SYNOPSIS

Project Name: PHONE BOOK DIRECTORY APP

(Everest Explorers)

Group Members:

- **Ravi Bist**
- **Sushil Shah**
- **Sudip Khatiwada**
- **Laxmi P. Awasthi**

Date: 11 March 2025

ABSTRACT

- **Problem Statement**

In the modern era, people rely heavily on smartphones and digital platforms for storing contact information. However, existing solutions often suffer from inefficiencies such as limited search capabilities, lack of categorization, data redundancy, and security vulnerabilities. Users may lose important contacts due to device failure, accidental deletion, or lack of backup options. Furthermore, traditional phonebooks or built-in contact apps lack advanced management features like tagging, sorting, and cloud synchronization.

- **Proposed Solution**

The Phonebook Directory App aims to provide a secure, efficient, and user-friendly solution for contact management. This web-based application allows users to store, update, delete, and search contacts with ease. The system integrates advanced search filters, contact categorization, and backup & restore functionalities to ensure seamless access to information. With Spring Boot for the backend, React.js for the frontend, and MySQL/PostgreSQL as the database, the app offers a robust and scalable platform.

- **Key Features**

User Authentication: Secure login system to protect personal contact data.

Contact Management: CRUD (Create, Read, Update, Delete) operations for managing contacts.

Advanced Search & Filters: Quick retrieval of contacts using name, number, or category.

Backup & Restore: Cloud-based storage options to prevent data loss.

Responsive Design: Works seamlessly across desktop and mobile devices.

- **Conclusion**

This app offers a modern and scalable contact management system, improving accessibility and security. Future upgrades include AI-based suggestions, voice search, and cloud sync for enhanced usability.

INTRODUCTION

❖ Overview of the Phonebook Directory App

The Phonebook Directory App is a web-based contact management system designed to help users store, organize, and retrieve contact details efficiently. It provides a structured and secure alternative to traditional phonebooks and scattered digital contact storage. Users can add, edit, delete, and search contacts with ease, ensuring quick access to essential information.

❖ Importance of Digital Contact Management

In today's fast-paced world, managing contacts efficiently is essential for both personal and professional use. Traditional methods such as physical diaries or basic phone contact lists lack features like categorization, quick search, and backup options. A dedicated digital phonebook enhances accessibility, reduces data loss risks, and ensures a seamless user experience across multiple devices.

With features like secure authentication, cloud backup, and smart categorization, the Phonebook Directory App offers a modern solution for efficient contact management.

OBJECTIVES

❖ Key Goals and Functionalities of the System

- **Efficient Contact Management:** Allow users to add, update, delete, and search contacts easily.
- **Advanced Search & Filtering:** Enable users to find contacts quickly using name, number, or category.
- **Secure Authentication:** Implement user authentication to protect sensitive contact data.
- **Backup & Restore:** Provide cloud-based backup to prevent data loss.
- **Categorization & Tagging:** Allow users to group contacts based on work, family, or other categories.
- **Cross-Device Accessibility:** Ensure seamless synchronization across mobile and desktop platforms.
- **User-Friendly Interface:** Design an intuitive UI for easy navigation and interaction.
- **Future Scalability:** Support future enhancements like AI-based suggestions and voice search.

SCOPE OF THE PROJECT

❖ Who Will Use It?

The Phonebook Directory App is designed for a variety of users:

- **Students:** To store academic and personal contacts securely.
- **Professionals:** To manage business contacts, clients, and colleagues efficiently.
- **Businesses:** To maintain a structured and categorized database of clients and suppliers.
- **General Users:** Anyone needing an organized and secure way to store contacts.

❖ Features Covered

- **User Authentication:** Secure login and access control.
- Contact Management: Add, edit, delete, and search contacts.
- **Advanced Search & Filters:** Quickly find contacts by name, number, or category.
- **Backup & Restore:** Cloud-based data storage to prevent data loss.
- **Categorization & Tagging:** Organize contacts based on work, family, or personal groups.
- **Cross-Device Synchronization:** Access contacts on mobile and desktop platforms.
- **Security Features:** Data encryption and secure authentication for privacy.
- **Future Upgrades:** AI-based suggestions, voice search, and cloud synchronization.

TECHNOLOGY USED

❖ Frontend (User Interface)

- **HTML, CSS, JavaScript:** Designs the layout and styles the app interface.
- **React.js:** Creates an interactive UI where users can add, edit, delete, and search contacts smoothly.

