Mr. Ashwini Kumar Saini, 2026, 14:6
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Social Interaction Website

Amisha Negi[1] Ankita Thapliyal[2] Priyanka Sati[3] Mr. Ashwini Kumar Saini[4] Mr. Vivek Kumar Tamta[5] Prof. Dr. H.S Bhadauria[6]

[1,2,3]Bachelor Of Technology In Computer Science & Engineering
G.B Pant Institute of Engineering & Technology, Pauri, Uttarakhand
[4]Assistant Professor Department Of Computer Science & Engineering
[5]Project Coordinator Computer Science & Engineering,
G.B Pant Institute of Engineering & Technology, Pauri, Uttarakhand
G.B Pant Institute of Engineering & Technology, Pauri (Uttarakhand).
[6]Head Computer Science & Engineering,
G.B Pant Institute of Engineering & Technology, Pauri (Uttarakhand).

ABSTRACT- A social network service focuses on the building and verifying of online social networks for communities of people who share interests and activities, or who are interested in exploring the interests and activities of others, and which necessitates the use of software. Most services are primarily web based and provide a collection of various ways for users to interact, such as messaging, queries, advices and file uploading. Alumni connect- the social networking site is an online networking site. Here users can create profiles with photos, lists of personal interests, contact information, and other personal information. Users can communicate with friends and other users through private or public messages and chat feature. To allay concerns about privacy, Alumni Connect enables users to choose their own privacy settings and choose who can see specific parts of their profile. The website will be free to all users. It requires a user's name and profile picture (if applicable) to be accessible by everyone. Social networks connect people at low cost; this can be beneficial for us to connect with the alumni of our college, know how and what they are doing. We can seek help from them and ask for their guidance regarding various fields. We can get referrals or other recruitment news from them. Several websites are beginning to tap into the power of social networking model for social good. Such models may be highly successful for connecting otherwise fragmented industries and small organizations without the resources to reach a broader audience with the interested ad passionate users.

Keywords- Alumni Connect, Social Networking Site, Alumni Network, Online Community, Social Media, Alumni Engagement, Profile Management, User Authentication, Privacy Settings, Information Sharing, Online Communication, Messaging System, Chat Feature, Professional Networking, Career Guidance, Job Opportunities, Recruitment, Employee Referrals, Knowledge Sharing, Networking Platform, Educational Institution, Alumni Association, Student Development, Community Building, Web-Based Application, Information Technology.

I. INTRODUCTION

1. General Introduction

Social networking has become one of the most significant means of communication in the 21st century. It refers to the process of connecting individuals, groups, and organizations through common interests, relationships, activities, or professional associations. Traditionally, social networking took place through face-to-face interactions in workplaces, educational institutions, local communities, and social gatherings. However, with the rapid growth of the Internet and digital technologies, online social networking has become an integral part of everyday life.

A social network can be defined as a structure that represents and measures relationships and interactions among individuals, groups, organizations, and other connected entities. In a social network, individuals or groups are represented as nodes, while the relationships or interactions between them are represented as links or connections. Social network analysis provides both visual and mathematical approaches to understanding relationships,

© 2026 Mr. Ashwini Kumar Saini, This is an Open Access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly credited.

communication patterns, and information flows among connected entities.

Since their introduction, social networking sites (SNSs) such as MySpace, Facebook, Cyworld, and Bebo have attracted millions of users worldwide. These platforms have gradually become an important part of people's daily activities. Today, numerous social networking applications provide a wide range of services, including instant messaging, photo and video sharing, online communities, blogging, and professional networking.

Although most social networking platforms offer similar basic functionalities, their purposes, features, and user communities may differ significantly. Some platforms focus on maintaining existing relationships among friends, relatives, and colleagues, whereas others help people connect with strangers who share similar interests, hobbies, professional goals, or opinions. Many platforms also provide specialized services based on language, geographical location, culture, and other common interests. The integration of mobile applications, multimedia sharing, and real-time communication has further increased the popularity and usefulness of these platforms.

Social networking sites have attracted considerable attention from researchers and scholars across various disciplines. Researchers study these platforms to understand their usage patterns, social impact, communication practices, security challenges, and influence on modern society. Their development has also introduced new opportunities for information sharing, collaboration, education, business promotion, and community building.

With the increasing demand for fast and convenient communication, instant messaging applications have become an essential component of modern social networking systems. These applications allow users to exchange messages and information in real time without geographical limitations. They provide a simple and efficient way to communicate with friends, family members, colleagues, and other registered users.

