ĐẠI HỌC BÁCH KHOA HÀ NỘI

Trường Công nghệ thông tin và Truyền thông

A red and yellow logo

AI-generated content may be incorrect.

Tài liệu thiết kế phần mềm

(Software Design Document – SDD)

Phiên bản 1

Kĩ thuật phần mềm

Đề tài: Quản lý chung cư BlueMoon

Nhóm 10

Nguyễn Văn Thành Đạt - 20225606

Nguyễn Mạnh Thái Hà - 20225621

Nguyễn Thanh Tân - 20225923

Nguyễn Minh Quân – 20235814

Nguyễn Trường Sơn - 20230097

*Hà Nội, ngày 10 tháng 05 năm 2025*

**Mục lục**

Mục lục 1

1 Giới thiệu 2

1.1 Mục đích 2

1.2 Phạm vi 2

1.3 Từ điển thuật ngữ 2

1.4 Tài liệu tham khảo 2

2 Thiết kế kiến trúc 3

2.1.1 Lựa chọn kiến trúc phần mềm 3

2.1.2 Thiết kế tổng quan 3

2.1.3 Thiết kế chi tiết gói 4

2.2 Thiết kế chi tiết 5

2.2.1 Thiết kế cơ sở dữ liệu 5

2.2.2 Thiết kế lớp 5

2.2.3 Thiết kế giao diện 6

# Giới thiệu

## Mục đích

Tài liệu này mô tả chi tiết thiết kế hệ thống quản lý và thu phí chung cư BlueMoon được phát triển bằng công nghệ web với Node.js và MongoDB. Tài liệu được tạo ra để cung cấp hướng dẫn chi tiết cho đội phát triển trong quá trình cài đặt phần mềm, làm cơ sở cho việc kiểm thử và bảo trì hệ thống, đảm bảo tính nhất quán trong thiết kế và phát triển.

Đối tượng đọc tài liệu:

* Đội phát triển phần mềm
* Kiến trúc sư phần mềm
* Tester và đội QA
* Người quản lý dự án

## Phạm vi

Tài liệu này mô tả thiết kế cho hệ thống quản lý chung cư BlueMoon phiên bản 1.0, bao gồm các module chính:

* Module xác thực và phân quyền: Đăng nhập, quản lý session, phân quyền theo vai trò
* Module Admin (Kế toán): Quản lý khoản thu, thu phí, thống kê tài chính
* Module Tổ trưởng/ Tổ phó: Quản lý hộ khẩu, nhân hẩu, tạm trú tạm vắng, báo cáo
* Module Cư dân: Thông tin cá nhân, thanh toán trực tuyến, phản hồi

Hệ thống được thiết kế như một ứng dụng web sử dụng:

Backend: Node.js với Express.js framework

Frontend: EJS template engine với Bootstrap CSS

Database: MongoDB với Mongoose ODM

Session Management: Express-session với MongoDBstore

## Từ điển thuật ngữ

|  |  |
| --- | --- |
| Thuật ngữ | Định nghĩa |
| Route Handler | Hàm xử lý HTTP request trong Express.js |
| Middleware | Hàm trung gian xử lý request trước khi đến route handler |
| Schema | Cấu trúc định nghĩa model trong Mongoose |
| Collection | Bảng dữ liệu trong MongoDB |
| Session | Phiên làm việc của người dung được lưu trên server |
| EJS Template | Template engine để render HTML động |
| Populate | Kĩ thuật join dữ liệu trong Mongoose |
| Hộ khẩu | Đơn vị quản lý gồm một hoặc nhiều nhân khẩu cùng sinh sống |
| Nhân khẩu | Cá nhân cư trú trong chung cư BlueMoon |
| Chủ hộ | Người đại diện hộ khẩu |
| Khoản thu | Các loại phí mà hộ cần nộp |
| Tạm trú/ Tạm vắng |  |

## Tài liệu tham khảo

# Thiết kế kiến trúc

### Lựa chọn kiến trúc phần mềm

Hệ thống được thiết kế theo kiến trúc monolithic web application sử dụng MVC pattern với các đặc điểm sau:

**Node.js + Express.js Framework:**

* **Routes**: Định nghĩa các endpoint API và web routes
* **Controllers**: Logic xử lý nghiệp vụ được tích hợp trong route handlers
* **Models**: Mongoose schemas định nghĩa cấu trúc dữ liệu
* **Views**: EJS templates để render giao diện người dùng

**Session-based Authentication:**

* Sử dụng express-session để quản lý phiên đăng nhập
* Session data được lưu trữ server-side
* Role-based access control với middleware functions

