Software Requirements Specification

for

PicVerse

Version <1.0>

Prepared by

Group ID: 14

Jain Tisha Navinkumar	**202412026**	**202412026@daiict.ac.in**
Kaival Shah	**202412032**	**202412032@daiict.ac.in**
Patel Dhruv Alkeshbhai	**202412066**	**202412066@daiict.ac.in**

Instructor: JayPrakash Lalchandani

Course: IT632 - Software Engineering

Date: 28/02/25

Contents

1 Introduction

1.1 Document Purpose

This Software Requirements Specification (SRS) document outlines the requirements for **PicVerse**, an online social media platform designed for content enthusiasts. The document details the functional and non-functional requirements of the application, providing a structured overview of its capabilities. This SRS serves as a guide for developers, project managers, testers, and other stakeholders. The goal is to ensure a clear understanding of PicVerse's functionalities, features, and future enhancements.

1.2 Product Scope

PicVerse is an interactive social media platform that enables users to share content, connect with influencers, and engage in discussions. The application is designed with a beginner-friendly approach, making it accessible to users with varying levels of technical expertise.

PicVerse offers a simplified social media experience with an easy-to-use interface similar to Instagram but more streamlined, ensuring a user-friendly environment. It enhances engagement and collaboration by enabling seamless communication through messaging, discussions, and content sharing. Designed with scalability in mind, the platform's architecture supports future expansion, including video support and advanced messaging features, making it adaptable to evolving user needs.

1.3 Intended Audience and Document Overview

This document is intended for the following readers:

- **Developers**: To understand the technical and functional requirements of PicVerse.
- **Project Managers**: To oversee the development process and ensure project alignment with objectives.
- **Testers**: To verify and validate the functionality of the application.
- **Clients & Professors**: To review the scope, feasibility, and expected outcomes of the project.

Document Organization:

1. **Introduction** – Provides an overview of the product, its purpose, and its scope.
2. **Overall Description** – Explains the key features, system architecture, and assumptions.

3. **Specific Requirements** – Lists the functional and non-functional requirements.
4. **Future Enhancements** – Discusses potential improvements and additional features.

1.4 Definitions, Acronyms and Abbreviations

- **SRS** – Software Requirements Specification
- **UI** – User Interface
- **SQL** – Structured Query Language

1.5 Document Conventions

Formatting Conventions:

- **Font**: Arial, size 11 or 12 for standard text.
- **Text Spacing**: Single-spaced with 1-inch margins on all sides.
- **Emphasis**: Italics are used for comments and special notes.
- **Section Titles**: Follow the IEEE standard format for headings and subheadings.

Naming Conventions:

- Tables follow uppercase notation (e.g. Lessons, Sheet Music etc.).
- Function names, variables, and database entities in the database follow a lowercase naming convention with underscores (e.g., msg_id, post_likes).

1.6 References and Acknowledgments

This project has been made possible with the guidance and support of our mentors, professors, and colleagues. We would like to extend our gratitude to:

- **Faculty Advisors & Professors** for their invaluable insights and feedback.
- **Development Team Members** for their dedication and contribution to building PicVerse.
- **Technical Resources & Online Communities** that provided knowledge and support.
- **IEEE Standard for Software Requirements Specifications (IEEE 830-1998)**

