**HIGH LEVEL DESIGN DOCUMENT**

**PROFESSIONAL SOCIAL NETWORK**

**Document Version:** 1.0

**Date:** 02/03/2024

**Prepared By:** Vajjala Srinandhini

Change Record

|  |  |  |  |
| --- | --- | --- | --- |
| **Revision** | **Date** | **Author** | **Changes** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**TABLE OF CONTENTS**

1. Introduction

* Purpose of the Document
* Scope of the Document
* Overview of the Social Media Platform
* Key Objectives and Goals of the System

1. Requirements and Goals

* Functional Requirements
* User Authentication and Authorization
* User Profile Management
* Content Creation and Publishing
* Social Interactions
* Messaging and Notifications
* Search and Discovery
* Non-functional Requirements
* Performance
* Scalability
* Security
* Availability
* Reliability
* Usability

1. Assumptions and Prerequisites

* Technology Stack
* Infrastructure Requirements
* Data Privacy and Compliance

1. Business Overview

* Business Goals and Objectives
* Target Audience
* Competitive Analysis

1. High-level Design

* Architectural Overview
* System Components
* Communication Protocols
* Deployment Architecture
* Data Model
* Entity Relationship Diagram (ERD)
* Data Storage Strategy (Database Schema)
* Application Design
* User Interface Design
* Client-Server Interaction
* Server Side Logic
* Caching Strategy
* Integration Points
* Third-Party Integrations
* API Design
* Security Design
* Authentication and Authorization Mechanisms
* Data Encryption
* Threat Modelling
* Performance Considerations
* Load Balancing
* Caching Strategies
* Content Delivery Network (CDN) Integration

1. Application Modules

* User Management Module
* Content Management Module
* Social Interaction Module
* Messaging Module
* Search and Discovery Module

1. Transactions and User Flows

* User Registration and Authentication Flow
* Profile Setup Flow
* Content Creation Flow
* Social Interaction Flow
* Messaging Flow
* Search and Discovery Flow

1. Future Enhancements

* Feature Roadmap
* Technology Upgrades
* Scalability Improvements

1. Conclusion

* Summary of Key Points
* Next Steps

**INTRODUCTION:**

* **Purpose of the document:**

This High-Level Design (HLD) document is created to provide a structured plan and overview of the architecture, features, and key components of a Professional Social Network platform. The HLD document plays a crucial role in guiding the development process, aligning stakeholders, mitigating risks, ensuring compliance and security, and ultimately, delivering a successful Professional Social Network platform that meets the needs and expectations of its users.

* **Scope of the document:**

The scope of this document encompasses the architectural overview, key features, functionalities, security measures, scalability considerations, and deployment strategies of the professional social network application. It aims to provide a broad understanding of the system's design and objectives without delving into specific implementation details.

* **Overview of the platform**:

The professional social network application is designed to connect professionals from diverse industries, facilitating networking, knowledge sharing, and career development opportunities. It provides a platform for users to create and manage their professional profiles, connect with peers and industry experts, share insights and expertise, access career resources, and engage in discussions and collaborations relevant to their fields.

* **Key Objectives and Goals of System:**

The main objectives and goals of Professional Social Network platform are

* Building Network: Enable professionals to expand their professional networks, connect with peers, mentors, and industry leaders.
* Promoting Knowledge Sharing: Facilitate the exchange of insights, expertise, and industry-related content to support continuous learning and professional growth.
* Supporting Career Advancement: Provide tools, resources, and opportunities for users to explore job openings, enhance their skills, and advance their careers.
* Ensuring User Privacy and Security: Implement robust authentication, authorization, and data protection mechanisms to safeguard user information and interactions, maintaining user trust and confidentiality.

