**Report: Device Management Software Using .NET Core MVC**

**Github link**

**https://github.com/thienvu290620002/CSE422.git**

**Project Overview**

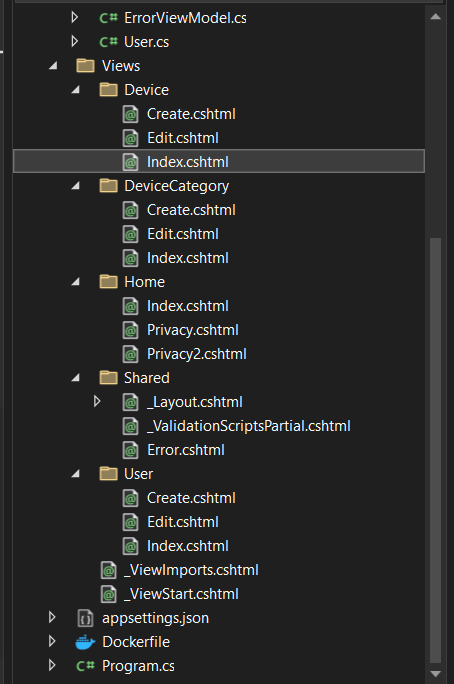
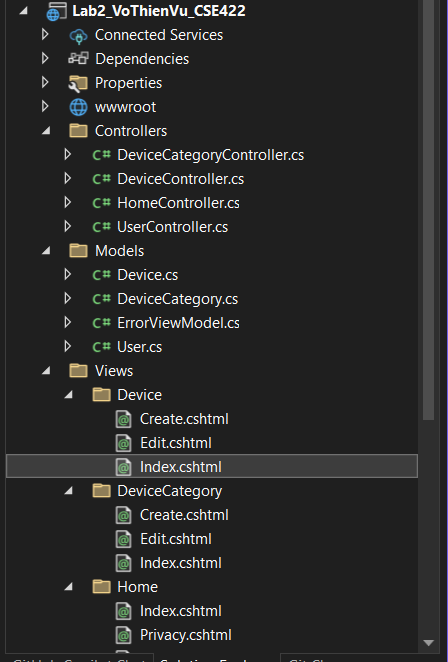
- The Device Management Software is a web application built using .NET Core MVC that allows users to manage device categories, devices, and users effectively. The application provides functionalities for adding, editing, deleting, and displaying device categories and devices, as well as managing user information. The software also includes search and filter capabilities to enhance user experience.

**Project Structure**

- The project is structured following the MVC (Model-View-Controller) design pattern, which promotes separation of concerns and enhances maintainability. The main components of the project are:

1. **Models:** Represents the data structure of the application.
2. **Views:** Contains the UI components that users interact with.
3. **Controllers:** Handles user input and interacts with the models to update the views.

**Directory Structure**



**Created Classes**

1. **DeviceCategory**

. Properties: Id, Name

. Represents the categories of devices.

1. **Device**

. Properties: Id, Name, Code, CategoryId, Status, DateOfEntry

. Represents the devices being managed.

1. **User**

. Properties: Id, FullName, Email, PhoneNumber

. Represents the users who utilize the devices.

**Controllers**

1. **DeviceCategoryController**

. Manages CRUD operations for device categories.

. Methods: Index(), Create(), Edit(), Delete()

1. **DeviceController**

. Manages CRUD operations for devices.

. Methods: Index(), Create(), Edit(), Delete(), Search(), Filter()

1. **UserController**

. Manages CRUD operations for users.

. Methods: Index(), Create(), Edit(), Delete()

**Services**

1. **DeviceService**

. Contains business logic for managing devices.

. Methods: AddDevice(), EditDevice(), DeleteDevice(), GetDevices(),

SearchDevices(), FilterDevices()

1. **UserService**

. Contains business logic for managing users.

. Methods: AddUser (), EditUser (), DeleteUser (), GetUsers()

**Implementation of Features**

1. **Device Category Management**

. Implemented CRUD operations for device categories in DeviceCategoryController.

. Used Entity Framework Core for database interactions.

. Created views for listing, adding, and editing categories.

**2. Device Management**

. Implemented CRUD operations for devices in DeviceController.

. Each device includes necessary fields such as name, code, category, status, and date of entry.

. Implemented search and filter functionalities to enhance usability.

**3. User Management**

. Implemented CRUD operations for users in User Controller.

. User information includes full name, email, and phone number.

. Created views for listing, adding, and editing users.

**4. Search and Filter Data**

. Implemented search functionality in DeviceController to find devices by name or code.

. Implemented filtering options to display devices based on status or category.

**UI Design**

- The user interface is designed to be simple and user-friendly, with clear navigation and intuitive forms for data entry. Each view is structured to provide a seamless experience for users, ensuring that they can easily manage device categories, devices, and users.

**Validation**

- Input validation is implemented using data annotations in the model classes. This ensures that all required fields are filled out correctly before submission. Custom validation messages are provided to guide users in correcting any errors.

**Coding Standards and Clean Code Principles**

- The code adheres to established coding standards, including:

. Meaningful naming conventions for classes, methods, and variables.

. Consistent code formatting for readability.

. Separation of concerns by organizing code into models, views, and controllers.

. Use of comments and documentation to explain complex logic.

- By following Clean Code principles, the codebase is kept maintainable and easy to understand, facilitating future enhancements and modifications.

**Conclusion**

- The Device Management Software project successfully implements the required functionalities while adhering to best practices in coding structure and Clean Code principles. The application is designed to be user-friendly and maintainable, ensuring that users can efficiently manage devices, categories, and user information. The project structure promotes clarity and organization, making it easier for developers to navigate and extend the application in the future. Overall, this project serves as a solid foundation for device management solutions, demonstrating the effective use of .NET Core MVC and modern software development practices.