2 Overall Description

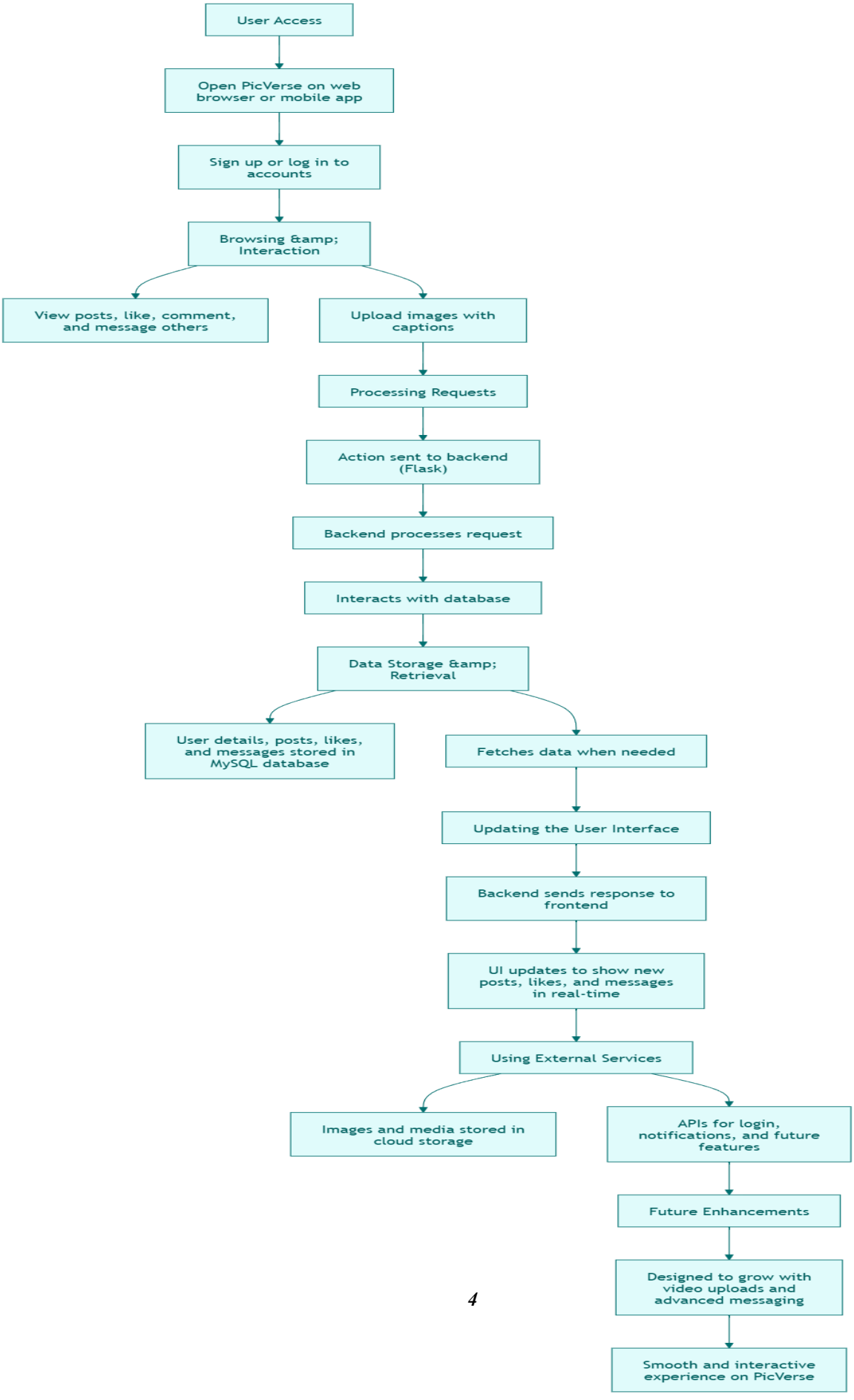
2.1 Product Overview

PicVerse is a brand-new social media platform designed for content lovers. Unlike other platforms, it is not a replacement or upgrade to an existing system but a fresh and independent product.

PicVerse aims to provide a simple and enjoyable experience, particularly for beginners, by offering a clean and easy-to-use interface. It takes inspiration from Instagram but focuses on keeping things straightforward and user-friendly. PicVerse allows users to share content, connect with influencers, and participate in engaging discussions.

The platform uses modern technologies to ensure smooth performance:

- **Frontend:** Built with ReactJS, HTML, CSS, and JavaScript to create a responsive and dynamic user interface. Displays the app and allows users to interact with content, share posts, and chat with others.
- **Backend:** Uses Python and Flask to handle app logic, user management, and data processing. Processes requests, manages data, and ensures secure login and messaging.
- **Database:** A MySQL database securely stores user profiles, posts, messages, and platform data. Stores all important information such as user accounts, posts, and conversations.
- **Future-Ready:** The system is designed to easily integrate with additional tools like analytics services or new content-sharing features in the future. Supports future add-ons, such as video-sharing and advanced analytics tools.


User Access
Open PicVerse on web browser or mobile app
Sign up or log in to accounts
Browsing & Interaction
View posts, like, comment, and message others
Upload images with captions
Processing Requests
Action sent to backend (Flask)
Backend processes request
Interacts with database
Data Storage & Retrieval
User details, posts, likes, and messages stored in MySQL database
Fetches data when needed
Updating the User Interface
Backend sends response to frontend
UI updates to show new posts, likes, and messages in real-time
Using External Services
Images and media stored in cloud storage
APIs for login, notifications, and future features
Future Enhancements
Designed to grow with video uploads and advanced messaging
Smooth and interactive experience on PicVerse

2.2 Product Functionality

PicVerse offers major functions of the system:

- **User Accounts:** Easy sign-up and secure login.
- **Content Sharing:** Users can post images and add captions.
- **Social Interactions:** Like and comment on posts to connect with others.
- **Messaging:** Send private messages to other users.
- **News Feed:** View posts from followed users in chronological order.
- **Future Growth:** The platform is built to support new features like videos and enhanced messaging.