❖ Backend (Server & Logic)

- **Java (Spring Boot):** Manages user requests, processes contact data, and ensures secure authentication.
- Handles user login, contact storage, and API requests.

❖ Git & GitHub

- Used for version control, collaboration, and code management. It helps track changes, work in teams, and deploy updates efficiently.

❖ Database (Data Storage)

- MySQL/PostgreSQL: Stores user accounts and contact details securely.

❖ Other Tools

- **APIs**: Used for features like importing contacts from Google, cloud backup, or SMS integration.

SYSTEM ARCHITECTURE

❖ High-Level System Design

- **Presentation Layer (Frontend):**

  Built using React.js for an interactive and responsive UI.

  Allows users to add, edit, delete, and search contacts easily.

- **Business Logic Layer (Backend):**

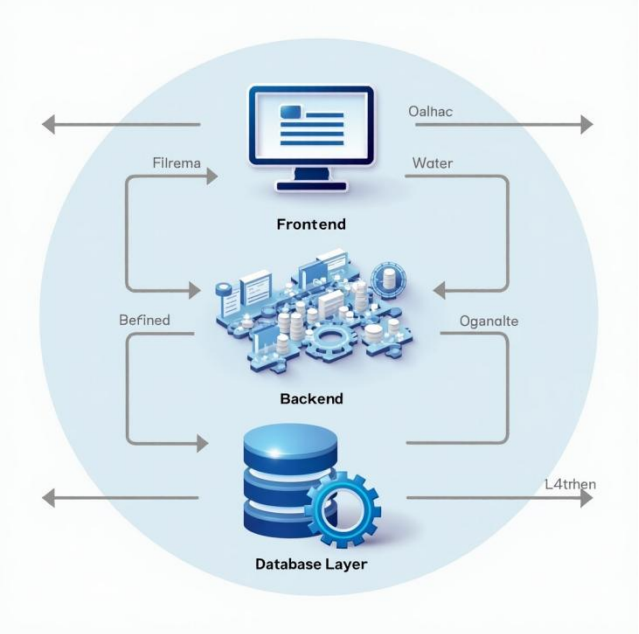
  Developed using Spring Boot, handling all business logic.

  Manages user authentication, contact operations, and API requests.

- **Data Layer (Database):**

  Uses MySQL/PostgreSQL for storing user details and contacts securely.

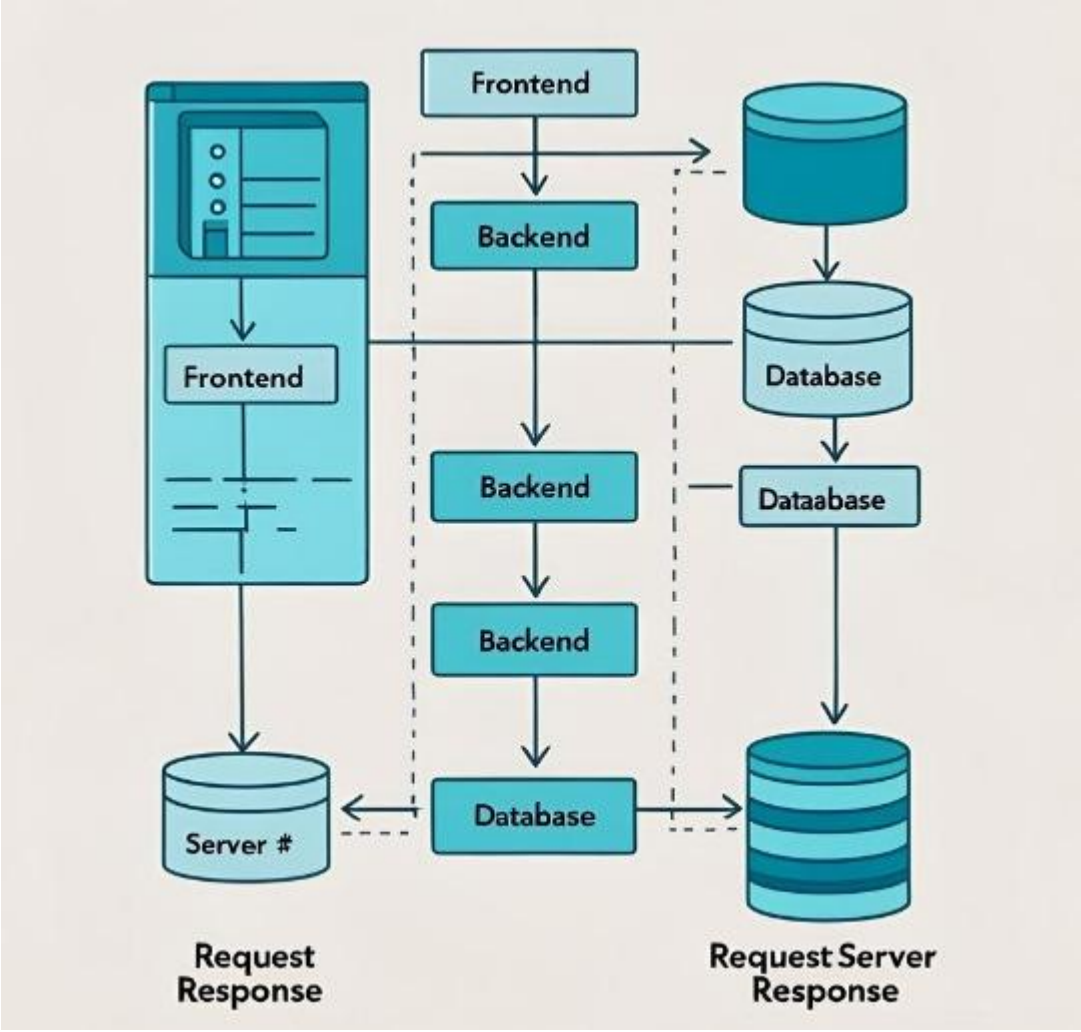
  Supports efficient data retrieval, searching, and filtering.



