بسم الله الرحمن الرحیم

گزارش پروژه ی درس مهندسی نرم افزار

فاز6 پیاده سازی نرم افزار

استاد درس:

دکتر مهدی سخایی نیا

ترم 4031

اسامی اعضای گروه:

فائزه حیدری

راحله بختیاری

زهرا دهقان

محدثه انصاری

فهرست

۱. مقدمه 3

۲. اهداف پروژه 3

۳. ساختار کلی سامانه 3

4.فناوری‌ها و ابزارهای مورد استفاده 4

5. نحوه کارکرد پروژه 4

6. تحلیل کدهای پروژه 5

کنترلرها (Controllers) 5

مدل‌های داده‌ای (Models) 11

Data (پیکربندی پایگاه داده) 13

Views (نمایش صفحات) 15

پایگاه داده (ApplicationDbContext.cs) 26

ارتباط کدهای پروژه با فرآیندهای وصول بیمه 27

۶. نتیجه‌گیری 33

۱. مقدمه

پروژه واحد وصول حق بیمه یک سامانه تحت وب برای مدیریت فرآیندهای بیمه‌ای کارفرمایان است که توسط ASP.NET Core MVC توسعه داده شده است. این سیستم به سازمان‌های بیمه‌ای کمک می‌کند تا فرآیندهای مربوط به تشکیل پرونده، دریافت و بررسی لیست بیمه، حسابرسی دفاتر قانونی، ثبت شکایات، تقسیط بدهی‌ها، صدور فیش‌های پرداختی و پیگیری مطالبات را به‌صورت یکپارچه و الکترونیکی مدیریت کنند.

در این گزارش، به بررسی ساختار فایل‌ها، تکنولوژی‌های استفاده شده، نحوه عملکرد سیستم، ارتباطات بین بخش‌های مختلف و پیاده‌سازی فرآیندهای وصول حق بیمه پرداخته می‌شود.

۲. اهداف پروژه

هدف از توسعه این سامانه:

1. بهینه‌سازی فرآیندهای بیمه‌ای: کاهش زمان پردازش پرونده‌ها، تسهیل دریافت اطلاعات و حذف فرآیندهای دستی.

2. افزایش دقت و شفافیت: مدیریت دقیق بدهی‌ها، جلوگیری از مغایرت‌ها و افزایش نظارت بر پرداخت‌ها.

3. اتصال به پایگاه‌های داده و ارائه گزارش‌های دقیق: امکان بررسی و تحلیل اطلاعات بدهی‌ها، پرداخت‌ها و شکایات.

4. ارتباط کارفرمایان با سازمان بیمه: ایجاد بستری برای تعامل الکترونیکی کارفرمایان با واحد وصول بیمه.

5. پیگیری سریع مطالبات و کاهش بدهی‌های معوقه: ارسال اخطاریه‌ها و تعیین مهلت پرداخت برای بدهکاران.

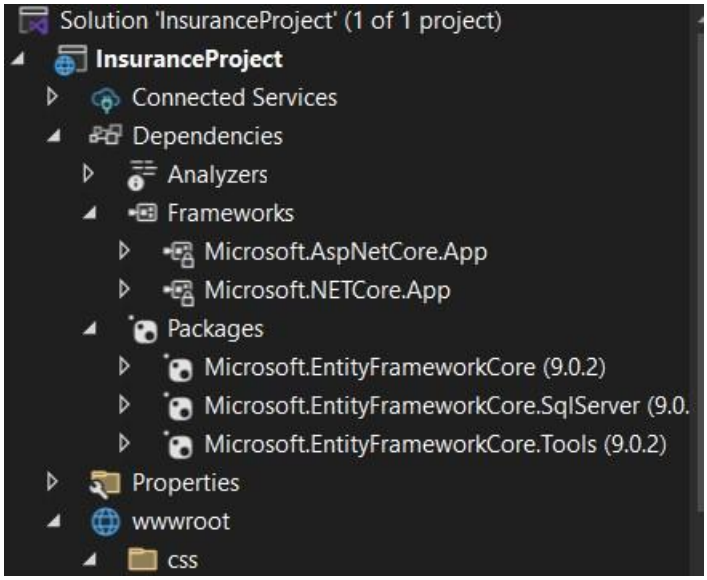
۳. ساختار کلی سامانه

پروژه شامل سه بخش اصلی است:

1. بخش بک‌اند (Back-end): شامل کنترلرها، مدل‌های داده‌ای، پایگاه داده و ارتباط با سرور.

2. بخش فرانت‌اند (Front-end): شامل ویوها (Razor Pages)، فرم‌های ورود داده و صفحات نمایش اطلاعات.

3. پایگاه داده (Database): ذخیره و مدیریت اطلاعات کارفرمایان، بدهی‌ها، پرداخت‌ها، اقساط، شکایات و فیش‌های پرداختی.



4.فناوری‌ها و ابزارهای مورد استفاده

زبان برنامه‌نویسی: C# (.NET Core 6+)

فریمورک بک‌اند: ASP.NET Core MVC

ORM: Entity Framework Core

پایگاه داده: SQL Server

فرانت‌اند: Razor Pages و HTML, CSS, Bootstrap

Dependency Injection: برای مدیریت سرویس‌ها و پایگاه داده

کنترل اعتبارسنجی: [ValidateAntiForgeryToken] برای جلوگیری از CSRF

5. نحوه کارکرد پروژه

5.۱. فرآیند کارفرما

1. ثبت اطلاعات کارفرما در سیستم

2. مشاهده و مدیریت بدهی‌های مربوط به کارفرما

3. پرداخت بدهی‌ها از طریق فیش‌های پرداختی

4. مشاهده و مدیریت پرونده‌های بیمه‌ای و شکایات

۵.۲. مدیریت شکایات

1. ثبت شکایت جدید

2. نمایش لیست شکایات ثبت شده

3. تغییر وضعیت شکایات (در حال بررسی، بررسی شده)

۵.۳. مدیریت بدهی و اقساط

1. تعریف بدهی جدید

2. ثبت اقساط مربوط به بدهی‌ها

3. نمایش وضعیت پرداخت اقساط

۵.۴. مدیریت پرداخت‌ها

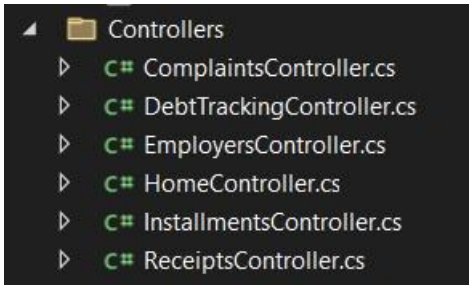
1. ثبت فیش‌های پرداختی

2. مشاهده و بررسی جزئیات پرداخت‌ها

6. تحلیل کدهای پروژه

کنترلرها (Controllers)

کنترلرها نقش واسط بین کاربران و پایگاه داده را دارند و شامل عملیات مربوط به ایجاد، خواندن، بروزرسانی و حذف داده‌ها (CRUD) هستند.



۱. ComplaintsController.cs (مدیریت شکایات)

نمایش لیست شکایات: درخواست‌ها از پایگاه داده دریافت و در ویو نمایش داده می‌شوند.

ایجاد شکایت جدید: فرم شکایت از طریق متد Create() پردازش می‌شود.

تغییر وضعیت شکایات: وضعیت شکایت‌ها به "در حال بررسی" یا "بررسی شده" تغییر می‌کند.

```
1 reference
public class ComplaintsController : Controller
{
    private readonly ApplicationDbContext _context;

    0 references
    public ComplaintsController(ApplicationDbContext context)
    {
        _context = context;
    }

    1 reference
    public async Task<IActionResult> Index()
    {
        var complaints = await _context.Complaints
            .Include(c => c.Employer)
            .ToListAsync();
        return View(complaints);
    }

    0 references
    public async Task<IActionResult> Create()
    {
        // دریافت لیست کارفرماها از پایگاه داده
        var employers = await _context.Employers.ToListAsync();
        // ارسال لیست کارفرماها به View
        ViewBag.Employers = new SelectList(employers, "EmployerId","EmployerName");
        return View();
    }


    [HttpPost]
    [ValidateAntiForgeryToken]
    0 references
    public async Task<IActionResult> Create([Bind("EmployerName,ComplaintDetails")] Complaint complaint)
    {
        if (ModelState.IsValid)
```

۲. DebtTrackingController.cs (مدیریت بدهی‌ها)

ثبت بدهی‌های کارفرمایان: بدهی‌های ایجاد شده در پایگاه داده ذخیره می‌شود.

نمایش لیست بدهی‌ها: کاربران می‌توانند بدهی‌های خود را مشاهده کنند.

تعیین مهلت پرداخت: تاریخ سررسید بدهی‌ها مشخص شده و در صورت عدم پرداخت، اخطاریه صادر می‌شود.

```
1 reference
public class DebtTrackingController : Controller
{
    private readonly ApplicationDbContext _context;

    0 references
    public DebtTrackingController(ApplicationDbContext context)
    {
        _context = context;
    }

    1 reference
    public async Task<IActionResult> Index()
    {
        var debtTrackings = await _context.DebtTrackings
            .Include(d => d.Employer)
            .ToListAsync();
        return View(debtTrackings);
    }

    0 references
    public async Task<IActionResult> Create()
    {
        // دریافت لیست کارفرماها از پایگاه داده
        var employers = await _context.Employers.ToListAsync();
        // ارسال لیست کارفرماها به View
        ViewBag.Employers = new SelectList(employers, "EmployerId", "EmployerName");
        return View();
    }


    [HttpPost]
    [ValidateAntiForgeryToken]
    0 references
    public async Task<IActionResult> Create([Bind("EmployerName,Amount,Status,NotificationDate,DeadlineDate,Remarks")] DebtTracking debtTracking)
    {
        if (ModelState.IsValid)
```

۳. EmployersController.cs (مدیریت کارفرمایان)

ایجاد و ثبت کارفرما: مشخصات کارفرمایان جدید در پایگاه داده ذخیره می‌شود.

نمایش لیست کارفرمایان: اطلاعات کارفرمایان قابل مشاهده و مدیریت است.

```
1 reference
public class EmployersController : Controller
{
    private readonly ApplicationDbContext _context;

    0 references
    public EmployersController(ApplicationDbContext context)
    {
        _context = context;
    }

    1 reference
    public async Task<IActionResult> Index()
    {
        return View(await _context.Employers.ToListAsync());
    }

    0 references
    public IActionResult Create()
    {
        return View();
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    0 references
    public async Task<IActionResult> Create([Bind("EmployerName,EmployerType,EmployerCode,Address,PhoneNumber,Email")] Employer employer)
    {
        if (ModelState.IsValid)
        {
            _context.Add(employer);
            await _context.SaveChangesAsync();
            return RedirectToAction(nameof(Index));
        }
        return View(employer);
    }
```

۴. InstallmentsController.cs (مدیریت اقساط)

ثبت اقساط جدید: کاربران می‌توانند برای بدهی‌های خود اقساط ثبت کنند.

