grid;
  grid-template-columns: 2fr 1fr;
  gap: 1.5rem;
}

.department-stats {
  display: flex;
  flex-direction: column;
  gap: 1rem;
}

.department-item {
  display: flex;
  flex-direction: column;
  gap: 0.5rem;
  padding: 1rem;
  background: rgba(248, 150, 30, 0.05);
  border-radius: 8px;
}

.dept-info {
  display: flex;
  justify-content: space-between;
  align-items: center;
}

.dept-name {
  font-weight: 600;
  color: #2b2d42;
}

.dept-count {
  color: #6c757d;
  font-size: 0.9rem;
}

.dept-amount {
  font-weight: 700;
  color: #27ae60;
  font-size: 1.1rem;
}

.progress-bar {
  height: 6px;
  background: rgba(0, 0, 0, 0.1);
  border-radius: 3px;
  overflow: hidden;
}

.progress {
  height: 100%;
  background: linear-gradient(135deg, #f8961e, #f9c74f);
  border-radius: 3px;
  transition: width 0.3s ease;
}

.quick-actions-vertical {
  display: flex;
  flex-direction: column;
  gap: 1rem;
}

.quick-action {
  display: flex;
  align-items: center;
  gap: 1rem;
  padding: 1rem;
  background: rgba(102, 126, 234, 0.05);
  border-radius: 12px;
  text-decoration: none;
  color: #2b2d42;
  transition: all 0.3s ease;
  border: 1px solid transparent;
}

  .quick-action:hover {
    background: rgba(102, 126, 234, 0.1);
    border-color: rgba(102, 126, 234, 0.2);
    transform: translateX(5px);
    color: #4361ee;
  }

.action-icon {
  width: 50px;
  height: 50px;
  border-radius: 12px;
  background: linear-gradient(135deg, #667eea, #764ba2);
  display: flex;
  align-items: center;
  justify-content: center;
  color: white;
  font-size: 1.25rem;
  flex-shrink: 0;
}

.action-content {
  flex: 1;
}
```

```
.action-title {
   display: block;
   font-weight: 600;
   margin-bottom: 0.25rem;
}

.action-desc {
   display: block;
   color: #6c757d;
   font-size: 0.8rem;
}

.empty-state {
   text-align: center;
   padding: 3rem 2rem;
}

.empty-icon {
   font-size: 4rem;
   color: #27ae60;
   margin-bottom: 1rem;
}

.empty-state h3 {
   color: #2b2d42;
   margin-bottom: 1rem;
}
```

```
.empty-state p {
   color: #6c757d;
   margin-bottom: 2rem;
}

.empty-actions {
   display: flex;
   gap: 1rem;
   justify-content: center;
   flex-wrap: wrap;
}

.modal {
   display: none;
   position: fixed;
   top: 0;
   left: 0;
   width: 100%;
   height: 100%;
   background: rgba(0, 0, 0, 0.5);
   backdrop-filter: blur(5px);
   z-index: 1000;
   align-items: center;
   justify-content: center;
}

.modal-content {
   background: white;
```

```
  border-radius: 16px;

  width: 90%;

  max-width: 500px;

  box-shadow: 0 20px 60px rgba(0, 0, 0, 0.3);

  overflow: hidden;

}


.modal-header {

  padding: 1.5rem;

  border-bottom: 1px solid rgba(0, 0, 0, 0.1);

  display: flex;

  justify-content: space-between;

  align-items: center;

}


  .modal-header h3 {

    margin: 0;

    display: flex;

    align-items: center;

    gap: 0.5rem;

    color: #2b2d42;

  }


.modal-close {

  background: none;

  border: none;

  font-size: 1.5rem;

  cursor: pointer;
  color: #6c757d;
}

.modal-body {
  padding: 1.5rem;
}

  .modal-body textarea {
    width: 100%;
    padding: 1rem;
    border: 1px solid rgba(0, 0, 0, 0.1);
    border-radius: 8px;
    font-family: inherit;
    resize: vertical;
    margin-top: 1rem;
  }

.modal-actions {
  padding: 1.5rem;
  border-top: 1px solid rgba(0, 0, 0, 0.1);
  display: flex;
  gap: 1rem;
  justify-content: flex-end;
}

.btn-danger {
  background: linear-gradient(135deg, #e74c3c, #f72585);
  color: white;
```

```
}

  .btn-danger:hover {
    transform: translateY(-2px);
    box-shadow: 0 4px 12px rgba(231, 76, 60, 0.3);
  }