**MongoDB Database:**

* NoSQL document database với flexibility cao
* Mongoose ODM để tương tác với database
* Schema validation và data relationships

**Lý do lựa chọn:**

1. **Rapid Development**: Express.js cung cấp framework minimalist dễ phát triển
2. **Scalability**: MongoDB hỗ trợ horizontal scaling tốt
3. **Flexibility**: NoSQL schema cho phép thay đổi cấu trúc dữ liệu dễ dàng
4. **JavaScript Ecosystem**: Sử dụng một ngôn ngữ cho cả frontend và backend
5. **Real-time Capabilities**: Node.js event-driven architecture phù hợp cho real-time features

### Thiết kế tổng quan

Sơ đồ kiến trúc phân cấp các chức năng và mô tả chi tiết cho các chức năng, quan hệ giữa các thành phần:

A diagram of a company

AI-generated content may be incorrect.

Package diagram:

A screenshot of a computer

AI-generated content may be incorrect.

### Thiết kế chi tiết gói

Giải thích ngắn gọn về thiết kế của mình :

Khoản Thu:

A diagram of a computer

AI-generated content may be incorrect.

Package controller1

- Chức năng: Đóng vai trò là tầng điều khiển trung gian giữa View1 và Model.

- Nhiệm vụ: Nhận dữ liệu từ View (giao diện người dùng), xử lý logic nghiệp vụ, tương tác với Model để lấy hoặc cập nhật dữ liệu, sau đó phản hồi lại cho View.

Package view1

- Chức năng: Hiển thị thông tin và nhận dữ liệu đầu vào từ người dùng.

- Nhiệm vụ: Giao tiếp trực tiếp với người dùng, gửi yêu cầu đến Controller và hiển thị kết quả phản hồi.

Package model1

- Chức năng: Quản lý dữ liệu và các thao tác liên quan đến thực thể Feehousehold.

- Nhiệm vụ: Lưu trữ và xử lý dữ liệu thực tế, không phụ thuộc vào View hoặc Controller, giúp đảm bảo tính độc lập và khả năng tái sử dụng.

Mối quan hệ phụ thuộc giữa các gói

- controller1 → model1: adminController tương tác với Model để lấy và lưu dữ liệu FeeHousehold.

- controller1 → view1: adminControler điều khiển quá trình cập nhật giao diện người dùng.

- view1 các lớp ↔ nhau: Các View có thể chuyển đổi qua lại, ví dụ từ danh sách sang form sửa hoặc thêm mới.

Thanh toán:

A diagram of a computer server

AI-generated content may be incorrect.

Package controller2

- Chức năng: Đóng vai trò là tầng điều khiển trung gian giữa View1 và Model.

- Nhiệm vụ: Nhận dữ liệu từ View (giao diện người dùng), xử lý logic nghiệp vụ, tương tác với Model để lấy hoặc cập nhật dữ liệu, sau đó phản hồi lại cho View.

Package view2

- Chức năng: Hiển thị thông tin và nhận dữ liệu đầu vào từ người dùng.

- Nhiệm vụ: Giao tiếp trực tiếp với người dùng, gửi yêu cầu đến Controller và hiển thị kết quả phản hồi.

Package model2

- Chức năng: Quản lý dữ liệu và các thao tác liên quan đến thực thể Paymentlist

- Nhiệm vụ: Lưu trữ và xử lý dữ liệu thực tế, không phụ thuộc vào View hoặc Controller, giúp đảm bảo tính độc lập và khả năng tái sử dụng.

Mối quan hệ phụ thuộc giữa các gói

- controller2 → model2: adminController tương tác với Model để lấy và lưu dữ liệu Paymentlist

- controller1 → view1: adminControler điều khiển quá trình cập nhật giao diện người dùng.

- view1 các lớp ↔ nhau: Các View có thể chuyển đổi qua lại, ví dụ từ danh sách sang form sửa hoặc thêm mới.

**Hộ khẩu/Nhân khẩu:**

A diagram of a computer server

AI-generated content may be incorrect.

Package Controller3

- Chức năng: Điều phối luồng dữ liệu giữa View và Model, xử lý logic nghiệp vụ.

- Lớp tiêu biểu:

adminController: Điều khiển các thao tác CRUD trên Hộ Khẩu và Nhân Khẩu.

Package View3

- Chức năng: Giao tiếp với người dùng, hiển thị thông tin và tiếp nhận đầu vào.