مشاهده وضعیت پرداخت اقساط: امکان مشاهده اقساط پرداخت‌شده و باقی‌مانده وجود دارد.

```
{
    private readonly ApplicationDbContext _context;

    0 references
    public InstallmentsController(ApplicationDbContext context)
    {
        _context = context;
    }

    1 reference
    public async Task<IActionResult> Index()
    {
        var installments = await _context.Installments
            .Include(i => i.DebtTracking)
            .ToListAsync();
        return View(installments);
    }

    0 references
    public IActionResult Create()
    {
        return View();
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    0 references
    public async Task<IActionResult> Create([Bind("DebtId,Amount,DueDate,IsPaid")] Installment installment)
    {
        if (ModelState.IsValid)
        {
            _context.Add(installment);
            await _context.SaveChangesAsync();
            return RedirectToAction(nameof(Index));
        }

        // دریافت لیست کارفرماها از پایگاه داده
        var installments = await _context.Installments.ToListAsync();
```

```
        _context.Add(installment);
        await _context.SaveChangesAsync();
        return RedirectToAction(nameof(Index));
    }

    // دریافت لیست کارفرماها از پایگاه داده
    var installments = await _context.Installments.ToListAsync();
    // ارسال لیست کارفرماها به View
    ViewBag.Installments = new SelectList(installments, "DeptId");

    return View(installment);
}

0 references
public async Task<IActionResult> Details(int? id)
{
    if (id == null)
    {
        return NotFound();
    }

    var installment = await _context.Installments
        .Include(i => i.DebtTracking)
        .FirstOrDefaultAsync(m => m.InstallmentId == id);
    if (installment == null)
    {
        return NotFound();
    }

    return View(installment);
}
```

۵. ReceiptsController.cs (مدیریت فیش‌های پرداختی)

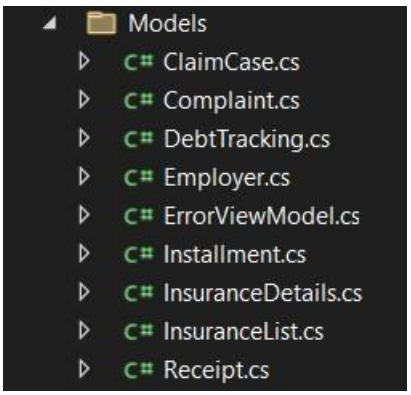
صدور فیش‌های پرداختی: فیش‌های بیمه‌ای به‌صورت الکترونیکی صادر می‌شوند.

مشاهده فیش‌های پرداخت شده: کاربران می‌توانند جزئیات فیش‌های خود را ببینند.

```
public class ReceiptsController : Controller
{
    private readonly ApplicationDbContext _context;

    0 references
    public ReceiptsController(ApplicationDbContext context)
    {
        _context = context;
    }

    1 reference
    public async Task<IActionResult> Index()
    {
        var receipts = await _context.Receipts
            .Include(r => r.Employer)
            .ToListAsync();
        return View(receipts);
    }

    0 references
    public async Task<IActionResult> Create()
    {
        // دریافت لیست کارفرماها از پایگاه داده
        var employers = await _context.Employers.ToListAsync();
        // ارسال لیست کارفرماها به View
        ViewBag.Employers = new SelectList(employers, "EmployerId", "EmployerName");

        return View();
    }

    [HttpPost]
    [ValidateAntiForgeryToken]
    0 references
    public async Task<IActionResult> Create([Bind("EmployerName,Amount,ReceiptType,ReceiptDate")] Receipt receipt)
    {
        if (ModelState.IsValid)
        {
            _context.Add(receipt);
            await _context.SaveChangesAsync();
            return RedirectToAction(nameof(Index));
        }

        // اگر مدل معتبر نباشد، دوباره لیست کارفرماها را به View ارسال کنید
        var employers = await _context.Employers.ToListAsync();
        ViewBag.Employers = new SelectList(employers, "EmployerId", "EmployerName");

        return View(receipt);
    }

    0 references
    public async Task<IActionResult> Details(int? id)
    {
        if (id == null)
        {
            return NotFound();
        }

        var receipt = await _context.Receipts
            .Include(r => r.Employer)
            .FirstOrDefaultAsync(m => m.ReceiptId == id);
        if (receipt == null)
        {
            return NotFound();
        }

        return View(receipt);
    }
```

مدل‌های داده‌ای (Models)

این مدل‌ها مستقیماً با جداول پایگاه داده مرتبط هستند و داده‌ها را سازماندهی می‌کنند.



1. Employer.cs: مشخصات کارفرما (نام، کد کارگاهی، نوع کارفرما، شماره تماس).

```
using System.Threading.Tasks;
using System.Collections.Generic;

16 references
public class Employer
{
    22 references
    public string EmployerName { get; set; }
    0 references
    public int EmployerId { get; set; }
    9 references
    public string EmployerType { get; set; } = null!; // حقیقی یا حقوقی
    9 references
    public string EmployerCode { get; set; } = null!;
    9 references
    public string Address { get; set; } = null!;
    9 references
    public string PhoneNumber { get; set; } = null!;
    8 references
    public string Email { get; set; } = null!;
    0 references
    public DateTime CreatedAt { get; set; }
    1 reference
    public ICollection<InsuranceList> InsuranceLists { get; set; } = new List<Insura
    1 reference
    public ICollection<ClaimCase> ClaimCases { get; set; } = new List<ClaimCase>();
    1 reference
    public ICollection<Complaint> Complaints { get; set; } = new List<Complaint>();
    1 reference
    public ICollection<DebtTracking> Debts { get; set; } = new List<DebtTracking>();
}
```

2. DebtTracking.cs: اطلاعات بدهی‌ها شامل مبلغ، وضعیت و مهلت پرداخت.

```
using Microsoft.EntityFrameworkCore;
using System.Threading.Tasks;
using System.Collections.Generic;

15 references
public class DebtTracking
{
    4 references
    public int DebtId { get; set; }
    5 references
    public string EmployerName { get; set; }
    5 references
    public Employer Employer { get; set; } = null!;
    11 references
    public decimal Amount { get; set; }
    9 references
    public string Status { get; set; } = null!;
    9 references
    public DateTime NotificationDate { get; set; }
    9 references
    public DateTime? DeadlineDate { get; set; }
    8 references
    public string Remarks { get; set; } = null!;
    0 references
    public DateTime StartDate { get; set; } // اضافه شده
    0 references
    public bool IsPaid { get; set; } // اضافه شده
    1 reference
    public ICollection<Installment> Installments { get; set; } = new List<Installment>();
}
```

3. Installment.cs: اطلاعات اقساط پرداختی، تاریخ سررسید و وضعیت پرداخت.

```
using Microsoft.EntityFrameworkCore;
using System.Threading.Tasks;
using System.Collections.Generic;

14 references
public class Installment
{
    5 references
    public int InstallmentId { get; set; }
    6 references
    public int DebtId { get; set; }
    5 references
    public DebtTracking DebtTracking { get; set; } = null!;
    9 references
    public decimal Amount { get; set; }
    9 references
    public DateTime DueDate { get; set; }
    9 references
    public bool IsPaid { get; set; }
}
```

4. Receipt.cs: اطلاعات فیش‌های پرداختی و نوع فیش بیمه‌ای.

```cs
using Microsoft.EntityFrameworkCore;
using System.Threading.Tasks;
using System.Collections.Generic;

11 references
public class Receipt
{
    5 references
    public int ReceiptId { get; set; }
    4 references
    public int EmployerName { get; set; }
    9 references
    public decimal Amount { get; set; }
    9 references
    public DateTime ReceiptDate { get; set; }
    9 references
    public string ReceiptType { get; set; } // حق بیمه، رانندگان، کارگران ساختمانی و غیره

    5 references
    public virtual Employer Employer { get; set; }
}
```

5. Complaint.cs: اطلاعات شکایات ثبت شده و وضعیت بررسی آن‌ها.

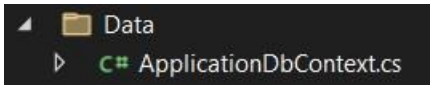
```cs
using Microsoft.EntityFrameworkCore;
using System.Threading.Tasks;
using System.Collections.Generic;

14 references
public class Complaint
{
    4 references
    public int ComplaintId { get; set; }
    6 references
    public string EmployerName { get; set; }
    5 references
    public Employer Employer { get; set; } = null!;
    4 references
    public DateTime ComplaintDate { get; set; }
    3 references
    public string ComplaintDetails { get; set; } = null!;
    7 references
    public string Status { get; set; } = null!; // در حال بررسی یا بررسی شده
    9 references
    public string Title { get; set; } = null!; // اضافه شده
    8 references
    public string Description { get; set; } = null!; // اضافه شده
}
```

Data (پیکربندی پایگاه داده)

فایل ApplicationDbContext.cs مسئول تنظیمات پایگاه داده است و شامل DbSetهایی برای جداول اصلی سیستم است.

از Entity Framework Core برای ORM استفاده شده است.