@@media (max-width: 768px) {
  .dashboard-container {
    padding: 1rem;
  }

  .dashboard-header {
    flex-direction: column;
    text-align: center;
  }

  .header-stats {
    justify-content: center;
  }

  .stats-grid {
    grid-template-columns: 1fr;
  }

  .card-header {
    flex-direction: column;
    gap: 1rem;
    align-items: stretch;
  }

  .filter-group {
    flex-direction: column;
  }

  .claim-header {
    flex-direction: column;
  }

  .claim-meta {
    flex-direction: column;
  }

  .claim-amount {
    margin-left: 0;
    text-align: left;
  }

  .claim-status {
    align-items: flex-start;
  }

  .claim-actions {
    flex-direction: column;
    align-items: stretch;
  }
```

```
      .card-grid {
        grid-template-columns: 1fr;
      }

      .empty-actions {
        flex-direction: column;
      }
    }
  </style>

  <script>
    let currentClaimId = null;

    function showRejectionModal(claimId) {
      currentClaimId = claimId;
      document.getElementById('rejectionModal').style.display = 'flex';
    }

    function closeRejectionModal() {
      document.getElementById('rejectionModal').style.display = 'none';
      document.getElementById('rejectionReason').value = '';
      currentClaimId = null;
    }

    function submitRejection() {
      const reason = document.getElementById('rejectionReason').value;
      if (!reason.trim()) {
```

```
      alert('Please provide a reason for flagging this claim.');
      return;
    }

    // Here you would typically send this to your server
    console.log(`Flagging claim ${currentClaimId}: ${reason}`);
    alert('Claim has been flagged for review. The lecturer will be notified.');
    closeRejectionModal();
  }