CLIENT SERVER INTERACTION

❖ Client-Server Interaction

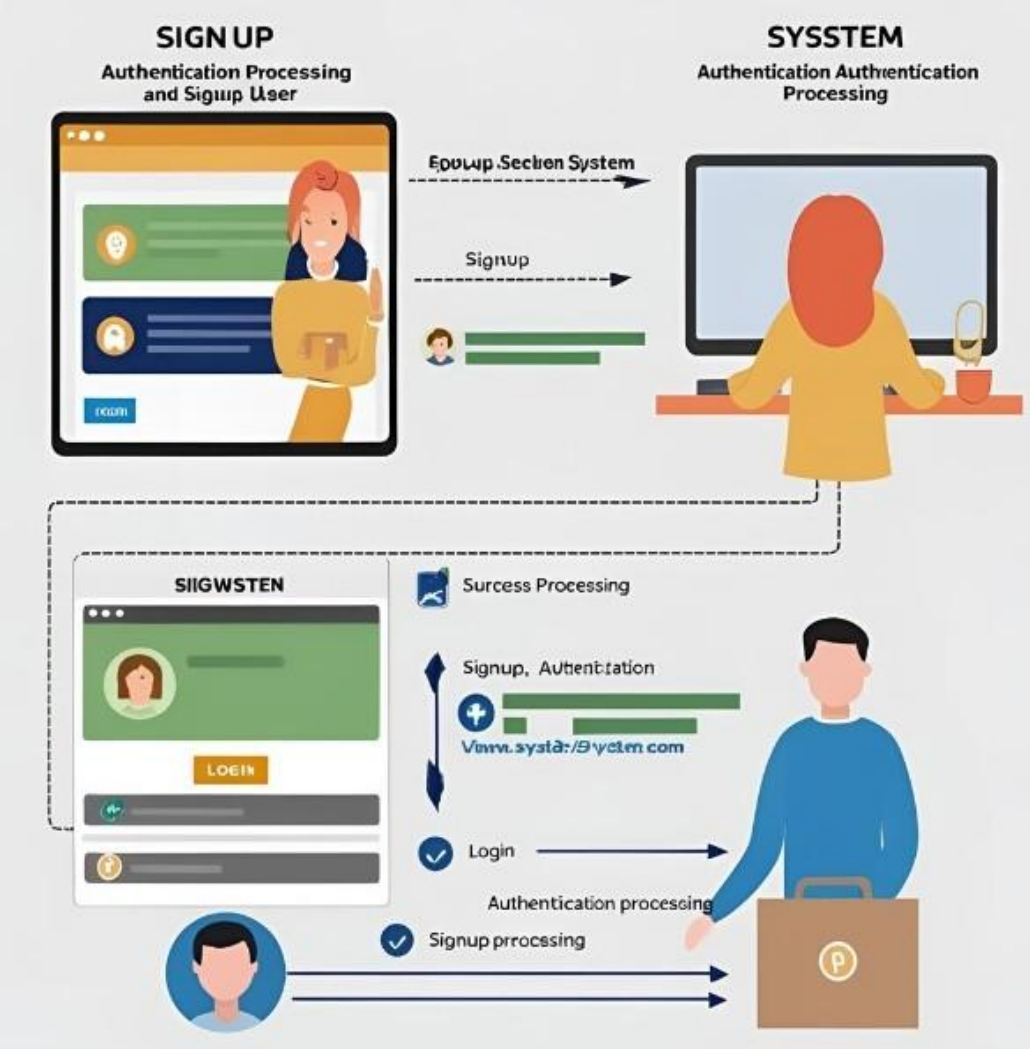
- **User Actions (Client Request):** The user interacts with the app (e.g., adding a contact) via the React.js frontend.
- **Request Processing (Server Side):** The request is sent to the Spring Boot backend, where authentication and business logic are applied.
- **Database Query:** The backend communicates with MySQL/PostgreSQL to fetch or update contact details.
- **Response to Client**: The processed data is sent back to the frontend for display**.**



