**Name: Đặng Thị Ánh Tuyết**

**ID: 2131200054**

**Lab 2: Device Management**

1. **Project Overview**

**Objective:** Develop device management software using .NET Core MVC.

**Main Features:** Manage device categories, devices, and users; implement search and filter functionality.

1. **Project Structure**

**Controllers**

CategoryController.cs: Handles requests related to device categories, including CRUD operations.

DeviceController.cs: Manages requests for adding, editing, deleting, and listing devices.

UserController.cs: Handles CRUD operations for users.

HomeController.cs: May include dashboard-related features or redirect users to specific views.

**Models**

Category.cs: Represents the data structure for device categories.

Device.cs: Represents a device, including properties like DeviceName, DeviceCode, CategoryId, Status, and DateOfEntry.

User.cs: Represents user data such as FullName, Email, and PhoneNumber.

**Enums**

Gender.cs (if needed): Represents user gender options.

Status.cs: Represents device statuses (e.g., InUse, Broken, UnderMaintenance).

**Views**

Organized by feature (e.g., Category, Device, User) with Razor pages for Create, Edit, and Index.

1. **Key Functionalities**

**Device Category Management**

Use CategoryController to manage CRUD actions for categories.

The Index view displays a list of all categories.

**Device Management**

DeviceController implements methods to create, edit, delete, and list devices.

Includes features like associating devices with categories and filtering by status or category.

**User Management**

UserController implements CRUD actions for users.

Validation ensures users' full names, email addresses, and phone numbers are valid.

**Search and Filter**

Implemented in DeviceController with parameters for filtering by status and searching by name or code.

1. **UI Design**

Keep the interface simple using Bootstrap for responsive design.

Create navigation links to each feature (e.g., Category Management, DeviceManagement, User Management).

1. **Coding Standards**

Follow naming conventions like PascalCase for classes and methods and camelCase for local variables.

Use dependency injection for database access.

Separate layers for models, controllers, and views.