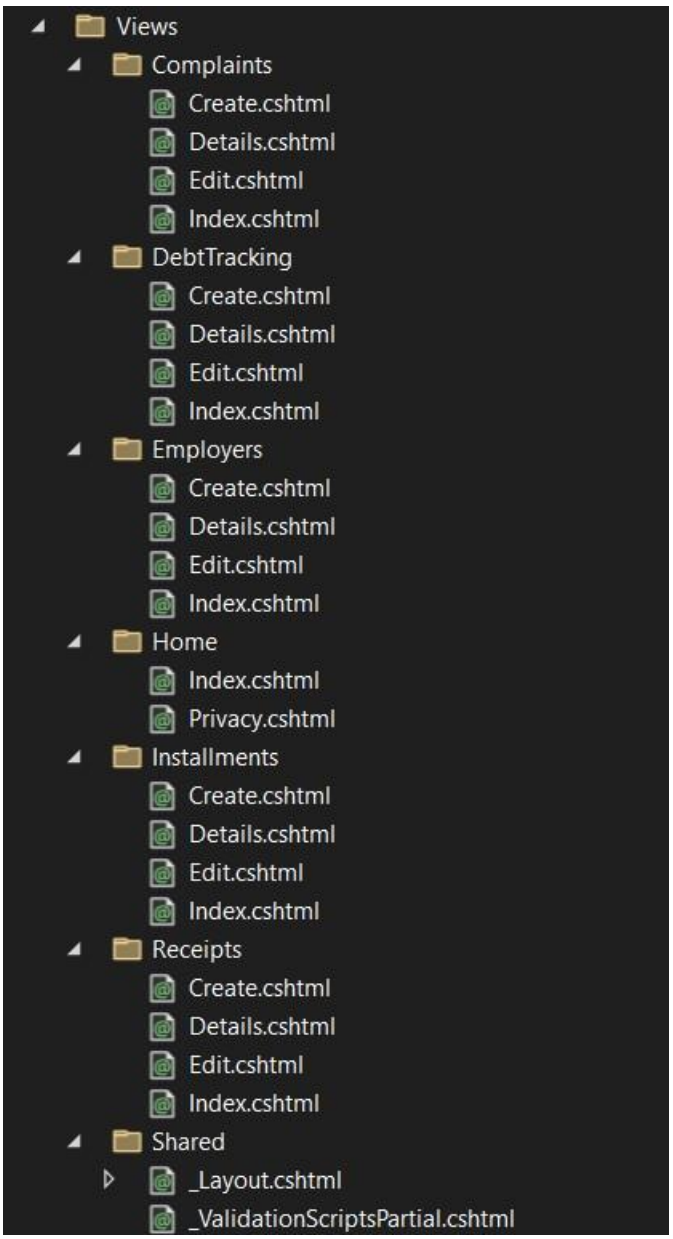
ارتباطات بین جداول با HasOne و HasMany تعریف شده‌اند.




```
public class ApplicationDbContext : DbContext
{
    0 references
    public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options)
    {
    }

    // جداول مربوط به سیستم
    7 references
    public DbSet<Employer> Employers { get; set; }
    0 references
    public DbSet<InsuranceList> InsuranceLists { get; set; }
    0 references
    public DbSet<ClaimCase> ClaimCases { get; set; }
    1 reference
    public DbSet<Complaint> Complaints { get; set; }
    3 references
    public DbSet<Installment> Installments { get; set; }
    2 references
    public DbSet<Receipt> Receipts { get; set; }
    1 reference
    public DbSet<DebtTracking> DebtTrackings { get; set; }

    0 references
    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        base.OnModelCreating(modelBuilder);

        // تعریف کلیدهای اصلی
        modelBuilder.Entity<Employer>().HasKey(e => e.EmployerName);
        modelBuilder.Entity<InsuranceList>().HasKey(il => il.InsuranceListId);
        modelBuilder.Entity<ClaimCase>().HasKey(cc => cc.ClaimCaseId);
        modelBuilder.Entity<Complaint>().HasKey(c => c.ComplaintId);
        modelBuilder.Entity<Installment>().HasKey(i => i.InstallmentId);
        modelBuilder.Entity<Receipt>().HasKey(r => r.ReceiptId);
        modelBuilder.Entity<DebtTracking>().HasKey(dt => dt.DebtId);

        // ارتباطات بین جداول
        modelBuilder.Entity<DebtTracking>().HasKey(dt => dt.DebtId);

        // ارتباطات بین جداول

        // ارتباط یک به چند بین کارفرما و لیست‌های بیمه
        modelBuilder.Entity<InsuranceList>()
            .HasOne(il => il.Employer)
            .WithMany(e => e.InsuranceLists)
            .HasForeignKey(il => il.EmployerName);

        // ارتباط یک به چند بین کارفرما و پرونده‌های مطالباتی
        modelBuilder.Entity<ClaimCase>()
            .HasOne(cc => cc.Employer)
            .WithMany(e => e.ClaimCases)
            .HasForeignKey(cc => cc.EmployerName);

        // ارتباط یک به چند بین کارفرما و شکایات
        modelBuilder.Entity<Complaint>()
            .HasOne(c => c.Employer)
            .WithMany(e => e.Complaints)
            .HasForeignKey(c => c.EmployerName);

        // ارتباط یک به چند بین بدهی و اقساط
        modelBuilder.Entity<Installment>()
            .HasOne(i => i.DebtTracking)
            .WithMany(dt => dt.Installments)
            .HasForeignKey(i => i.DebtId);

        // ارتباط یک به چند بین کارفرما و بدهی‌ها
        modelBuilder.Entity<DebtTracking>()
            .HasOne(dt => dt.Employer)
            .WithMany(e => e.Debts)
            .HasForeignKey(dt => dt.EmployerName);
    }
```

Views (نمایش صفحات)

صفحات فرانت‌اند در پوشه **Views** قرار دارند و از **Razor Pages** برای نمایش داده‌ها استفاده می‌کنند.



Views/Complaints: صفحات مدیریت شکایات

```
    ViewData["Title"] = "ایجاد شکایت جدید";
}

<h2>ایجاد شکایت جدید</h2>

<form asp-action="Create" method="post">
    <div class="form-group">
        <label asp-for="EmployerName" class="control-label"></label>
        <select asp-for="EmployerName" class="form-control" asp-items="ViewBag.Employers"></select>
        <span asp-validation-for="EmployerName" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="ComplaintDetails" class="control-label"></label>
        <textarea asp-for="ComplaintDetails" class="form-control"></textarea>
        <span asp-validation-for="ComplaintDetails" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Title" class="control-label"></label>
        <input asp-for="Title" class="form-control" />
        <span asp-validation-for="Title" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Description" class="control-label"></label>
        <textarea asp-for="Description" class="form-control"></textarea>
        <span asp-validation-for="Description" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
    </div>
</form>

<div>
    <a asp-action="Index">بازگشت به لیست</a>
</div>

@section Scripts {
    @{
        await Html.RenderPartialAsync("_ValidationScriptsPartial");
```

```
    <h4>شکایت</h4>
    <hr />
    <dl class="row">
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Employer.EmployerName)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Employer.EmployerName)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Title)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Title)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Description)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Description)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.ComplaintDate)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.ComplaintDate)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Status)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Status)
        </dd>
    </dl>
</div>
<div>
    <a asp-action="Edit" asp-route-id="@Model.ComplaintId">ویرایش</a> |
    <a asp-action="Index">بازگشت به لیست</a>
```

```
<h2>ویرایش شکایت</h2>

<form asp-action="Edit" method="post">
    <div class="form-group">
        <label asp-for="EmployerName" class="control-label"></label>
        <select asp-for="EmployerName" class="form-control" asp-items="ViewBag.Employers"></select>
    </div>
    <div class="form-group">
        <label asp-for="Title" class="control-label"></label>
        <input asp-for="Title" class="form-control" />
        <span asp-validation-for="Title" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Description" class="control-label"></label>
        <textarea asp-for="Description" class="form-control"></textarea>
        <span asp-validation-for="Description" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Status" class="control-label"></label>
        <input asp-for="Status" class="form-control" />
        <span asp-validation-for="Status" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
    </div>
</form>

<div>
    <a asp-action="Index">بازگشت به لیست</a>
</div>

@section Scripts {
    @{
        await Html.RenderPartialAsync("_ValidationScriptsPartial");
    }
```

```
@model IEnumerable<Complaint>

@{
    ViewData["Title"] = "لیست شکایات";
}

<h2>لیست شکایات</h2>

<table class="table">
    <thead>
        <tr>
            <th>نام کارفرما</th>
            <th>عنوان شکایت</th>
            <th>تاریخ ثبت</th>
            <th>وضعیت</th>
            <th>عملیات</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var complaint in Model)
        {
            <tr>
                <td>@complaint.Employer.EmployerName</td>
                <td>@complaint.Title</td>
                <td>@complaint.ComplaintDate.ToString("yyyy-MM-dd")</td>
                <td>@complaint.Status</td>
                <td>
                    <a asp-action="Details" asp-route-id="@complaint.ComplaintId">جزئیات</a> |
                    <a asp-action="Edit" asp-route-id="@complaint.ComplaintId">ویرایش</a>
                </td>
            </tr>
        }
    </tbody>
</table>

<a asp-action="Create" class="btn btn-primary">ایجاد شکایت جدید</a>
```

Views/DebtTracking: صفحات مدیریت بدهی‌ها

```
<h2>ایجاد بدهی جدید</h2>

<form asp-action="Create" method="post">
    <div class="form-group">
        <label asp-for="EmployerName" class="control-label"></label>
        <select asp-for="EmployerName" class="form-control" asp-items="ViewBag.Employers"></select>
    </div>
    <div class="form-group">
        <label asp-for="Amount" class="control-label"></label>
        <input asp-for="Amount" class="form-control" />
        <span asp-validation-for="Amount" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="NotificationDate" class="control-label"></label>
        <input asp-for="NotificationDate" class="form-control" type="date" />
        <span asp-validation-for="NotificationDate" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="DeadlineDate" class="control-label"></label>
        <input asp-for="DeadlineDate" class="form-control" type="date" />
        <span asp-validation-for="DeadlineDate" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Status" class="control-label"></label>
        <input asp-for="Status" class="form-control" />
        <span asp-validation-for="Status" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Remarks" class="control-label"></label>
        <textarea asp-for="Remarks" class="form-control"></textarea>
        <span asp-validation-for="Remarks" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
    </div>
```

```
<h4>بدهی</h4>
<hr />
<dl class="row">
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.Employer.EmployerName)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.Employer.EmployerName)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.Amount)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.Amount)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.NotificationDate)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.NotificationDate)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.DeadlineDate)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.DeadlineDate)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.Status)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.Status)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.Remarks)
    </dt>
    <dd class="col-sm-10">
```

```razor
<h2>ویرایش بدهی</h2>

<form asp-action="Edit" method="post">
    <div class="form-group">
        <label asp-for="EmployerName" class="control-label"></label>
        <select asp-for="EmployerName" class="form-control" asp-items="ViewBag.Employers"></select>
    </div>
    <div class="form-group">
        <label asp-for="Amount" class="control-label"></label>
        <input asp-for="Amount" class="form-control" />
        <span asp-validation-for="Amount" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="NotificationDate" class="control-label"></label>
        <input asp-for="NotificationDate" class="form-control" type="date" />
        <span asp-validation-for="NotificationDate" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="DeadlineDate" class="control-label"></label>
        <input asp-for="DeadlineDate" class="form-control" type="date" />
        <span asp-validation-for="DeadlineDate" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Status" class="control-label"></label>
        <input asp-for="Status" class="form-control" />
        <span asp-validation-for="Status" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Remarks" class="control-label"></label>
        <textarea asp-for="Remarks" class="form-control"></textarea>
        <span asp-validation-for="Remarks" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
    </div>
</form>
```

```razor
@{
    ViewData["Title"] = "لیست بدهی‌ها";
}

<h2>لیست بدهی‌ها</h2>

<table class="table">
    <thead>
        <tr>
            <th>نام کارفرما</th>
            <th>مبلغ بدهی</th>
            <th>تاریخ اخطار</th>
            <th>تاریخ مهلت</th>
            <th>وضعیت</th>
            <th>عملیات</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var debt in Model)
        {
            <tr>
                <td>@debt.Employer.EmployerName</td>
                <td>@debt.Amount</td>
                <td>@debt.NotificationDate.ToString("yyyy-MM-dd")</td>
                <td>@debt.DeadlineDate?.ToString("yyyy-MM-dd")</td>
                <td>@debt.Status</td>
                <td>
                    <a asp-action="Details" asp-route-id="@debt.DebtId">جزئیات</a> |
                    <a asp-action="Edit" asp-route-id="@debt.DebtId">ویرایش</a>
                </td>
            </tr>
        }
    </tbody>
</table>

<a asp-action="Create" class="btn btn-primary">ایجاد بدهی جدید</a>
```