USER AUTHENTICATION MODULE

❖ Signup and Login

- Allows users to create an account and log in securely.
- Ensures only authorized users can access and manage contacts.
- Uses JWT or session-based authentication for security.



CONTACT MANAGEMENT MODULE

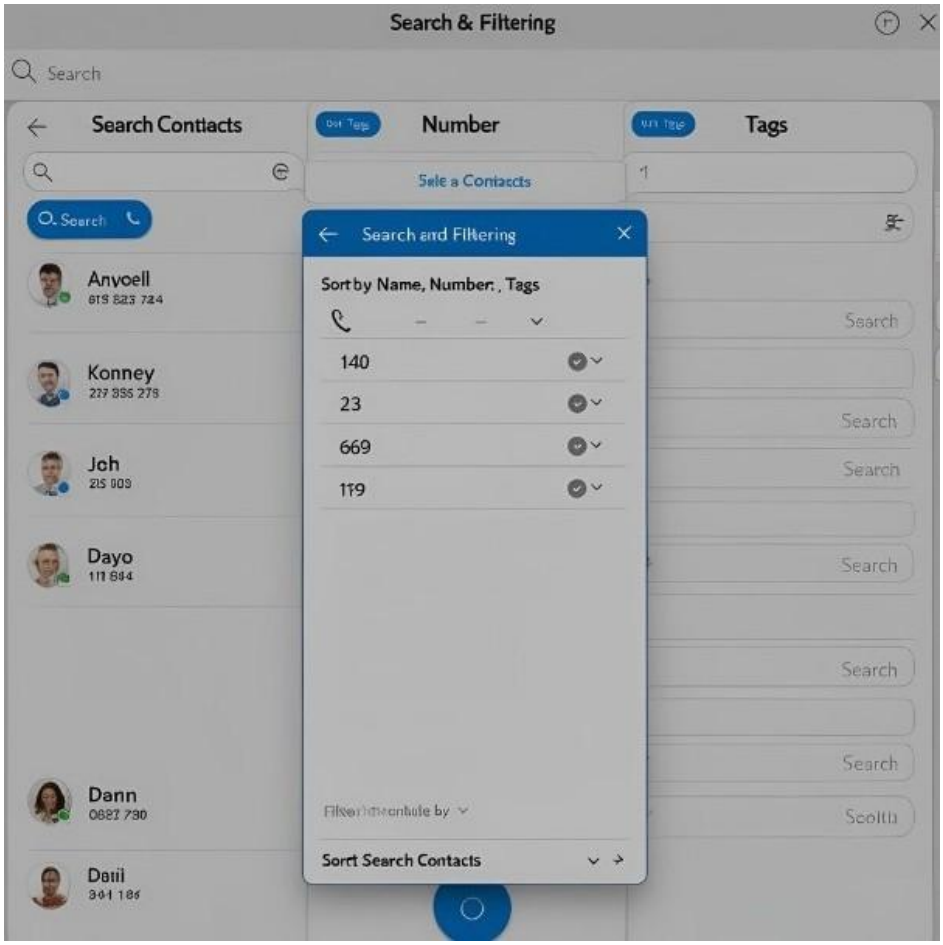
❖ Add, Edit, Delete, Search Contacts

- Enables users to add new contacts with details like name, phone number, and email.
- Users can edit, update, or delete contacts as needed.
- Supports real-time updates for a smooth experience.