  // Close modal when clicking outside
  window.onclick = function(event) {
    const modal = document.getElementById('rejectionModal');
    if (event.target === modal) {
      closeRejectionModal();
    }
  }
</script>
```

Shared (Folder)

_Layout.cshtml

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8"
/>
  <meta
name="viewport"
```

```
content="width=device-
width" />
    <title>CMCS -
Contract Monthly Claim
System</title>
    <link rel="stylesheet"
href="/css/site.css" />
    <link rel="stylesheet"
href="https://cdnjs.clou
dflare.com/ajax/libs/font
-
awesome/6.4.0/css/all.
min.css">

    <!-- CSS Animations --
>
    <style>
        @@keyframes spin {
            0% {
                transform:
rotate(0deg);
            }

            100% {
                transform:
rotate(360deg);
            }
        }
    .pulse {
      animation: pulse 2s infinite;
    }

    @@keyframes pulse {
      0%, 100% {
        opacity: 1;
      }

      50% {
        opacity: 0.7;
      }
    }
  </style>
</head>

<body>
  <header class="site-header">
    <div class="container">
      <h1 class="logo">Contract Monthly Claim System (CMCS)</h1>
      <nav class="navbar">
```

```
        <a
href="/">Home</a>

        @if
(Context.Session.GetStri
ng("Role") == "Lecturer")
        {
          <a
href="/Lecturer">My
Claims</a>
          <a
href="/Lecturer/Create"
>Submit Claim</a>
        }
        else if
(Context.Session.GetStri
ng("Role") == "HR")
        {
          <a
href="/HR">User
Management</a>
          <a
href="/HR/Reports">Rep
orts</a>
        }
        else if
(Context.Session.GetStri
ng("Role") ==
"ProgrammeCoordinator
")
```

```
            {
               <a
href="/ProgrammeCoord
inator">Pending
Claims</a>
            }
            else if
(Context.Session.GetStri
ng("Role") ==
"AcademicManager")
            {
               <a
href="/AcademicManage
r">Verified Claims</a>
            }

            <a
href="/Claims">All
Claims</a>

            <!-- Logout form
-->
            <form
method="post"
action="@Url.Action("Lo
gout", "Login")"
class="logout-form">

@Html.AntiForgeryToke
n()
```

```
          <button
type="submit"
class="btn-logout">
            <i class="fas
fa-sign-out-alt"></i>
Logout
          </button>
        </form>
      </nav>
    </div>
  </header>

  <main
class="container">
    @RenderBody()
  </main>

  <footer class="site-
footer">
    <div
class="container">
      <small>Contract
Monthly Claim System
@DateTime.Now.Year</s
mall>
    </div>
  </footer>
```

```
   <script
src="https://cdnjs.cloud
flare.com/ajax/libs/jquer
y/3.6.0/jquery.min.js"></
script>
   <script
src="https://cdnjs.cloud
flare.com/ajax/libs/jquer
y-
validate/1.19.3/jquery.va
lidate.min.js"></script>
   <script
src="https://cdnjs.cloud
flare.com/ajax/libs/jquer
y-validation-
unobtrusive/3.2.12/jque
ry.validate.unobtrusive.
min.js"></script>

   <!-- Enhanced
interactions and
animations -->
   <script>

document.addEventList
ener('DOMContentLoad
ed', function() {
         // Add loading
animation to all buttons
except logout
```

```
        const buttons =
document.querySelecto
rAll('button:not(.btn-
logout), .btn-primary,
.btn-secondary');

buttons.forEach(button
=> {

button.addEventListene
r('click', function(e) {
            if ((this.type
=== 'submit' ||
this.getAttribute('type')
=== 'submit') &&
!this.classList.contains('
btn-logout')) {
                const
originalText =
this.innerHTML;

this.innerHTML = `
                    <span
class="loading-spinner"
style="display: inline-
flex; align-items:
center;">
                        <span
style="width: 16px;
height: 16px; border: 2px
solid transparent;
border-top: 2px solid
currentColor; border-
radius: 50%; animation:
spin 1s linear infinite;
margin-right:
8px;"></span>

Processing...
                  </span>
                `;
                this.disabled
= true;

                // Revert
after 3 seconds if still on
same page
                setTimeout(()
=> {

this.innerHTML =
originalText;

this.disabled = false;
                }, 3000);
             }
          });
        });

        // Add fade-in
animation to cards
```

```
        const cards =
document.querySelecto
rAll('.card');

cards.forEach((card,
index) => {

card.style.opacity = '0';

card.style.transform =
'translateY(20px)';
          setTimeout(() =>
{

card.style.transition =
'all 0.6s cubic-
bezier(0.4, 0, 0.2, 1)';

card.style.opacity = '1';

card.style.transform =
'translateY(0)';
          }, index * 100);
        });

        // Enhanced table
row interactions
        const tableRows =
document.querySelecto
```

```
rAll('.styled-table tbody
tr');

tableRows.forEach(row
=> {

row.addEventListener('
mouseenter', function() {

this.style.transform =
'scale(1.01)';
            });



row.addEventListener('
mouseleave', function() {

this.style.transform =
'scale(1)';
            });
         });


         // Auto-hide flash
messages
         const
flashMessages =
document.querySelecto
rAll('.alert');
```

```
flashMessages.forEach(
message => {
        setTimeout(() =>
{

message.style.transition
= 'all 0.5s ease';

message.style.opacity =
'0';

message.style.transform
= 'translateY(-20px)';
          setTimeout(()
=> message.remove(),
500);
        }, 5000);
      });
    });
  </script>

  @await
RenderSectionAsync("S
cripts", required: false)
</body>
</html>
```

Appsettings.json

```
{ "ConnectionStrings": { "CMCSDatabase":
"Server=lab7L95SR\SQLEXPRESS;Database=CMCS_New;Trusted_Connection=true;
TrustS erverCertificate=true;" }, "Logging": { "LogLevel": { "Default": "Information",
"Microsoft.AspNetCore": "Warning" } }, "AllowedHosts": "*" }
```

