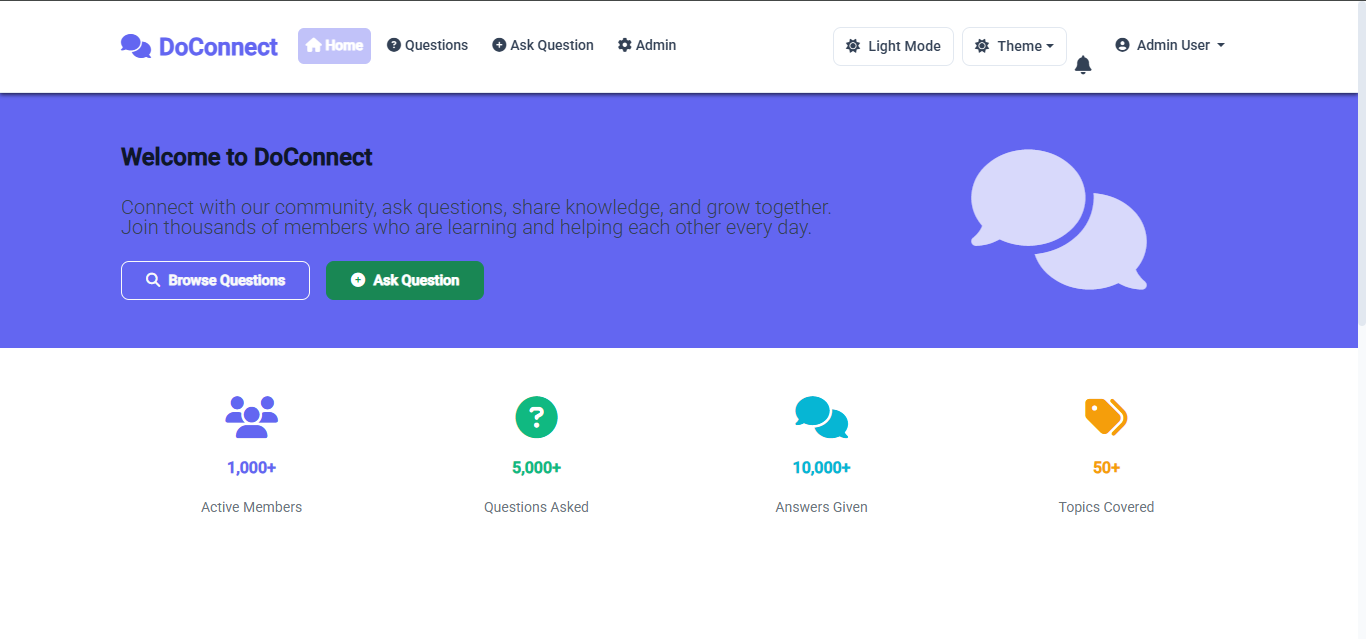
**DoConnect - Q&A Platform**

**📋 Table of Contents**

* [Project Overview](#project-overview)
* [Technology Stack](#technology-stack)
* [System Architecture](#system-architecture)
* [Database Design](#database-design)
* [Features](#features)
* [Sprint Implementation](#sprint-implementation)
* [Setup and Installation](#setup-and-installation)
* [API Documentation](#api-documentation)
* [User Interface](#user-interface)
* [Admin Panel](#admin-panel)
* [Testing](#testing)
* [Deployment](#deployment)
* [Future Enhancements](#future-enhancements)



**Project Overview**

**DoConnect** is a comprehensive Q&A platform that allows users to ask and answer technical questions while providing administrators with powerful content moderation capabilities. The platform features real-time notifications, image upload functionality, and a robust approval workflow.

**Problem Statement**

In today's fast-paced technical environment, developers and tech enthusiasts need a reliable platform to share knowledge, ask questions, and get expert answers. DoConnect bridges this gap by providing a structured, moderated environment for technical discussions.

**Project Objectives**

* Create an intuitive Q&A platform for technical discussions
* Implement role-based access control (User/Admin)
* Provide real-time notifications for administrators
* Enable content moderation and approval workflows
* Support multimedia content through image uploads
* Ensure scalable and secure architecture

**🛠 Technology Stack**

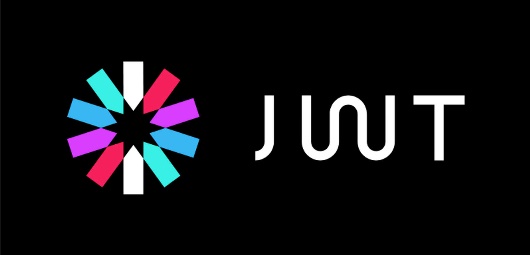
**Frontend**

* **Framework:** Angular 14+
* **UI Library:** Tailwind css
* **State Management:** Angular Services with RxJS
* **Routing:** Angular Router
* **HTTP Client:** Angular HttpClient
* **Authentication:** JWT Token-based
* **Real-time:** SignalR Client







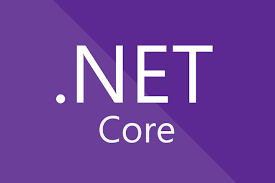


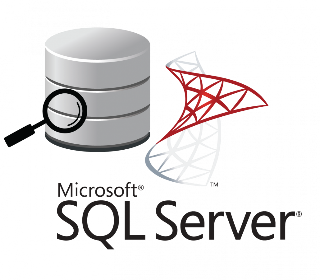
**Backend**

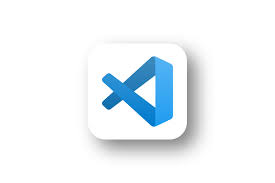
* **Framework:** [ASP.NET](http://ASP.NET) Core 8 Web API
* **ORM:** Entity Framework Core
* **Database:** SQL Server
* **Authentication:** JWT Bearer Authentication
* **Real-time:** SignalR
* **File Upload:** IFormFile
* **API Documentation:** Swagger / OpenAPI

**Development Tools**

* **IDE:** VS Code
* **Version Control:** Git/GitHub
* **Package Manager:** NuGet (Backend), npm (Frontend)
* **Testing:** xUnit (Backend), Jasmine/Karma (Frontend)



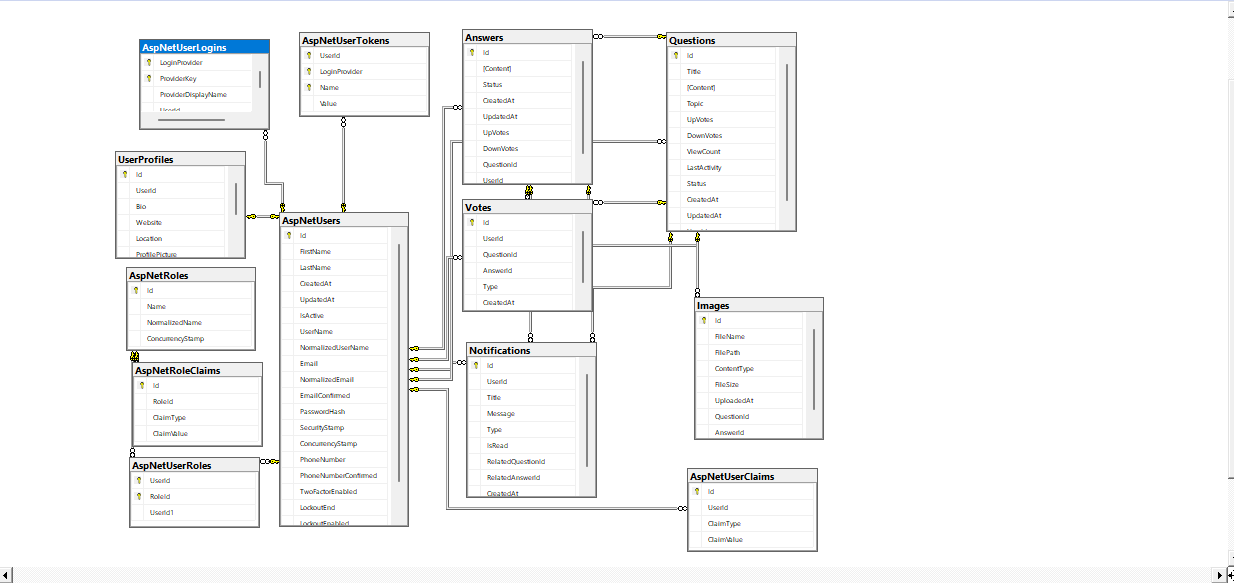




**System Architecture**

**Architecture Components**

1. **Client Layer (Angular)**
   * User Interface Components
   * Authentication Guards
   * Service Layer for API Communication
   * State Management
2. **API Layer (**[**ASP.NET**](http://ASP.NET) **Core)**
   * Controllers for handling requests
   * JWT Authentication Middleware
   * SignalR Hubs for real-time communication
   * File Upload Services
3. **Business Logic Layer**
   * User Management Services
   * Question/Answer Processing
   * Content Moderation Logic
   * Notification Services
4. **Data Access Layer**
   * Entity Framework Core
   * Repository Pattern
   * Database Context
5. **Database Layer**
   * SQL Server Database
   * Stored Procedures (if needed)
   * Indexes for Performance