SEARCH AND FILTERING MODULE

❖ Sort by Name, Number, Tags

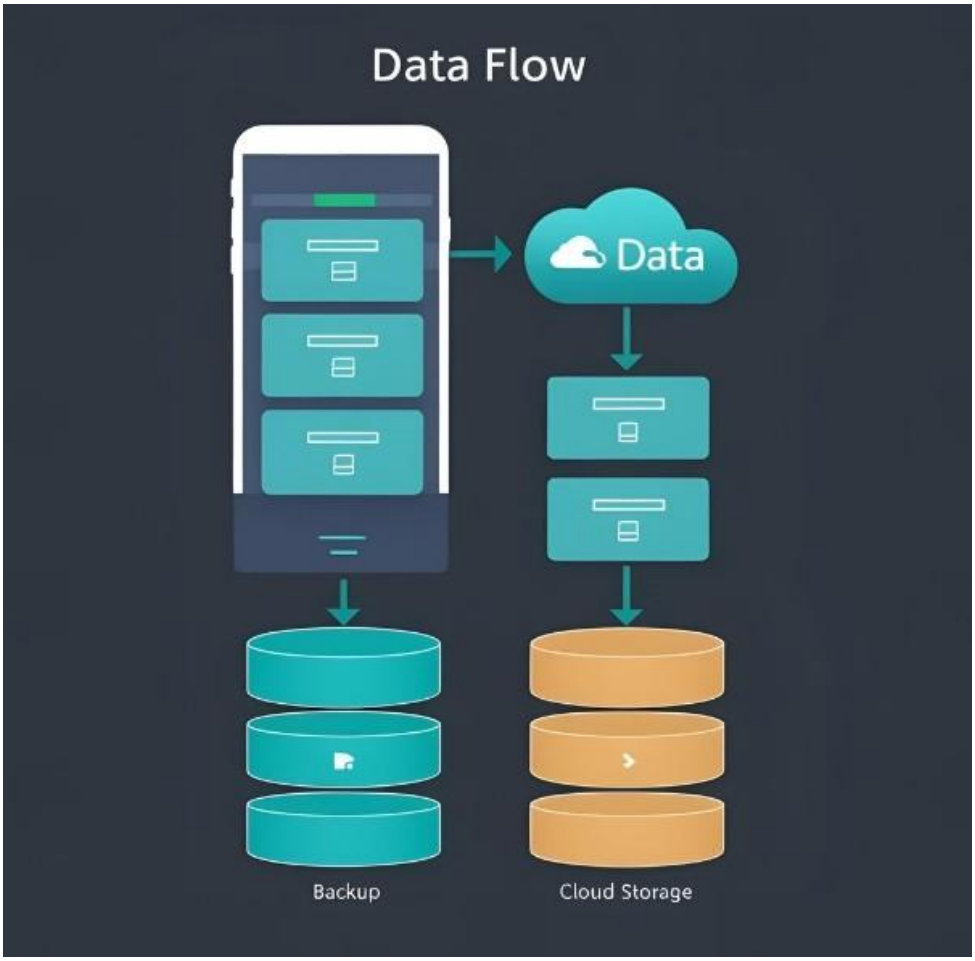
- Users can quickly search contacts by name, phone number, or tags.
- Provides sorting and filtering options to organize contacts better.
- Helps in easy and quick retrieval of specific contact.



BACKUP & RESTORE MODULE

❖ Cloud Storage Integration

- Ensures automatic or manual backup of contacts.
- Allows users to restore lost contacts in case of accidental deletion.
- Can integrate Google Drive or Firebase for cloud storage.