2.3 Design and Implementation Constraints

While building PicVerse, developers need to keep the following issues and limitations in mind:

- **Device Support:** The platform should work well on both desktop and mobile devices.
- **Tech Stack:** Must use ReactJS for the frontend, Python with Flask for the backend, and MySQL for the database.
- **Design Methodology:** Follow COMET software design practices and use UML for system design visuals.
- **Security Measures:** Ensure safe logins, protect user data, and maintain privacy.
- **Programming Standards:** Use consistent and clear coding practices, such as naming variables and database tables in lowercase with underscores (e.g., post_likes, msg_id).

2.4 Assumptions and Dependencies

To successfully build and maintain PicVerse, the following conditions are expected:

- **Third-Party Tools:** Reliable external libraries and SocketIO for messaging and analytics will be available.
- **Development Setup:** All team members will have the necessary software and tools to contribute to the project.
- **Handling Growth:** The initial setup can manage a moderate number of users, with plans to scale up as needed.

- **Data Management:** The MySQL database will efficiently store and manage user information, posts, and messages.
- **User Engagement:** Users will actively use the platform, helping to build a vibrant and interactive community.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

PicVerse provides a user-friendly and intuitive interface for seamless interaction. The main user interface consists of:

- A **home feed** displaying posts from followed users in a chronological order.
- A **navigation bar** with icons for home, search, post creation, messages, and user profile.
- A **post creation interface** allowing users to upload images, add captions, and tag other users.
- A **messaging interface** featuring real-time private messaging.
- A **profile page** displaying user information, posts, and follower details.

Users will interact with the platform primarily through **touchscreens on mobile devices** and **mouse/keyboard inputs on desktops**. The interface includes dropdown menus, modals, and intuitive icons for smooth navigation.

3.1.2 Hardware Interfaces

PicVerse interacts with the following hardware components:

- **Smartphones & Tablets**: Android and iOS devices with internet connectivity for accessing the platform.
- **Personal Computers & Laptops**: Windows, macOS, and Linux devices capable of running a web browser.

- **Camera & Media Storage**: Utilized for capturing and storing images before uploading.
- **Internet Servers**: Hosting PicVerse's backend services, storing data, and managing communications.
- **Notification System**: Integrated with device notification services (e.g., push notifications on mobile devices).

3.1.3 Software Interfaces

PicVerse communicates with various software components, including:

- **Web Browsers**: Chrome, Firefox, Edge, and Safari for accessing the web-based platform.
- **Mobile Application**: A native mobile app (future enhancement) for iOS and Android.
- **APIs & External Libraries**: Integration with third-party APIs for image processing, authentication, and analytics.
- **Database (MySQL)**: Storing user data, posts, messages, and interactions.
- **Authentication Services**: OAuth-based authentication for secure user login.
- **Messaging System (Socket.IO)**: Enables real-time private messaging and notifications.

3.2 Functional Requirements

3.2.1 F1: User Authentication

- The system shall allow users to sign up using email or social media accounts.
- The system shall support secure login with username/password or OAuth authentication.
- The system shall provide password recovery options via email.

3.2.2 F2: Content Posting and Sharing

- The system shall allow users to upload images and add captions.
- The system shall support tagging other users in posts.
- The system shall allow users to edit or delete their posts.

3.2.3 F3: Social Interactions

- The system shall enable users to like and comment on posts.

- The system shall support following and unfollowing other users.
- The system shall provide notifications for interactions such as likes, comments, and follows.

3.2.4 F4: Messaging System

- The system shall allow users to send and receive private messages.
- The system shall provide real-time message notifications.
- The system shall store chat history securely.

3.2.5 F5: User Profiles

- The system shall allow users to customize their profile with a picture, bio, and personal details.
- The system shall display user posts and followers on the profile page.
- The system shall allow users to manage their account settings, including privacy options.

3.2.6 F6: News Feed

- The system shall display posts from followed users in chronological order.
- The system shall allow users to filter and sort posts based on categories or trending topics.

3.2.7 F7: Search and Discovery

- The system shall allow users to search for other users, hashtags, and trending content.
- The system shall provide personalized recommendations based on user activity.

