Software Architecture (SA)
for
‘Connecta’
An online social networking system
Group 4

ANISH MANANDHAR (CS23BTKMU11001)
ANKUSH NIROULA (CS23BTKMU11002)
VIGNAN KOTA (CS21BTECH11029)
DAVID MALOTH (CS21BTECH11035)

1. Overview

1.1 System Overview

Connecta has both frontend and backend elements. Users engage with the platform via web browsers, desktop apps, or mobile apps on the front end. The user interface (UI) of the frontend oversees displaying elements like news feeds, profiles, chat, and notifications. Additionally, the architecture of client-side logic and the layered architecture have been proposed to manage user interactions with the cache data or the data repository.

1.2 System Context

The system context is defined clearly in the SRS. The relationship between users and posts defines the system context in the design of social media platforms.

1.3 Stakeholders of Connecta

New users and registered users are the general stakeholders of Connecta. Most users that access the content and services provided by Connecta are general users. Users interact with several features include sharing media, liking, commenting, as well as seeing feeds and updating their profiles.

Concerns of various stakeholders:

- Privacy: There can be concern about the security and privacy of personal information. Data security: People want to know that their personal data is safe from misuse or illegal access.
- Content Moderation: People rely on social media to efficiently control content to stop hateful, offensive or dangerous content from spreading.
- User Experience: Users appreciate a seamless, easy-to-use interface with quick loading times, straightforward navigation, and responsive design.
- Security: User will want to know that their personal data is safe from misuse or illegal access.

1.4. Scope of this Document

Connecta is a general social media platform that aims to provide a different experience to social networking itself. The system aims to have thousands of users in its space who can share information with their circle. In this document, two architectures are discussed that meets the aim of Connecta, and then choose the best alternative which we plan to extend to this project throughout. For both, Component and Connector views are prioritized in making this software architecture document.

2. Architecture Design

2.1 Architecture 1: Shared Data Style Architecture with Cached Repository

The system's components are arranged in a Shared Data Style Architecture for the Component & Connector View to make data sharing and manipulation across numerous related pieces easier. The Browser, acts as the main interface via which users interact with the system, is the center of this architecture. Here node.js is the central hub.

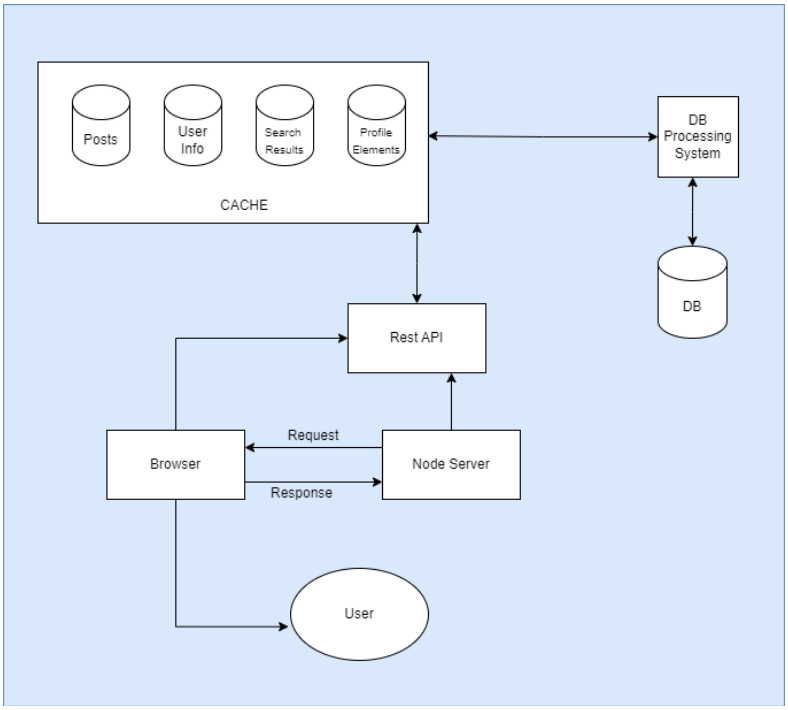

Figure 1: Architecture with Cached Repository

When a user takes an action, such as viewing a webpage or completing a form, the browser sends an HTTP request to the server. After the required data has been obtained, the API will use frontend technologies like HTML, CSS, and JavaScript to create the client-side view and renders the response in accordance with the requested action or query. After rendering, this page is contained in an HTTP response and will be sent back to the browser. The interaction cycle is concluded when the Browser shows the user the content it has received. This design allows for smooth data flow and system interaction because the components (Browser, Node.js server, RestAPI, and database) are connected by a cached data model.