**🗄 Database Design:  
**

**Core Tables**

|  |  |  |
| --- | --- | --- |
| Table | Columns | Details |
| **Users** | Id (PK, int), FirstName, LastName, Email (Unique), UserName (Unique), PasswordHash, IsActive, CreatedAt, UpdatedAt | Stores user information |
| **Questions** | Id (PK), Title, Content, Topic, Status (0=Pending,1=Approved,2=Rejected), UserId (FK), CreatedAt, UpdatedAt | User questions |
| **Answers** | Id (PK), Content, Status, QuestionId (FK), UserId (FK), CreatedAt, UpdatedAt | Answers to questions |
| **Images** | Id (PK), FileName, FilePath, ContentType, FileSize, QuestionId (nullable FK), AnswerId (nullable FK), UploadedAt | Image metadata |

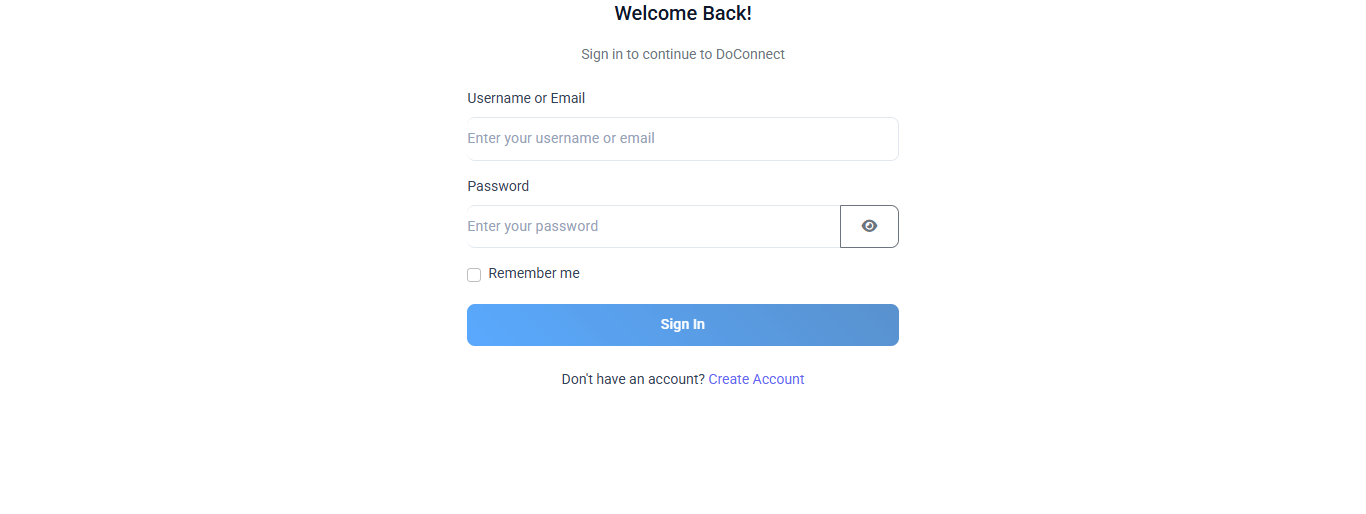
**Entity Relationships**

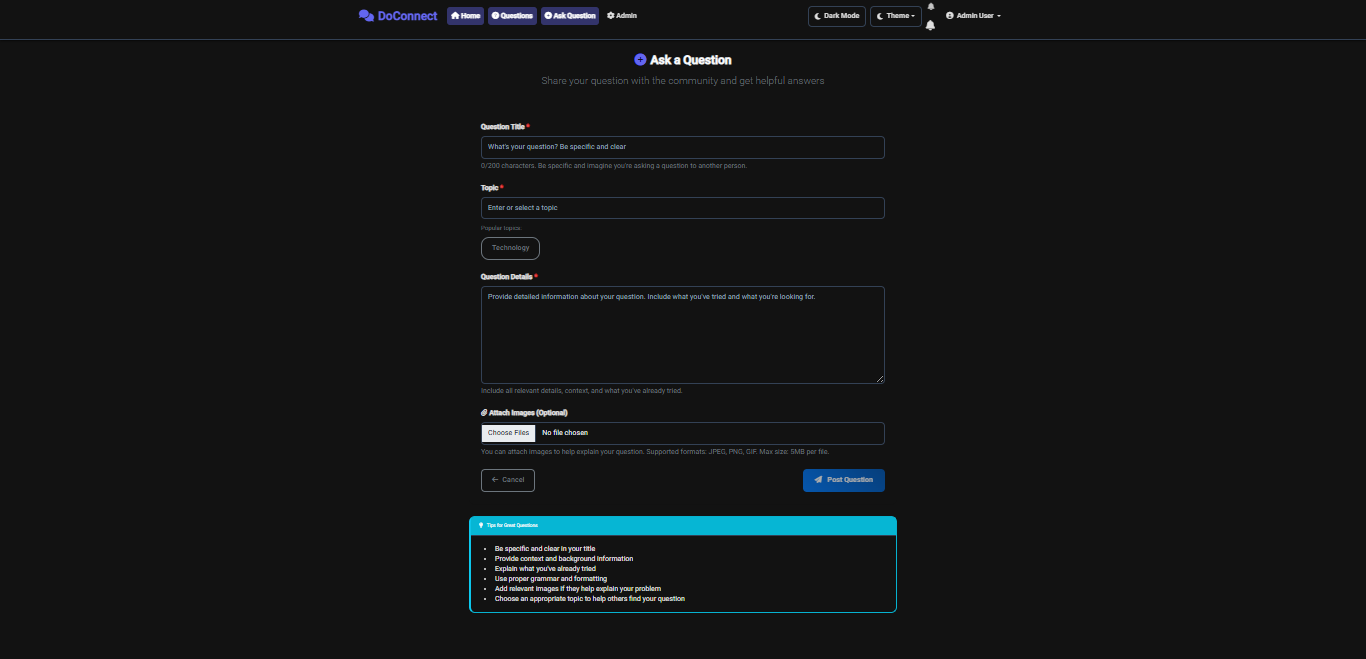
* Users → Questions (One-to-Many)
* Users → Answers (One-to-Many)
* Questions → Answers (One-to-Many)
* Questions → Images (One-to-Many)