DATABASE DESIGN

❖ Tables:

- Users Table (Stores user account details):

  user_id (Primary Key)

  username

  email

  password

  created_at

- Contacts Table (Stores contact details for each user) :

  contact_id (Primary Key)

  user_id (Foreign Key → Users)

  name

  phone_number

  email

  category_id (Foreign Key → Categories)

  created_at

- Categories Table (Stores different contact categories) :

  category_id (Primary Key)

  category_name

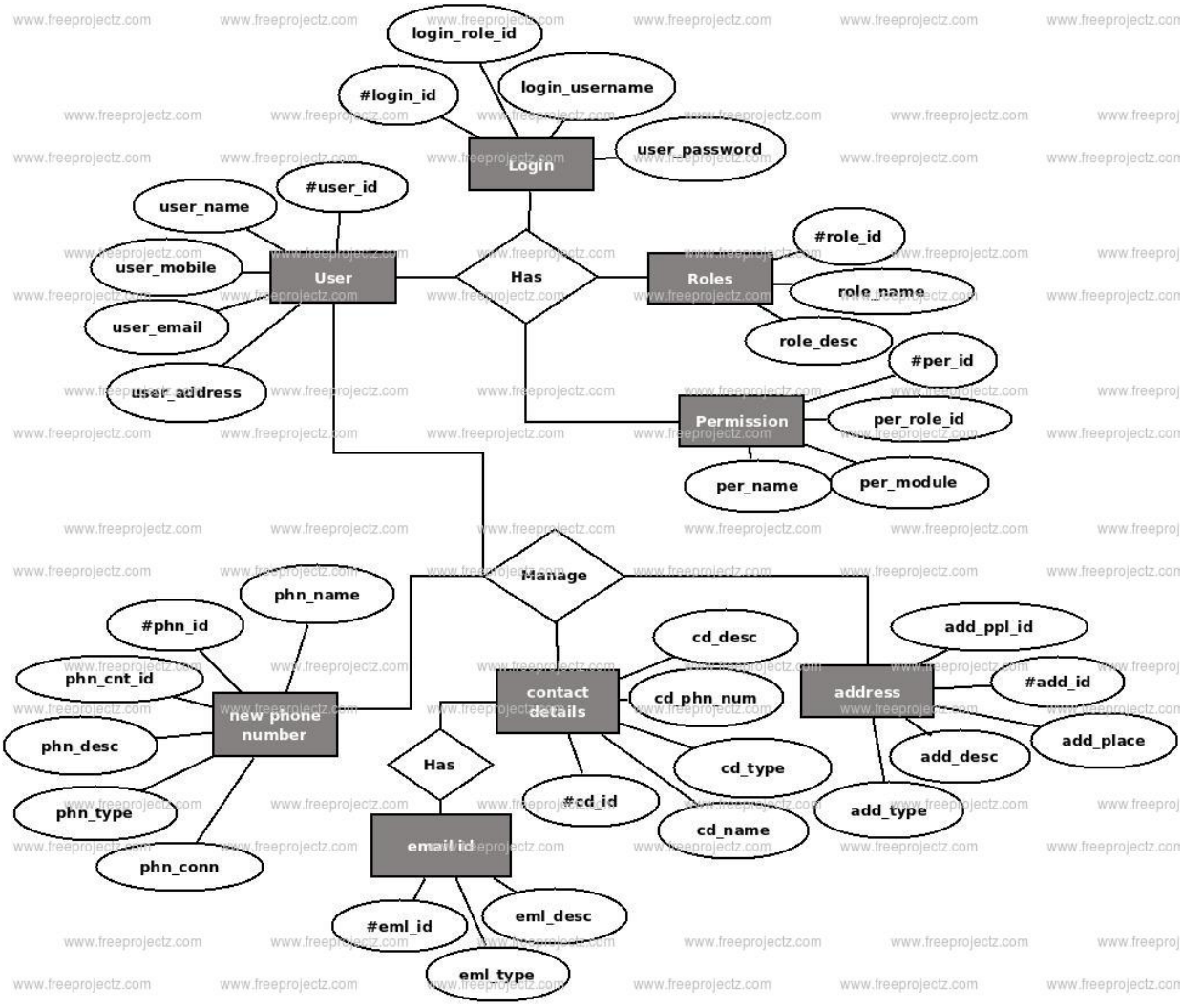
❖ Relationships Between Tables:

- Users → Contacts → (One-to-Many) → A user can have multiple contacts.
- Contacts → Categories → (Many-to-One) → A contact belongs to one category, but a category can have multiple contacts.

