Device Management System - Implementation Report

Name: Nguyen Cong Minh

IRN: 2131200085

Course: CSE422 - Lab 2

Project Overview

The Device Management System is a web-based application developed using ASP.NET Core MVC that enables efficient management of devices within an organization. The system provides features for tracking devices, their categories, status, and assignments to users.

Project Structure

The project follows a standard ASP.NET Core MVC architecture with the following main components:

Models

- **Device**: Core entity representing a device with properties like Name, Code, Status, and relationships to Category and User
- **DeviceCategory**: Represents device categories for organization
- **User**: Represents system users who can be assigned devices
- **Enums**: Contains DeviceStatus for tracking device states
- **ViewModels**: Custom models for view-specific data like DeviceFilterViewModel
- **Interfaces**: IRepository and IDeviceRepository defining data access contracts

Controllers

- **DeviceController**: Handles device-related operations (CRUD, search, filter)
- **UserController**: Manages user-related operations
- **HomeController**: Handles basic navigation and home page

Views

Organized by controller with views for:

- Device listing, creation, editing, and details
- User management
- Home page and navigation

Data Access

- **DeviceManagementContext**: EF Core DbContext for database operations
- **Repository Pattern**: Implementation of generic and specific repositories
- **BaseRepository**: Generic implementation of common data operations
- **DeviceRepository**: Device-specific data access implementation

Key Features Implementation

1. Device Management

- CRUD operations implemented in DeviceController
- Form validation using Data Annotations and Model Validation
- Proper error handling and user feedback
- Related data loading using Entity Framework Include statements

2. Search and Filtering

- Combined search functionality across device names and codes
- Category-based filtering
- Status-based filtering
- All filters can be applied simultaneously using LINQ
- Case-insensitive search implementation

3. User Assignment

- Devices can be assigned to users
- User assignments tracked in database
- Proper relationship management between Device and User entities

4. Data Access Layer

- Repository pattern implementation for separation of concerns
- Generic repository for common operations
- Specialized device repository for device-specific operations
- Async/await pattern used throughout for better performance

5. User Interface

- Clean and responsive design using Bootstrap
- Intuitive navigation and user feedback
- Form validation with client and server-side checks
- Proper error handling and display

Technical Implementation Details

Database Design

- SQL Server database with proper relationships
- Foreign key constraints for data integrity
- Indexes for improved query performance

Code Organization

- Proper separation of concerns (MVC pattern)
- Use of interfaces for dependency injection
- ViewModels for specialized view requirements
- Shared layouts for consistent UI

Performance Considerations

- Async operations for database access
- Efficient LINQ queries for filtering
- Proper use of Include statements to avoid N+1 queries
- Client-side validation to reduce server load

Conclusion

The Device Management System successfully implements all required features while following best practices in software development. The use of repository pattern, proper separation of concerns, and efficient data access patterns ensures the application is maintainable and performant.