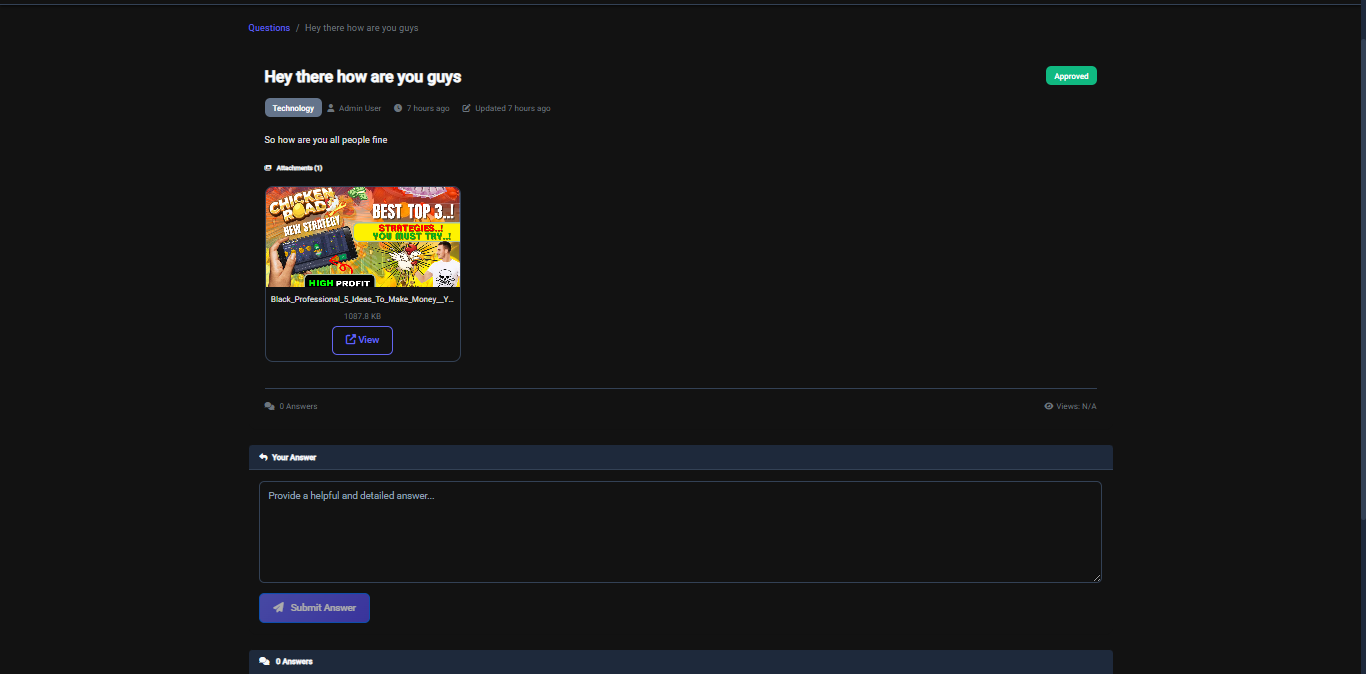
**Features**

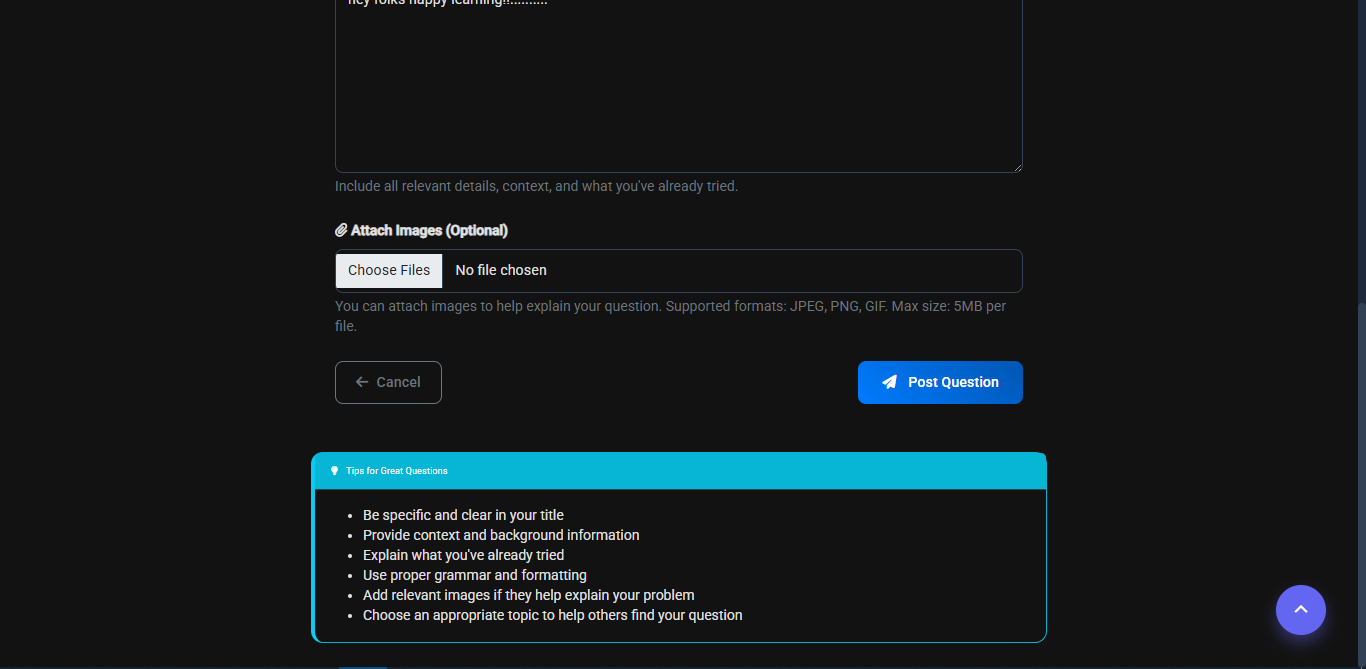
**User Features**

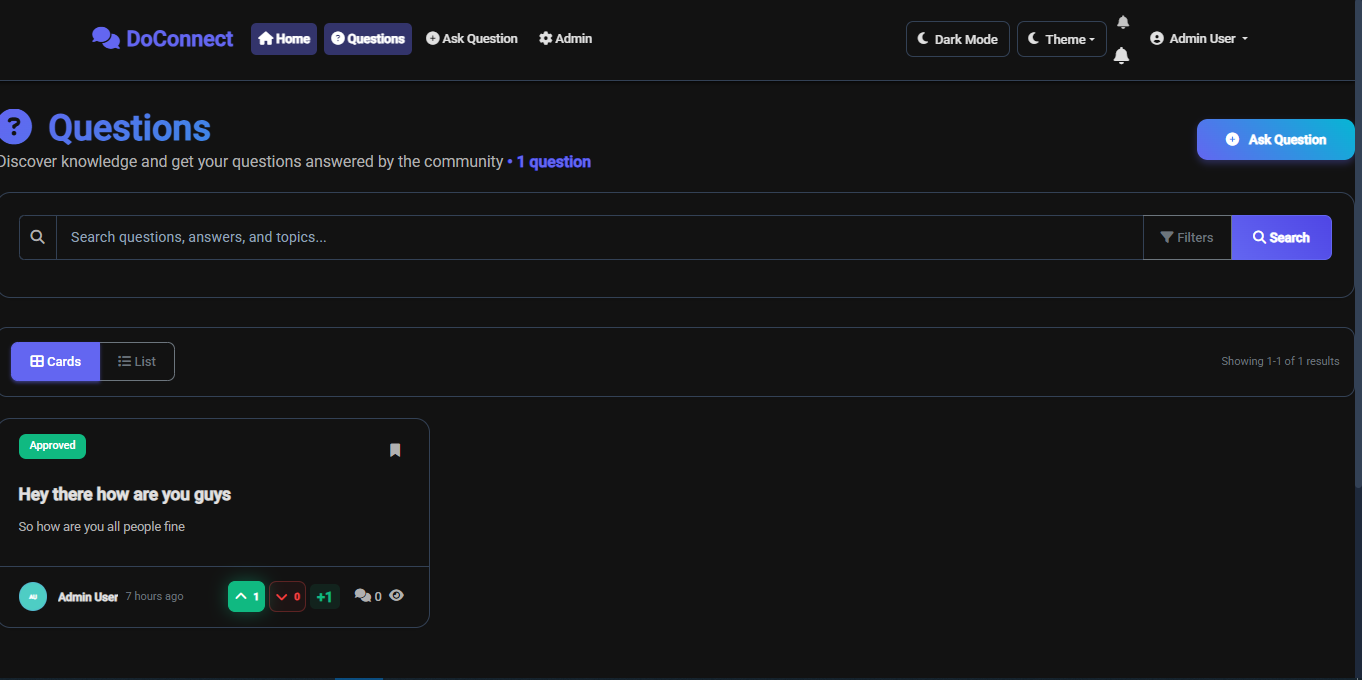
* User Registration and Authentication
* Ask Questions with Rich Text Editor
* Answer Questions
* Upload Images with Questions/Answers
* Search Questions by Keywords
* View Question Details
* User Profile Management
* Dark/Light Theme Toggle