Views/Employers: صفحات مربوط به کارفرمایان

```
<h2>ایجاد کارفرما جدید</h2>

<form asp-action="Create" method="post">
    <div class="form-group">
        <label asp-for="EmployerName" class="control-label"></label>
        <input asp-for="EmployerName" class="form-control" />
        <span asp-validation-for="EmployerName" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="EmployerType" class="control-label"></label>
        <select asp-for="EmployerType" class="form-control">
            <option value="حقیقی">حقیقی</option>
            <option value="حقوقی">حقوقی</option>
        </select>
        <span asp-validation-for="EmployerType" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="EmployerCode" class="control-label"></label>
        <input asp-for="EmployerCode" class="form-control" />
        <span asp-validation-for="EmployerCode" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Address" class="control-label"></label>
        <input asp-for="Address" class="form-control" />
        <span asp-validation-for="Address" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="PhoneNumber" class="control-label"></label>
        <input asp-for="PhoneNumber" class="form-control" />
        <span asp-validation-for="PhoneNumber" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Email" class="control-label"></label>
        <input asp-for="Email" class="form-control" />
        <span asp-validation-for="Email" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
```

```
<h4>کارفرما</h4>
<hr />
<dl class="row">
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.EmployerName)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.EmployerName)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.EmployerType)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.EmployerType)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.EmployerCode)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.EmployerCode)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.Address)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.Address)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.PhoneNumber)
    </dt>
    <dd class="col-sm-10">
        @Html.DisplayFor(model => model.PhoneNumber)
    </dd>
    <dt class="col-sm-2">
        @Html.DisplayNameFor(model => model.Email)
    </dt>
    <dd class="col-sm-10">
```

```
<h2>ویرایش کارفرما</h2>

<form asp-action="Edit" method="post">
    <div class="form-group">
        <label asp-for="EmployerName" class="control-label"></label>
        <input asp-for="EmployerName" class="form-control" />
        <span asp-validation-for="EmployerName" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="EmployerType" class="control-label"></label>
        <select asp-for="EmployerType" class="form-control">
            <option value="حقیقی">حقیقی</option>
            <option value="حقوقی">حقوقی</option>
        </select>
        <span asp-validation-for="EmployerType" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="EmployerCode" class="control-label"></label>
        <input asp-for="EmployerCode" class="form-control" />
        <span asp-validation-for="EmployerCode" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Address" class="control-label"></label>
        <input asp-for="Address" class="form-control" />
        <span asp-validation-for="Address" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="PhoneNumber" class="control-label"></label>
        <input asp-for="PhoneNumber" class="form-control" />
        <span asp-validation-for="PhoneNumber" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Email" class="control-label"></label>
        <input asp-for="Email" class="form-control" />
        <span asp-validation-for="Email" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
```

```
@{
    ViewData["Title"] = "لیست کارفرمایان";
}

<h2>لیست کارفرمایان</h2>

<table class="table">
    <thead>
        <tr>
            <th>نام کارفرما</th>
            <th>نوع کارفرما</th>
            <th>کد کارگاهی</th>
            <th>آدرس</th>
            <th>شماره تلفن</th>
            <th>عملیات</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var employer in Model)
        {
            <tr>
                <td>@employer.EmployerName</td>
                <td>@employer.EmployerType</td>
                <td>@employer.EmployerCode</td>
                <td>@employer.Address</td>
                <td>@employer.PhoneNumber</td>
                <td>
                    <a asp-action="Details" asp-route-id="@employer.EmployerName">جزئیات</a> |
                    <a asp-action="Edit" asp-route-id="@employer.EmployerName">ویرایش</a>
                </td>
            </tr>
        }
    </tbody>
</table>

<a asp-action="Create" class="btn btn-primary">ایجاد کارفرما جدید</a>
```

Views/Installment: صفحات مدیریت اقساط

```razor
<h2>ایجاد قسط جدید</h2>

<form asp-action="Create" method="post">
    <div class="form-group">
        <label asp-for="DebtId" class="control-label"></label>
        <select asp-for="DebtId" class="form-control" asp-items="ViewBag.Debts"></select>
        <span asp-validation-for="DebtId" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Amount" class="control-label"></label>
        <input asp-for="Amount" class="form-control" />
        <span asp-validation-for="Amount" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="DueDate" class="control-label"></label>
        <input asp-for="DueDate" class="form-control" type="date" />
        <span asp-validation-for="DueDate" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="IsPaid" class="control-label"></label>
        <input asp-for="IsPaid" class="form-control" type="checkbox" />
        <span asp-validation-for="IsPaid" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
    </div>
</form>

<div>
    <a asp-action="Index">بازگشت به لیست</a>
</div>

@section Scripts {
    @{
        await Html.RenderPartialAsync("_ValidationScriptsPartial");
    }
}
```

```razor
<h2>جزئیات قسط</h2>

<div>
    <h4>قسط</h4>
    <hr />
    <dl class="row">
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.DebtTracking.Amount)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.DebtTracking.Amount)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Amount)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Amount)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.DueDate)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.DueDate)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.IsPaid)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.IsPaid)
        </dd>
    </dl>
</div>
<div>
    <a asp-action="Edit" asp-route-id="@Model.InstallmentId">ویرایش</a> |
    <a asp-action="Index">بازگشت به لیست</a>
</div>
```

```
    ViewData["Title"] = "ویرایش قسط";
}

<h2>ویرایش قسط</h2>

<form asp-action="Edit" method="post">
    <div class="form-group">
        <label asp-for="DebtId" class="control-label"></label>
        <select asp-for="DebtId" class="form-control" asp-items="ViewBag.Debts"></select>
    </div>
    <div class="form-group">
        <label asp-for="Amount" class="control-label"></label>
        <input asp-for="Amount" class="form-control" />
        <span asp-validation-for="Amount" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="DueDate" class="control-label"></label>
        <input asp-for="DueDate" class="form-control" type="date" />
        <span asp-validation-for="DueDate" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="IsPaid" class="control-label"></label>
        <input asp-for="IsPaid" class="form-control" type="checkbox" />
        <span asp-validation-for="IsPaid" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
    </div>
</form>

<div>
    <a asp-action="Index">بازگشت به لیست</a>
</div>

@section Scripts {
    @{
        await Html.RenderPartialAsync("_ValidationScriptsPartial");
    }
```

```
@model IEnumerable<Installment>

@{
    ViewData["Title"] = "لیست اقساط";
}

<h2>لیست اقساط</h2>

<table class="table">
    <thead>
        <tr>
            <th>مبلغ قسط</th>
            <th>تاریخ سررسید</th>
            <th>وضعیت پرداخت</th>
            <th>عملیات</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var installment in Model)
        {
            <tr>
                <td>@installment.Amount</td>
                <td>@installment.DueDate.ToString("yyyy-MM-dd")</td>
                <td>@(installment.IsPaid ? "پرداخت شده" : "پرداخت نشده")</td>
                <td>
                    <a asp-action="Details" asp-route-id="@installment.InstallmentId">جزئیات</a> |
                    <a asp-action="Edit" asp-route-id="@installment.InstallmentId">ویرایش</a>
                </td>
            </tr>
        }
    </tbody>
</table>

<a asp-action="Create" class="btn btn-primary">ایجاد قسط جدید</a>
```

Views/Receipts: صفحات نمایش و ایجاد فیش‌های پرداختی

```
@{
    ViewData["Title"] = "ایجاد فیش جدید";
}

<h2>ایجاد فیش جدید</h2>

<form asp-action="Create" method="post">
    <div class="form-group">
        <label asp-for="EmployerName" class="control-label"></label>
        <select asp-for="EmployerName" class="form-control" asp-items="ViewBag.Employers"></select>
    </div>
    <div class="form-group">
        <label asp-for="ReceiptType" class="control-label"></label>
        <select asp-for="ReceiptType" class="form-control">
            <option value="حق بیمه">حق بیمه</option>
            <option value="رانندگان">رانندگان</option>
            <option value="کارگران ساختمانی">کارگران ساختمانی</option>
            <option value="گواهی ساختمانی">گواهی ساختمانی</option>
        </select>
        <span asp-validation-for="ReceiptType" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Amount" class="control-label"></label>
        <input asp-for="Amount" class="form-control" />
        <span asp-validation-for="Amount" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="ReceiptDate" class="control-label"></label>
        <input asp-for="ReceiptDate" class="form-control" type="date" />
        <span asp-validation-for="ReceiptDate" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
    </div>
</form>
```

