MPB PORTAL WEB APP - Documentation

1. Project Overview

The MPB PORTAL WEB APP is a web-based system developed using ASP.NET Core MVC for managing user activities, reports, statistics, and case management. It allows different user roles (Admin, Organization, User) to perform specific operations securely and efficiently.

2. Features

- User Authentication & Authorization
- User Management
- Case Management
- Reports Management
- Activity Management
- Statistics & Data Visualizations
- Error Handling & Logging

3. Technologies Used

Backend: ASP.NET Core 6.0+, Entity Framework Core, ASP.NET Identity, SQL Server

Frontend: Bootstrap 5, Chart.js, jQuery & AJAX

Libraries: EPPlus, CsvHelper, iTextSharp

4. Software Requirements

- .NET SDK 6.0+
- Visual Studio 2022+
- SQL Server 2019+
- Microsoft SQL Server Management Studio (SSMS)
- Node.js (optional)

5. Installation Guide

1. Clone the Repository:

   git clone https://github.com/your-repository-url/mpb-portal.git

   cd mpb-portal

2. Set Up Database in SQL Server
3. Update `appsettings.json` with Database Connection
4. Run Database Migrations:

dotnet ef migrations add InitialCreate

dotnet ef database update

5. Run the Project:

dotnet run

6. Open https://localhost:7195/ in browser

6. Configuration

Email Settings (SMTP) - Update `appsettings.json` for SMTP configurations.

Add email sender in `Program.cs` using `builder.Services.AddScoped<IEmailSender, EmailSender>();`

7. Database Schema

- Users
- Roles
- UserRoles
- Cases
- Reports
- Activities

8. System Modules

- Authentication Module
- User Management
- Case Management
- Report Management
- Activity Management
- Statistics & Reports

9. User Roles and Permissions

Admin - Manage Users, Approve/Deny Reports, Update Cases

Organization - Create Activities, Submit Reports

User - Submit Reports, View Activities

10. API Endpoints

Authentication:

- POST /Account/Login
- POST /Account/Register
- POST /Account/VerifyEmail

- POST /Account/ChangePassword

Case Management:

- GET /Cases
- POST /Cases/Create
- PUT /Cases/Edit/{id}
- DELETE /Cases/Delete/{id}

Report Management:

- GET /Reports
- POST /Reports/Create
- PUT /Reports/Edit/{id}
- DELETE /Reports/Delete/{id}
- GET /Reports/Download?type=pdf

11. Error Handling and Logging

Logs errors in `logs/error.log`

Uses ASP.NET Core built-in logging

Bootstrap alerts display errors to users

12. Security Features

Authentication & Authorization using ASP.NET Identity

Data Validation using FluentValidation

Hashed Passwords, CSRF Protection, SQL Injection Prevention

13. Future Enhancements

- Real-time notifications
- Multi-Factor Authentication (MFA)
- Enhanced audit logging
- Improved mobile responsiveness

14. Contributors

- Lead Developer: [Your Name]
- Project Manager: [Your Team Lead]
- QA & Testing: [QA Engineer Name]