The proposed project focuses on the development of a secure and user-friendly social networking and instant messaging application. The primary objective of this application is to provide users with a convenient platform for communicating, sharing information, and maintaining social connections through the Internet.

The application provides facilities for user registration and authentication. Every user must register and obtain valid login credentials before accessing the system. After successful login, users can search for other registered members using their email addresses or contact numbers and establish connections with them.

One of the important features of this application is its temporary messaging facility. Users can select a message-disappearance duration ranging from 5 seconds to 10 seconds or choose an infinite duration. Once a message is received, it automatically disappears after the selected time interval, provided that a temporary duration has been selected. This feature is intended to provide greater control over the visibility and retention of private conversations.

The application also supports file sharing, allowing users to exchange documents and other data files. Additionally, users can capture photographs using a connected webcam and share them through the platform. The application incorporates AES (Advanced Encryption Standard) encryption to protect sensitive information and improve the security of data exchanged between users.

The proposed system follows a client-server architecture. The client application runs on the user's computer, while the server application manages communication and data exchange among connected clients. To initiate communication, the client must establish a connection with the server through a network socket. Once connected, users can participate in both private and group conversations.

The system is designed to support multiple users simultaneously and provide an efficient communication environment. It aims to combine the essential features of social networking and instant messaging into a single application. By integrating secure communication, temporary messages, multimedia sharing, and user connectivity, the proposed system seeks to offer a convenient and reliable platform for online interaction.

2. Scope

The scope of the proposed system is to develop an interactive social networking and instant messaging application that enables registered users to communicate, share information, and establish connections with other users through the Internet.

The system provides facilities for users to create and maintain their personal profiles. Registered members can search for other users, send friend requests, and establish social connections. Once a connection is established, users can exchange private messages, participate in group conversations, and share different types of digital content.

The application also provides a platform for sharing text messages, photographs, videos, and data files. Users can maintain a personal collection of shared content and upload photographs to create and manage their own albums. The system may also provide community-based features that allow users to join groups according to their interests, preferences, and activities.

An important aspect of the application is its instant communication functionality. Users can communicate through one-to-one private chats as well as group chats. Online users can receive messages instantly, while messages sent during offline periods can be made available when they reconnect, subject to the application's message-retention settings.

The client-side application is designed to run on machines equipped with a Java Virtual Machine (JVM), while the server-side application manages communication between connected clients through socket programming. This architecture enables multiple clients to communicate with one another through a centralized server.

The major features and functionalities covered within the scope of this project are as follows:

- **User Registration and Login:** The system allows new users to register by providing the required information and subsequently access the application using valid login credentials.
- **Private Chat Facility:** Registered users can communicate individually through private conversations.
- **Group Communication:** The application enables multiple users to participate in group conversations and exchange information simultaneously.
- **Instant Messaging:** Users can send and receive text messages in real time through an Internet-based communication system.
- **Temporary Messages:** Users can select a message-disappearance duration of 5 seconds, 10 seconds, or infinite time according to their requirements.
- **File Sharing:** Users can exchange documents, photographs, and other supported data files up to a maximum size of 30 MB.
- **Webcam Support:** Users can capture photographs through a connected webcam and share them with other users.
- **Social Content Sharing:** The application allows users to share posts, photographs, videos, and other content through their feeds. Other users can view and like the shared content.
- **User Search and Friend Requests:** Users can search for other registered members using their email addresses or contact numbers and send friend requests to establish connections.
- **Profile and Album Management:** Users can maintain their personal profiles and upload photographs to create and manage their own albums.
- **Data Security:** The application uses AES encryption to enhance the confidentiality and security of sensitive user information and communication data.
- **Offline Message Access:** The system supports the availability of messages sent while a user is offline, subject to the configured message-retention policy.
- **Administrative Management:** The administrator has the facility to view, edit, and delete relevant records to maintain the proper functioning of the system.

The proposed application aims to provide a simple, accessible, and secure environment for online communication. It can also help users reconnect with old friends, communicate with alumni, exchange useful information, and participate in online communities.

By combining communication and social networking functionalities, the system aims to make digital interaction more convenient, efficient, and accessible.

3. Purpose

The primary purpose of a social networking service is to provide an online platform that facilitates the creation,

development, and maintenance of social relationships among individuals and groups. These relationships may be based on common interests, activities, educational backgrounds, professional associations, geographical locations, or existing personal connections.

A social networking service generally consists of individual user profiles, social connections, and various communication and information-sharing facilities. Most modern social networking platforms are web-based or application-based and provide different ways for users to interact through the Internet, including instant messaging, multimedia sharing, notifications, and online discussions.