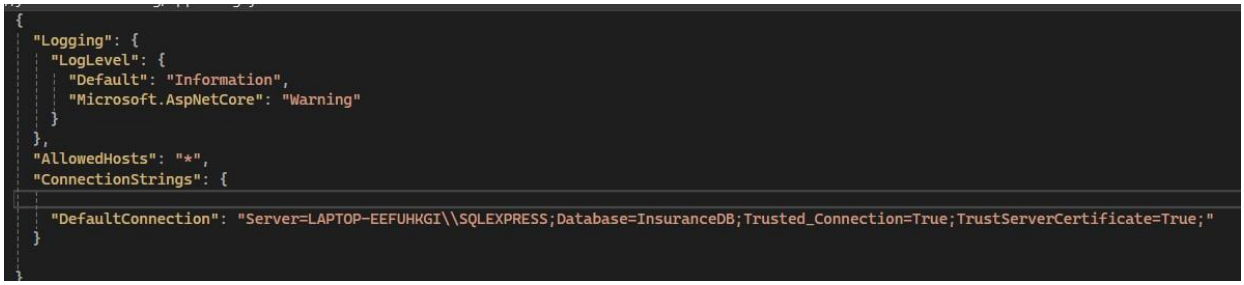
```
@{
    ViewData["Title"] = "جزئیات فیش";
}

<h2>جزئیات فیش</h2>

<div>
    <h4>فیش</h4>
    <hr />
    <dl class="row">
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Employer.EmployerName)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Employer.EmployerName)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.ReceiptType)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.ReceiptType)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.Amount)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.Amount)
        </dd>
        <dt class="col-sm-2">
            @Html.DisplayNameFor(model => model.ReceiptDate)
        </dt>
        <dd class="col-sm-10">
            @Html.DisplayFor(model => model.ReceiptDate)
        </dd>
    </dl>
</div>
```

```
<h2>ویرایش فیش</h2>

<form asp-action="Edit" method="post">
    <div class="form-group">
        <label asp-for="EmployerName" class="control-label"></label>
        <select asp-for="EmployerName" class="form-control" asp-items="ViewBag.Employers"></select>
    </div>
    <div class="form-group">
        <label asp-for="ReceiptType" class="control-label"></label>
        <select asp-for="ReceiptType" class="form-control">
            <option value="حق بیمه">حق بیمه</option>
            <option value="رانندگان">رانندگان</option>
            <option value="کارگران ساختمانی">کارگران ساختمانی</option>
            <option value="گواهی ساختمانی">گواهی ساختمانی</option>
        </select>
        <span asp-validation-for="ReceiptType" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="Amount" class="control-label"></label>
        <input asp-for="Amount" class="form-control" />
        <span asp-validation-for="Amount" class="text-danger"></span>
    </div>
    <div class="form-group">
        <label asp-for="ReceiptDate" class="control-label"></label>
        <input asp-for="ReceiptDate" class="form-control" type="date" />
        <span asp-validation-for="ReceiptDate" class="text-danger"></span>
    </div>
    <div class="form-group">
        <input type="submit" value="ذخیره" class="btn btn-primary" />
    </div>
</form>

<div>
    <a asp-action="Index">بازگشت به لیست</a>
</div>
```

```
@model IEnumerable<Receipt>

@{
    ViewData["Title"] = "لیست فیش‌ها";
}

<h2>لیست فیش‌ها</h2>

<table class="table">
    <thead>
        <tr>
            <th>نام کارفرما</th>
            <th>نوع فیش</th>
            <th>مبلغ</th>
            <th>تاریخ</th>
            <th>عملیات</th>
        </tr>
    </thead>
    <tbody>
        @foreach (var receipt in Model)
        {
            <tr>
                <td>@receipt.Employer.EmployerName</td>
                <td>@receipt.ReceiptType</td>
                <td>@receipt.Amount</td>
                <td>@receipt.ReceiptDate.ToString("yyyy-MM-dd")</td>
                <td>
                    <a asp-action="Details" asp-route-id="@receipt.ReceiptId">جزئیات</a> |
                    <a asp-action="Edit" asp-route-id="@receipt.ReceiptId">ویرایش</a>
                </td>
            </tr>
        }
    </tbody>
</table>

<a asp-action="Create" class="btn btn-primary">ایجاد فیش جدید</a>
```

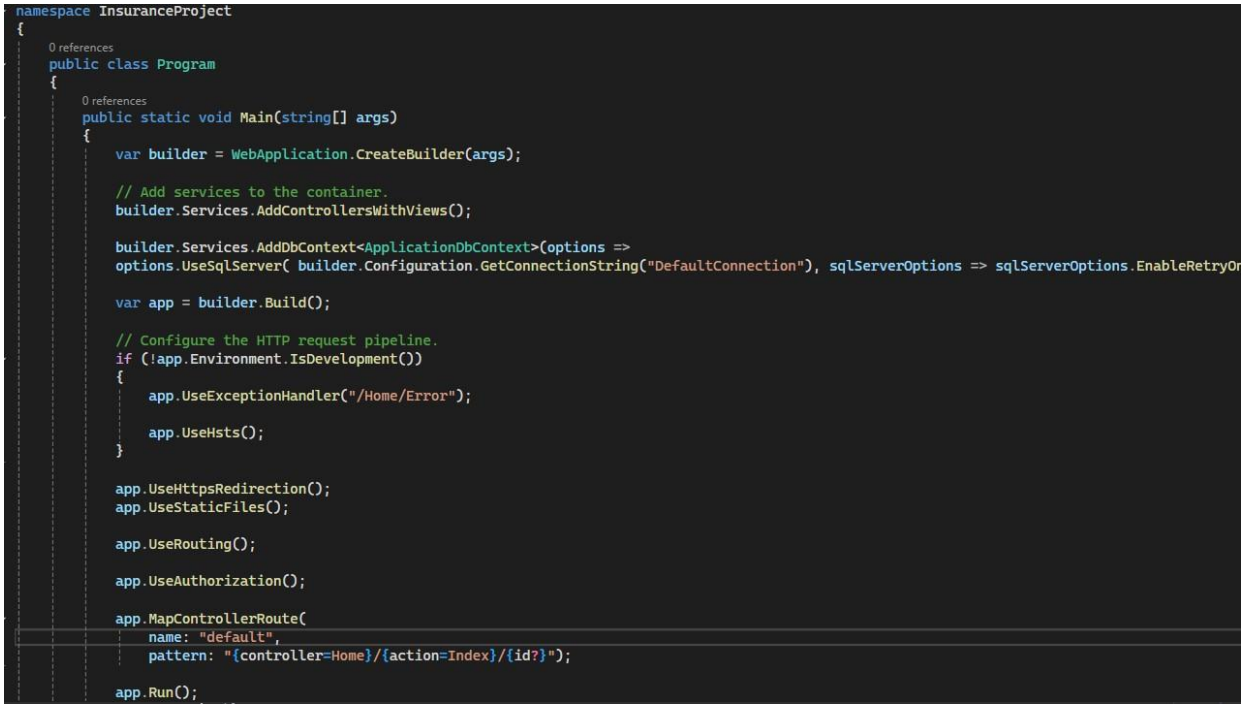
پیکربندی و تنظیمات

appsettings.json: شامل تنظیمات اتصال به پایگاه داده SQL Server



```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {

    "DefaultConnection": "Server=LAPTOP-EEFUHKGI\\SQLEXPRESS;Database=InsuranceDB;Trusted_Connection=True;TrustServerCertificate=True;"
  }

}
```

Program.cs: تنظیمات راه‌اندازی برنامه، شامل تنظیمات MVC و پایگاه داده



```
namespace InsuranceProject
{
    0 references
    public class Program
    {
        0 references
        public static void Main(string[] args)
        {
            var builder = WebApplication.CreateBuilder(args);

            // Add services to the container.
            builder.Services.AddControllersWithViews();

            builder.Services.AddDbContext<ApplicationDbContext>(options =>
            options.UseSqlServer( builder.Configuration.GetConnectionString("DefaultConnection"), sqlServerOptions => sqlServerOptions.EnableRetryOn

            var app = builder.Build();

            // Configure the HTTP request pipeline.
            if (!app.Environment.IsDevelopment())
            {
                app.UseExceptionHandler("/Home/Error");

                app.UseHsts();
            }

            app.UseHttpsRedirection();
            app.UseStaticFiles();

            app.UseRouting();

            app.UseAuthorization();

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

            app.Run();
```

پایگاه داده (ApplicationDbContext.cs)

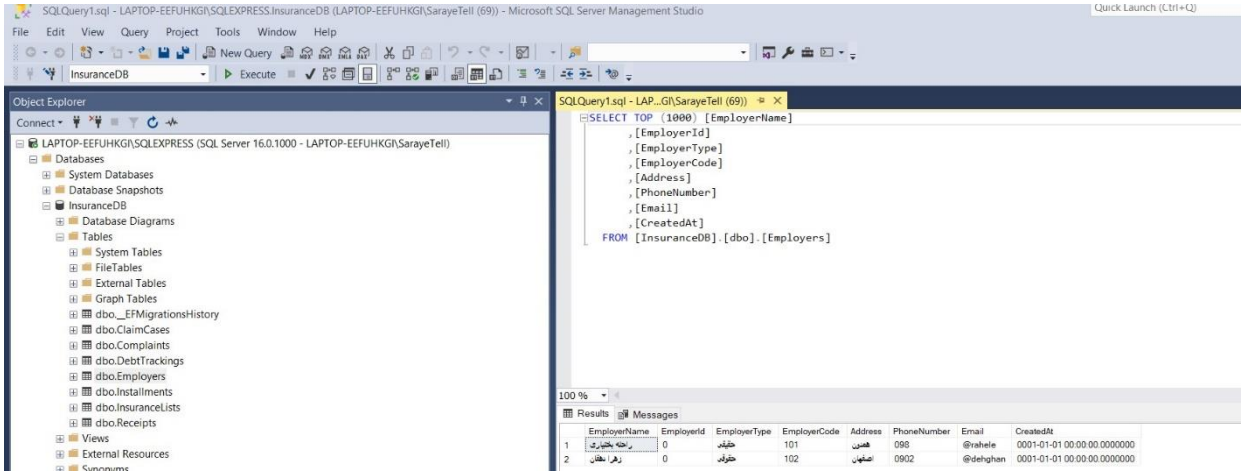
در این فایل ارتباط بین مدل‌ها و جداول پایگاه داده مشخص شده است.