SQL database

```
-- Create Database (if it doesn't exist) IF NOT EXISTS (SELECT name FROM
sys.databases
WHERE name = 'CMCS_New') BEGIN CREATE DATABASE CMCS_New; END GO

USE CMCS_New; GO

-- Drop tables in correct order (due to foreign key constraints) IF EXISTS (SELECT *
FROM
INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME = 'ProofDocuments') DROP
TABLE ProofDocuments;

IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME =
'Claims') DROP TABLE Claims;

IF EXISTS (SELECT * FROM INFORMATION_SCHEMA.TABLES WHERE TABLE_NAME =
'Users') DROP TABLE Users; GO

-- Create Users table FIRST (no foreign key dependencies) CREATE TABLE Users ( Id
INT
IDENTITY(1,1) PRIMARY KEY, Username NVARCHAR(100) NOT NULL UNIQUE,
Password
NVARCHAR(255) NOT NULL, Role NVARCHAR(50) NOT NULL, Name NVARCHAR(100)
NOT
NULL, Surname NVARCHAR(100) NOT NULL, Email NVARCHAR(255) NOT NULL,
HourlyRate DECIMAL(18,2) NOT NULL DEFAULT 0, Department NVARCHAR(100),
CreatedDate DATETIME2 DEFAULT GETDATE(), IsActive BIT DEFAULT 1 ); GO

-- Create Claims Table (depends on Users) CREATE TABLE Claims ( Id INT
IDENTITY(1,1)
PRIMARY KEY, UserId INT NOT NULL, Title NVARCHAR(200) NOT NULL, Description
NVARCHAR(MAX), Hours INT NOT NULL DEFAULT 0, Rate DECIMAL(18,2) NOT NULL
DEFAULT 0, FilePath NVARCHAR(500), Status NVARCHAR(50) NOT NULL DEFAULT
'Pending', CreatedDate DATETIME2 DEFAULT GETDATE(), ModifiedDate DATETIME2
DEFAULT GETDATE(), FOREIGN KEY (UserId) REFERENCES Users(Id) ); GO

-- Create ProofDocuments Table (depends on Claims) CREATE TABLE
ProofDocuments ( Id
INT IDENTITY(1,1) PRIMARY KEY, ClaimId INT NOT NULL, FileName NVARCHAR(255)
NOT
NULL, FilePath NVARCHAR(500) NOT NULL, UploadedAt DATETIME2 DEFAULT
GETDATE(), FileSize BIGINT, FOREIGN KEY (ClaimId) REFERENCES Claims(Id) ON
DELETE CASCADE ); GO
-- Insert default users for each role INSERT INTO Users (Username, Password, Role,
Name,
Surname, Email, HourlyRate, Department) VALUES ('hr@cmcs.com', 'hr123', 'HR',
'HR',
'Manager', 'hr@cmcs.com', 0, 'Human Resources'), ('lecturer1@cmcs.com',
'lecturer123',
'Lecturer', 'John', 'Smith', 'lecturer1@cmcs.com', 85.00, 'Computer Science'),
('lecturer2@cmcs.com', 'lecturer123', 'Lecturer', 'Sarah', 'Johnson',
'lecturer2@cmcs.com', 75.00, 'Mathematics'), ('lecturer3@cmcs.com', 'lecturer123',
'Lecturer', 'Michael', 'Chen', 'lecturer3@cmcs.com', 90.00, 'Physics'),
('coordinator@cmcs.com', 'coordinator123', 'ProgrammeCoordinator', 'Programme',
'Coordinator', 'coordinator@cmcs.com', 0, 'Administration'), ('manager@cmcs.com',
'manager123', 'AcademicManager', 'Academic', 'Manager', 'manager@cmcs.com', 0,
'Administration'); GO

-- Insert Sample Claims (using UserId instead of LecturerId) INSERT INTO Claims
(UserId, Title, Description, Hours, Rate, Status) VALUES (2, 'August Research Hours',
'Research work for August 2025', 40, 85.00, 'Pending'), (3, 'Course Development', 'New
course material development', 30, 75.00, 'Verified'), (4, 'Student Supervision',
'Graduate student supervision', 20, 90.00, 'Approved'), (2, 'September Teaching',
'Undergraduate course teaching', 45, 85.00, 'Pending'), (3, 'Curriculum Design', 'New
curriculum development', 35, 75.00, 'Rejected'); GO

-- Verify tables and data SELECT 'Users table row count: ' + CAST(COUNT() AS
VARCHAR(10)) FROM Users UNION ALL SELECT 'Claims table row count: ' + CAST(COUNT()
AS VARCHAR(10)) FROM Claims UNION ALL SELECT 'ProofDocuments table row
count: ' + CAST(COUNT(*) AS VARCHAR(10)) FROM ProofDocuments; GO

-- Show table structure SELECT TABLE_NAME, COLUMN_NAME, DATA_TYPE,
IS_NULLABLE
FROM INFORMATION_SCHEMA.COLUMNS WHERE TABLE_NAME IN ('Users', 'Claims',
'ProofDocuments') ORDER BY TABLE_NAME, ORDINAL_POSITION; GO
```

```
-- Show sample data for verification SELECT '=== USERS ===' AS Info; SELECT Id, Username, Role, Name, Surname, Email, HourlyRate, Department FROM Users;

SELECT '=== CLAIMS ===' AS Info; SELECT c.Id, u.Name + ' ' + u.Surname AS Lecturer, c.Title, c.Hours, c.Rate, c.Amount, c.Status FROM Claims c INNER JOIN Users u ON c.UserId = u.Id; GO
```