Online community services are also closely associated with social networking. However, social networking services generally focus on individual users and their personal connections, whereas online community platforms primarily emphasize interaction among members of a particular group. Both types of services contribute to the development of online relationships and information exchange.

Social networking platforms allow users to share ideas, opinions, activities, photographs, videos, and important events with people across geographical boundaries. They have significantly reduced

communication barriers and made it easier for individuals to remain connected regardless of their physical location.Apart from personal communication, social networking has also become an important tool for education, professional development, business promotion, and public awareness. Educational institutions use these platforms to share academic information and communicate with students. Businesses use social networking services to promote their products, interact with customers, receive feedback, and provide customer support.

The proposed project aims to develop a secure and efficient social networking and instant messaging application that addresses the growing need for convenient online communication. Its main purpose is to provide users with a common platform where they can communicate privately or in groups, exchange files, share multimedia content, and maintain their social connections.

Another important objective of the project is to enhance communication privacy through AES encryption and temporary message functionality. The temporary messaging feature allows users to control how long their messages remain available, thereby offering an additional level of control over private conversations. However, message expiration alone cannot guarantee that information has not been copied or captured by recipients.

The specific objectives of the proposed system are:

- To develop a user-friendly platform for online social interaction and instant communication.
- To provide secure user registration and authentication facilities.
- To enable real-time private and group messaging.
- To facilitate the sharing of photographs, videos, documents, and other data files.
- To implement temporary messaging with user-selectable message-disappearance intervals.
- To provide user profile management and friend-request facilities.
- To support social interaction through posts, feeds, and online communities.
- To improve the confidentiality of communication through AES encryption.
- To provide an efficient client-server communication mechanism using socket programming.
- To simplify information sharing and help users maintain their social connections.

In conclusion, the proposed system aims to integrate social networking and instant messaging functionalities into a single application. It is intended to provide a reliable, convenient, and secure communication environment that meets the basic requirements of modern online users.

II. LITERATURE REVIEW

1. Literature Survey

The development of web-based social networking services has significantly transformed the way people communicate, interact, and share information. These services enable individuals with common interests, activities, educational backgrounds, and professional goals to connect with one another, irrespective of geographical, political, or economic boundaries.

Through facilities such as e-mail, instant messaging, discussion forums, and multimedia sharing, social networking platforms have created virtual communities where users can exchange information, express their opinions, and establish social relationships.

Information sharing is one of the fundamental characteristics of social networking platforms. Unlike physical resources, digital information can be reproduced and distributed at a very low cost. This characteristic encourages users to share knowledge, experiences, photographs, documents, and other digital resources with their networks. Such interactions promote cooperation, reciprocal communication, and the development of online communities.

Social networking sites have also become an important subject of academic and industrial research. Researchers from various disciplines have studied their influence on individual identity, privacy, social relationships, social capital, youth culture, education, and professional communication. These studies highlight the growing importance of social networking platforms in modern society and their impact on personal as well as professional activities.

According to a report published by the Pew Research Center, approximately 72% of online adults in the United States were using social networking sites in May 2013, compared with 67% in late 2012. This represented a substantial increase from February 2005, when only 8% of online adults reported using such platforms. These historical figures demonstrate the rapid growth of social networking during its early stages of mainstream adoption.

The use of social networking platforms has expanded across different age groups. Although younger users have traditionally been among the most active participants, older adults have also increasingly adopted these services. The adoption rate among internet users aged 65 and above increased considerably during the period covered by the report. This indicates that social networking is not limited to a particular age group but serves a wide range of users with different communication needs.

Social networking services provide several features, including user registration, profile creation, friend lists, messaging, content sharing, community participation, and recommendation systems.

Some platforms primarily focus on maintaining existing relationships, while others help users discover new people based on shared interests, professional backgrounds, geographical locations, or activities.

Several social networking platforms have emerged over time to serve different user requirements. Popular examples include Facebook, LinkedIn, Instagram, YouTube, Pinterest, Tumblr, and X (formerly Twitter). Other platforms, such as VKontakte in Russia, Badoo in Europe, and Weibo and Renren in China, have also contributed to the development of online social communities. Each platform offers a different combination of communication, networking, and content-sharing facilities.

The early online communities primarily focused on bringing people together through chat rooms, discussion forums, and personal web pages. These platforms provided simple publishing tools and free or inexpensive web space, allowing users to share their personal information, interests, and ideas. Some services, such as Classmates.com, initially concentrated on reconnecting former school and college classmates through online directories.