Entity-Relationship (ER) Diagram & Data Flow Diagram (DFD)

❖ ER Diagram

- Represents how data is structured in the system.
- Users can have multiple Contacts, and each contact belongs to a Category.
- Include ER Diagram: Shows relationships between Users, Contacts, and Categories.

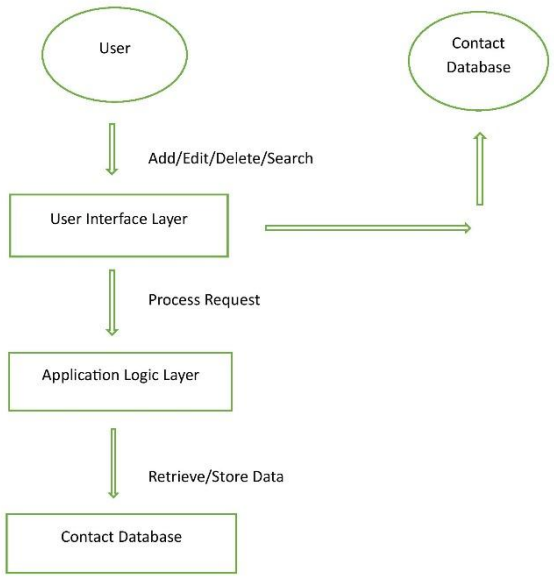


ER Diagram For Telephone Directory Management System

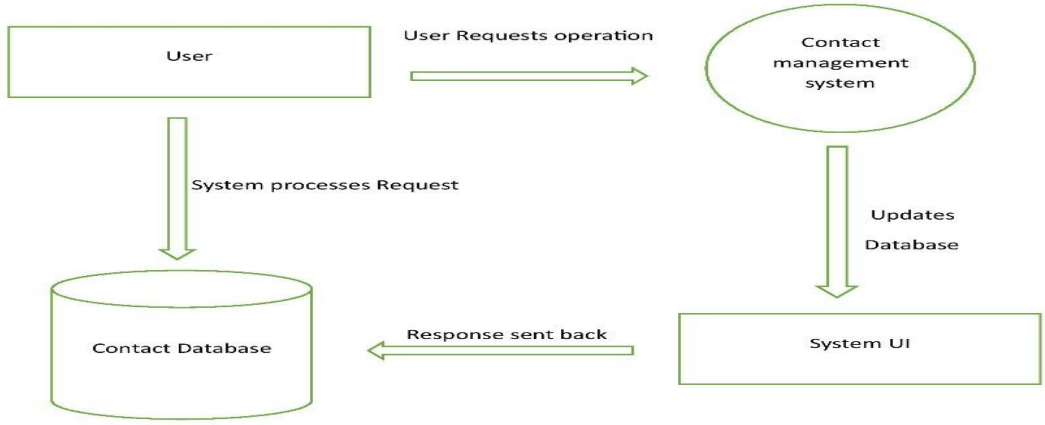
❖ DFD (Data Flow Diagram):

- Shows how data moves within the system.
- Level 0: User interacts with the system for login, contact management, and search.
- Level 1: Detailed flow of adding, searching, and backing up contacts.

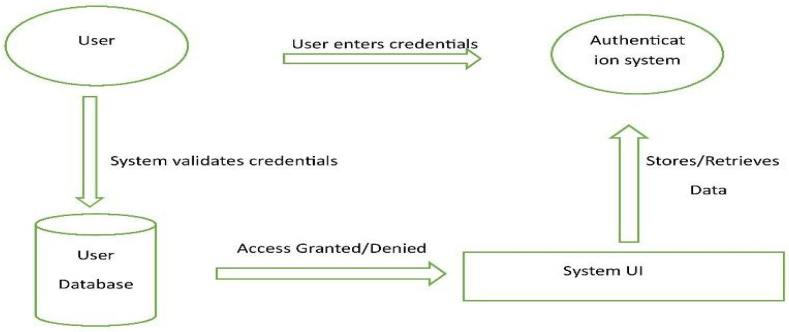
Data Flow Diagram (DFD)