**REQUIREMENTS AND GOALS:**

* **Functional Requirements**
* User Authentication and Authorisation:
* Users should be able to create accounts and log in securely.
* Different levels of access (e.g., user, admin) should be implemented.
* Authentication mechanisms like email verification and password encryption should be in place.
* User Profile Management:
* Users should have the ability to create and manage their profiles.
* Profile information should include personal details, work experience, skills, education, etc.
* Users should be able to customize the privacy settings for their profiles.
* Content Creation and Publishing:
* Users should be able to create various types of content such as posts, articles, and multimedia.
* Content should support rich formatting options.
* Users should have the ability to publish and share their content with specific audiences or publicly.
* Social Interactions:
* Users should be able to connect with other users.
* Features such as following, friending, and networking should be available.
* Users should be able to like, comment, and share content created by others.
* Messaging and Notifications:
* Users should have the ability to send private messages to each other.
* Notifications should be sent for actions like new connections, comments, mentions, etc.
* Notification preferences should be customizable for each user.
* Search and Discovery:
* Users should be able to search for other users, content, and groups.
* Advanced search filters should be available, such as by location, industry, or skill.
* Recommendations for people to connect with or content to follow should be provided.
* **Non-Functional Requirements**
* Performance:
* The system should respond quickly to user actions.
* Page load times should be minimized.
* Response times for requests should be within acceptable limits.
* Scalability:
* The system should be able to handle a growing number of users and content.
* Horizontal scaling should be supported to distribute the load across multiple servers.
* Security:
* User data should be encrypted both in transit and at rest.
* Measures should be in place to prevent unauthorized access to user accounts and data.
* Security best practices should be followed for authentication, authorization, and data protection.
* Availability:
* The system should be available and accessible to users at all times.
* Redundancy and failover mechanisms should be in place to minimize downtime.
* Reliability:
* The system should operate reliably without frequent failures or errors.
* Regular backups of data should be performed to prevent data loss.
* Usability:
* The user interface should be intuitive and easy to navigate.
* Features should be clearly labelled and easily accessible.
* Accessibility features should be implemented to accommodate users with disabilities.

**ASSUMPTIONS AND PREREQUISITES**

* **Technology Stack:**

1. Backend Development:
   * Java: Core programming language for backend development.
   * Spring Boot: Framework for building Java-based web applications, providing features like dependency injection, MVC architecture, and auto-configuration.
   * Node.js: Optionally used alongside Java/Spring Boot for specific functionalities or microservices, leveraging its event-driven, non-blocking nature.
2. Frontend Development:

* AngularJS: Frontend JavaScript framework for building dynamic and interactive user interfaces.
* HTML5/CSS3: Markup and styling languages for structuring and styling the frontend user interface.

1. Testing and Quality Assurance:
   * JUnit/TestNG: Java testing frameworks for unit and integration testing of backend services.
   * Selenium/WebDriver: Automated testing tools for web applications, used for end-to-end testing of frontend components.
   * Postman/SOAPUI: Tools for API testing and automation, facilitating testing of RESTful or SOAP APIs.
2. DevOps and Deployment:
   * Docker: Containerization platform for packaging, distributing, and running applications in isolated environments.
   * Jenkins: Continuous Integration and Continuous Deployment (CI/CD) tool for automating the build, test, and deployment processes.
   * AWS (Amazon Web Services): Cloud computing platform for hosting, scaling, and managing the application infrastructure.
3. Database:
   * MongoDB: NoSQL database for storing unstructured or semi-structured data, offering flexibility and scalability for certain data storage needs.
   * MySQL: Relational database management system for storing user data, profiles, posts, and other structured data.
4. Design and Prototyping:
   * Figma/Canva: Design and prototyping tools for creating wireframes, mockups, and user interface designs.

* **Infrastructure Requirements:**

1. Web Servers:
   * Deploy web servers (e.g., Apache HTTP Server, Nginx) to host the frontend components of the application.
   * Configure web servers to handle user requests, render web pages, and respond to user interactions.
   * Ensure high availability and scalability by deploying multiple instances behind a load balancer.
2. Application Servers:
   * Set up application servers (e.g., Tomcat, Jetty) to host the backend business logic of the application.
   * Deploy microservices architecture to encapsulate and manage individual functionalities.
   * Utilize containers (e.g., Docker) for deploying and managing microservices independently.
   * Implement an API Gateway to manage API traffic, enforce security policies, and facilitate API versioning.
3. Databases:
   * Provision both relational (MySQL) and NoSQL (MongoDB) databases to store user data, posts, comments, and connections.
   * Utilize database clusters or replicas to ensure data availability, durability, and scalability.
   * Implement database sharding or partitioning strategies to distribute data across multiple nodes for improved performance.
4. Media Storage:
   * Use external storage solutions like Amazon S3 for storing multimedia content uploaded by users.
   * Implement content filtering services (e.g., Amazon Rekognition) to scan and filter user-generated content before storing it in S3.
5. Monitoring and Analysis:
   * Set up monitoring and logging solutions (e.g., AWS CloudWatch, ELK stack) to track system performance, detect anomalies, and troubleshoot issues.
   * Analyze user activity using AWS Kinesis streams to capture and process clickstream data for real-time behavior analysis.
6. Security:
   * Implement identity and access management (IAM) using AWS Cognito or IAM to manage user access and permissions securely.
   * Encrypt data at rest and in transit using AWS Key Management Service (KMS) and secure HTTPS connections with Amazon ACM.
   * Configure AWS Web Application Firewall (WAF) to protect against common web application attacks.
7. Deployment and DevOps Pipeline:
   * Utilize AWS services like EC2, ECS, or Lambda for deploying and managing application instances.
   * Set up a CI/CD pipeline using Jenkins to automate the build, test, and deployment processes.
   * Implement infrastructure as code (IaC) using tools like AWS CloudFormation or Terraform to define and provision AWS resources.