During the late 1990s, user profiles became an important feature of social networking websites. These profiles allowed users to provide personal details, describe their interests, and maintain lists of friends or contacts. The ability to search for other users and view their connections gradually became a defining characteristic of social networking platforms.

A significant development in this field occurred with the launch of SixDegrees.com in 1997. It combined user profiles, friend lists, and the ability to browse the connections of other users. This was followed by platforms such as LiveJournal, Friendster, MySpace, and LinkedIn, which introduced or popularized additional features for online interaction and networking. Orkut also gained considerable popularity in countries such as Brazil and India.

With the advancement of internet technologies, social networking platforms evolved beyond simple profile-based websites. They began incorporating instant messaging, multimedia sharing, mobile connectivity, online communities, activity feeds, and real-time notifications. These developments made social networking more interactive and accessible.

Instant messaging has become one of the most important components of modern social networking systems. It allows users to exchange text messages and other digital content in real time. Unlike traditional e-mail communication, instant messaging provides a faster and more interactive method of communication. Features such as private conversations, group chats, file sharing, and online status indicators have further improved the user experience.

The increasing use of social networking and instant messaging applications has also highlighted the importance of information security and user privacy. As users exchange personal messages,

photographs, and documents, platforms must provide appropriate mechanisms to protect information from unauthorized access. Authentication, access control, secure communication protocols, and encryption techniques are therefore important considerations in the development of modern social networking applications.

Overall, the literature indicates that social networking systems have evolved from basic online communities into comprehensive communication platforms. The integration of social networking features with instant messaging, multimedia sharing, and security mechanisms provides the foundation for developing more efficient and user-friendly communication systems.

2. Problem Statement

A social networking site can be defined as a web-based service that enables individuals to create public or semi-public profiles within a defined system, establish connections with other users, and view or explore their own connections as well as the connections of other users. The nature of these relationships may vary depending on the purpose and functionality of the platform.

Since their introduction, social networking sites such as MySpace, Facebook, Cyworld, and Hi5 have attracted millions of users. Many individuals have incorporated these platforms into their daily routines for communication, information sharing, entertainment, education, and professional networking. Different social networking platforms support different user requirements and provide a variety of technological features.

Most social networking platforms allow users to maintain existing social relationships. However, some platforms also help individuals connect with unfamiliar users who share common interests, professional goals, educational backgrounds, political views, or other activities. Certain platforms are designed for general audiences, whereas others focus on specific communities based on language, geographical location, professional interests, or cultural backgrounds.

Modern social networking systems provide several communication and content-sharing facilities, including instant messaging, blogging, photo sharing, video sharing, and mobile connectivity. These features have made online communication more convenient and interactive. Social networking platforms are also widely used by businesses for product promotion, customer engagement, marketing, and responding to customer queries.

Despite the availability of several social networking applications, users may still require a single platform that integrates multiple communication and networking facilities in an organized and convenient manner. Managing personal profiles, maintaining friend lists, exchanging messages, sharing photographs and documents, and participating in communities may involve different interfaces and services. Another important concern is the security and privacy of information shared through online platforms. Users frequently exchange personal messages, photographs, and

documents, making it necessary to implement suitable security mechanisms. Without proper authentication and access-control mechanisms, sensitive information may be exposed to unauthorized users.

The proposed project aims to address these requirements by developing a web-based social networking and instant messaging application. The system is intended to provide an integrated environment in which registered users can create and manage profiles, connect with friends, exchange messages, share multimedia content, and participate in online communities.

The major problem addressed by the project is the need for a user-friendly and integrated communication

platform that combines social networking facilities with instant messaging and content-sharing features. The system also aims to provide appropriate mechanisms for managing user information and controlling access to shared resources.

3. Existing System

The development of social networking platforms has taken place through several stages. One of the earliest recognizable social networking websites was SixDegrees.com, launched in 1997. It allowed users to create personal profiles, maintain lists of friends, and browse the friend lists of other users. Although some of these features existed independently on earlier websites, SixDegrees.com was among the first platforms to combine them into a single social networking service.

SixDegrees.com was developed to help people connect with others and communicate through online messaging facilities. The platform attracted millions of users but could not establish a sustainable business model and was eventually closed in 2000. One possible reason for its limited long-term success was that internet usage and online social networking were still at an early stage. Many users did not have sufficiently large online networks, and the platform offered limited activities beyond connecting with other members. Another early platform, Classmates.com, focused on reconnecting people with their former school and college classmates. It allowed users to search for individuals associated with particular educational institutions. However, its initial functionality was more limited than that of later social networking platforms, as user profiles and friend-list features were introduced gradually.

