VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

B.Tech. Winter Semester 2024-25
School Of Computer Science and Engineering
(SCOPE)

Digital Assignment - IV
Software Engineering Lab

Apurva Mishra: 22BCE2791
Date: 14 April, 2025

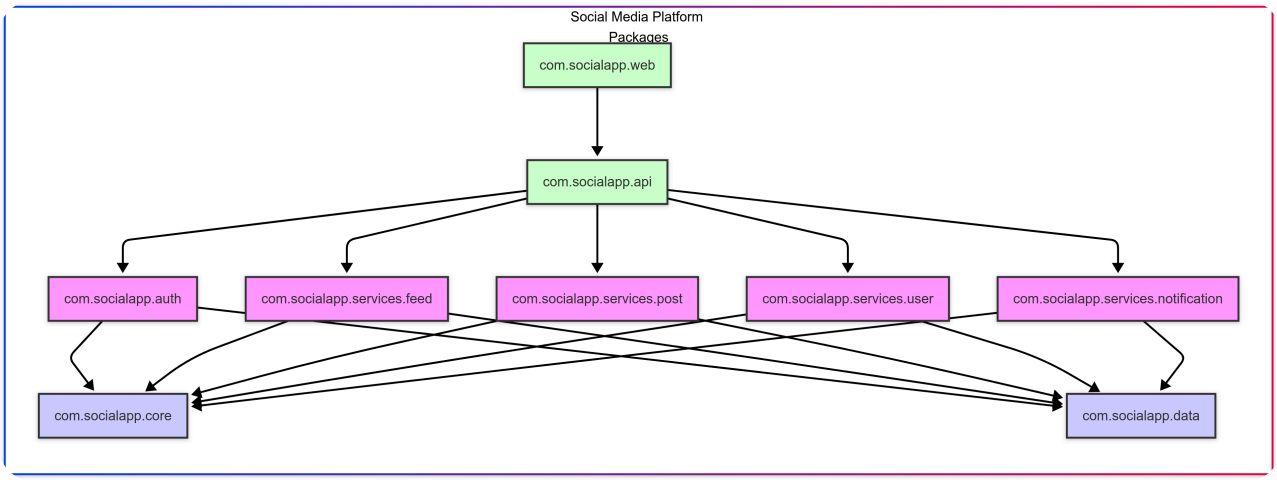
Contents

1 Question

Package, Component, and deployment models

1.1 Answer

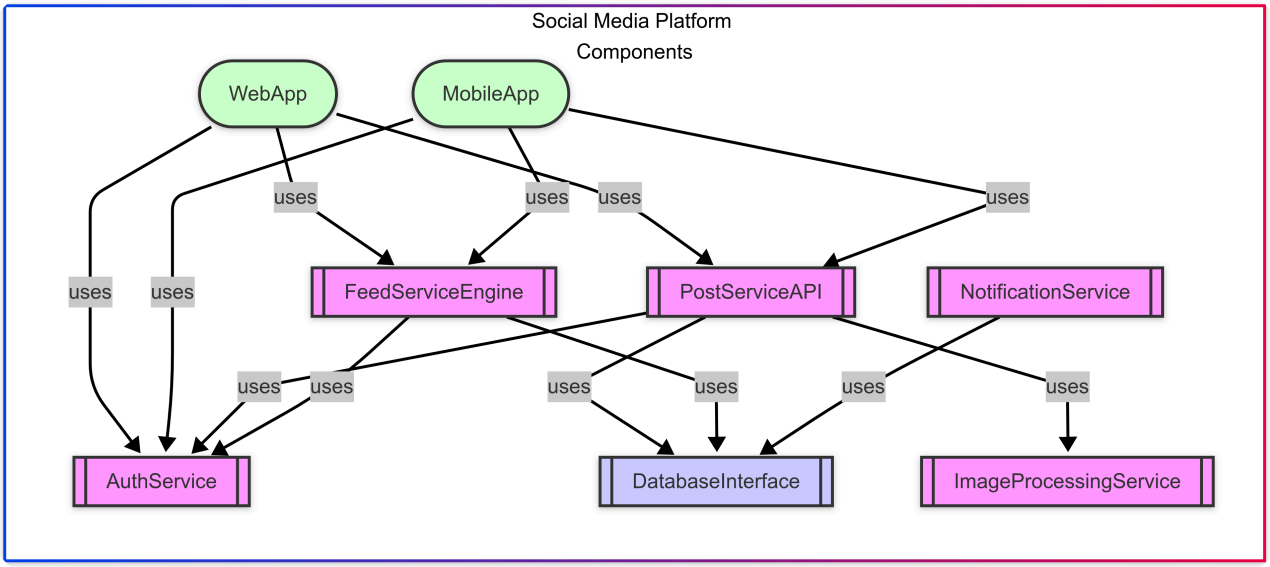
1. **Package Diagram**
   - com.socialapp.core (Core entities like User, Post)
   - com.socialapp.auth (Authentication & Authorization logic)
   - com.socialapp.services.feed (Feed generation logic)
   - com.socialapp.services.post (Post creation/management logic)
   - com.socialapp.services.user (User profile management)
   - com.socialapp.services.notification (Notification handling)
   - com.socialapp.data (Data access layer/repositories)
   - com.socialapp.web (Web application UI and controllers)
   - com.socialapp.api (REST API controllers/endpoints)