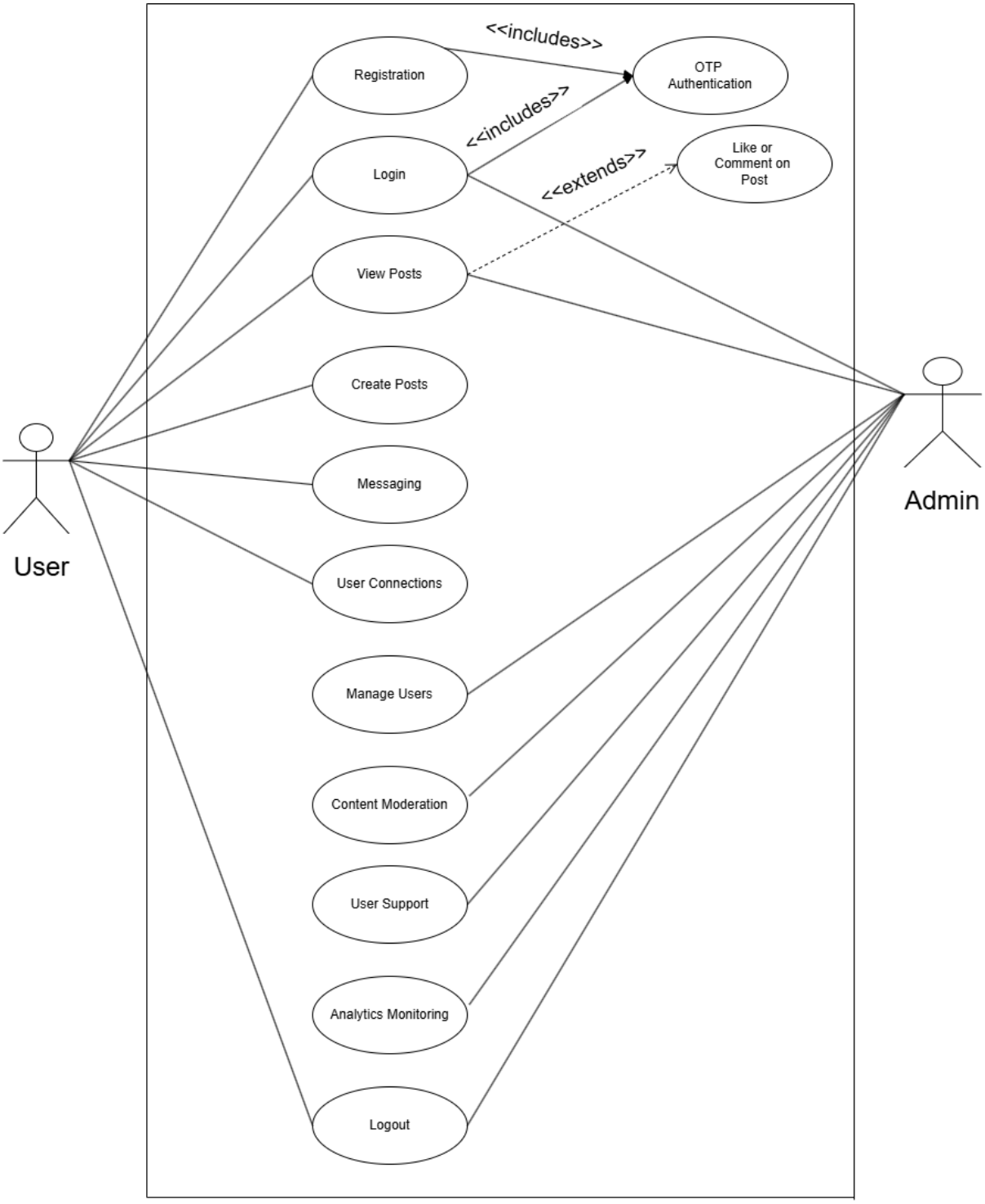
3.2.8 F8: Security and Privacy

- The system shall encrypt user passwords and sensitive data.
- The system shall allow users to report inappropriate content or block other users.
- The system shall comply with GDPR and data privacy regulations.

3.2.9 F9: Scalability and Future Enhancements

- The system shall support future enhancements such as video uploads and advanced analytics.
- The system shall allow seamless integration with third-party applications for extended functionalities.

3.3 Use Case Model



3.3.1 Use Case #1 - User Registration (U1)

Author: Kaival
Purpose: Allows a new user to create an account on the system.
Requirements Traceability: R1, R2
Priority: High
Preconditions: The user must have valid credentials (email/phone number).
Postconditions: The user account is created, and the user can log in.
Actors: User
Extends: None

Flow of Events

1. Basic Flow:

1. User navigates to the registration page.
2. User enters details (username, email/phone number, password).
3. System validates the input.
4. System sends an OTP for authentication.
5. User enters OTP.
6. System verifies OTP and creates the account.
7. User is redirected to the login page.

2. Alternative Flow:

- If OTP is not received, the user can request a resend.

3. Exceptions:

- Invalid email/phone number format.
- Weak password entered.
- OTP verification fails multiple times.

Includes: U2 (OTP Authentication)
Notes/Issues: Ensure password security measures like encryption.

3.3.2 Use Case #2 - OTP Authentication (U2)

Author: Dhruv
Purpose: Provides a secure authentication mechanism via OTP.
Requirements Traceability: R3
Priority: High
Preconditions: The user must have entered a valid email/phone number.
Postconditions: OTP is verified, and the user can proceed with the action.