Between 1997 and 2001, several online community platforms introduced different combinations of user profiles and social connections. Websites such as AsianAvenue, BlackPlanet, and MiGente provided users with opportunities to create personal, professional, and dating profiles. These platforms also allowed users to identify and maintain connections with other members.

LiveJournal, launched in 1999, introduced a different approach to online social interaction. It allowed users to maintain online journals and follow the activities of other users. Its friend-list feature helped users manage the journals they followed and control access to their content through privacy settings.

Similarly, Cyworld, a South Korean online community platform launched in 1999, later incorporated social networking features. It provided users with virtual spaces and facilities for interacting with other members. LunarStorm, a Swedish online community, also developed social networking features such as friend lists, guestbooks, and personal diary pages.

With the growth of internet connectivity and increasing user participation, social networking platforms became more advanced. Friendster, launched in 2002, gained popularity by allowing users to create profiles and establish connections. MySpace and LinkedIn emerged in 2003, serving different requirements. MySpace focused primarily on social interaction and personal expression, whereas LinkedIn concentrated on professional networking and career-related connections.

Over time, Facebook and other platforms expanded the concept of social networking by introducing features such as activity feeds, photo albums, groups, messaging, notifications, and content sharing. These facilities made online interaction more dynamic and helped users maintain their social and professional relationships.

Although existing social networking systems provide a wide range of facilities, their features and working approaches differ. Some platforms primarily focus on social networking, while others emphasize instant messaging, professional connections, multimedia sharing, or community discussions. Users may therefore need to depend on multiple applications to fulfil different communication requirements.

The following table summarizes some commonly available features of existing social networking platforms:

Feature	General Description
User Registration	Allows new users to create accounts on the platform.
User Profile	Enables users to maintain personal and professional information.

Feature	General Description
Friend Management	Allows users to send, accept, and manage friend requests.
Instant Messaging	Supports real-time communication between users.
Photo Sharing	Allows users to upload and share photographs.
File Sharing	Enables users to exchange documents and other files.
Online Communities	Allows users to participate in groups based on shared interests.
Privacy Settings	Provides control over the visibility of user information and shared content.

The proposed system takes inspiration from these existing platforms and aims to integrate important social networking and communication facilities into a single web-based application.

4. Project Scope

The proposed system is designed to provide an integrated web-based environment for social networking and instant messaging. It allows registered users to create their profiles, connect with other users, exchange information, and participate in online communities.The system focuses on providing a simple and convenient interface through which users can manage their social connections and communicate with friends. The major features and functional areas included in the project are described below.

- **User Registration and Profile Management**
  The system allows new users to register by providing the required information. After successful registration and authentication, users can create and manage different types of profile information, such as personal, general, social, and professional details. Users can also update their profile information whenever required.
- **Friend Management**
  Users can search for other registered members and send them friend requests. The recipient can accept or reject the request. Once a request is accepted, the users can become part of each other's friend lists and communicate through the facilities provided by the system.
- **Instant Messaging**
  The application provides a facility for users to exchange messages with their friends. This feature enables direct communication between registered members and makes the system more interactive. Depending on the implemented functionality, users may communicate through private conversations or group discussions.
- **Scrap and Message Management**
  The system provides a facility to send scrap messages to friends. Users can also maintain a record of the messages or scraps they have sent and received. This helps users review their previous interactions and maintain a history of communication.

5. Photo Uploading And Album Management

Users can upload photographs to their accounts and organize them into personal albums. This feature enables users to store and share photographs with their friends, subject to the access permissions provided by the system.

6. File And Data Sharing

The system allows users to share digital files and other data with their friends. This facility makes it easier to exchange documents and information through the social networking platform without depending entirely on separate communication services.

7. Community Participation

Users can join communities according to their interests, preferences, and areas of activity. These communities provide an opportunity for members to interact, exchange ideas, and share information with people having similar interests.

8. Friend Invitations

The application allows users to invite other individuals to join the platform. Users can send invitations to potential members and add them to their friend network after the invitation or friend request is accepted.

9. Friend List Maintenance

Users can view and manage their existing friend lists. They can maintain their social connections and update

their lists according to their requirements. This feature helps users organize their online relationships.

10. User-Friendly Interface

The proposed application aims to provide a simple, understandable, and user-friendly interface. The major functions are organized so that users can access profile management, messaging, file sharing, and community features conveniently.