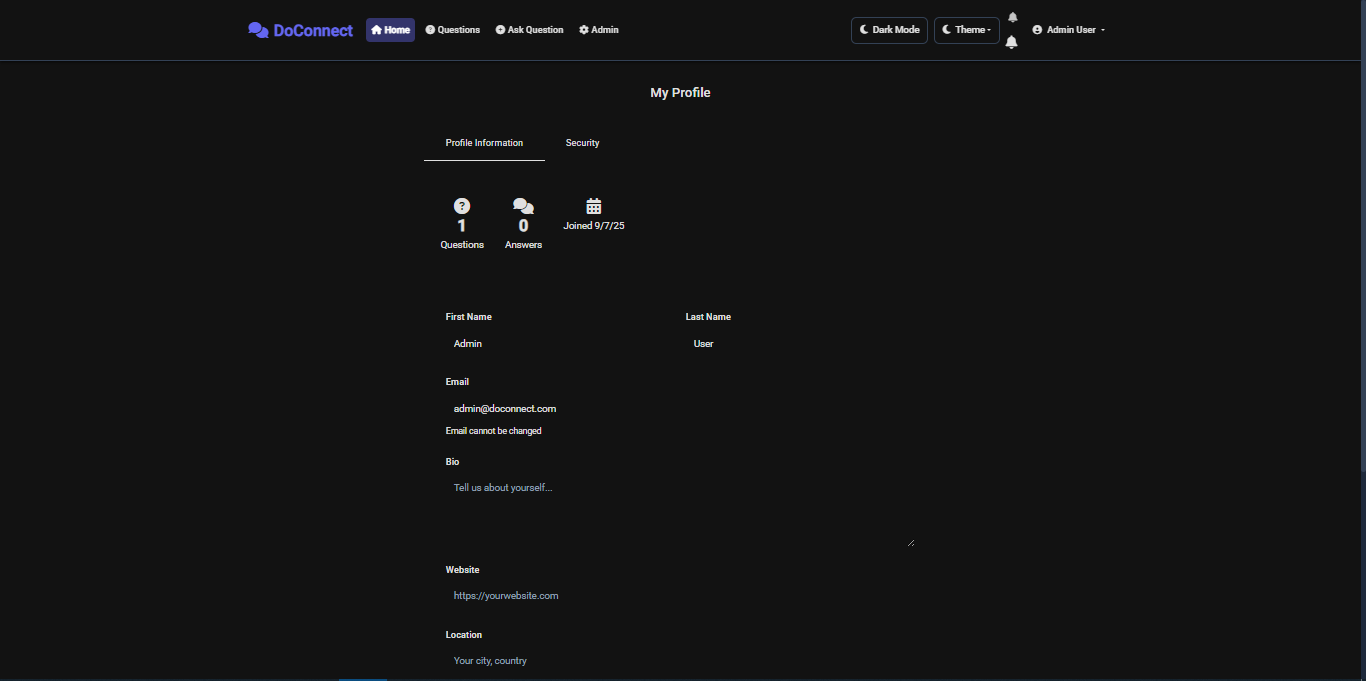


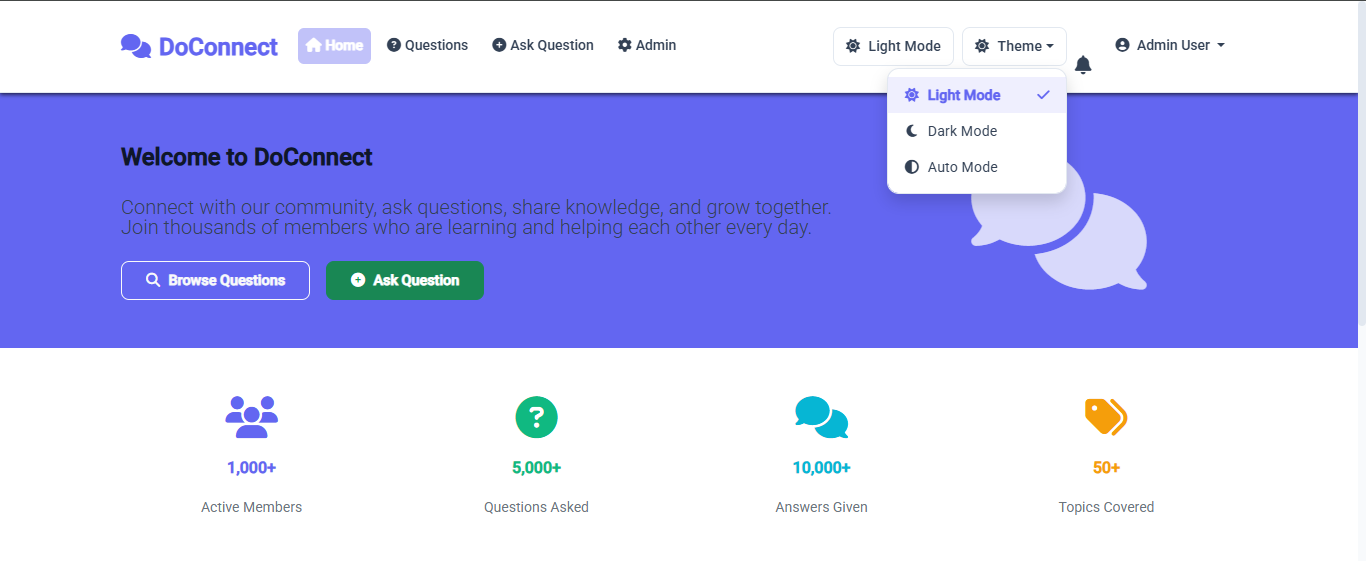


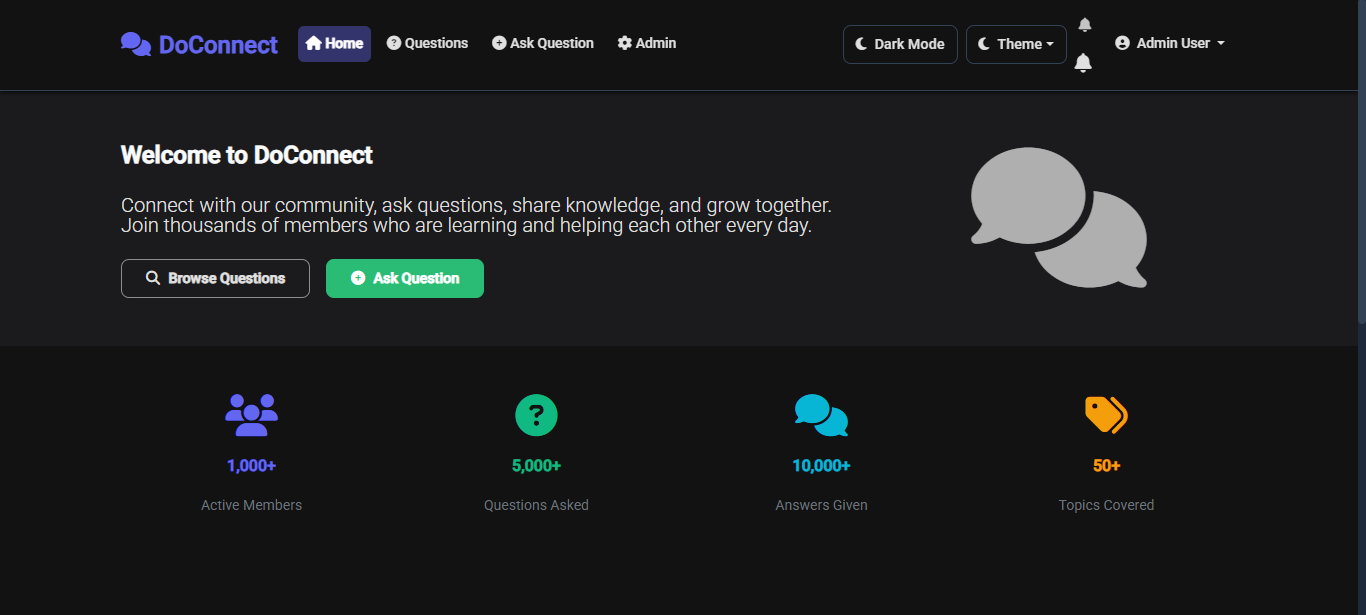


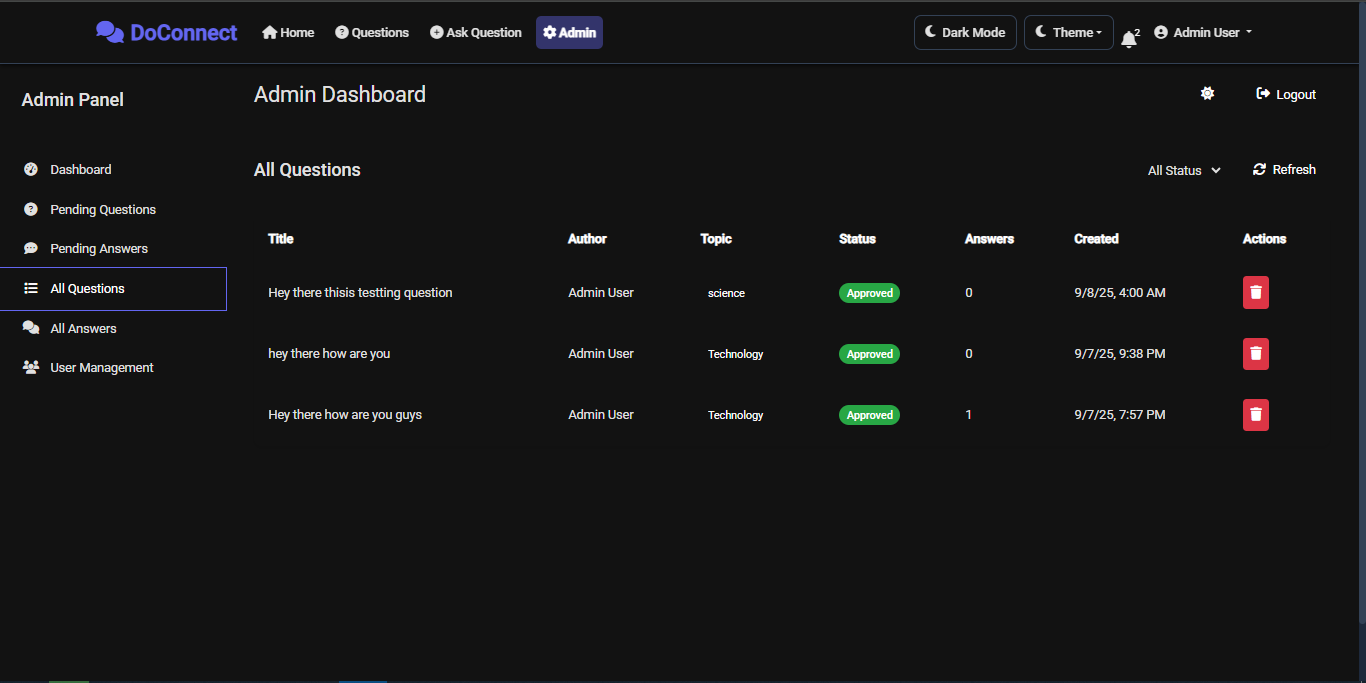


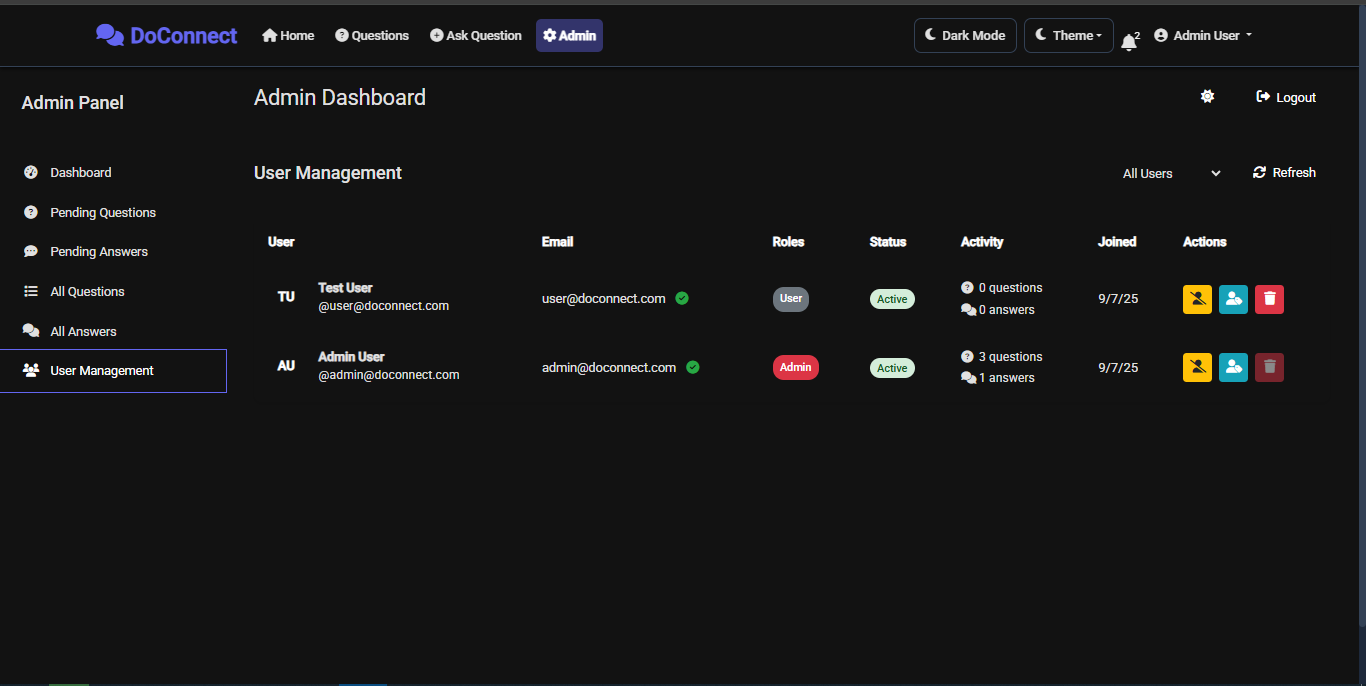






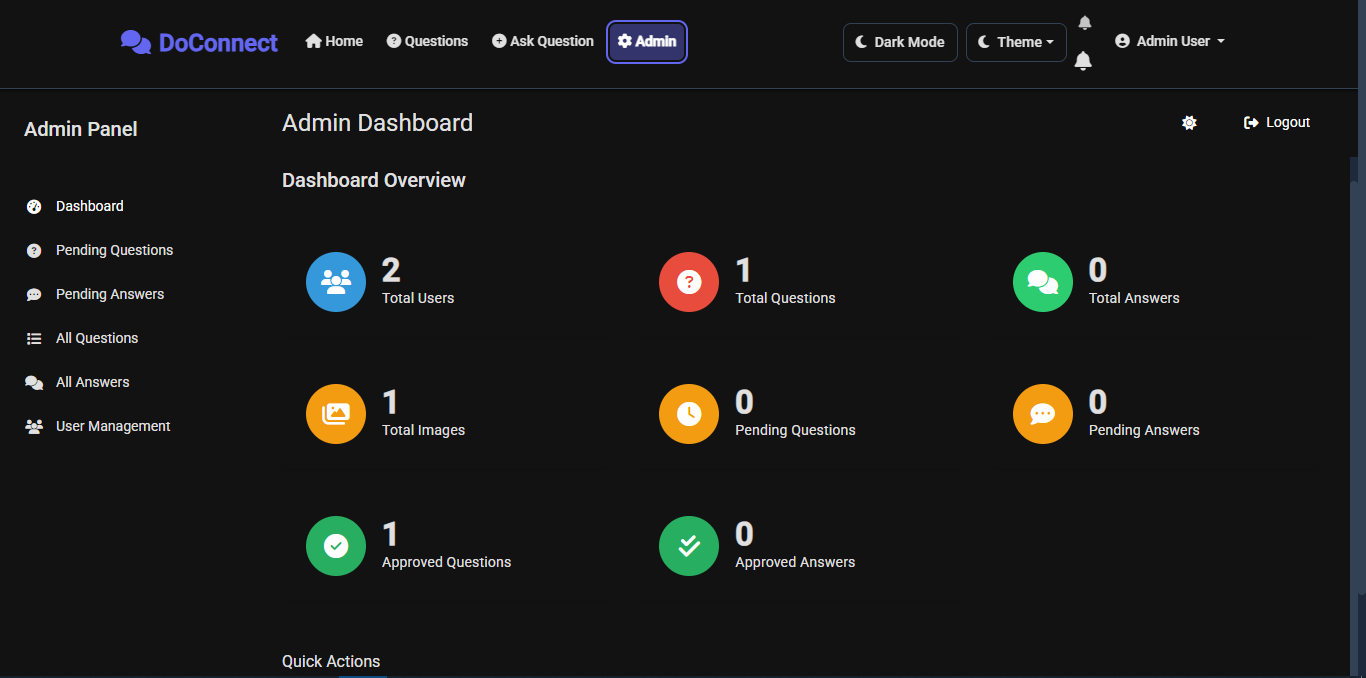


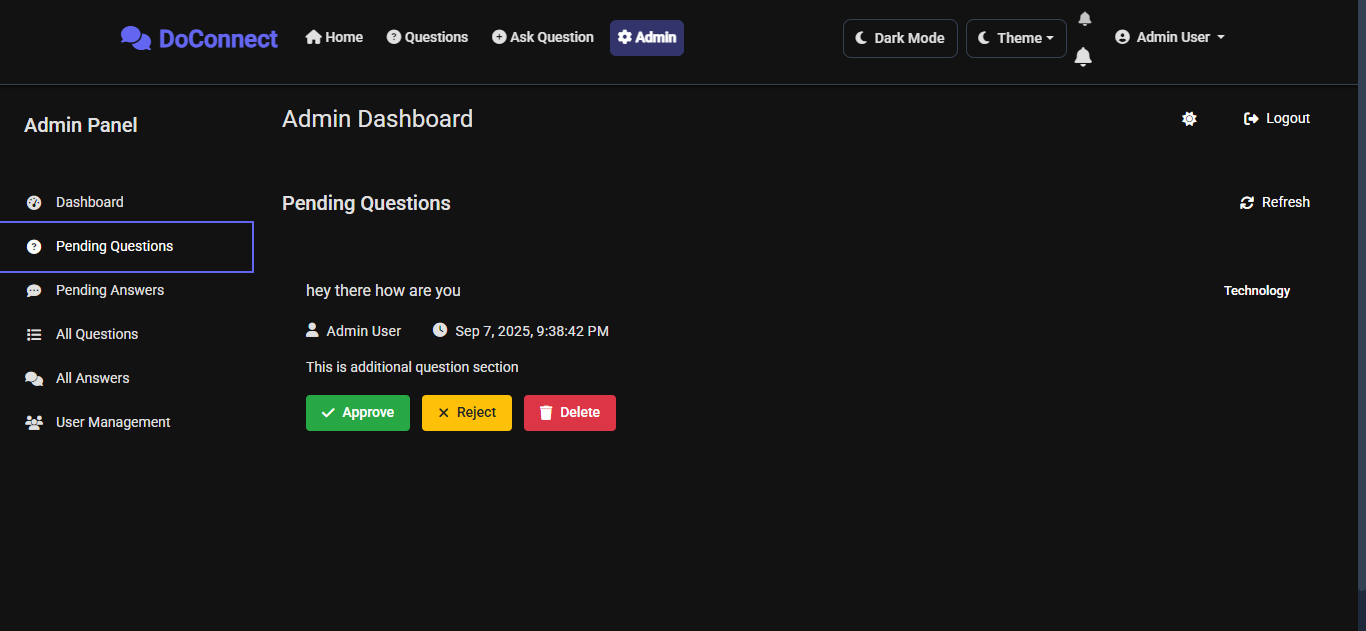
****

****

**Admin Features**

* Admin Dashboard with Statistics
* Content Moderation (Approve/Reject)
* User Management
* Real-time Notifications
* Content Deletion
* Role Management
* System Analytics



****

**Sprint Implementation**

**Sprint I: Foundation and Planning**

**Objectives**

* Use Case Documentation
* Database Schema Design
* Technology Stack Selection
* Project Setup

**Deliverables**

* Use case document
* Entity Relationship Diagram
* Database schema scripts
* Backend project structure
* Frontend project initialization

**Sprint II: Core Functionality**

**Objectives**

* User Authentication System
* Question & Answer CRUD Operations
* Image Upload Functionality
* Basic UI Components

**Deliverables**

* JWT Authentication
* User registration and login
* Question creation and viewing
* Answer submission
* Image upload API and UI
* Responsive Angular components

*User registration page*

*User login interface*

*Ask question form with image upload*

*Questions listing page*