مهم‌ترین روابط پایگاه داده:

کارفرما -> بدهی‌ها (یک به چند)

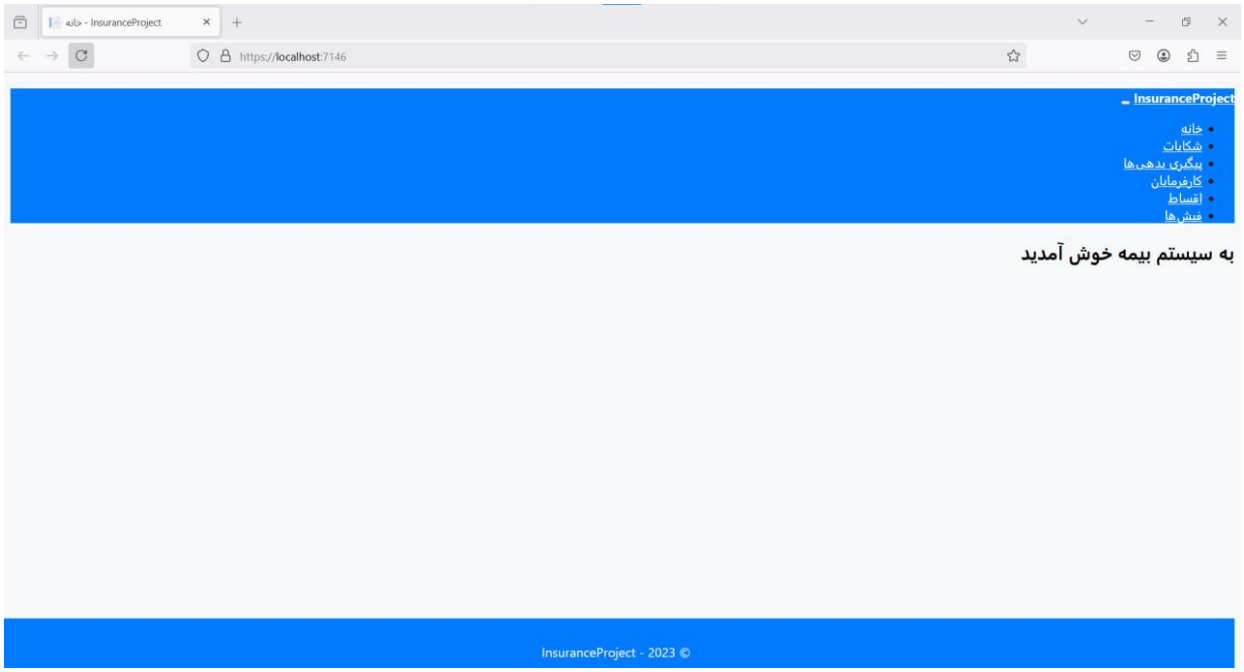
بدهی -> اقساط (یک به چند)

کارفرما -> فیش‌های پرداختی (یک به چند)



ارتباط کدهای پروژه با فرآیندهای وصول بیمه

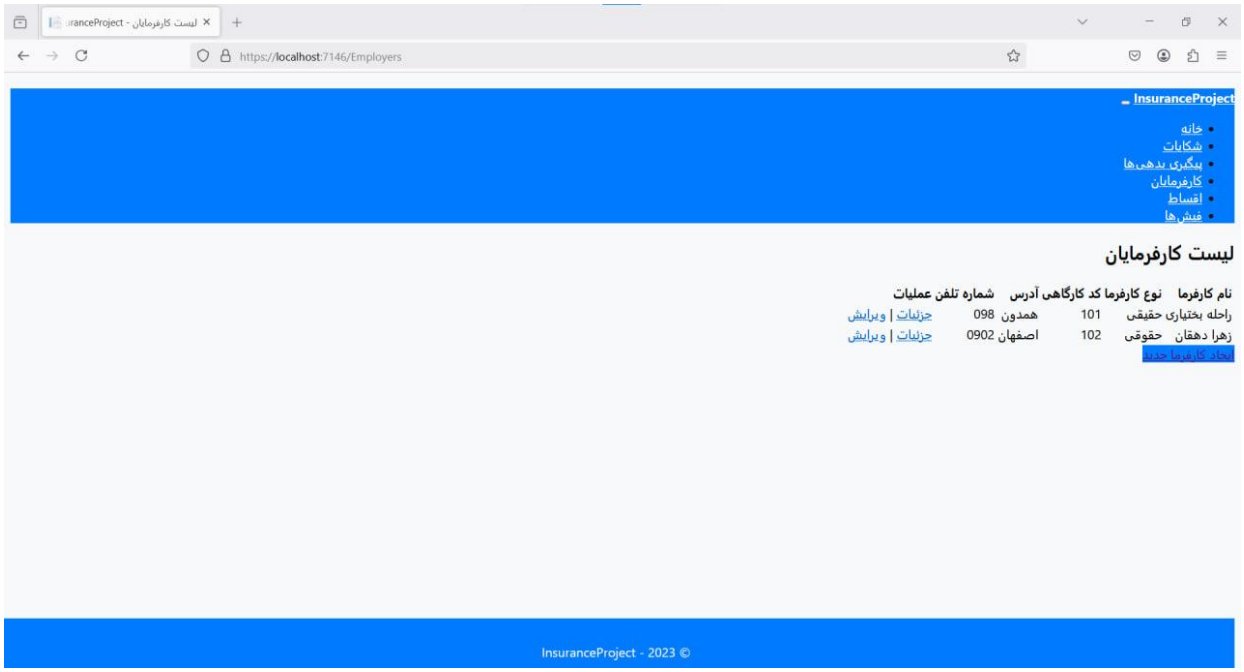
بر اساس سند فرآیند وصول حق بیمه، این پروژه مراحل مختلف را پیاده‌سازی کرده است:

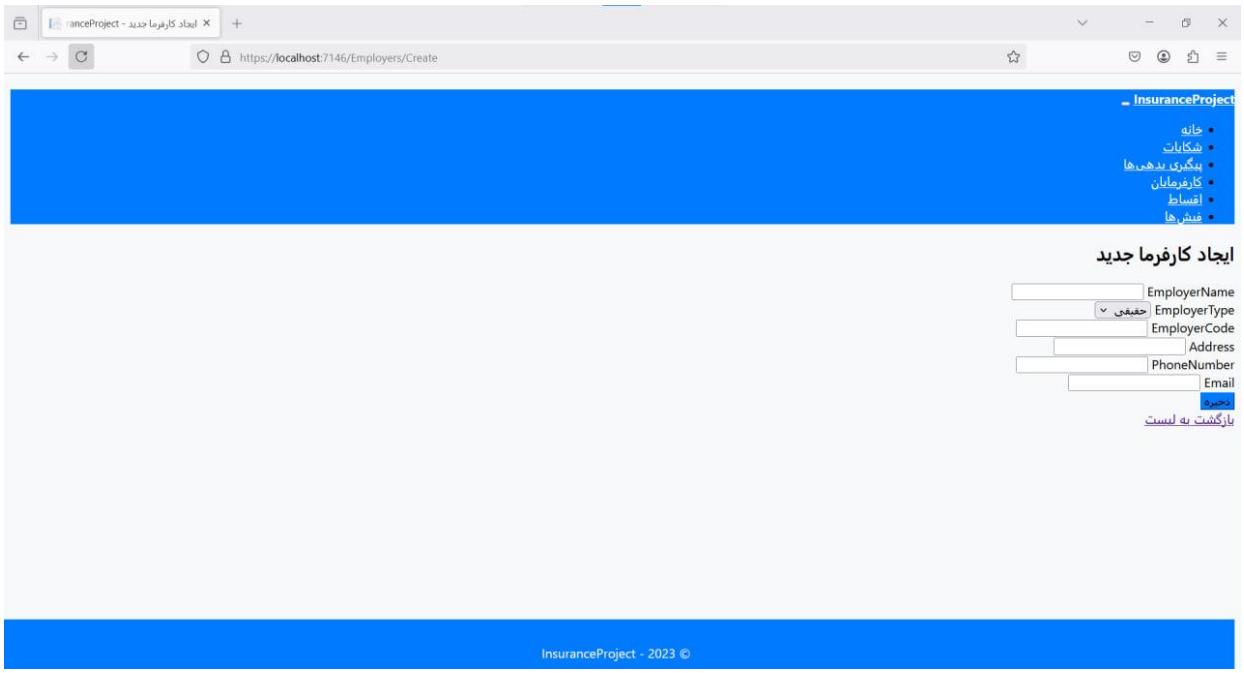


۱. تشکیل پرونده مطالباتی

در EmployersController.cs امکان ثبت اطلاعات کارفرمایان فراهم شده است.

اطلاعات مربوط به کد کارگاهی، فعالیت کارگاه، مشمول بیمه بودن و تاریخ ایجاد کارگاه ذخیره می‌شود.