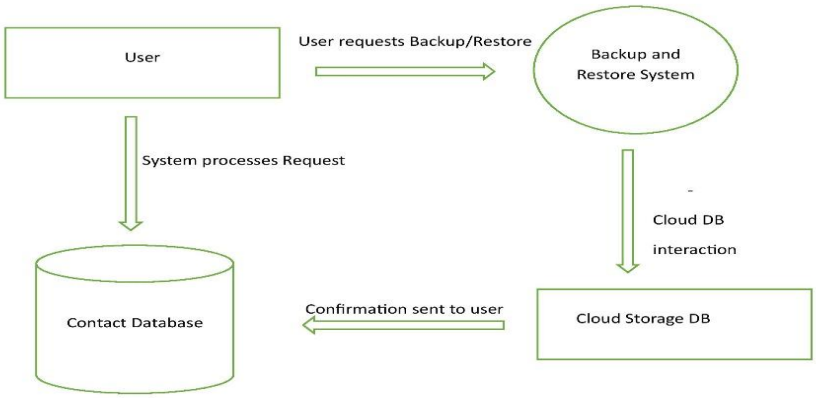
DFD of Contact Management Module.



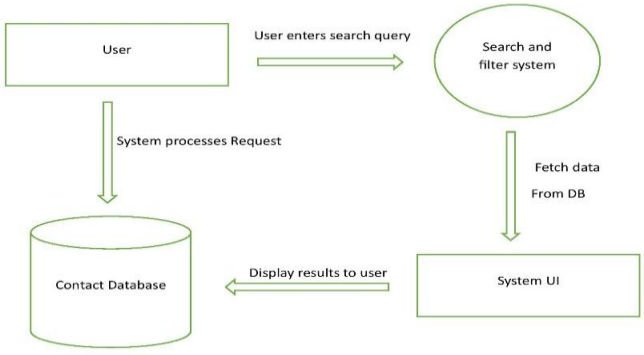
User Authentication Module DFD



DFD of Backup & Restore Module(Cloud Storage Integration):



DFD of Search and Filtering Module:



IMPLEMENTATION PLAN

- **Requirement Analysis**: Define features (Login, Contact Management, Search, Backup) and choose technologies (React.js, Spring Boot, MySQL).
- **System Design:** Create ER Diagram, DFD, and UI wireframes.

❖ Development

  - **Frontend:** Build React.js interface.
  - **Backend:** Develop APIs in Spring Boot.
  - **Database:** Set up MySQL/PostgreSQL.
  - **Testing & Integration:** Connect frontend and backend, perform unit & security testing.
  - **Deployment & Maintenance:** Host on GitHub, AWS, or Firebase and ensure updates.

❖ Testing and Validation

- **Unit Testing:** Test individual components like login, contact addition, search, and backup.
- **Integration Testing:** Ensure smooth interaction between frontend, backend, and database.
- **Functional Testing:** Verify that all features (CRUD operations, search, authentication) work correctly.
- **Security Testing:** Check data encryption, authentication, and unauthorized access prevention.
- **Performance Testing:** Ensure fast response times for contact retrieval and management.
- **User Acceptance Testing (UAT):** Get feedback from users to refine the system.

SECURITY MEASURES

❖ **Authentication & Authorization**

- JWT (JSON Web Token): Ensures secure user login by generating unique tokens.
- OAuth: Enables third-party authentication (e.g., Google login).
- **Data Encryption:**
  - Encrypt contact details before storing them in the database.
  - Use AES encryption to protect sensitive data.

FUTURE ENHANCEMENTS

- **Voice Search:** Allow users to search contacts using voice commands for faster access.
- **AI-Based Contact Suggestions:** Use AI to suggest frequently contacted or important contacts based on usage patterns.
- **Cloud Sync:** Enable automatic synchronization of contacts across multiple devices using cloud storage.

EXPECTED OUTCOME

- **Better Contact Management:** Users can efficiently store, search, and categorize contacts.
- **Enhanced Accessibility:** Contacts are available anytime, anywhere with cloud sync.
- **Improved Security:** Authentication and encryption ensure data privacy.
- **User-Friendly Experience:** Simple UI with advanced features like AI suggestions and voice search

CONCLUSION

- The Phonebook Directory App provides a secure, scalable, and user-friendly solution for managing contacts. With features like search, backup, cloud sync, and AI-based suggestions, it enhances user experience and efficiency. Future enhancements like voice search and advanced AI features will further improve functionality.

BIBLOGRAPGY

- ***"Spring Boot in Action" – Craig Walls***
- ***"React.js Essentials" – Artemij Fedosejev***
- ***"Database System Concepts" – Abraham Silberschatz***
- ***Spring Boot Official Documentation***
- ***React.js Official Docs***
- ***MDN Web Docs***