*Question detail view with answers*

**Sprint III: Advanced Features and Admin Panel (3 weeks)**

**Objectives**

* Search Functionality
* Admin Dashboard
* Content Moderation
* Real-time Notifications
* Final Integration and Testing

**Deliverables**

* Advanced search with filters
* Complete admin panel
* Approval workflow
* SignalR notifications
* Swagger API documentation
* Comprehensive testing

*Search functionality with filters*

*Admin dashboard with statistics*

*Admin pending questions management*

*Admin user management interface*

*SignalR real-time notifications*

**Setup and Installation**

**Prerequisites**

* .NET 8 SDK
* Node.js 18+ and npm
* SQL Server 2019+
* Visual Studio 2022 or VS Code
* Git

**Backend Setup**

git clone https://github.com/yourusername/doconnect.git  
cd doconnect/Backend  
dotnet restore  
dotnet ef database update  
dotnet run

Backend API will be available at https://localhost:5001

**Frontend Setup**

cd ../Frontend  
npm install  
ng serve

Frontend app will be available at http://localhost:4200

**Default Admin Credentials**

|  |  |  |
| --- | --- | --- |
| Role | Email | Password |
| Admin | [admin@doconnect.com](mailto:admin@doconnect.com) | Admin@123 |
| User | [user@doconnect.com](mailto:user@doconnect.com) | User@123 |