* **Data Privacy and Compliance:**
* Data Encryption:

Sensitive user data, including personal information, passwords, and private messages, must be encrypted both in transit and at rest using strong encryption algorithms. Utilize industry-standard encryption protocols such as AES (Advanced Encryption Standard) with robust key management practices to ensure the confidentiality and integrity of user data.

* Compliance:

The platform must ensure compliance with relevant data privacy regulations such as GDPR, CCPA, or HIPAA, depending on the regions and user demographics targeted. This involves implementing measures to protect user privacy rights, including data protection, user consent, data access controls, and breach notification procedures, in accordance with applicable laws and regulations.

* Data Retention Policies:

Define clear data retention policies specifying how long user data will be stored and when it should be deleted. Ensure that data retention practices align with legal requirements, user consent, and the purpose for which the data was collected. Regularly review and update data retention policies to reflect changes in business needs and regulatory obligations.

* User Consent:

Implement features to obtain explicit user consent for data processing and sharing activities. Users should be informed about the types of data collected, the purposes of data processing, and any third parties with whom their data may be shared. Provide clear options for users to grant or revoke consent and ensure that their preferences are respected throughout their interactions with the platform.

* Privacy Controls:

Provide users with granular privacy controls to manage the visibility of their profiles, content, and interactions. Enable users to customize privacy settings for individual posts, restrict access to specific users or groups, and control the visibility of personal information on their profiles. Empower users to maintain control over their privacy preferences and ensure transparency in how their data is used and shared on the platform.

**BUSINESS OVERVIEW**

* Business Goals and Objectives:

The professional social network platform aims to create a robust online community for professionals across various industries, fostering networking, knowledge sharing, and career development opportunities. Key objectives include:

* Facilitating meaningful connections between professionals to expand their networks.
* Providing a platform for sharing insights, expertise, and industry-related content.
* Supporting career advancement by offering resources for skill development, job opportunities, and mentorship.
* Ensuring compliance with data privacy regulations such as GDPR, CCPA, or HIPAA to protect user privacy and maintain trust.
* Target Audience:

The platform targets professionals from diverse backgrounds, including but not limited to:

* Corporate professionals seeking networking opportunities and career growth.
* Freelancers and independent consultants looking to expand their client base and collaborate with peers.
* Students and recent graduates seeking mentorship, internships, and job opportunities.
* Industry experts and thought leaders interested in sharing knowledge and contributing to professional discussions.
* Competitive Analysis:

The professional social network platform faces competition from existing social networking platforms such as LinkedIn, as well as niche professional communities and forums. To differentiate itself, the platform will focus on:

* Offering granular privacy controls and user consent features to empower users and ensure data privacy compliance.
* Providing a seamless user experience with intuitive navigation, responsive design, and interactive features.
* Curating high-quality content and fostering meaningful interactions to create value for users and encourage engagement.
* Leveraging innovative technologies such as microservices architecture, AI-powered recommendations, and real-time analytics to enhance platform capabilities and user satisfaction.

**HIGH-LEVEL DESIGN**

* **Architectural Overview:**

The professional social network platform will be designed using a microservices architecture to ensure scalability, flexibility, and maintainability. Each microservice will encapsulate specific functionalities, promoting modularity and independent deployment.

