**Spacyn**

A campus-centric social networking platform for the Computer Science and Engineering department.

****

**Mentor:**

**TEAM MEMBERS**

## Abstract

Spacyn is a mobile-first social interface designed to foster community engagement within the CSE department. , it enables students and faculty to share updates, form interest-based groups, and build vibrant communities. The app supports secure authentication, real-time posting, and scalable backend services, ensuring a polished and professional experience.

## Project Description

Spacyn aims to:

* Share college activities, events, and announcements.
* Enable group creation for clubs, projects, and study circles.
* Foster community interaction through posts, comments, and likes.
* Provide admin tools for moderation and content curation.

## Tech Stack

| **Layer** | **Technology** | **Purpose** |
| --- | --- | --- |
| Frontend | Flutter (Dart) | Cross-platform mobile UI |
| Backend | Java + Spring Boot | REST APIs, business logic |
| Database | MySQL / PostgreSQL | Structured data storage |
| ORM | Spring Data JPA / Hibernate | Object-relational mapping |
| API Docs | Swagger / SpringDoc OpenAPI | Auto-generated API documentation |
| Authentication | JWT + Spring Security | Secure login, token-based access |
| CI/CD | GitHub Actions | Automated testing and deployment |
| Backend Hosting | Render / Railway / Fly.io | Hosting backend and database |
| Mobile Deployment | Play Store / App Store | Publishing mobile app |

## Key Features

| **Feature** | **Description** |
| --- | --- |
| Multimedia Posts | Users can share images, videos, and text updates with captions. |
| Comments & Replies | Interactive comment threads for each post, with nested replies. |
| Likes & Reactions | Emoji-based reactions to posts and comments for expressive feedback. |
| Search & Discovery | Search for users, groups, and posts by keywords, tags, or categories. |
| Group Creation | Students and faculty can create public or private groups for collaboration. |
| Tags & Categories | Posts can be tagged by topic (e.g., #Hackathon, #Placement, #Seminar). |
| Event Announcements | Create and share events with RSVP options and calendar integration. |
| Push Notifications | Real-time alerts for likes, comments, group invites, and announcements. |
| Role-Based Access | Different privileges for students, faculty, and admins. |
| Content Moderation | Admin tools to approve posts, flag inappropriate content, and manage users. |
| Analytics Dashboard | Admin view of user engagement, post reach, and group activity. |
| Profile Customization | Users can add bio, profile photo, interests, and achievements. |
| Offline Mode (Optional) | Cached content for browsing recent posts without internet. |
| AI-Powered Suggestions | Recommend groups, events, or posts based on user interests (future scope). |
| Resource Sharing | Upload and share PDFs, slides, and links within groups. |
| Responsive Design | Optimized for mobile and tablet screens using Flutter. |
| Community-Generated User ID | Communities can create shared user IDs with authorization passwords to post official updates to all users. |
| Post Authorization | Only verified community IDs with secure passwords can broadcast posts to the entire department feed. |

## Key Roles

| **Role** | **Responsibilities** |
| --- | --- |
| Project Lead | Architecture, roadmap, team coordination |
| Backend Developer | API design, database modeling, security implementation |
| Frontend Developer | UI/UX design, Flutter development, integration |
| DevOps Engineer | CI/CD setup, hosting, monitoring |
| QA Engineer | Manual and automated testing |
| Content Moderator | Community management, post approvals |

## Detailed Implementation

### 1. ****Frontend (Flutter)****

* Build responsive UI for feed, groups, profile, and notifications.
* Integrate REST APIs using http or dio.
* Use Provider or Riverpod for state management.

### 2. ****Backend (Spring Boot)****

* Define REST endpoints for posts, users, groups, and comments.
* Implement JWT-based authentication and role-based access.
* Use Spring Data JPA for ORM and database operations.

### 3. ****Database (MySQL/PostgreSQL)****

* Tables: Users, Posts, Comments, Groups, Notifications.
* Relationships: One-to-many (User → Posts), many-to-many (Users ↔ Groups).
* Special table for community-generated user IDs with password hashes.

### 4. ****API Documentation****

* Annotate controllers with Swagger/OpenAPI annotations.
* Auto-generate interactive docs for testing and onboarding.

### 5. ****Authentication****

* Secure endpoints with Spring Security.
* Issue JWT tokens on login, validate on each request.
* Community IDs require password verification before posting to public feed.

## Testing

### Unit Testing

* Backend: JUnit + Mockito for service and controller layers.
* Frontend: Flutter test for widgets and logic.

### Integration Testing

* End-to-end API tests using Postman or RestAssured.
* UI testing with Flutter Driver or integration\_test package.

### Manual Testing

* User flows: login, post creation, group joining.
* Edge cases: invalid inputs, unauthorized access.

## CI/CD

**GitHub Actions**:

* + Linting and unit tests on every push.
  + Build and deploy backend to Render/Railway.
  + Build Flutter APK/IPA for release branches.

**Versioning**:

* + Semantic versioning for releases.
  + Tagging and changelogs for visibility.

## Hosting and Deployment

### Backend

* Deploy Spring Boot app to Render or Railway.
* Use managed PostgreSQL/MySQL for persistence.

### Mobile App

* Generate release builds for Android/iOS.
* Publish to Play Store and App Store with proper metadata and screenshots.

## Outcome and Results

* A fully functional social app tailored for CSE department needs.
* Increased student engagement and visibility of departmental activities.
* Scalable architecture for future features like chat, polls, or event RSVP.
* A showcase project for your portfolio and departmental innovation.