Overall, the scope of the project is to develop a web-based social networking application that combines profile management, friend connections, instant messaging, multimedia sharing, and community participation. The system is intended to provide an organized and convenient platform for online communication and social interaction.

11. Research Gap And Project Motivation

The review of existing social networking platforms indicates that considerable progress has been made in online communication and social interaction. Modern applications provide facilities for maintaining social connections, exchanging messages, sharing photographs, and participating in online communities. However, different platforms often emphasize different functionalities, and their features may not always be available in a single integrated environment.

Another important consideration is the management of user-generated content. Users may need facilities to organize their photographs, maintain records of messages, manage friend lists, and control access to shared information. An integrated application can make these activities easier by providing related functions through a common interface.

Security and privacy are also important considerations in the design of social networking applications. Since users exchange personal information and digital content, suitable authentication, authorization, and data-protection mechanisms should be considered during system development. The application should ensure that users can access only those resources for which they have appropriate permissions.

The motivation behind the proposed project is therefore to develop a simple and integrated web-based social networking and instant messaging application. The system combines essential features such as user registration, profile management, friend requests, private communication, photo albums, file sharing, and online communities.

The project also provides an opportunity to understand and implement important concepts of web application development, database management, user authentication, client-server communication, and information security. Through the integration of these concepts, the proposed system aims to demonstrate the practical development of a social networking application suitable for academic and learning purposes.

III. SYSTEM REQUIREMENTS

1. Functional Specifications

Functional requirements describe the operations and services that a system must provide to its users. They define how the application should behave when users interact with different features. The proposed Social Networking and Instant Messaging System consists of several functional components responsible for managing user accounts, profiles, privacy settings, messages, chat sessions, events, and media files.

The major functional components of the proposed system are explained below.

2. Server Object

The Server component acts as an interface between the application and its database. It is responsible for managing the exchange of information related to user accounts, account details, pages, notes, messages, events, and other application resources.

Whenever an application component requests information from the database, the required data is retrieved and made available to that component for processing. If any changes are made to the retrieved information, the updated data must be submitted to the database so that the permanent record can be modified.

This approach helps separate application-level operations from database-related operations. It also makes the system easier to maintain because individual components can communicate with the database

through a common interface rather than implementing separate database operations for every activity.

3. Account Object

The Account component is one of the most important components of the social networking application. Every user must create an account before accessing the services provided by the system. The account maintains essential information required to identify and manage a registered user. Its major responsibilities include:

- Maintaining user login credentials.
- Managing user authentication.
- Maintaining the user's friend list.
- Connecting the user account with profile information.
- Associating the account with privacy settings.
- Identifying the activities performed by the user.

Since an account contains several types of information, managing all the information through a single component may make the application complex. Therefore, the account-related functionality is logically divided into separate components.

The Account component primarily manages authentication-related information and user connections. Profile information is maintained through the Account Details component, while privacy-related information is handled by the Privacy Settings component.

This separation improves modularity and makes the application easier to understand, develop, and maintain.

4. Account Details Object

The Account Details component maintains the information displayed on a user's profile. It is associated with the corresponding Account component and provides facilities for storing and updating profile information. The information maintained by this component may include:

1. User's full name. Gender. Location. Personal information. Professional details. Interests and hobbies. Other profile-related information. Users can modify their profile details whenever required. The updated information is stored in the database and displayed whenever an authorized user visits the corresponding profile. The main purpose of this component is to separate profile management from account authentication, thereby improving the organization of the application.

5. Privacy Settings Object

Privacy is an important consideration in any social networking application because users frequently share personal information, photographs, messages, and other digital content. The Privacy Settings component maintains the privacy preferences associated with each user account. It determines who is permitted to access a particular profile or resource.

The system provides three levels of privacy:

- **Open Privacy**
  Under this setting, the selected information is publicly accessible. Any user who visits the relevant section can view the information.
- **Friends-Only Privacy**

Under this setting, only users who are connected through an accepted friendship can access the selected information.

- **Closed Privacy**

Under this setting, the information is restricted to the account owner. Other users are not permitted to access the protected content.

The privacy settings may be applied to different sections of the application, including:

- User profiles.
- Media files and photo albums.
- Account walls.
- User-created pages.
- Notes and other shared content.

When a user modifies a privacy preference, the corresponding settings are updated in the database. Whenever another user attempts to access a protected resource, the application checks the privacy settings of the resource owner. The system compares the requested access level with the relationship between the resource owner and the requesting user.If the requesting user satisfies the privacy conditions, access

is granted. Otherwise, the application displays an appropriate access-denied message or redirects the user to an error page. This mechanism helps users maintain control over their personal information and shared content.