- Lớp tiêu biểu:

Account/Dashboard: Cung cấp các màn hình giao diện tương ứng với chức năng quản lý hộ khẩu và nhân khẩu .

Package Model3

- Chức năng: Quản lý dữ liệu cốt lõi và các thao tác truy xuất, cập nhật dữ liệu.

- Lớp tiêu biểu:

Household/user : Thực thể chính của chức năng quản lý hộ khẩu và nhân khẩu.

Mối quan hệ giữa các gói

**Controller3 và View3**: Các lớp trong gói controller phụ thuộc vào các lớp trong gói view nhằm điều khiển giao diện và xử lý sự kiện từ người dùng.

**Controller3 và Model3**: Các lớp controller thao tác dữ liệu thông qua các lớp entity trong gói model

**Household và User**: Giữa hai lớp này tồn tại mối quan hệ Composition, tức là mỗi hộ khẩu bao gồm nhiều nhân khẩu . Mỗi nhân khẩu không thể tồn tại độc lập ngoài hộ khẩu .

Cư Dân:

A diagram of a computer file

AI-generated content may be incorrect.

**Package Controller4**

-Chức năng: Điều phối luồng dữ liệu giữa View và Model, xử lý logic nghiệp vụ.

Các lớp này đều phụ thuộc vào gói View để nhận thao tác từ giao diện người dùng, đồng thời tương tác với gói Service để xử lý nghiệp vụ.

**Package View4**

-Bao gồm các lớp giao diện:

Resident: giao diện hồ sơ người dùng.

Report: giao diện gửi phản hồi.

.

**Package Model4**

- Chức năng: Quản lý dữ liệu cư dân

- Lớp tiêu biểu:

user : Thực thể chính của chức năng quản lý cư dân

**Quan hệ giữa các thành phần**

**- Controller → View**: mối quan hệ điều phối, controller nhận dữ liệu từ view

**- Controller→ Model**: tương tác với model để truy xuất, xử lý và lưu trữ dữ liệu.

## Thiết kế chi tiết

### Thiết kế cơ sở dữ liệu

A diagram of a company

AI-generated content may be incorrect.

*ERD Quản Lý chung cư*

Phần này có độ dài từ hai đến bốn trang. Sinh viên thiết kế, vẽ và giải thích biểu đồ thực thể liên kết (E-R diagram). Từ đó, sinh viên thiết kế cơ sở dữ liệu tùy theo hệ quản trị cơ sở dữ liệu mà mình sử dụng (SQL, NoSQL, Firebase, v.v.)

Ví dụ: Xác định các thực thể dữ liệu, liên kết và thuộc tính, xây dựng sơ đồ thực thể liên kết 🡪 thiết kế các bảng dữ liệu theo mô hình quan hệ, xây dựng sơ đồ các bảng quan hệ và đặc tả chi tiết cho từng bảng

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **dc: Bảng Địa chỉ** | | | | |
| **STT** | **Tên trường** | **Kiểu dữ liệu** | **Ghi chú** | **Ràng buộc** |
| 1 | Ma | Int | Khóa định danh | Khoá chính |
| 2 | Huyen | String | Quận/Huyện |  |
| 3 | Tinh | String | Thành phố/Tỉnh |  |

1. USER

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| useId | INT | PRIMARY KEY |
| name | VARCHAR | NOT NULL |
| email | VARCHAR | UNIQUE |
| password | VARCHAR | NOT NULL |
| phone | VARCHAR |  |
| position | VARCHAR |  |
| role | VARCHAR |  |
| createAt | DATETIME |  |

2. Resident

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| ResidentId | INT | PRIMARY KEY |
| Fullname | VARCHAR | NOT NULL |
| gender | VARCHAR |  |
| birthdate | DATE |  |
| phone | VARCHAR |  |
| email | VARCHAR |  |
| idNumber | VARCHAR | UNIQUE |
| familyMembers | INT |  |
| moveInDate | DATE |  |
| leaseEndDate | DATE |  |
| residentType | VARCHAR |  |
| notes | TEXT |  |
| isActive | BOOLEAN |  |
| createdAt | DATETIME |  |
| updatedAt | DATETIME |  |

3. Apartment

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| apartmentId | INT | PRIMARY KEY |
| number | VARCHAR |  |
| floor | INT |  |
| block | VARCHAR |  |
| area | FLOAT |  |
| type | VARCHAR |  |
| status | VARCHAR |  |
| isOccupied | BOOLEAN |  |
| handoverDate | DATE |  |
| createAt | DATETIME |  |