**📚 API Documentation**

**Authentication Endpoints**

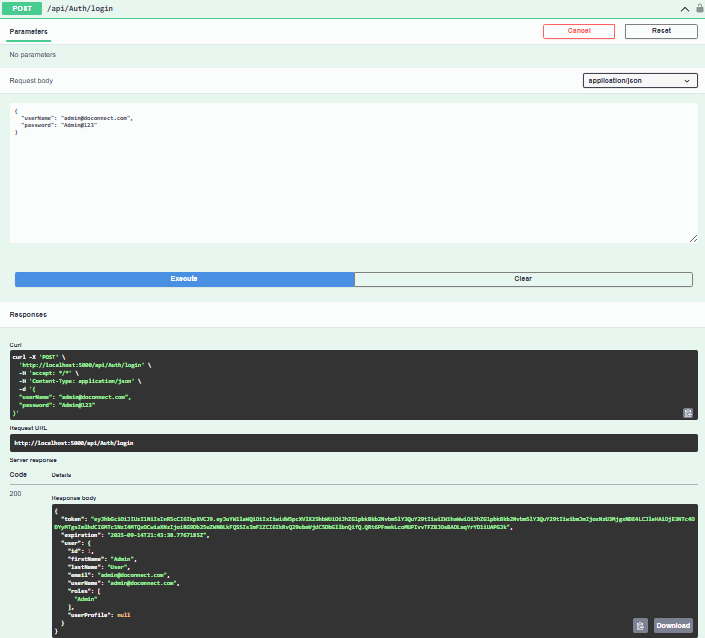
**POST** /api/auth/login

Request body example:

{  
 "email": "user@example.com",  
 "password": "password123"  
}

Response example:

{  
 "success": true,  
 "token": "jwt\_token\_here",  
 "user": {  
 "id": 1,  
 "firstName": "John",  
 "lastName": "Doe",  
 "email": "user@example.com",  
 "roles": ["User"]  
 }  
}

****

**POST** /api/auth/register

Request body example:

{  
 "firstName": "John",  
 "lastName": "Doe",  
 "email": "user@example.com",  
 "password": "password123",  
 "confirmPassword": "password123"  
}

**Questions Endpoints**

**GET** /api/questions

Retrieve paginated question list with optional filters.

Query parameters:

* page (int): page number
* pageSize (int): number of items per page
* search (string): keyword search
* topic (string): topic filter

**POST** /api/questions

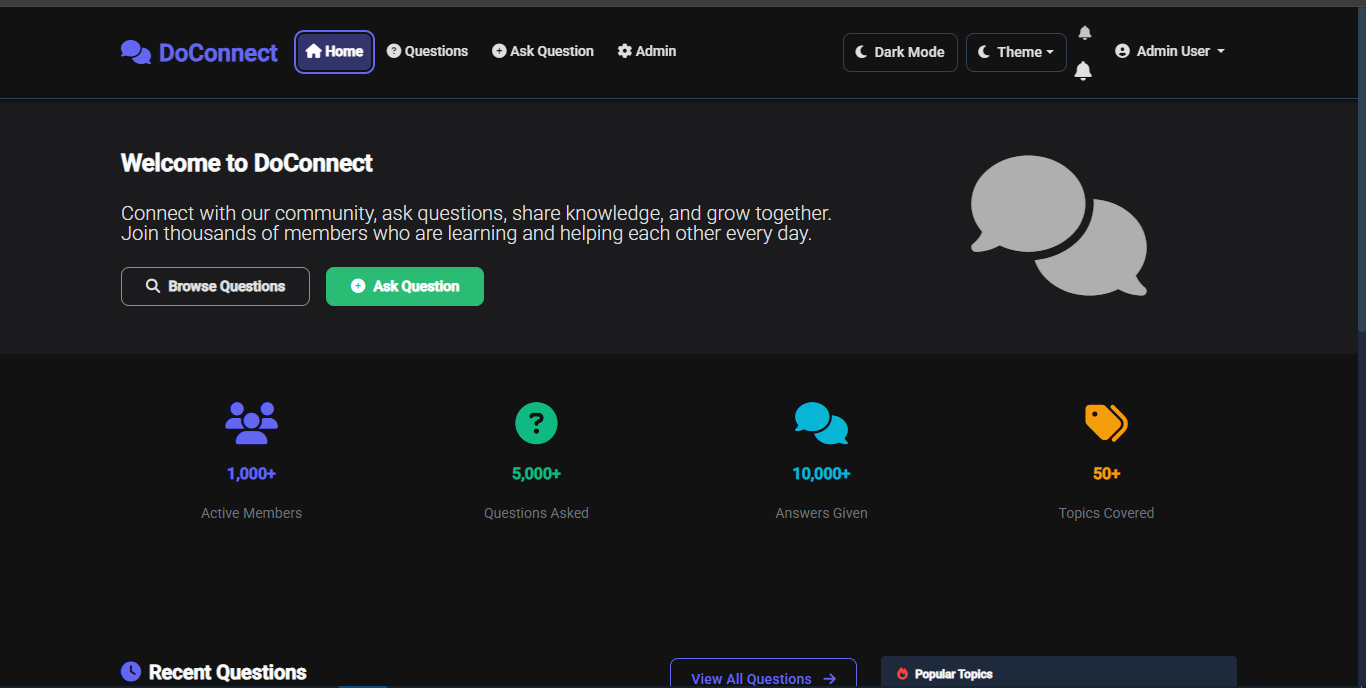
Create a new question (authorization required). Submit as multipart/form-data with fields:

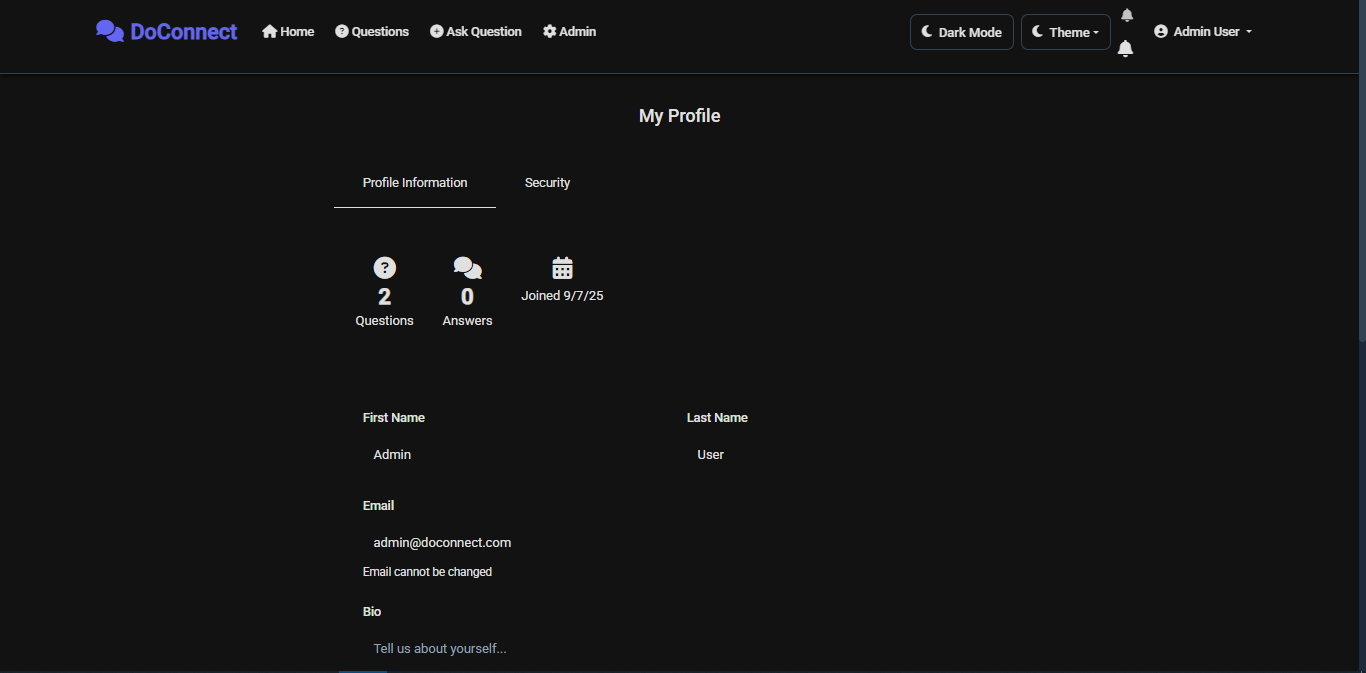
* title
* content
* topic
* files (optional images)