Actors: User, System
Extends: None
Flow of Events

1. Basic Flow:

1. System generates an OTP and sends it to the user's email/phone.
2. User enters the OTP in the input field.
3. System verifies the OTP.
4. If correct, authentication is successful.

2. Alternative Flow:

- User requests OTP resend.

3. Exceptions:

- Expired OTP.
- Incorrect OTP entered multiple times.

Includes: None
Notes/Issues: OTP expiration time should be set.

3.3.3 Use Case #3 - User Login (U3)

Author: Tisha
Purpose: Allows an existing user to log into the system.
Requirements Traceability: R4
Priority: High
Preconditions: The user must be registered.
Postconditions: The user is successfully logged in.
Actors: User
Extends: None
Flow of Events

1. Basic Flow:

1. User enters username/email and password.
2. System validates credentials.
3. System grants access and redirects to the homepage.

2. Alternative Flow:

- If 2FA is enabled, OTP verification is required.

3. Exceptions:

- Incorrect username/password.
- Account locked due to multiple failed attempts.

Includes: U2 (OTP Authentication)
Notes/Issues: Implement account lockout after multiple failed attempts.

3.3.4 Use Case #4 - Create Post (U4)

Author: Tisha
Purpose: Allows users to create and publish a post.
Requirements Traceability: R5
Priority: High
Preconditions: The user must be logged in.
Postconditions: The post is successfully created and visible to others.
Actors: User
Extends: None
Flow of Events

1. Basic Flow:

1. User navigates to the "Create Post" section.
2. User uploads an image/video and writes a caption.
3. System validates content and stores the post.
4. Post is published successfully.

2. Alternative Flow:

- User saves the post as a draft instead of publishing.

3. Exceptions:

- Invalid file format.
- File size exceeds the limit.

Includes: None
Notes/Issues: Consider implementing AI-based content moderation.

3.3.5 Use Case #5 - View Posts (U5)

Author: Kaival
Purpose: Allows users to view posts from others.
Requirements Traceability: R6
Priority: High
Preconditions: The user must be logged in.
Postconditions: Posts are displayed on the user's feed.
Actors: User
Extends: None
Flow of Events

1. Basic Flow:

1. User navigates to the homepage.
2. System fetches posts based on relevance.
3. Posts are displayed in the feed.

2. Alternative Flow:

- User searches for a specific post.

3. Exceptions:

- Network failure.
- No posts available.

Includes: U6 (Like or Comment on Post)

Notes/Issues: Implement personalized post recommendations.

3.3.6 Use Case #6 - Like or Comment on Post (U6)

Author: Dhruv
Purpose: Allows users to engage with posts by liking or commenting.
Requirements Traceability: R7
Priority: Medium
Preconditions: The user must be logged in and viewing a post.
Postconditions: Like or comment is successfully added.
Actors: User
Extends: U5 (View Posts)
Flow of Events

1. Basic Flow:

1. User clicks the "Like" button or enters a comment.
2. System updates like/comment count.
3. If a comment is added, it appears under the post.

2. Alternative Flow:

- User edits or deletes their comment.

3. Exceptions:

- Spam detection blocks comment.
- Network failure prevents interaction.

Includes: None

Notes/Issues: Implement comment moderation for abusive content.

3.3.7 Use Case #7 - Messaging (U7)

Author: Kaival
Purpose: Enables users to send and receive messages.
Requirements Traceability: R8
Priority: High
Preconditions: Both users must be connected.
Postconditions: Message is delivered to the recipient.

Actors: User
Extends: None
Flow of Events

1. Basic Flow:

1. User opens the messaging section.
2. User selects a contact and types a message.
3. System sends the message to the recipient.
4. Recipient receives a notification.

2. Alternative Flow:

- User sends an image or video.

3. Exceptions:

- Message not delivered due to network failure.

Includes: None
Notes/Issues: Consider implementing end-to-end encryption.

3.3.8 Use Case #8 - Manage Users (U8)

Author: Tisha
Purpose: Allows admins to manage users.
Requirements Traceability: R9
Priority: High
Preconditions: Admin must be logged in.
Postconditions: Users are successfully managed (approved, banned, etc.).
Actors: Admin
Extends: None
Flow of Events

1. Basic Flow:

1. Admin navigates to the user management panel.
2. Admin searches for a user.
3. Admin performs actions (edit, ban, approve).
4. Changes are updated in the system.

2. Alternative Flow:

- Admin temporarily suspends a user instead of banning.

