

COMMAND 1- Create Constitution

/speckit.constitution

Create principles focused on code quality, testing standards, user experience consistency, and performance requirements for the "Project Management System" with these project details:

Frontend: React (HTML, CSS, JS)

Mobile App: Flutter

Backend API: PHP Laravel

Database: MySQL via XAMPP

Authentication: Laravel Breeze with Sanctum

Notifications: Laravel Notifications (Database + Mail)

Reports & Exports: Chart.js and Laravel Excel

Features:

- 1) Admin (Web): Create groups (approval), assign team leaders, view and print reports, send deadline notifications (group and individual).
- 2) Project Leader: Assign members, check deadlines, send notifications, print reports.
- 3) Individual (Mobile): Mark tasks, report difficulties to leader, receive notifications.
- 4) Security: Role-based authentication and data protection.

Include principles for:

- Code Quality
- Testing Standards
- UX Consistency
- Performance Requirements
- Security & Maintenance

COMMAND 2 - Create Specification

/speckit.specify

Build a "Project Management System" for academic and institutional use that allows structured task management between Admin, Project Leaders, and Individual Members.

Purpose:

To make project coordination, reporting, and deadline tracking easier by providing role-based tools.

System Functions:

- Admin: Creates and approves project groups, assigns leaders, tracks progress, generates printable reports, and sends deadline notifications.
- Project Leader: Assigns tasks to members, checks deadlines, sends notifications, and prints reports.
- Individual Member: Marks task completion, reports difficulties to the leader, and receives deadline alerts.

Platform:

- Admin & Leader (Web): React frontend communicating with Laravel API.
- Member (Mobile): Flutter app communicating with same Laravel backend.
- Database: MySQL (XAMPP local environment).

Non-Functional Goals:

- Secure access via Laravel Breeze (Sanctum tokens).
- Notifications via Laravel Notifications (database/mail).
- Reports with Chart.js and Laravel Excel.
- Fast performance, responsive UI, clean UX.

Focus only on features above (no chat or AI modules).

COMMAND 3 - Technical Plan

/speckit.plan

Define the technical implementation plan for the "Project Management System" using this stack:

Frontend (Web): React (HTML, CSS, JS)

Mobile: Flutter

Backend API: Laravel (PHP)

Database: MySQL (XAMPP)

Authentication: Laravel Breeze with Sanctum

Notifications: Laravel Notifications (Database + Mail)

Reports & Exports: Chart.js and Laravel Excel

Architecture:

- RESTful API layer in Laravel.
- React frontend connects via Axios to Laravel API.
- Flutter app uses HTTP client to same API.
- Role-based middleware for Admin, Leader, Member.
- Database includes tables for users, groups, projects, tasks, notifications, and reports.

Define data flow, API routes, and basic ER diagram connection plan.

COMMAND 4 - Task Breakdown

/speckit.tasks

Break down the "Project Management System" into development tasks according to the technical plan.

Main Tasks:

- 1) Setup Laravel project with Breeze, Sanctum, and database connection.
- 2) Create MySQL tables (users, groups, projects, tasks, notifications, reports).
- 3) Implement backend routes (admin, leader, member, authentication, reporting).
- 4) Build React frontend for admin and leader dashboards.
- 5) Build Flutter frontend for individual members.
- 6) Integrate Laravel Notifications (database + mail).
- 7) Integrate Chart.js and Laravel Excel for reports.
- 8) Test each role's functionality (API + UI).
- 9) Finalize documentation and deployment on XAMPP (backend) and localhost (frontend).

COMMAND 5 - Implementation Execution

/speckit.implement

Execute the implementation plan for the "Project Management System" using the defined tech stack and tasks.

Expected Deliverables:

- Fully functional backend (Laravel) with REST APIs.
- Working React web app for admin and project leader.
- Flutter mobile app for individual members.
- Authentication (Laravel Breeze + Sanctum).
- Notifications (Database + Mail).
- Reports & Charts (Chart.js + Laravel Excel).
- Testing and demo-ready local environment (XAMPP).