Project IS7012

Awesome! Starting to think I was getting picked last for the kickball game!  Two people have less work than 4+ so I suggest we call this Team Us.

**Team size of two/three Deliverables:**

1. Including authentication
2. Entities - must have a minimum of 4.
3. Relationships
4. 3 out of 4 must be related to at least one other entity.
5. 1 entity has multiple other entities in it
6. 1 relationship must be 1 to many.
7. Include a search page
8. Include a custom page such as statistics

Pick something that at least 1 person on the team is familiar with.  Examples: Cookbook, Real estate for sale, and Book ranking site.

I only have an Android Catalog project that's written in Kotlin. I looked up Bookstore but that's a lot of work.  The AI model offered Task Management App.  So that's below.  If you have anything, that's great!

Availability:

I'm available to meet tomorrow after 7 PM, Thursday would be 8 PM, Friday 5-8 PM.  Weekends open after 1 PM.

I'm on Teams for work- feel free to message me- Longeaa or email-longeaa@ucmail.uc.edu.   DTS Scrum Master/AI Enablement Team

Thanks for picking me!

Anisa

DRAFT Task Management App - Project Plan

Project Overview

Duration: 3-4 days (4-6 hours/day for 2 developers)  
Tech Stack: [ASP.NETLinks to an external site.](https://asp.net/" \t "_blank) Core MVC, Entity Framework, SQL Server, Bootstrap

Entities & Relationships

text

User (Authentication)

├── Projects (1-to-Many)

   ├── Tasks (1-to-Many)

       └── Comments (1-to-Many)

Entity Details:

1. User - [ASP.NETLinks to an external site.](https://asp.net/" \t "_blank) Identity
2. Project - Relates to User and Tasks
3. Task - Relates to Project and Comments (entity with multiple relationships)
4. Comment - Relates to Task

Development Timeline

Day 1: Foundation & Core Entities (4-6 hours)

Hour 1-2: Project Setup

bash

dotnet new mvc -n TaskManager

cd TaskManager

dotnet add package Microsoft.EntityFrameworkCore.SqlServer

dotnet add package Microsoft.AspNetCore.Identity.EntityFrameworkCore

* Create solution structure
* Configure database context
* Set up basic layout with Bootstrap

Hour 3-4: Authentication Setup

csharp

*// Implement Identity*

public class ApplicationUser : IdentityUser

{

   public ICollection<Project> Projects { get; set; }

}

Hour 5-6: Project Entity CRUD

* Create Project model with User relationship
* Implement Project Controller with basic views
* Test create/read operations

Day 2: Task System & Relationships (4-6 hours)

Hour 1-2: Task Entity Development

csharp

public class Task

{

   public int Id { get; set; }

   public string Title { get; set; }

   public string Description { get; set; }

   public DateTime DueDate { get; set; }

   public bool IsCompleted { get; set; }

*// Relationships*

   public int ProjectId { get; set; }

   public Project Project { get; set; }

   public ICollection<Comment> Comments { get; set; }

}

Hour 3-4: Task-Project Relationship

* Implement Task CRUD operations
* Create views within Project context
* Add navigation properties

Hour 5-6: Comment System

* Comment entity with Task relationship
* Basic comment functionality

Day 3: Search & Advanced Features (4-6 hours)

Hour 1-2: Search Functionality

csharp

*// SearchViewModel*

public class TaskSearchViewModel

{

   public string SearchTerm { get; set; }

   public bool? ShowCompleted { get; set; }

   public DateTime? DueDateFrom { get; set; }

   public List<Task> Results { get; set; }

}

Hour 3-4: Statistics Page

* Project completion statistics
* Task overdue alerts
* User productivity metrics

Hour 5-6: UI/UX Polish

* Responsive design improvements
* Navigation enhancements
* Form validation

Day 4: Testing & Polish (4-6 hours)

Hour 1-2: Relationship Testing

* Verify all 1-to-Many relationships work
* Test cascade operations
* Data integrity checks

Hour 3-4: Authentication Testing

* User registration/login flow
* Authorization rules (users only see their projects)
* Role-based access if time permits

Hour 5-6: Final Polish & Deployment

* Bug fixes
* Performance optimization
* Local deployment testing

Database Schema

sql

*-- Simplified schema*

Users (Id, Email, PasswordHash)

Projects (Id, Name, Description, UserId)

Tasks (Id, Title, Description, DueDate, IsCompleted, ProjectId)

Comments (Id, Content, CreatedDate, TaskId)

Division of Work (2 Developers)

Developer A: Backend Focus

* Entity Framework models and relationships
* Database context and migrations
* Controller logic
* Authentication backend

Developer B: Frontend Focus

* Razor views and layouts
* Bootstrap styling
* JavaScript interactions
* Search interface

Risk Mitigation

Time-Saving Strategies:

1. Use Scaffolding: dotnet aspnet-codegenerator for quick CRUD
2. Bootstrap Templates: Start with pre-built admin templates
3. EF Core Conventions: Leverage default behaviors
4. Parallel Development: Work on different entities simultaneously

Complexity Management:

* Start with basic CRUD, enhance incrementally
* Implement search with simple LINQ queries first
* Use Chart.js for simple statistics instead of complex libraries

Deliverables Checklist

* Authentication system
* 4+ entities with proper relationships
* 1-to-Many relationships implemented
* Search functionality
* Statistics page
* Responsive UI
* Data validation
* Error handling

This plan ensures you meet all requirements while staying within the time constraints. The task management domain is intuitive and allows for natural entity relationships that satisfy your criteria.