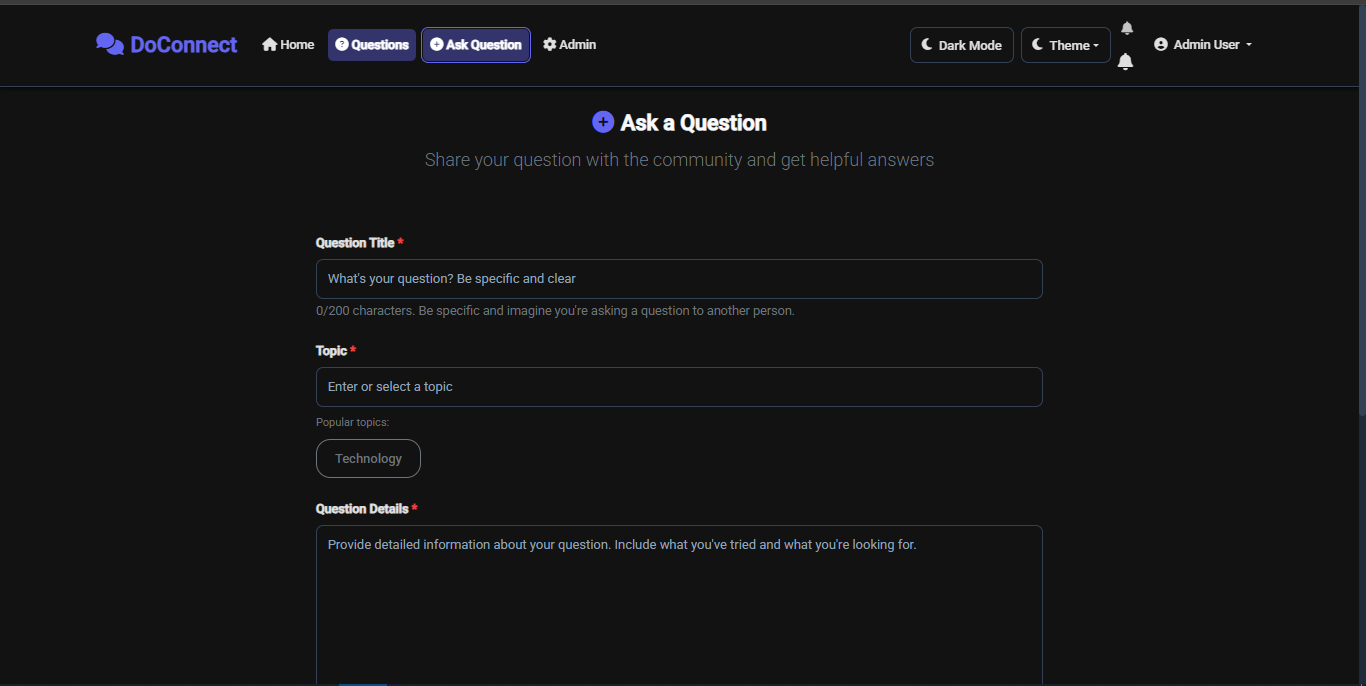
**🎨 User Interface**

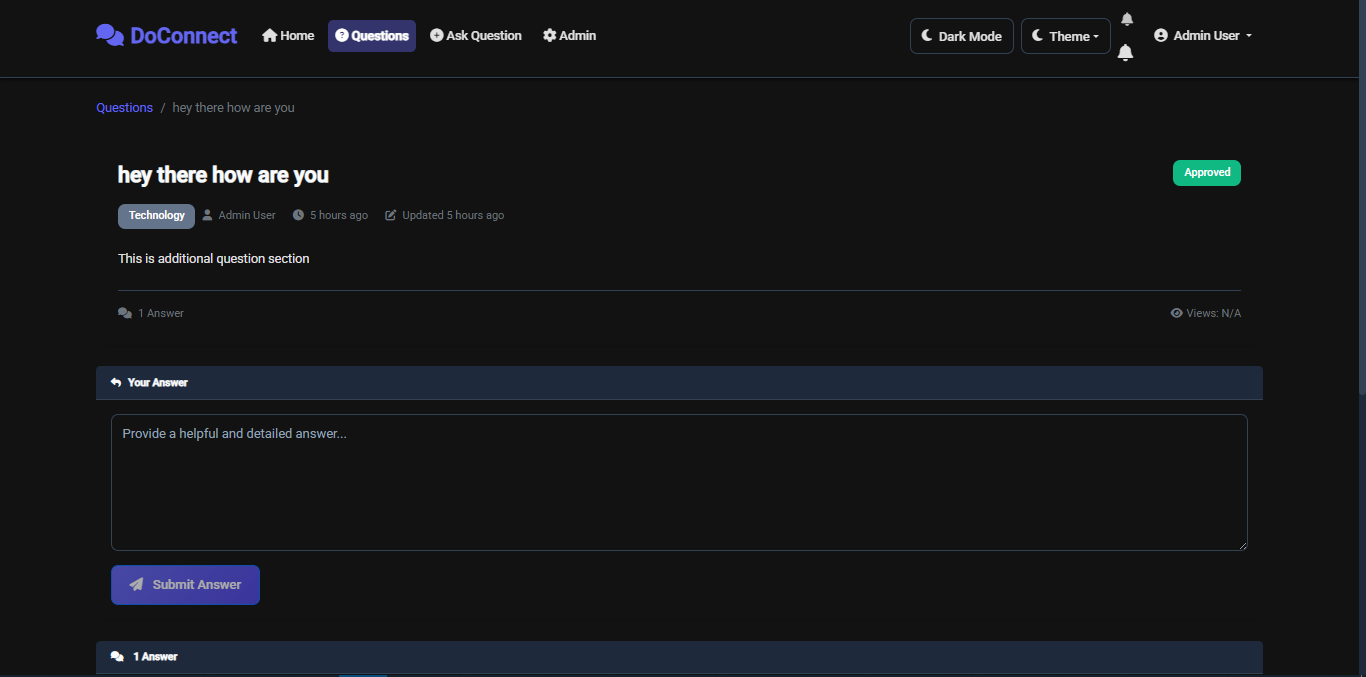
Screenshots included for:

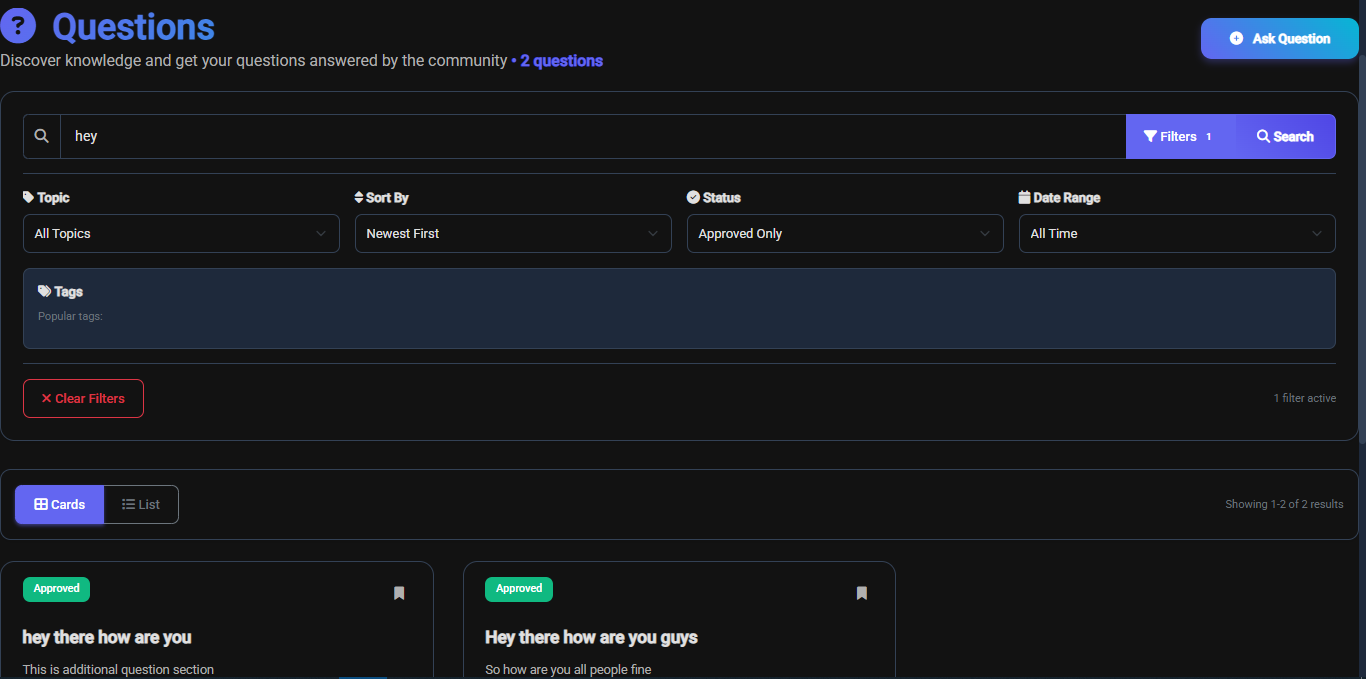
* Home Page
* User Dashboard
* Question Creation Form
* Question Detail View
* Search Results
* User Profile
* Responsive Mobile View