* **System Components:**
* User Management Service:
  + Handles user authentication, registration, and profile management.
  + Utilizes a relational database (e.g., MySQL) to store user credentials and profile information.
  + Provides RESTful APIs for user-related operations.
* Content Management Service:
  + Manages creation, publishing, and interaction with user-generated content (posts, comments).
  + Utilizes both relational (e.g., MySQL) and NoSQL (e.g., MongoDB) databases for storing content data.
  + Provides RESTful APIs for CRUD operations on posts and comments.
* Messaging Service:
  + Facilitates real-time messaging and notifications between users.
  + Utilizes WebSocket protocol for bidirectional communication.
  + Stores message data in a relational database (e.g., MySQL).
* Search and Discovery Service:
  + Enables users to search for other users, content, and groups.
  + Utilizes full-text search capabilities of the database or integrates with search engines like Elasticsearch.
  + Provides RESTful APIs for search functionality.
* API Gateway:
  + Acts as a single entry point for client requests, routing them to appropriate microservices.
  + Provides authentication and authorization checks before forwarding requests to backend services.
  + Implements rate limiting and throttling mechanisms to protect against abuse.
* Data Storage:
  + Utilizes a combination of relational (e.g., MySQL) and NoSQL (e.g., MongoDB) databases to store user data, content, and metadata.
  + Ensures data consistency, durability, and scalability based on the nature of the data and access patterns.
* **Communication Protocols:**
* HTTP/HTTPS for client-server communication.
* WebSocket for real-time messaging and notifications.
* **Deployment Architecture:**
* Utilizes containerization with Docker for deploying microservices.
* Orchestrates containers using Kubernetes for automatic scaling, load balancing, and fault tolerance.
* Deploys services across multiple availability zones for high availability and fault tolerance.
* **Data Model:**
* Defines data entities such as users, profiles, posts, comments, messages, and relationships between them.
* Utilizes a combination of relational and NoSQL databases to store structured and unstructured data efficiently.
* **Entity Relationship Diagram (ERD):**
* Represents the relationships between different entities in the data model visually, depicting primary/foreign key associations and cardinality.
* **Data Storage Strategy (Database Schema):**
* Utilizes normalized relational database schemas for structured data and denormalized schemas or document-based storage for unstructured data.
* **Application Design:**
* Follows a modular design approach with separate components for frontend and backend, facilitating easier maintenance and scalability.
* **User Interface Design:**
* Utilizes modern UI/UX design principles to create a user-friendly interface with intuitive navigation, responsive layout, and interactive features.
* **Client-Server Interaction:**
* Client-side interactions with the server are facilitated through RESTful APIs or GraphQL, enabling seamless communication and data exchange.
* **Server Side Logic:**
* Implements business logic within individual microservices, ensuring separation of concerns and scalability.
* **Caching Strategy:**
* Utilizes in-memory caching (e.g., Redis) to cache frequently accessed data and improve performance.
* **Integration Points:**
* Integrates with third-party services for functionalities such as authentication (OAuth), messaging (Twilio), and analytics (Google Analytics).
* **Third-Party Integrations:**
* Includes integrations with popular services like LinkedIn, Google, and GitHub for seamless user authentication and content sharing.
* **API Design:**
* Follows RESTful API design principles with clear endpoints, resource representations, and HTTP methods for efficient communication between client and server.
* **Security Design:**
* Implements robust security measures including authentication, authorization, data encryption, and threat modelling to protect user data and ensure platform security.
* **Authentication and Authorization Mechanisms:**
* Utilizes OAuth 2.0 for authentication and role-based access control (RBAC) for authorization, ensuring secure access to resources based on user roles and permissions.
* **Threat Modelling:**
* Identifies potential security threats and vulnerabilities, implementing countermeasures to mitigate risks and safeguard the platform against attacks.
* **Performance Considerations:**
* Optimizes performance through efficient database queries, caching strategies, and horizontal scaling to handle increasing user loads.

**Load Balancing:**

* Employs load balancers to distribute incoming traffic across multiple instances or microservices, ensuring high availability and reliability.

**Content Delivery Network (CDN) Integration:**

* Integrates with CDNs to cache and deliver static assets (e.g., images, videos) closer to users, reducing latency and improving content delivery speed.

**APPLICATION MODULES**

### User Management Module:

Responsible for user authentication, registration, profile management, and account settings. This module handles user-related functionalities such as user signup, login, profile updates, password resets, and account deletion.

### Content Management Module:

Manages the creation, publishing, and interaction with user-generated content. This module allows users to create, edit, delete, and share various types of content, such as posts, articles, images, videos, and documents. It also includes features for content moderation, reporting, and analytics.

### Social Interaction Module:

Facilitates social interactions between users, including likes, comments, shares, and follows. This module enables users to engage with each other's content, build connections, and foster communities within the platform. It includes features for notifications, activity feeds, trending topics, and user engagement metrics.

### Messaging Module:

Enables real-time messaging and communication between users. This module includes features for one-on-one messaging, group chats, multimedia sharing, and message delivery status. It ensures seamless communication and collaboration among users, fostering meaningful interactions and relationships.

### Search and Discovery Module:

Provides search and discovery functionalities to help users find relevant content, people, and communities within the platform. This module includes features for keyword search, advanced filtering, personalized recommendations, trending topics, and content discovery algorithms. It enhances user experience by facilitating content exploration and discovery based on user interests and preferences.