1. chat session object

The Chat Session component provides real-time communication between users who are connected to the social networking platform.

When a user selects an available friend from the chat list, the system initiates a chat session. The selected friend receives a notification indicating that a new chat session has been requested.

Once the chat session is established, both users can exchange messages through the chat interface. The session remains active until one or both participants leave the conversation or log out of the application.The chat functionality is an important feature of the proposed system because it enables users to communicate quickly without requiring them to navigate to a separate messaging service.

2. Event Object

The Event Component Allows Users To Create And Share Information About Upcoming Activities Or Occasions With Their Friends.

For Example, A User May Create An Event For A Birthday Celebration, Meeting, Social Gathering, Or Other Important Activity. The Event Information Is Stored In The Database And Made Available According To The Applicable Privacy Settings.

The Major Operations Supported By The Event Component Include:

- Creating A New Event.
- Entering Event-Related Information.
- Viewing Existing Events.
- Editing Event Details.
- Sharing Event Information With Friends.

When An Event Is Created, The System Associates It With The Account Of The User Who Created It. The Event Information Is Then Stored In The Database For Future Retrieval.Users Can Access The Events Section To View The Available Events And Obtain Information About Activities Shared By Other Users.

6. Link Object

The Link Component Is Intended To Manage Links Or Web Addresses Shared By Users Through The Social Networking Application. It Provides A Structured Way To Store And Access Shared Web Resources. Users Can Share Useful Websites, Educational Resources, Articles, And Other Relevant Online Information With Their Friends.Each Link Is Associated With The User Who Created Or Shared It. The Stored Information Can Be Retrieved Whenever An Authorized User Accesses The Corresponding Section.

1. **Note Object**

The Note Component Allows Users To Create And Maintain Text-Based Information Within The Application.Users May Prepare Notes Containing Personal Information, Thoughts, Reminders, Or Other Content That They Wish To Store Or Share. Each Note Is Associated With The Account Of Its Creator. The Application Can Retrieve And Display Notes According To The Privacy Settings Defined By The User. This Allows Users To Maintain Their Information In An Organized Manner While Controlling Its Visibility.

7. Page Object

The Page component manages user-created pages within the social networking application. A page can contain information related to a particular subject, interest, activity, or other topic. Users can create pages and maintain their associated content.Each page is linked to the account of its creator. The application retrieves and displays page information according to the applicable privacy settings.

1. Message Object

The Message component is responsible for handling messages exchanged between users.

A user can send a message to another registered user through the Message Center. To compose a message, the sender provides the recipient's email address, message subject, and message content.

Before storing the message, the application verifies whether the recipient exists in the system. If the recipient is valid, the message is associated with both

the sender's and receiver's accounts and stored in the database.

When a user opens the inbox, the application retrieves the messages addressed to that user. The messages can be displayed in descending order according to their received date and time, allowing the latest messages to appear first.

The Message component provides a convenient method of communication between users, even when the recipient is not currently available for a live chat session.

2. Wall Post Object

The Wall Post component allows users to publish information, thoughts, announcements, and status updates on their social networking profiles.

Unlike private messages, wall posts are intended to be visible to the user's friends or other permitted viewers, depending on the privacy settings.

Users can create posts and interact with existing posts through comments. Each wall post is associated with the account of the user who created it. The application stores the post content and

its related information in the database. Whenever an authorized user visits the relevant profile or activity section, the available posts are retrieved and displayed. This feature encourages interaction among users and makes the social networking application more engaging.

8. Account Creation And Registration

Account creation is the first step required to access the services provided by the application. A new user must complete the registration process before using the system. The registration process consists of the following major stages:

1. Login Information

The user is required to provide basic authentication details, including:

- Email address.
- Password.
- Password confirmation.

The password confirmation field helps prevent accidental typing errors while creating an account.

2. Personal Information

The user provides basic profile information, such as name, gender, and location. Certain fields may be optional depending on the registration requirements.

9. Interests And Activities

Users may also provide information about their interests, hobbies, preferences, and activities. This information helps create a more informative profile and supports interaction among users with similar interests.

After the required information has been submitted, the system validates the registration details and creates the user's account. The associated profile information and default privacy settings are also initialized.

Once registration is completed successfully, the user can proceed to the login page and access the application using the registered credentials.

1. Privacy Management

The application provides users with the facility to modify the visibility of their profiles and shared information.