2. **Component Diagram**

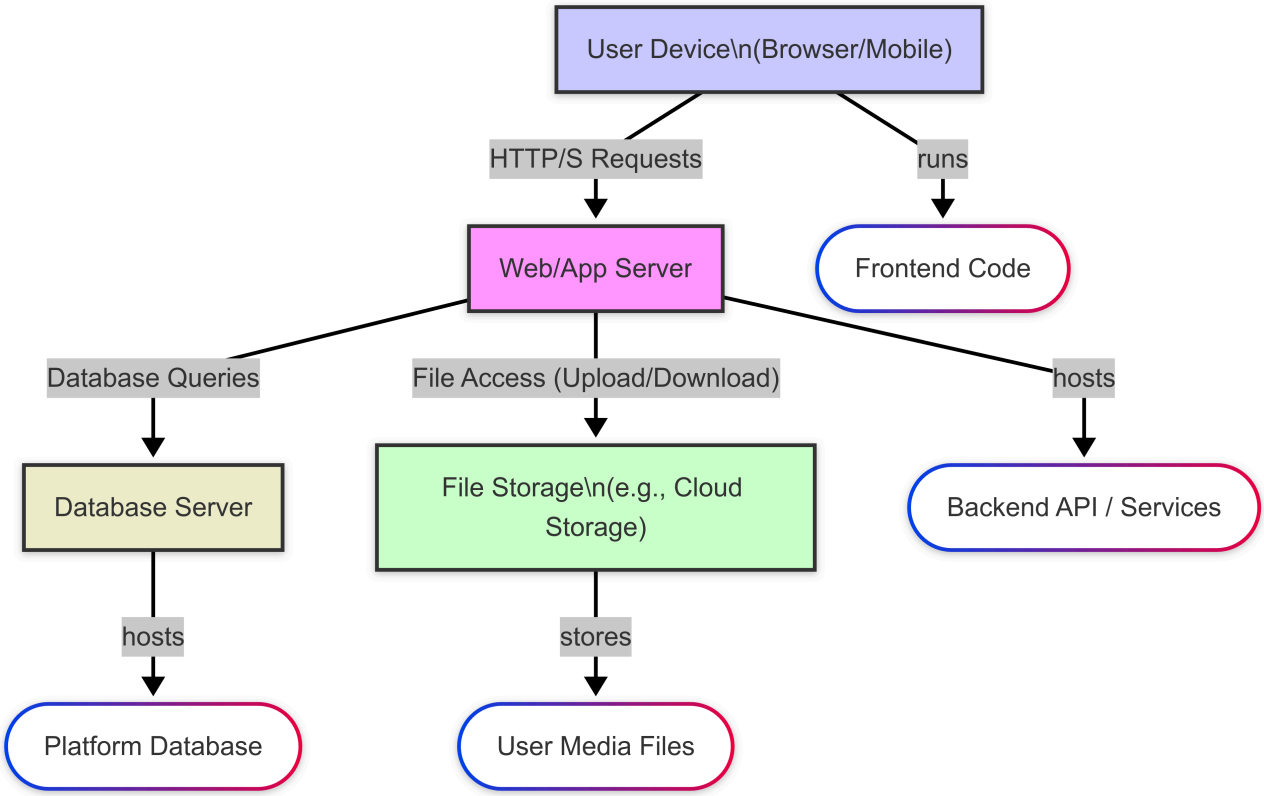
Diagram Modelling:

- Rectangles with a “component” icon,
- Interfaces as “lollipops” (provided) or “sockets” (required)
- Dependencies as arrows.



3. **Deployment Diagram**

- **User Device:** Combines both browser access (where frontend code runs dynamically) and the installed mobile app into a single node representing the user's point of interaction. Web/App Server: Represents the entire backend compute infrastructure as a single node. In reality, this could be multiple servers, load balancers, etc., but here it's simplified to where the main backend logic runs. Database Server: Represents the node(s) hosting the database system.
- **File Storage:** Represents a dedicated system (like AWS S3, Azure Blob Storage, or a local NAS) for storing user-uploaded images/videos.
- **Artifacts**: Only the most critical deployable units or data stores are shown (Frontend, Backend API, Database, Media Files). Specific .jar, .war, or script names are omitted.
- **Communication:** Shows the essential network paths between the tiers with simple labels describing the interaction type.



