**Đề Cương Dự Án Phần Mềm - Tuần 1**

**1. Mô tả bài toán**

* **Tên dự án**: Cộng đồng sinh viên
* **Mô tả ngắn gọn**: Xây dựng một ứng dụng web dành cho cộng đồng học sinh, sinh viên, bao gồm quản lý các trường đại học, đưa ra review, đưa ra thắc mắc và có thể trả lời thắc mắc đó, có thể tương tác với các bình luận( thắc mắc) và có thể đăng bài viết hoặc đọc các bài viết từ những người dùng khác.
* **Mục tiêu**:
  + Cung cấp giao diện thân thiện cho người dùng (Frontend).
  + Xử lý logic nghiệp vụ như thêm/sửa/xóa thắc mắc, bài viết, quản lý và kiểm tra những thứ người dùng đăng không phù hợp (Backend).
  + Đảm bảo dữ liệu được lưu trữ an toàn và truy vấn hiệu quả (Database).
* **Phạm vi**:
  + Người dùng: Admin, người dùng( sinh viên, học sinh).
  + Chức năng chính: Đăng nhập, đăng ký, quản lý bài viết, tìm kiếm trường mong muốn theo nguyện vọng, đưa ra nhận xét hoặc thắc mắc với những người dùng khác, chia sẻ thông tin( bài đăng) trên trang cá nhân, theo dõi người dùng mình yêu thích, like hoặc dislike những bài mình thích hoặc không đồng tình,….
* **Yêu cầu phi chức năng**:
  + Hiệu suất: Phản hồi trong vòng 2-3 giây cho các thao tác cơ bản.
  + Khả năng mở rộng: Hỗ trợ thêm tính năng trong tương lai.
  + Bảo mật: Xác thực người dùng, mã hóa dữ liệu nhạy cảm.

**2. Phân chia nhiệm vụ**

Dựa trên nhóm 3 thành viên, công việc được phân bổ như sau:

**Thành viên 1: Tài liệu phân tích thiết kế + Database**

* **Nhiệm vụ**:
  + Phân tích yêu cầu: Xác định các thực thể (entities) và mối quan hệ (relationships).
  + Thiết kế cơ sở dữ liệu: Vẽ sơ đồ ERD (Entity-Relationship Diagram), chuẩn hóa bảng.
  + Chuẩn bị tài liệu phân tích thiết kế: Bao gồm sơ đồ use-case, mô tả luồng hoạt động.
  + Đề xuất cấu trúc bảng cơ sở dữ liệu (SQL schema).

**Thành viên 2: Xử lý tương tác (Frontend + Sử dụng API)**

* **Nhiệm vụ**:
  + Thiết kế giao diện người dùng (UI/UX) cơ bản.
  + Xây dựng giao diện tĩnh (HTML, CSS, ReactJS).
  + Tích hợp API: Gọi các endpoint từ Backend để hiển thị dữ liệu.
  + Đảm bảo giao diện responsive, tương thích với các thiết bị.

**Thành viên 3: Xử lý logic nghiệp vụ (Code API)**

* **Nhiệm vụ**:
  + Xây dựng API xử lý logic nghiệp vụ.
  + Định nghĩa các endpoint API (RESTful).
  + Xử lý kết nối cơ sở dữ liệu, thực hiện các truy vấn CRUD.

**3. Lựa chọn công nghệ**

* Dựa trên bài toán, các công nghệ được đề xuất như sau:
* **Frontend**:
* Framework: **ReactJS** (với **Vite** để build nhanh và tối ưu hóa).
* Styling: **Tailwind CSS, Bootstrap** (tăng tốc độ thiết kế giao diện).
* **Backend**:
* Framework: **Laravel** (PHP-based, tích hợp Eloquent ORM, xác thực, Blade templating, và hỗ trợ tích hợp React).
* Xác thực: **Laravel Sanctum** hoặc **Laravel Authentication** (cho session-based authentication).
* View rendering: **Blade** kết hợp **React components** (React được render thông qua route trả về view).
* **Database**:
* Hệ quản trị: **MySQL** (miễn phí, tích hợp tốt với Laravel).
* Công cụ quản lý: **phpMyAdmin** (cho migration và seeding).
* **Công cụ hỗ trợ**:
* Quản lý mã nguồn: **Git** (GitHub/GitLab).
* Công cụ vẽ sơ đồ: **Draw.io**.
* IDE: **VS Code** (với extension Laravel Blade Snippets, PHP Intelephense, Vite, React Snippets).
* Build tool: **Vite** (tích hợp với Laravel qua laravel-vite plugin).
* **Lý do lựa chọn**:
* **ReactJS + Vite**: React cung cấp khả năng xây dựng giao diện động, tái sử dụng component. Vite giúp build nhanh, hỗ trợ hot module replacement (HMR), và tích hợp tốt với Laravel.
* **Laravel**: Hệ sinh thái mạnh mẽ, tích hợp sẵn Eloquent ORM, xác thực, và Blade templating. Laravel hỗ trợ tích hợp React trực tiếp qua Blade hoặc Inertia.js (nếu mở rộng sau này).
* **MySQL**: Miễn phí, đáng tin cậy, hoạt động tốt với Laravel qua Eloquent ORM.
* **Tailwind CSS, Bootstrap**: Tăng tốc độ phát triển giao diện, dễ tùy chỉnh, phù hợp với React components.
* **Authentication**: Đơn giản hóa xác thực người dùng, phù hợp với ứng dụng tích hợp trực tiếp Frontend-Backend.