2.2 Architecture 2: Shared Data Style Architecture for Connector & Component View (4-Layered model with blackboard variation)

The framework in figure 2 shows a 4 layered model with shared data style architecture of C&C view. Here, the data repo is active and thus acts as a blackboard variation of shared data architecture. In the topmost level, we have a shared data repo which is also called blackboard. Then the next layer consists of all data accessors that can update or get from the blackboard.

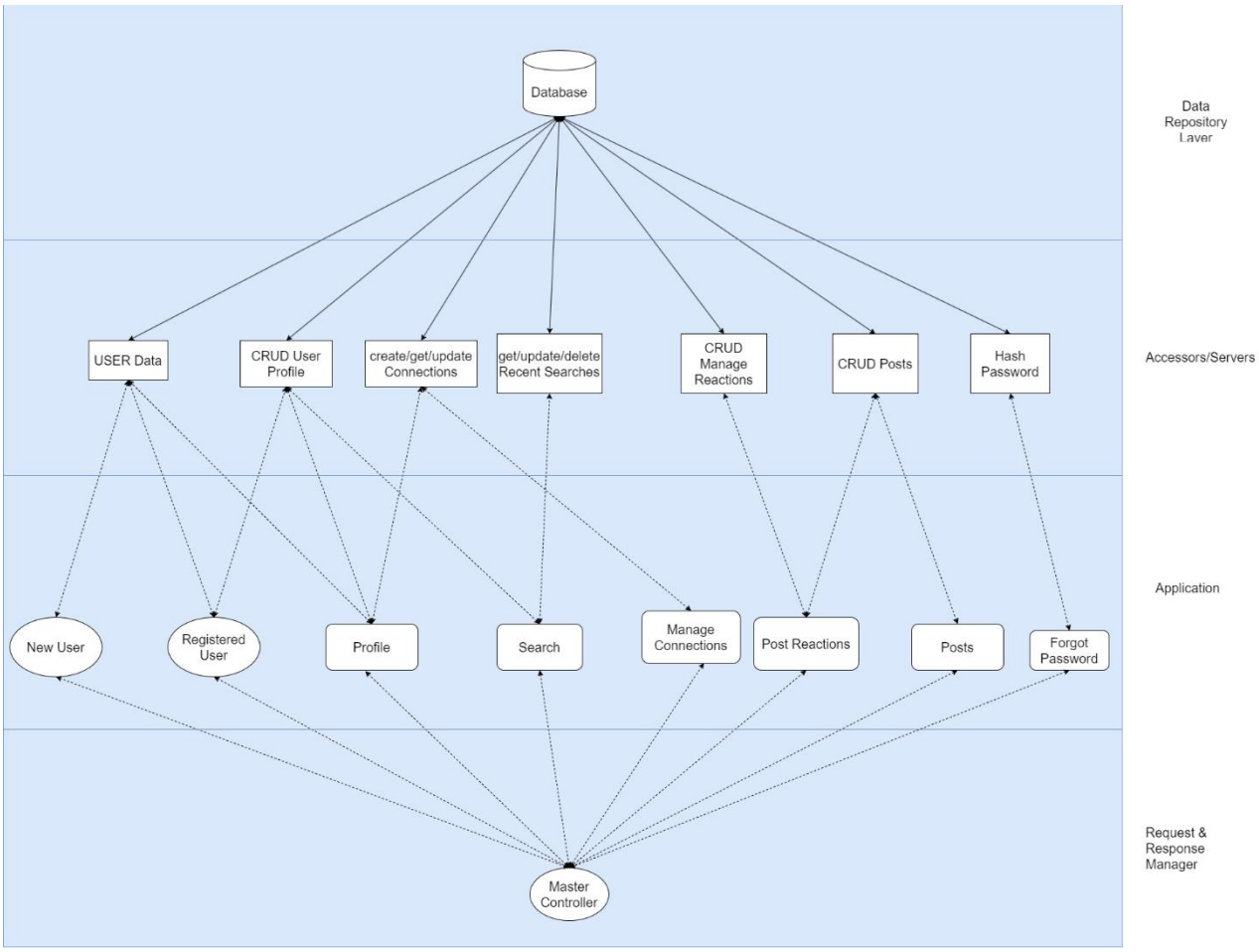

Figure 2: 4 Layered model Blackboard variation

The third layer shows the application component that interacts with the server or data accessors and shows it to the client. Connecta, as a social media platform, can be best shown using this architecture because of its nature of data retrieval and update that happens frequently. The final layer has a master controller which acts as a request and response manager that requests an application service from the user. Each layer has components that perform a particular task. The top and bottom layers are not that detailed due to the changes that could come during design and implementation.