**TRANSACTIONS AND USER FLOWS**

* **User Registration and Authentication Flow:**

1. **Registration:** User enters required details and submits registration form.
2. **Email Verification:** System sends verification email with unique token.
3. **Verification:** User clicks verification link and verifies email.
4. **Login:** User enters credentials and submits login form.
5. **Authentication:** System validates credentials and generates authentication token.
6. **Access Granted:** User gains access to platform features upon successful authentication.

* **Profile Setup Flow:**

1. **Profile Creation:** User navigates to profile settings and fills out profile information.
2. **Profile Picture:** User uploads profile picture or selects avatar.
3. **Bio and Details:** User adds bio, location, job title, skills, and other optional details.
4. **Save Profile:** User saves changes and completes profile setup.

* **Content Creation Flow:**

1. **Post Creation:** User navigates to content creation interface.
2. **Compose Post:** User writes or uploads content (text, image, video, etc.).
3. **Add Details:** User adds title, tags, and other metadata.
4. **Publish:** User submits post for publishing.
5. **Post Published:** System publishes post and makes it visible to other users.

* **Social Interaction Flow:**

1. **Like/Comment:** User views content and interacts with it by liking or commenting.
2. **Notification:** Author receives notification about likes and comments on their post.
3. **Engagement:** Author responds to comments and engages with users.

* **Messaging Flow:**

1. **Initiate Conversation:** User selects recipient and initiates new conversation.
2. **Compose Message:** User writes message and attaches any files or media.
3. **Send Message:** User sends message to recipient.
4. **Receive Message:** Recipient receives message and can respond accordingly.

* **Search and Discovery Flow:**

1. **Keyword Search:** User enters keywords in search bar.
2. **Filtering:** User applies filters such as category, date, location, etc.
3. **Browse Results:** User views search results and explores content.
4. **Engagement:** User interacts with content, profiles, or communities discovered through search.

**FUTURE ENHANCEMENTS**

* **Feature Roadmap:**
* **Advanced Analytics:** Implement advanced analytics to provide insights into user behavior, content engagement, and platform performance.
* **Enhanced Personalization:** Develop personalized recommendations and content curation algorithms based on user preferences, behavior, and interests.
* **Community Building Tools:** Introduce features for creating and managing user communities, including group discussions, events, and collaborative projects.
* **Monetization Options:** Explore monetization strategies such as premium memberships, sponsored content, and targeted advertising.
* **Integration with External Services:** Integrate with external services and APIs to offer additional functionalities such as job listings, professional development courses, and industry-specific tools.
* **Technology Upgrades:**
* **Adoption of Latest Frameworks:** Evaluate and adopt newer frameworks and libraries to improve development efficiency, performance, and maintainability.
* **Migration to Serverless Architecture:** Explore the possibility of migrating certain services to serverless architecture for cost optimization and scalability.
* **Upgrade to Latest Database Technologies:** Consider migrating to newer database technologies or implementing additional data storage solutions to enhance scalability, reliability, and performance.
* **Scalability Improvements:**
* **Horizontal Scaling:** Implement horizontal scaling techniques such as container orchestration (e.g., Kubernetes) to handle increased user traffic and workload.
* **Optimized Resource Allocation:** Fine-tune resource allocation and utilization to optimize performance and minimize resource wastage.
* **Load Testing and Performance Tuning:** Conduct regular load testing and performance tuning exercises to identify bottlenecks and optimize system performance.

**CONCLUSION**

* **Summary of Key Points:**
* The professional social network platform is designed to provide users with a comprehensive suite of features for networking, content sharing, and communication.
* Key modules include user management, content management, social interaction, messaging, and search and discovery.
* The platform follows a microservices architecture, ensuring scalability, modularity, and maintainability.
* Data privacy and security are paramount, with encryption, authentication, and authorization mechanisms in place to protect user data.
* Future enhancements include advanced analytics, enhanced personalization, community building tools, and technology upgrades for improved scalability and performance.
* **Next Steps:**

1. **Implementation:** Begin development based on the outlined high-level design and user stories.
2. **Testing:** Conduct thorough testing, including unit testing, integration testing, and user acceptance testing.
3. **Deployment:** Deploy the platform to production environment, ensuring scalability, reliability, and security.
4. **Monitoring and Maintenance:** Continuously monitor system performance, user feedback, and market trends. Implement regular updates and improvements to address issues and incorporate new features.
5. **Growth and Expansion:** Explore opportunities for growth and expansion, such as partnerships, marketing initiatives, and additional features to attract and retain users.