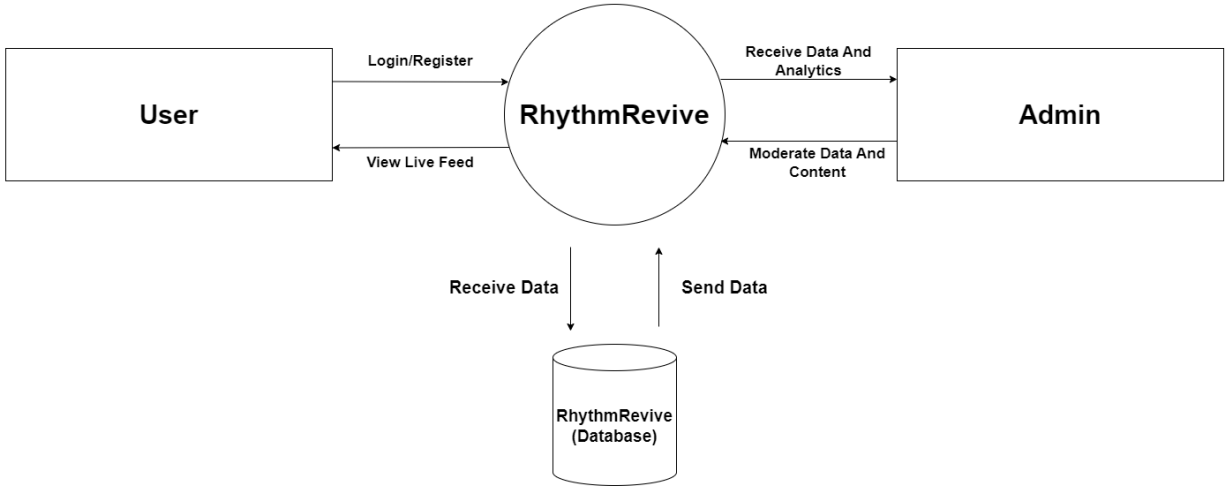
3. Exceptions:

- User data cannot be retrieved.

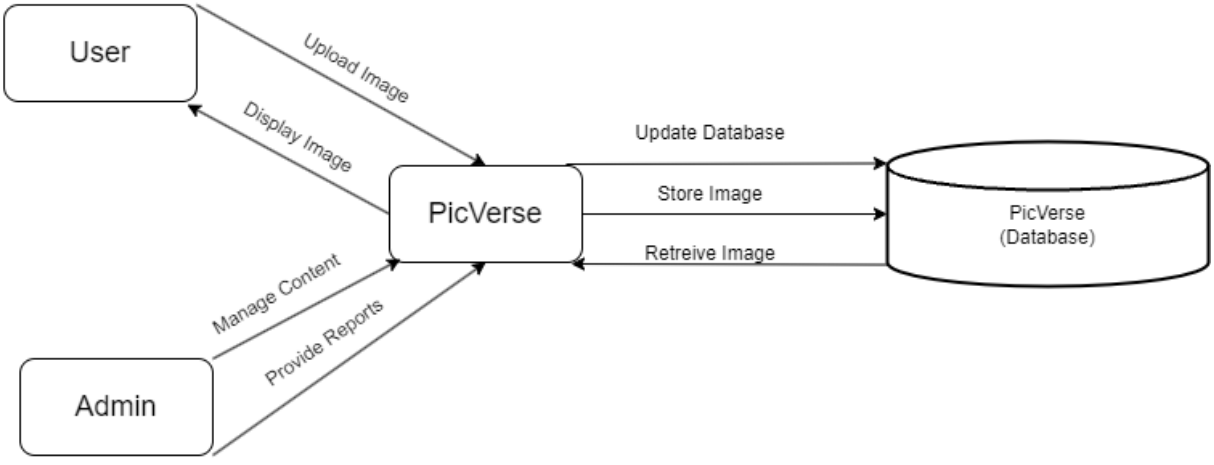
Includes: None
Notes/Issues: Ensure audit logs are maintained for admin actions.