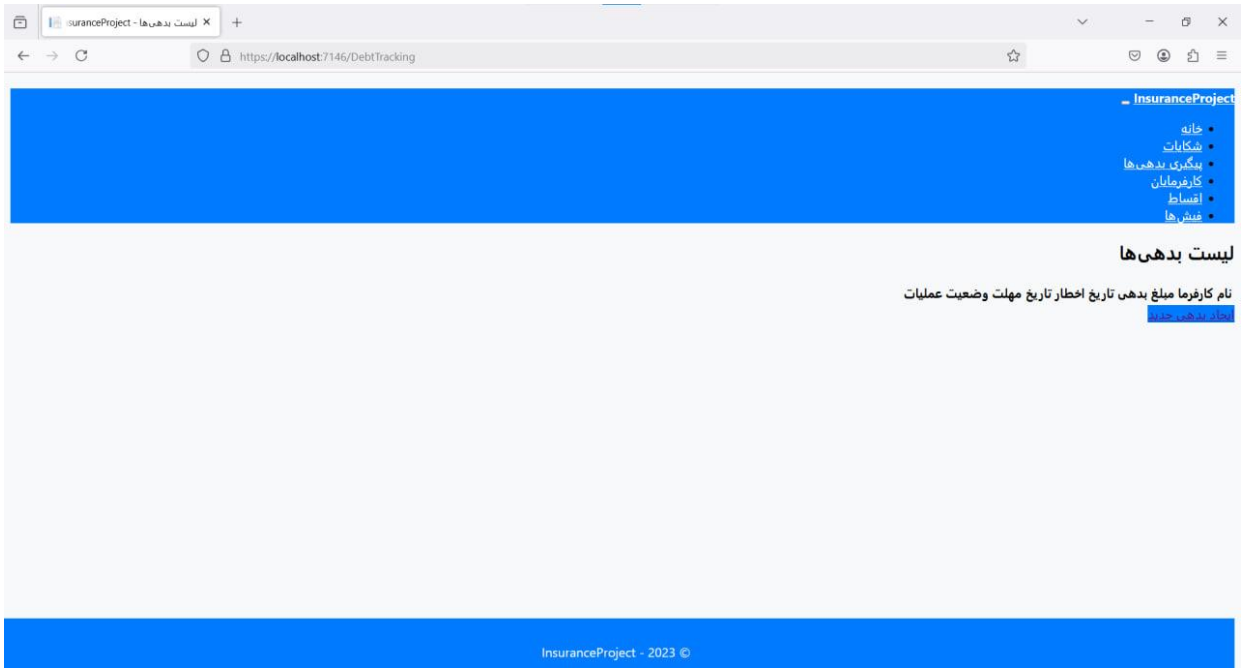


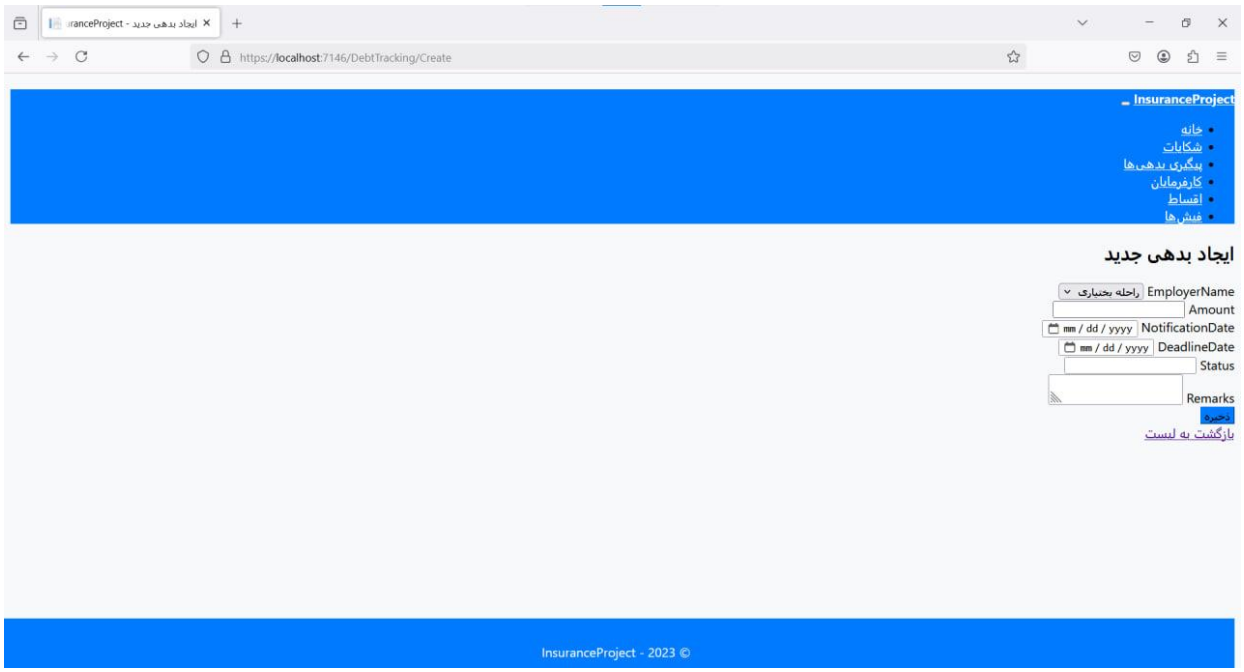


۲. دریافت لیست بیمه و حق بیمه

در DebtTrackingController.cs، اطلاعات بدهی‌ها ثبت شده و قابلیت بررسی لیست‌های بیمه‌ای فراهم شده است.

جرائم عدم ارسال لیست (۱۰٪ جریمه) محاسبه و ثبت می‌شود.





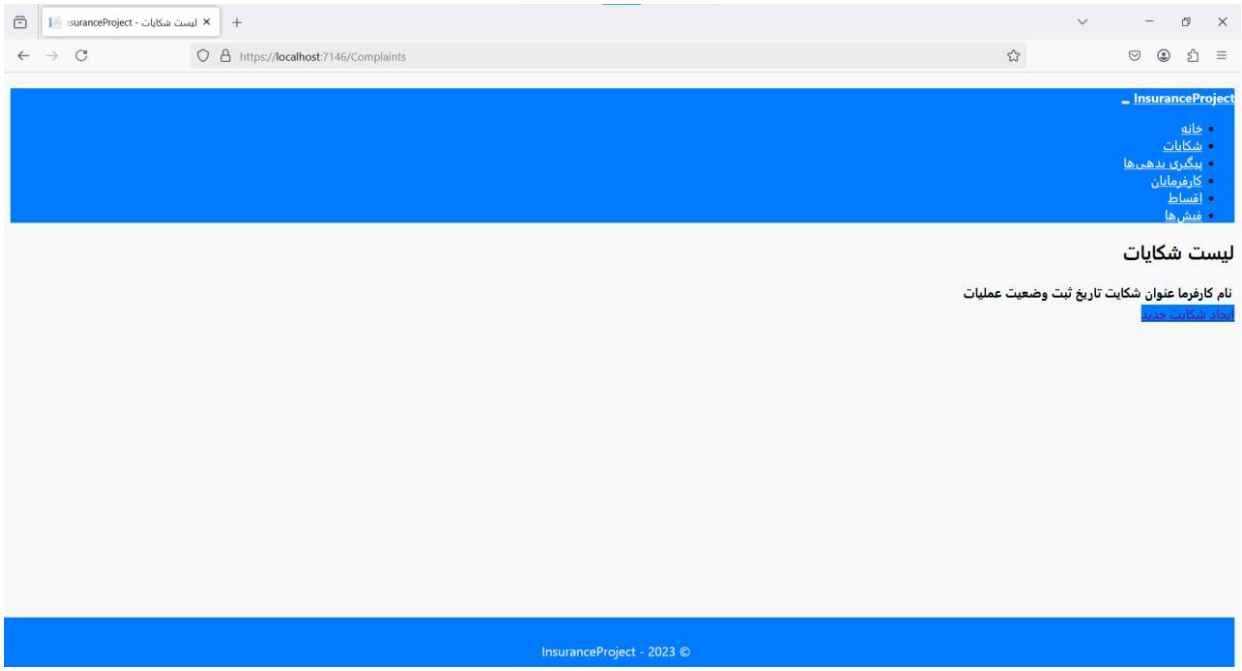
۳. حسابرسی دفاتر قانونی

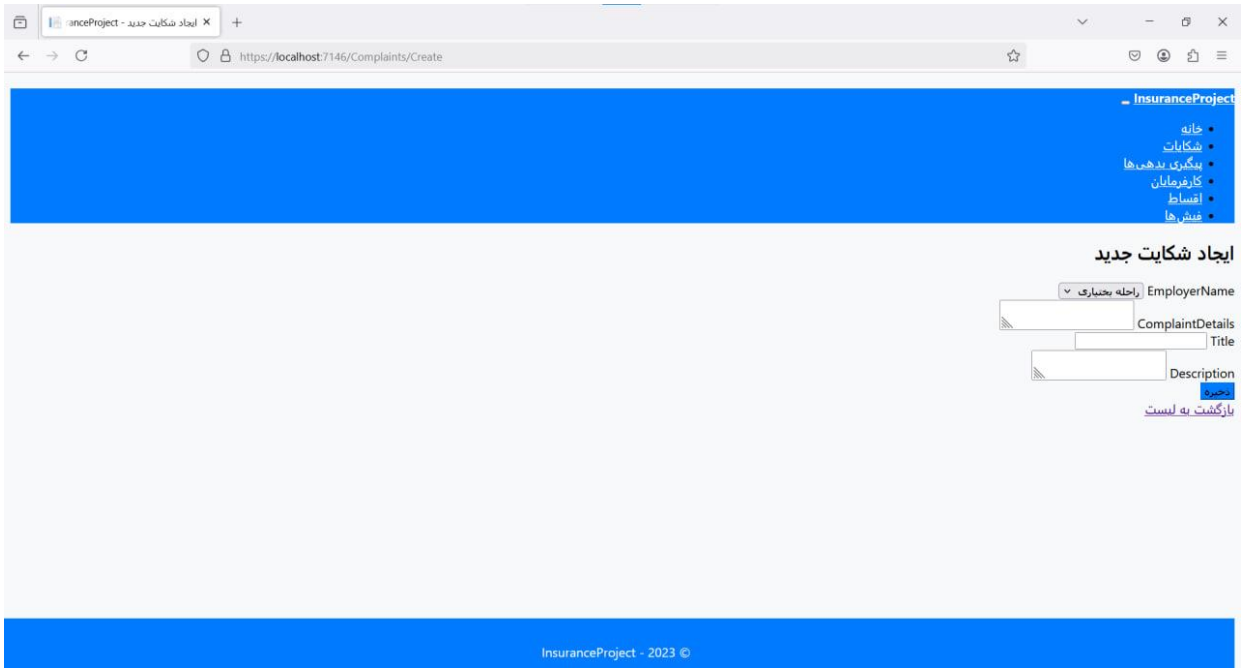
از طریق DebtTracking.cs امکان بررسی بدهی‌های معوق و عدم تطابق لیست‌های ارسالی وجود دارد.

۴. ثبت شکایات

ComplaintsController.cs مدیریت شکایات را برعهده دارد.

کارفرمایان می‌توانند شکایات خود را ثبت و وضعیت آن را پیگیری کنند.