4. Payment

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| PaymentId | INT | PRIMARY KEY |
| type | VARCHAR |  |
| amount | FLOAT |  |
| dueDate | DATE |  |
| paidDate | DATE |  |
| notes | TEXT |  |
| status | VARCHAR |  |
| createdAt | DATETIME |  |
| apartmentId | INT | FOREIGN KEY |
| residentId | INT | FOREIGN KEY |

5. Feedback

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| FeedbackId | INT | PRIMARY KEY |
| title | VARCHAR |  |
| description | TEXT |  |
| category | VARCHAR |  |
| response | TEXT |  |
| attachments | TEXT |  |
| status | VARCHAR |  |
| createdAt | DATETIME |  |
| updatedAt | DATETIME |  |
| residentId | INT | FOREIGN KEY |
| apartmentId | INT | FOREIGN KEY |

6.HoKhau

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| HoKhauId | INT | PRIMARY KEY |
| soHoKhau | VARCHAR |  |
| hoTenChuHo | VARCHAR |  |
| diaChi | TEXT |  |
| khuVuc | VARCHAR |  |
| ngayLamHoKhau | DATE |  |
| ghiChu | TEXT |  |

7. NhanKhau

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| NhanKhauId | INT | PRIMARY KEY |
| hoTen | VARCHAR |  |
| biDanh | VARCHAR |  |
| ngaySinh | DATE |  |
| gioiTinh | VARCHAR |  |
| danToc | VARCHAR |  |
| tonGiao | VARCHAR |  |
| quocTich | VARCHAR |  |
| noiSinh | TEXT |  |
| cccd | VARCHAR | UNIQUE |
| ngayCap | DATE |  |
| noiCap | TEXT |  |
| ngheNghiep | TEXT |  |
| noiLamViec | TEXT |  |
| quanHeVoiChuHo | VARCHAR |  |
| diaChiTruoc | TEXT |  |
| ghiChu | TEXT |  |
| hoKhauId | INT | FOREIGN KEY |

8. BieuDoiNhanKhau

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| bienDoiId | INT | PRIMARY KEY |
| nhanKhauId | INT | FOREIGN KEY |
| loaiThayDoi | VARCHAR |  |
| ngayThayDoi | DATE |  |
| nguoiThucHien | VARCHAR |  |
| noiDung | TEXT |  |
| hoKhauId | INT | FOREIGN KEY |

9. TamTru

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| TamTruId | INT | PRIMARY KEY |
| nhanKhauId | INT | FOREIGN KEY |
| diaChiTamTru | TEXT |  |
| tuNgay | DATE |  |
| denNgay | DATE |  |
| lyDo | TEXT |  |
| trangThai | VARCHAR |  |

10. TamVang

|  |  |  |
| --- | --- | --- |
| Trường | Kiểu dữ liệu | Ràng buộc |
| TamVangId | INT | PRIMARY KEY |
| nhanKhauId | INT | FOREIGN KEY |
| diaChiTamTru | TEXT |  |
| tuNgay | DATE |  |
| denNgay | DATE |  |
| lyDo | TEXT |  |
| trangThai | VARCHAR |  |

### Thiết kế lớp

1. Tổ trưởng
   1. Biểu đồ trình tự Cập nhật nhân khẩu

A diagram of a project

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Sửa hộ khẩu

A diagram of a diagram

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Thêm hộ khẩu

A diagram of a diagram

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Xoá hộ khẩu

A diagram of a project

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Thêm nhân khẩu

A diagram of a diagram

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Xoá nhân khẩu

A diagram with text and arrows

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Xem thống kê

A diagram of a diagram

AI-generated content may be incorrect.

1. Cư dân
   1. Biểu đồ trình tự Đăng nhập

A white background with black text

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Gửi phản ánh

A diagram of a diagram

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Nộp phí

A diagram of a diagram

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Xem thông tin cá nhân

A close-up of a blueprint

AI-generated content may be incorrect.

1. Kế toán
   1. Biểu đồ trình tự Thêm khoản thu

A diagram of a flowchart

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Xoá khoản thu

A screenshot of a computer screen

AI-generated content may be incorrect.