3.4 DFD Diagrams

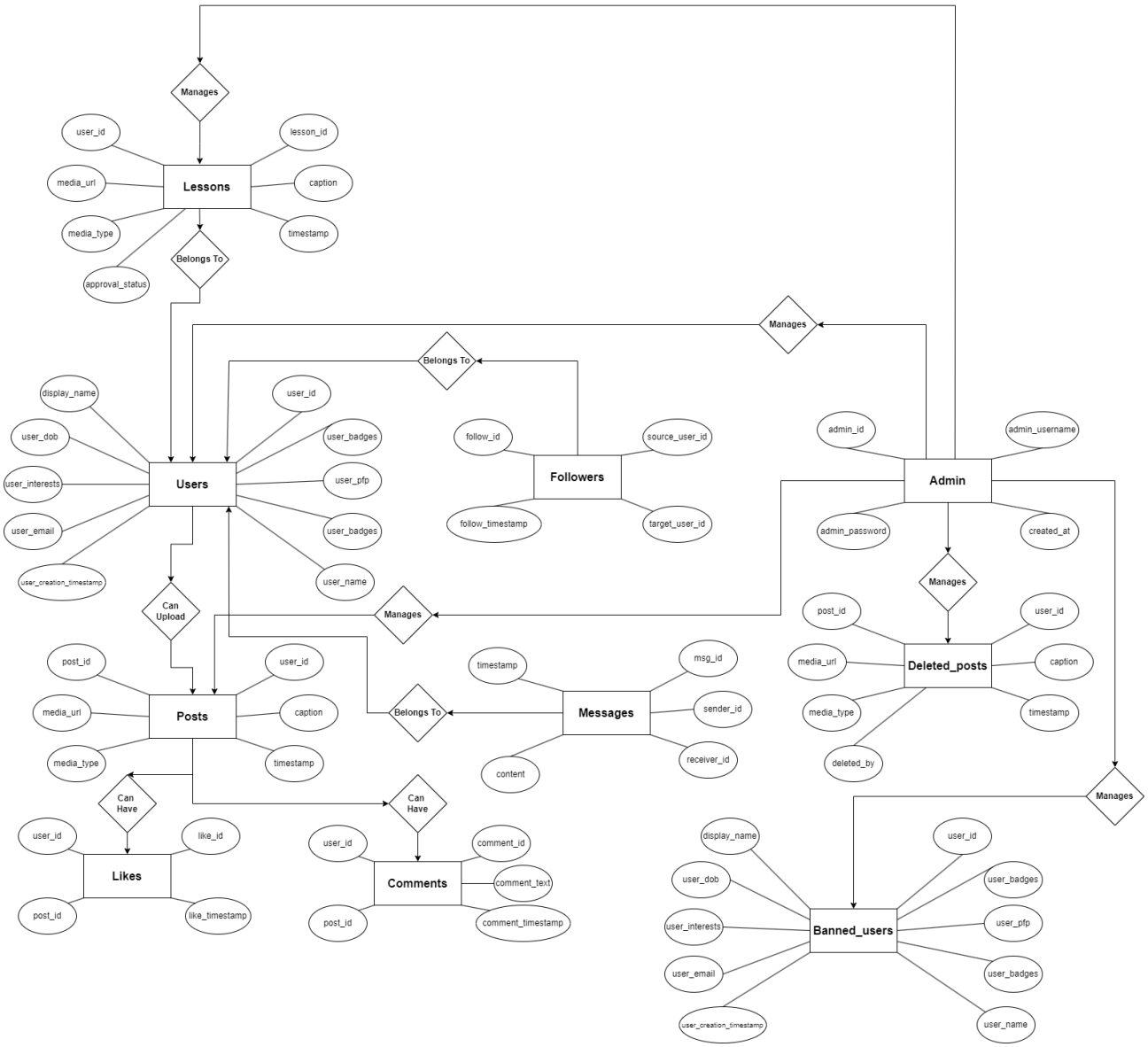
Level 0 DFD



Level 1 DFD



3.5 ER Diagram



4 Other Non-functional Requirements

4.1 Performance Requirements

P1. The system shall load the homepage feed within **2 seconds** under normal network conditions to ensure a seamless user experience.

P2. The system shall handle **up to 500 concurrent users** without significant performance degradation.

P3. The messaging feature shall deliver messages in **real-time** with a maximum delay of **500ms**.

P4. The system shall support image uploads of up to **5MB** and process them within **3 seconds** before making them available in the feed.

P5. The search functionality shall return results within **2 second** for queries with fewer than **10,000 database entries**.

P6. The system shall maintain **95% uptime** to ensure continuous availability.

4.2 Safety and Security Requirements

- User authentication shall be done using **email and password** with encrypted storage.
- Passwords shall be hashed using **SHA-256** before storage.
- Users shall have the option to **block and report** inappropriate content or users.
- Basic **HTTPS encryption** shall be implemented to secure data transmission.
- The system shall provide a simple **privacy settings page** to allow users to manage their data visibility.
- Only logged-in users shall be able to interact (like, comment, message) on posts.
- A basic security audit shall be conducted **before deployment** to check for vulnerabilities.

4.3 Software Quality Attributes

4.3.1 Reliability

- The system shall be able to restart automatically in case of a minor crash.
- User data shall be stored in a **database with regular backups** to prevent data loss.

4.3.2 Usability

- The user interface shall be **simple and easy to navigate**, designed for beginner users.
- The system shall use standard icons and layouts to ensure familiarity.