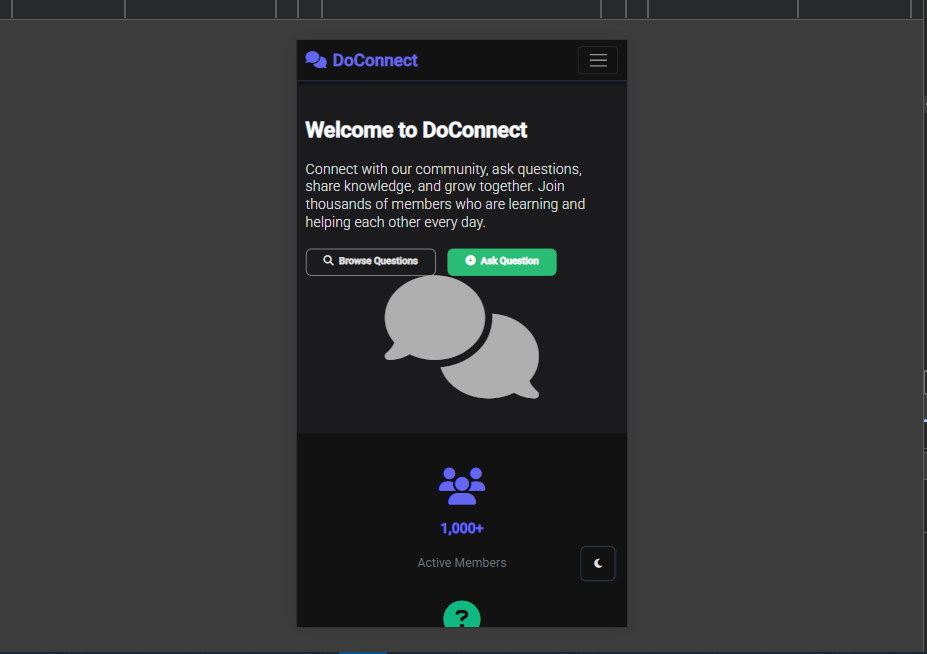
****

****

****

****

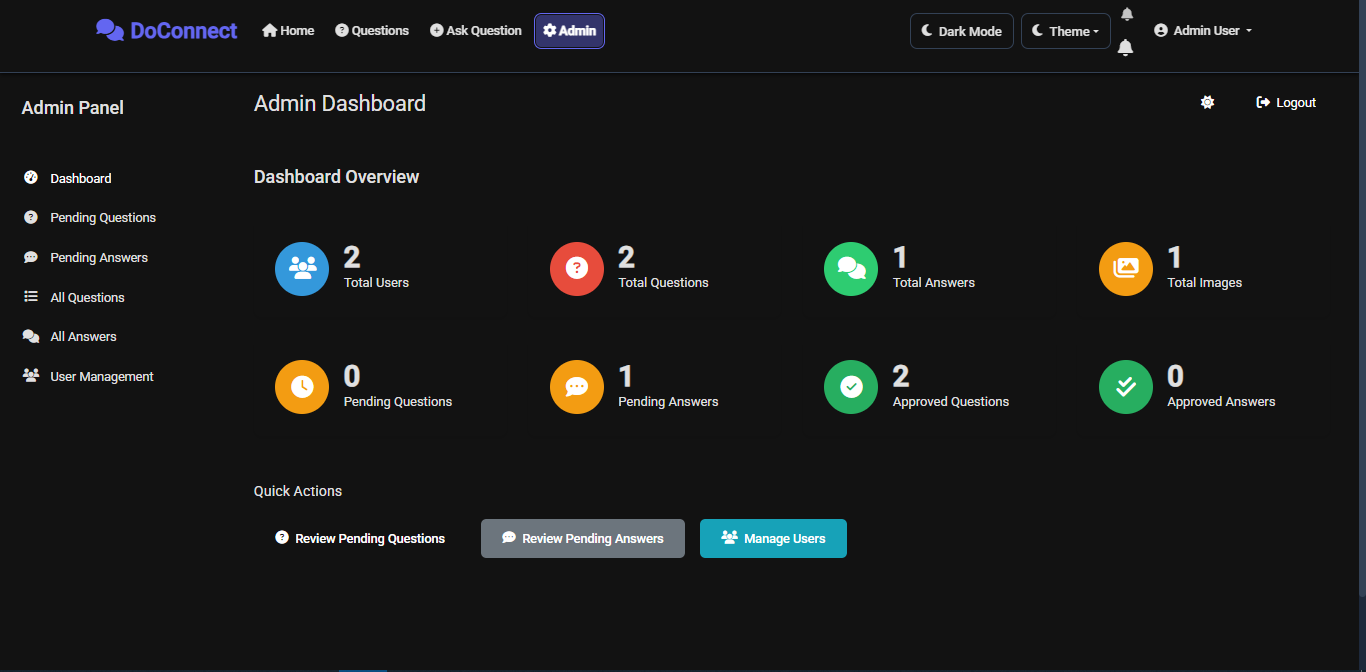
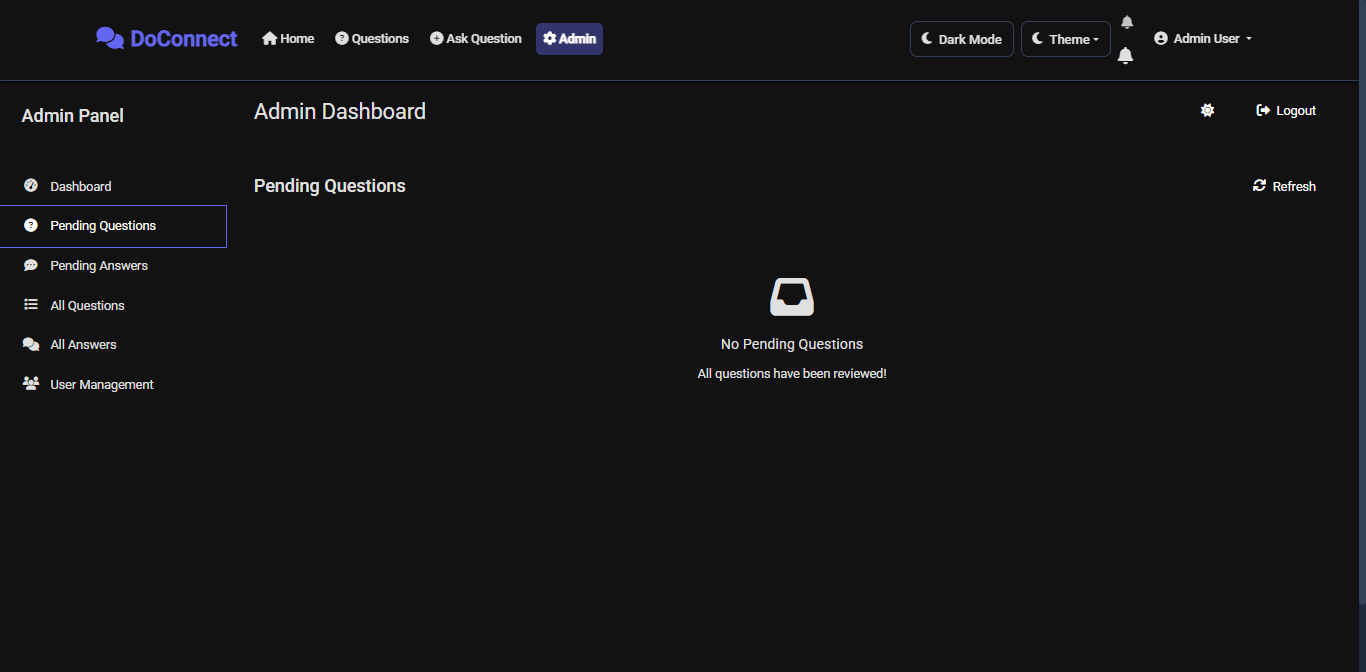
****

****

**Admin Panel**

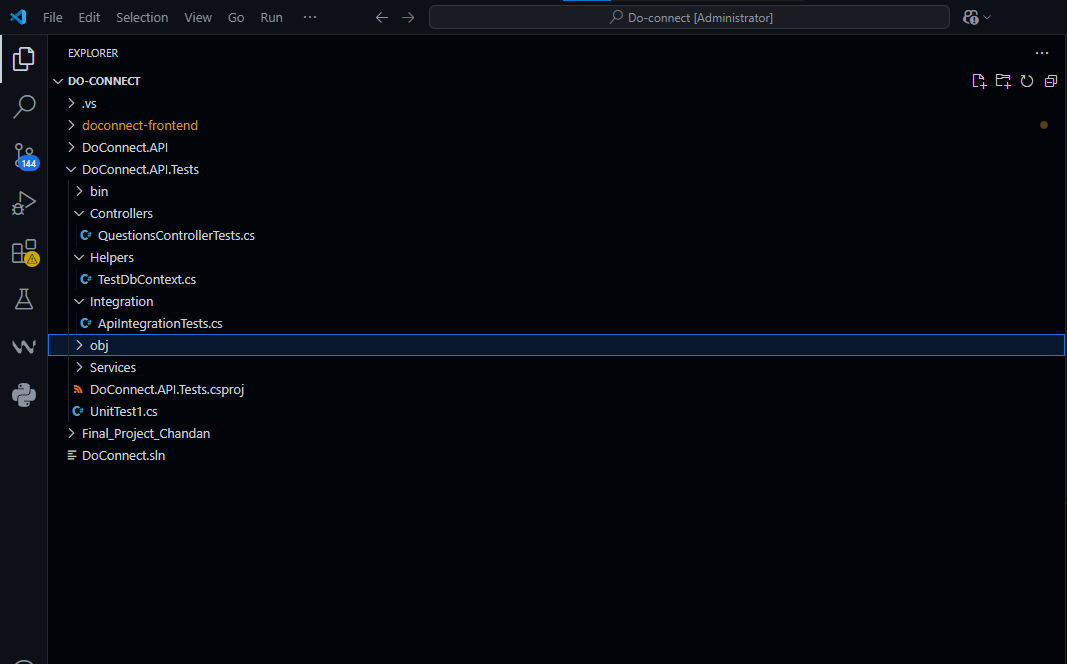
Includes:

* Dashboard overview
* Content moderation interface
* User management
* Analytics
* Real-time notifications

****

**Testing**

* Backend unit tests with xUnit
* Frontend tests with Jasmine/Karma
* Integration tests for API endpoints and authentication
* Coverage and test results screenshots included



**Future Enhancements**

* Advanced search (Elasticsearch)
* Email notifications
* Reputation system
* Question tagging
* Mobile app development
* AI-powered content suggestions
* Video upload support
* Multi-language support
* Machine learning for moderation
* Real-time collaboration
* API rate limiting and caching
* Microservices architecture

**👥 Team Members**

* Developer: **CHANDAN A S**
* Duration: 1 WEEK
* Instructor: **PATIL MAM**

**📞 Support**

* **Email:** helloworld.chandan@gmail.com
* **GitHub Issues:** [Repository Issues Link]

**License**

This project is developed for educational purposes as part of a capstone project.  
© 2025 **DoConnect** — All Rights Reserved.