* 1. Biểu đồ trình tự Sửa khoản thu

A diagram of a project

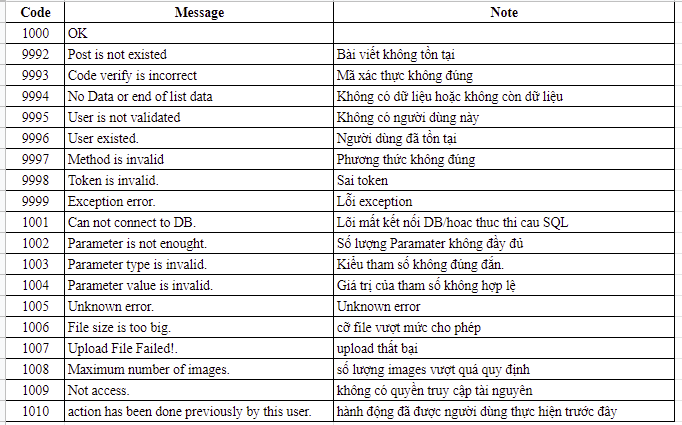
AI-generated content may be incorrect.

### Thiết kế giao diện

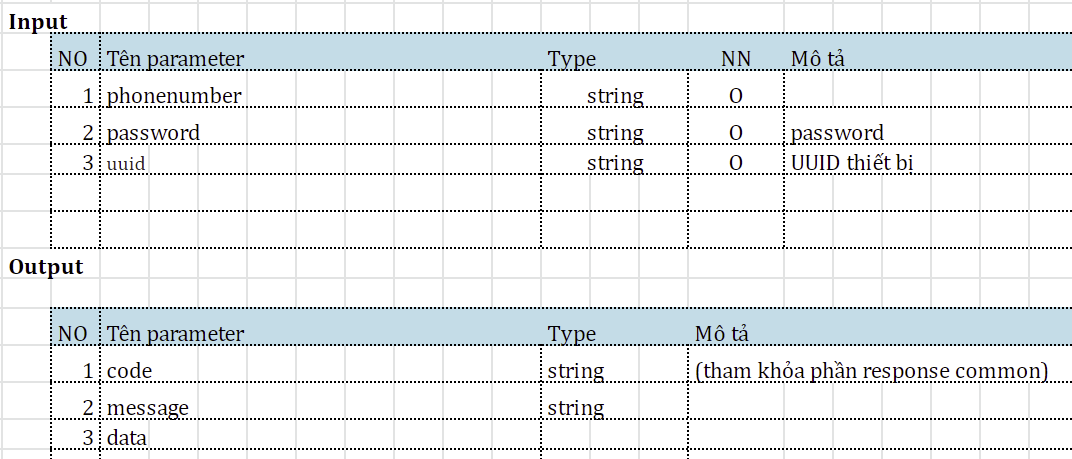
Phần này gồm 2 nội dung:

**Thiết kế giao diện API của ứng dụng**: đặc tả chi tiết các API của phía Backend

* Mô tả tổng quan về danh sách API:
  + Ví dụ: Đường dẫn cơ bản: [**https://ABC.def/it4788/**](https://abc.def/it4788/)
  + Tên miền ABC.def  sinh viên tự thiết lập, cho phép local
  + Với việc sử dụng API Login thì truy cập [**https://ABC.def/it4788/login**](https://abc.def/it4788/login&)**?**...  với các tham số đi kèm
* Danh sách mã Response khi xử lý API phía Backend, các mã này được xây dựng theo chức năng / nghiệp vụ của sản phẩm.
  + Ví dụ:



* Đặc tả chi tiết cho từng API, ví dụ:
  + API: /signup
  + Mô tả: API cho phép đăng ký một tài khoản mới của người dùng
    - Request dạng : **POST**
    - Input:
    - **- phonenumber** (số điện thoại của người dùng)
    - **- password** (mật khẩu của người dùng)
  + Kết quả: 1000|OK Nếu đăng ký thành công thì hiển thị giao diện bắt đầu ứng dụng. Nếu lỗi thì gửi mã lỗi kèm thông điệp.
  + Chi tiết input / output (O: bắt buộc, X: không bắt buộc)



**Thiết kế giao diện người dùng GUI**:

* Sinh viên đặc tả thông tin về màn hình mà ứng dụng của mình hướng tới, bao gồm độ phân giải màn hình, kích thước màn hình, số lượng màu sắc hỗ trợ, v.v. Tiếp đến, sinh viên đưa ra các thống nhất/chuẩn hóa của mình khi thiết kế giao diện như thiết kế nút, điều khiển, vị trí hiển thị thông điệp phản hồi, phối màu, v.v.
* Sinh viên xây dựng các thiết kế giao diện mockup cho các màn hình sản phẩm.
* Sau cùng sinh viên đưa ra một số hình ảnh minh họa thiết kế giao diện cho các chức năng. Lưu ý, sinh viên không nhầm lẫn giao diện thiết kế với giao diện của sản phẩm sau cùng.