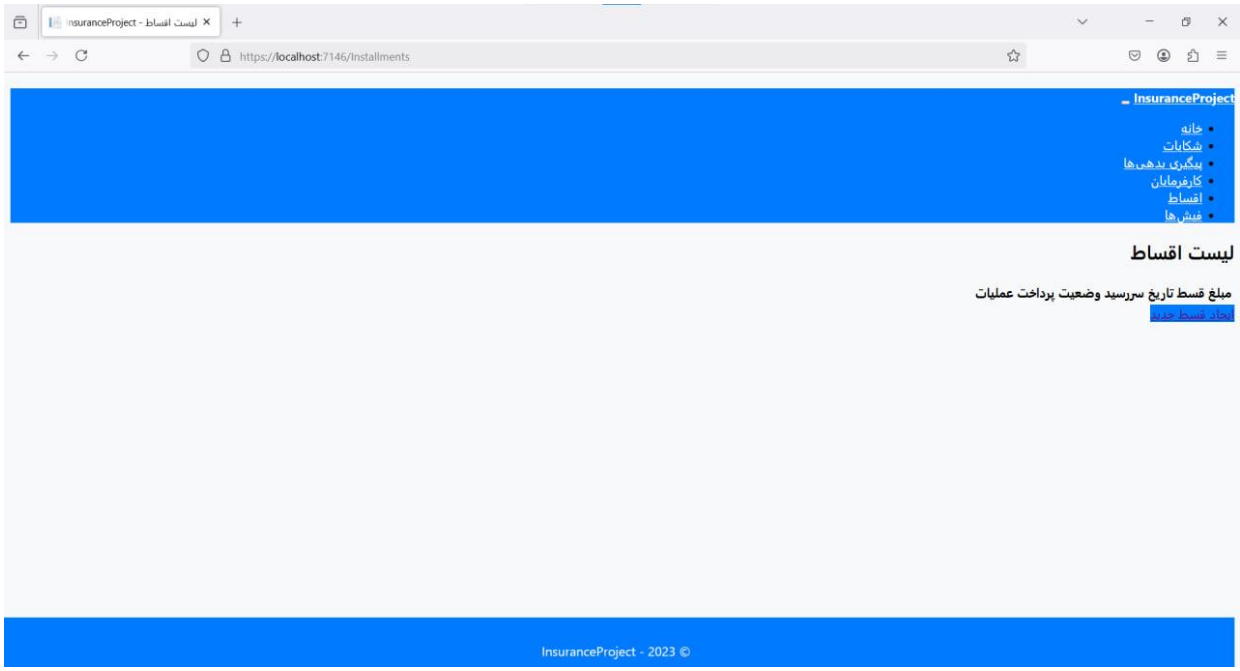


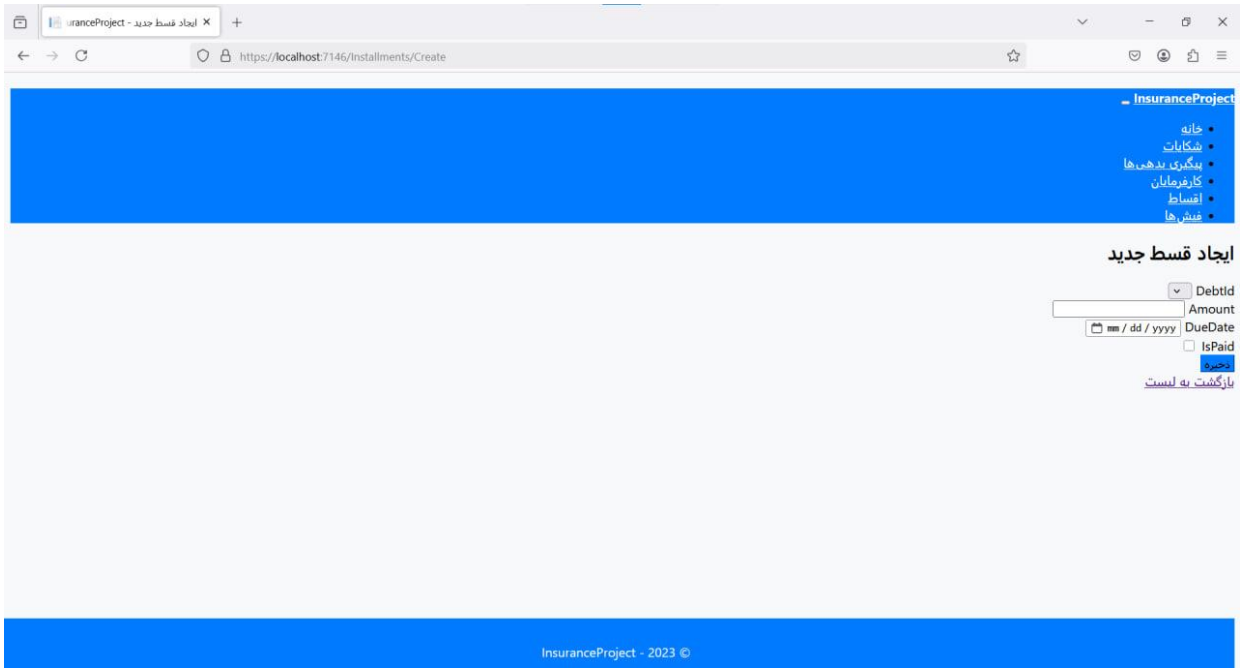


۵. تقسیط بدهی‌ها

InstallmentsController.cs وظیفه مدیریت و ثبت اقساط را دارد.

در صورت تأخیر در پرداخت اقساط، جریمه محاسبه و ثبت می‌شود.

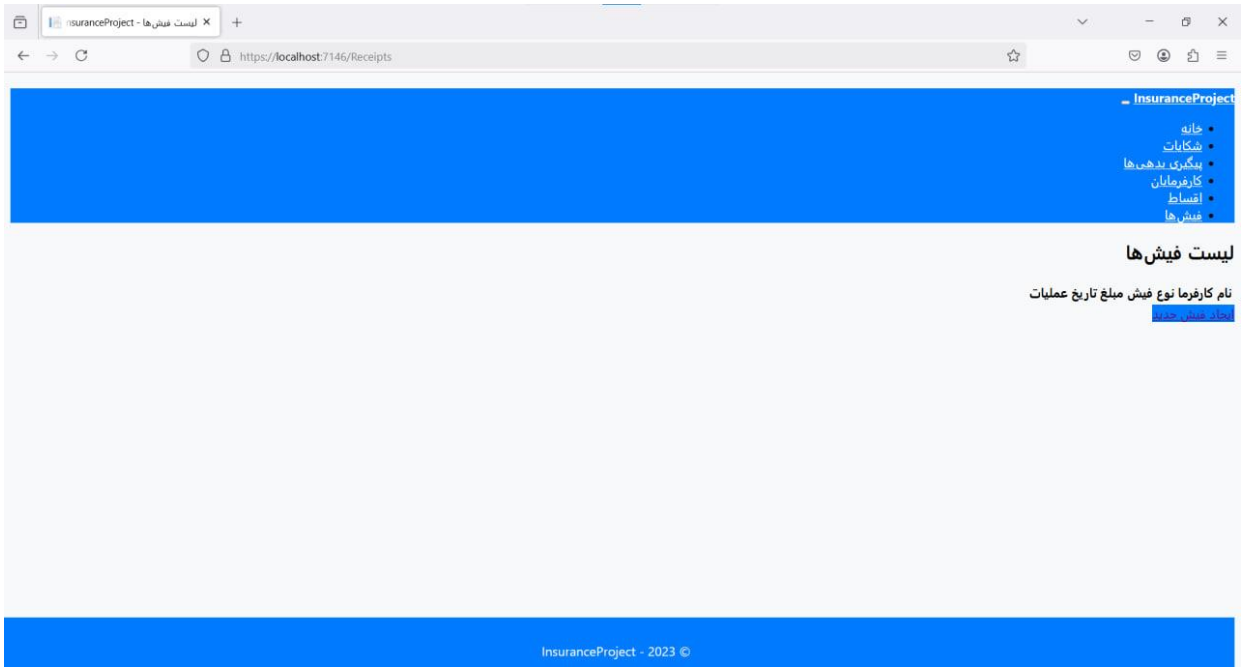


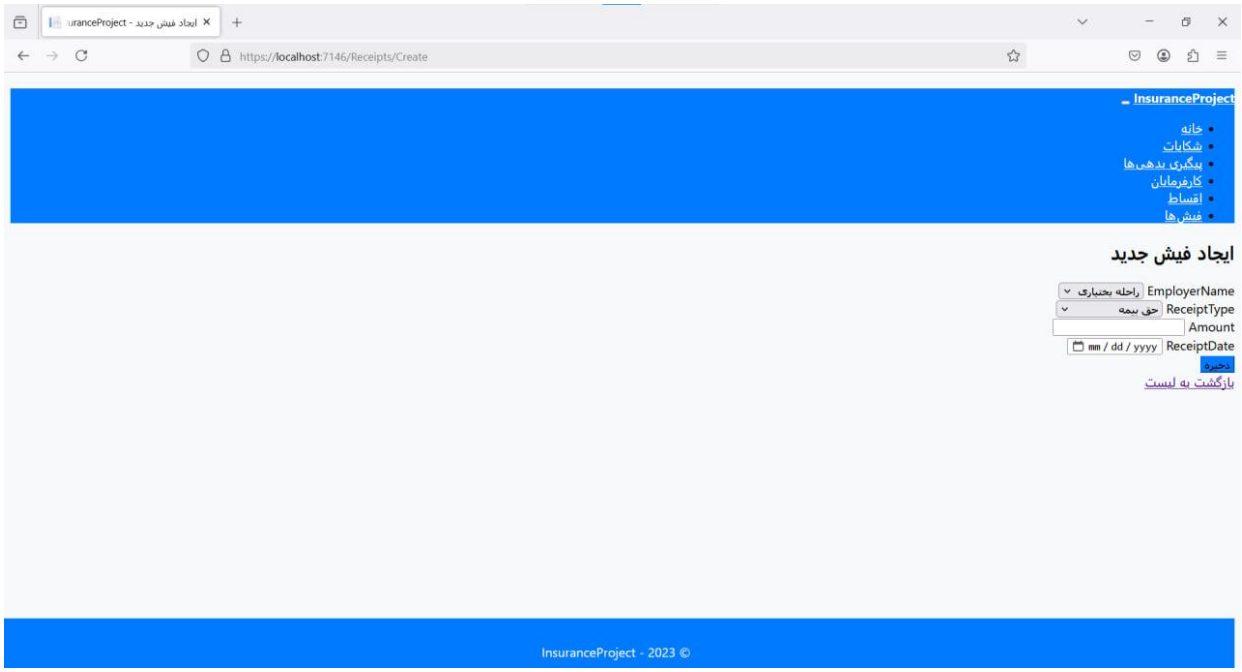


۶. باجه دریافت و صدور فیش‌ها

در ReceiptsController.cs فیش‌های پرداختی صادر و نمایش داده می‌شوند.

امکان دریافت و پرداخت آنلاین فراهم شده است.





۷. پیگیری مطالبات

DebtTrackingController.cs شامل مکانیزم ارسال اخطاریه و اجرای فرآیندهای وصول مطالبات است.

بدهی‌های معوقه به کارفرمایان اطلاع‌رسانی شده و در صورت عدم پرداخت، اقدامات اجرایی انجام می‌شود.

۶. نتیجه‌گیری

پروژه واحد وصول حق بیمه یک سامانه یکپارچه، دقیق و کارآمد برای مدیریت پرداخت‌های بیمه‌ای و پیگیری بدهی‌های کارفرمایان است. با بهره‌گیری از ASP.NET Core MVC و Entity Framework Core، این سامانه تمامی نیازهای سازمان‌های بیمه‌ای را برآورده کرده و باعث افزایش دقت، شفافیت و سرعت در فرآیند وصول بیمه می‌شود.

این پروژه می‌تواند به‌عنوان مدل پایه‌ای برای سیستم‌های بیمه‌ای آینده استفاده شود و قابلیت گسترش و سفارشی‌سازی را داراست.