4.3.3 Maintainability

- The code shall be written in a **modular** way so that features can be added or removed easily.

- Proper **comments and documentation** shall be maintained for future modifications.

4.3.4 Scalability

- The system shall be able to handle **increasing users** by optimizing queries and using caching mechanisms.
- Additional servers may be added in the future if needed.

4.3.5 Interoperability

- The system shall allow users to log in using **Google authentication** as an alternative.
- The web and mobile versions shall synchronize user data using a **common database**.

Appendix A – Data Dictionary

1. Admin Table

Field	Data Type	Size	Description	Constraint
admin_id	Integer	5	Unique identifier for admin	Primary Key
admin_username	Varchar	255	Username for admin login	Not Null
admin_password	Varchar	255	Password for admin login	Not Null
created_at	Timestamp	8	Timestamp of admin creation	Not Null

2. Users Table

Field	Data Type	Size	Description	Constraint
user_id	Integer	5	Unique identifier for user	Primary Key
user_name	Varchar	255	Username for user login	Not Null
user_pass	Varchar	255	Password for user login	Not Null
display_name	Varchar	255	Display name of user	Null
user_email	Varchar	255	Email of user	Not Null
user_dob	Date	8	Date of Birth of user	Null
user_bio	Varchar	500	Bio of user	Null
user_pfp	Varchar	255	Profile picture path of user	Not Null
user_badges	JSON	-	Badges obtained by user	Null
user_interests	JSON	-	Interests of the user	Null
user_creation_timestamp	Timestamp	8	Timestamp of user creation	Not Null

3. Banned Users Table

Field	Data Type	Size	Description	Constraint
user_id	Integer	5	Unique identifier for user	Foreign Key
user_name	Varchar	255	Username for user login	Not Null
user_email	Varchar	255	Email of user	Not Null
banTimestamp	Timestamp	8	Timestamp of Ban	Not Null
banReason	Varchar	500	Reason for Ban	Null

4. Posts Table

Field	Data Type	Size	Description	Constraint
post_id	Integer	5	Unique identifier for post	Primary Key
user_id	Integer	5	User id from Users Table	Foreign Key
caption	Varchar	200	Caption of the post	Null
tags	Varchar	50	Tags for the post	Null
media_url	Varchar	255	Path of the media stored	Null
media_type	Enumerated	-	Type of the media posted	Null
timestamp	Timestamp	8	Timestamp of post upload	Not Null

5. Likes Table

Field	Data Type	Size	Description	Constraint
like_id	Integer	5	Unique identifier for like	Primary Key
user_id	Integer	5	User id from Users Table	Foreign Key
post_id	Integer	5	Post id from Posts Table	Foreign Key
Like_timestamp	Timestamp	8	Timestamp of the like	Not Null

6. Comments Table

Field	Data Type	Size	Description	Constraint
comment_id	Integer	5	Unique identifier for comment	Primary Key
user_id	Integer	5	User id from Users Table	Foreign Key
post_id	Integer	5	Post id from Posts Table	Foreign Key
comment_text	Varchar	200	Comment text	Not Null
comment_timestamp	Timestamp	8	Timestamp of the comment	Not Null

7. Followers Table

Field	Data Type	Size	Description	Constraint
follow_id	Integer	5	Unique identifier for follow	Primary Key
source_user_id	Integer	5	User id from Users Table	Foreign Key
target_user_id	Integer	5	User id from Users Table	Foreign Key
follow_timestamp	Timestamp	8	Timestamp of the follow	Not Null

8. Messages Table

Field	Data Type	Size	Description	Constraint
msg_id	Integer	5	Unique identifier for message	Primary Key
sender_id	Integer	5	Sender user id from Users Table	Foreign Key
receiver_id	Integer	5	Receiver user id from Users Table	Foreign Key
content	Varchar	300	Message content (text)	Not Null
timestamp	Timestamp	8	Timestamp of the message	Not Null

Appendix B - Group Log

Date	Time	Activity	Details
02/25/2025	16:30	Initial Brainstorming Session	Defined project scope and assigned tasks to team members.
02/25/2025	17:00	Discussion of Requirements	Reviewed functional and non-functional requirements. Discussed UI/UX design overview.
02/25/2025	19:30	Backend and Database Decisions	Finalized database schema and confirmed backend framework decisions.
02/25/2025	22:00	Finalization of Requirements	Finalized project requirements. Discussed potential challenges and mitigation strategies.
02/26/2025	21:30	Group Activities	Conducted research on similar platforms. Refined software requirements and created UI/UX wireframes.
02/26/2025	21:00	Group Activities	Designed the database schema, backend logic, and implemented initial frontend components using ReactJS.
02/27/2025	18:30	Group Activities	Prepared SRS document and incorporated feedback from peers and mentors.