2.3 Comparing the Architectures

The **shared data style** architecture is our intended choice for the Component and Connector (**C&C**) view. The shared data style is better whenever various data items have multiple accessors. For Connecta, we've planned to consider elements like feed, connection requests, and user profiles which are better shown using **blackboard** variation.

The shared-data style is organized around one or more shared-data stores, which store data that other components may read and write. Component types include shared data stores and data accessors. The data-store components of a shared-data system provide shared access to data, support data persistence, manage concurrent access to data, provide fault tolerance, and support access control. This strategy encourages flexibility, scalability, and modularity, making it possible to manage and manipulate data effectively across many layers.

Below are the points that justify the advantages of using this architectural style and variation:

- Modifiability: The shared-data model lets you change individual components by keeping data producers and consumers apart. Changes to one component won't directly impact others.
- Concurrent Access: The shared data store manages concurrent access to data, ensuring consistency and avoiding conflicts.
- Fault Tolerance: Data persistence and fault tolerance are inherent in this style. If one component fails, others can still access the shared data.
- Access Control: You can enforce access control policies at the shared-data store level, securing sensitive information.

Table 1: Comparison of the architectures

Criteria	Architecture 1	Architecture 2
Security	No	Yes, as the password is hashed
Privacy	No	Yes
Debugging	Difficult	Easier
Concurrency	Yes	Yes
Division of Tasks	Relatively difficult	Easier
Switching Database	Relatively easier	Difficult
Creating New Features	Relatively easier	Difficult

2.4 ATAM Analysis

The scenarios of interests are as follows:

1. User Registration and Profile Creation: A new user registers to the Connecta and creates a profile and posts text, images to the profile.
2. Registration Failed due to invalid information.
3. Unauthorized Access.
4. Highly loaded system.

- Security: The password of users should not be leaked.
- Response Time:

Connecta would strive to offer users a prompt and responsive experience to facilitate seamless interactions and engagement.

Response Time	Normal	Heavily Loaded
Server	200ms	400ms
Database	800ms	1500ms
Application	300ms	400ms

- Availability: For Connecta, the optimal availability objective usually surpasses the minimal criterion of 0.80, particularly for the significant levels of user engagement. Social media platforms aim for high availability to guarantee continuous service access because they are essential tools for users' networking and communication.
    - Probability that both server and database are up for the first architecture = 0.8 * 0.8 = 0.64
    - Probability that server and database are up for the second architecture

      Extra Availability = 0.8 * 1 * 0.2 = 0.16, Total Availability = 0.64 + 0.16 = 0.80

    - For architecture 1
        - Normal = 200 + 1500 + 300 = 2000ms
        - Data repository down = 400 + 1500 + 400 = 2300ms
    - For architecture 2
        - Normal = 200 + 800 + 400 = 1600 ms
        - Data repository down = N/A
- Data Currency: The posts shown to the users should be relevant and accessible.

2.5 Evaluation Summary

The summary of the evaluation is summarized in table 2.

Table 2: Evaluation Summary

	Architecture 1	**Architecture 2**
Security(S1)	Yes	Yes
Security(S3)	No	Yes
Response Time (S1)	2000	1600
Response Time (S3)	2300	N/A
Response Time (S4)	4000	3200
Availability (S1,S2)	0.64	0.80
Data Currency (S1)	Yes	Yes
Data Currency (S2)	No	No

3. Use Case of Proposed Architecture

The use case of the proposed architectural design which shows the view is as follow:

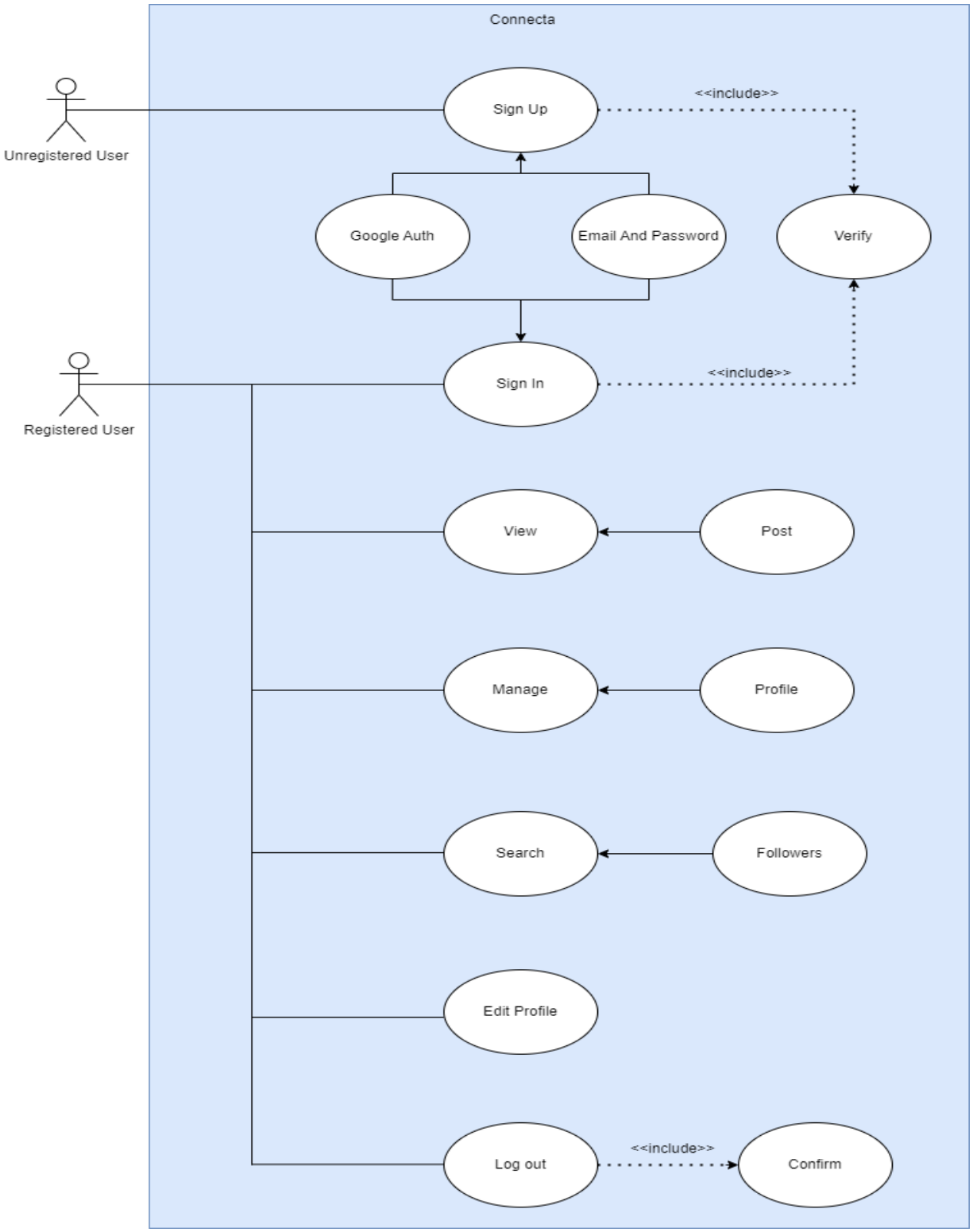

Figure 3: Use Case Diagram of Proposed Architecture

Users can be classified as either registered or unregistered. Only the user who has registered can access the application's different parts. For the unregistered user to access the application, they must first create an account. They can register using an email address and password or with Google Auth. The registered user can log in with their email address and password or with Google Auth. The registrant has access to view post, manage profile, search connections, edit profile and log out.
Users can use a range of services designed to improve their Connecta’s experience once they've enrolled. This involves having the capacity to see and engage with posts made by both themselves and other users, including sharing, liking, and commenting on information. Furthermore, registered users have the ability to make and manage their own posts. These posts can include text, photographs, videos, or links.

4. Non-functional Attributes:

- Security: The architecture shows measures implemented to protect users' personal information, prevent unauthorized access, and ensure the safety of interactions on the platform. Security breaches can have serious consequences, including data theft, identity theft, and privacy violations, so robust security measures are developed for Connecta in the architecture.

- Privacy: The extent to which users are able to manage who may see their interactions, content, and personal data on the site is highlighted in the architecture. Users will have sufficient privacy settings on Connecta to safeguard their information and control who can access it. Passwords will be saved in hashed form rather than plaintext. Hashing is a cryptographic technique that turns a password into a fixed-length string of characters, making it computationally infeasible to reverse the process and get the original password and is proposed in the second architecture. The privacy of posts on Connecta will be typically managed through access visibility settings.

- Performance: The Connecta's responsiveness and speed in handling user activities, loading data is essential for keeping users and promoting engagement because slow performance can irritate users and result in a bad user experience so the response time is handled with the mechanism shown in the architecture.

- Reliability: The Connecta will have the capacity to consistently supply its services without any disruptions or outages. Connecta will be dependable in order to continue earning the trust and satisfaction of users, who expect them to be available at all times.