2 Question

Design and demonstration of test cases. Functional Testing and Non-Functional Testing

2.1 Answer

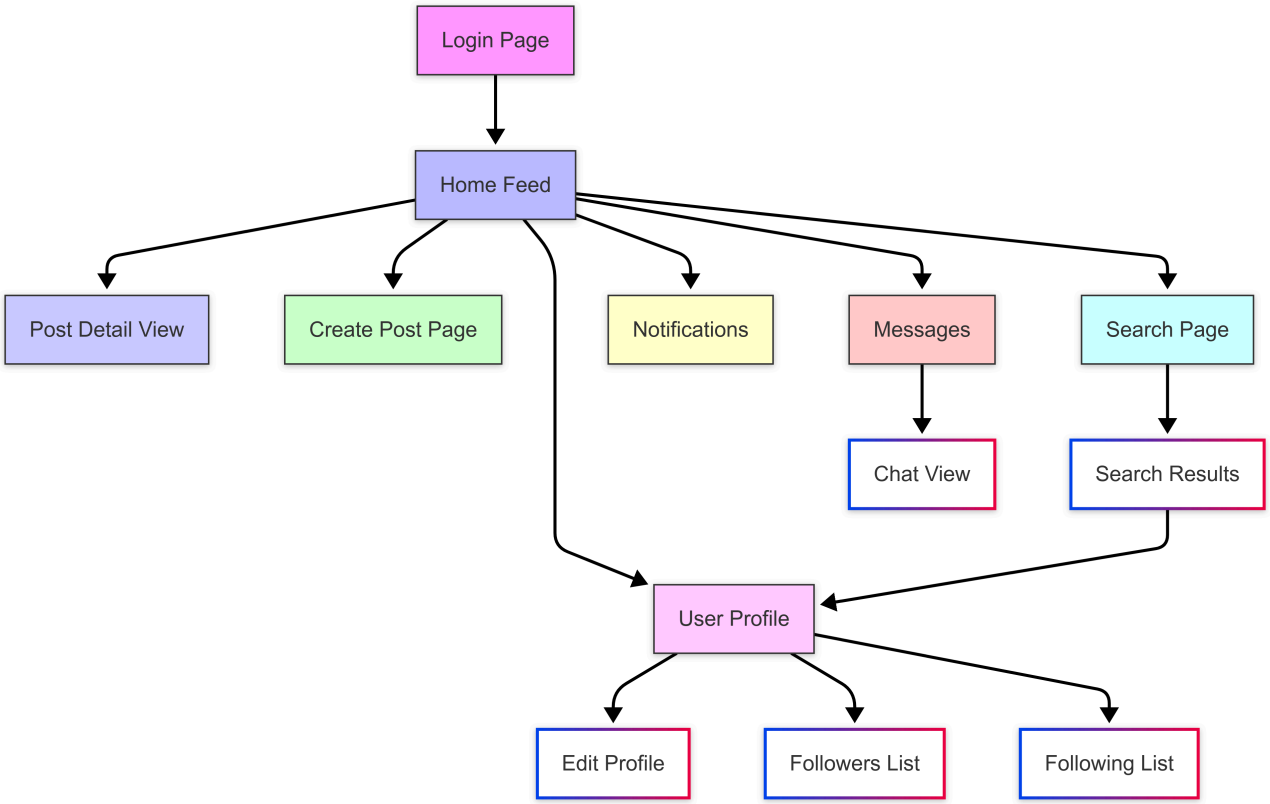
1. **Functional Testing:** Verifying what the system does against requirements.
   - User Registration: Can a new user sign up successfully? Are validations (email format, password strength) working?
   - Posting: Can users create text posts? Image posts? Video posts (if applicable)? Are character limits enforced?
   - Feed: Does the feed display posts from followed users? Is it chronological or algorithm-based as designed? Does infinite scroll work?
   - Following: Can users follow/unfollow others? Does the follower/following count update correctly?
   - Likes/Comments: Can users like/unlike posts? Can they add comments?
   - Search: Can users search for other users or content (hashtags)?
   - Profile: Can users view their own and others' profiles? Can they edit their profile information?
2. **Non-Functional Testing:** Verifying how well the system performs based on quality attributes.
   - Performance: How quickly does the feed load? What is the response time for creating a post? Image upload time? (e.g., Feed load time < 2 seconds under normal load).

- Load/Stress: How many concurrent users can the system support before response times degrade significantly? What happens during peak usage? (e.g., System supports 10,000 concurrent users with < 3s average response time).
- Scalability: Can the system handle increasing numbers of users and data by adding more server resources?
- Security: Are passwords hashed securely? Is data transmitted over HTTPS? Are common vulnerabilities (SQL Injection, XSS, CSRF) prevented? Is access control enforced correctly (e.g., can't see private profiles)?
- Usability: Is the interface intuitive and easy to navigate? Is it easy for new users to understand how to perform basic actions? (Often tested via user surveys or observation).
- Compatibility: Does the web interface work correctly on major browsers (Chrome, Firefox, Safari, Edge)? Does the mobile app work on target iOS/ Android versions and different screen sizes?
- Reliability: Does the system operate without frequent crashes or errors? What is the uptime percentage?

3 Question

User Interface design Modelling

- **Login Page** is the entry point.
- **Home Feed** is the dashboard after login, leading to:
  1. Viewing posts
  2. Creating a post
  3. Accessing profile, search, messages, and notifications
- **User Profile** leads to editing profile or exploring followers/following
- **Search Page and Search Results** lead to discovering other users
- **Messages** opens the chat view


Login Page
Home Feed
Post Detail View
Create Post Page
Notifications
Messages
Search Page
Chat View
Search Results
User Profile
Edit Profile
Followers List
Following List