Whenever a user changes a privacy setting, the updated preference is stored in the database. When another user attempts to view the protected information, the application checks whether the requested access is permitted.

The access-control mechanism follows these basic steps:

- The requesting user selects a profile, page, or other resource.
- The application identifies the owner of the requested resource.
- The system retrieves the privacy settings associated with that resource.
- The relationship between the requesting user and the resource owner is checked.
- Access is granted or denied according to the configured privacy level.

This process ensures that the visibility of user information is controlled according to individual preferences.

2. Real-Time Chatting

The chatting facility enables users to communicate with their friends through an interactive messaging interface.

When a user selects an online friend, the application initiates a chat request. The receiving user is notified about the incoming request and can participate in the conversation.

The chat interface displays the messages exchanged between the participants. Whenever a user sends a new message, the application updates the corresponding chat session so that the other participant can receive the latest information.

Real-time communication requires continuous synchronization between the user interface and the database or communication service. The application must therefore retrieve new messages and update the conversation without requiring the user to reload the entire webpage. This functionality improves the responsiveness of the application and provides users with a convenient communication experience.

10. Messaging System

The messaging system provides a facility for sending messages to registered users. Unlike real-time chatting, the messaging system does not require both users to be online simultaneously. The sender can compose and submit a message, which is stored in the database for the recipient.

The basic message-sending process involves the following steps:

- The sender opens the Message Center.
- The sender selects the option to compose a new message.
- The recipient's email address is entered.
- The message subject and content are provided.
- The application validates the recipient's account.
- The message is stored in the database.
- The recipient can access the message through the inbox.

The system also maintains the association between the sender, receiver, and message. This helps retrieve messages belonging to a particular user.

11. Event Management

The Event Management facility enables users to create, view, and modify information about upcoming events. To create an event, the user enters the required details through an event creation form. After submitting the form, the application validates the information and stores the event in the database.

The user can subsequently access the Events section to view the created event. If any modification is required, the user can edit the existing information through the event management interface. The event information is associated with the account of the user who created it. This helps maintain ownership and allows the application to apply appropriate access permissions.

12. Media Uploading

Media uploading is an important feature of social networking applications because it allows users to share photographs, videos, audio files, and documents. The proposed system provides a media uploading facility through which users can select a file from their local devices and submit it to the application.

The user may also provide a short description of the uploaded file. The application validates the selected file before storing it in the appropriate location.

The validation process may include checking:

- File extension.
- Supported file format.
- File size.
- User permissions.
- Availability of sufficient storage space.

The system should allow only permitted file types to reduce the risk of uploading harmful or unsupported files. Depending on the implementation, supported formats may include common image, audio, video, and document formats.

After successful validation, the file is stored in the designated storage location and associated with the uploading user's account. The application can then

retrieve and display the media according to the user's privacy settings.This feature provides users with a convenient way to maintain and share their digital content.

13. Non-Functional Requirements

Non-functional requirements describe the quality attributes and operational characteristics of a software system. They specify how efficiently, securely, and reliably the application should perform its functions.

The major non-functional requirements of the proposed social networking application are described below.

- **Security**
  Security Is One Of The Most Important Requirements Of The Proposed Application. Since The System Maintains Personal Information, User Profiles, Messages, And Uploaded Files, Suitable Security Mechanisms Must Be Implemented.
  The Application Should Provide Secure Authentication Through Registered Usernames Or Email Addresses And Passwords. Unauthorized Users Must Not Be Allowed To Access Protected Information.
  User Sessions Should Be Managed Properly, And Users Should Be Able To Log Out Securely. Appropriate Access-Control Mechanisms Should Also Be Implemented To Protect Private Profiles And Shared Content.

- **Availability**
  The Application Should Be Designed To Remain Accessible Whenever Users Need Its Services. Since It Is A Web-Based System, Users Should Be Able To Access The Application Through An Internet Connection Without Depending On A Particular Physical Location.
  The Availability Of The Application Also Depends On The Reliability Of The Hosting Environment, Database Service, And Network Connectivity.

- **Performance**
  The System Should Provide A Satisfactory Response Time For Common Operations Such As Login, Profile Viewing, Message Retrieval, And Media Access.
  Database Operations Should Be Designed Efficiently To Minimize Unnecessary Processing And Reduce Waiting Time. The Application Should Also Be Optimized To Provide A Responsive Experience When Multiple Users Access It.

- **Reliability**
  The Application Should Perform Its Intended Operations Accurately And Consistently. User Information, Messages, And Other Important Records Should Be Stored Reliably.
  The System Should Handle Unexpected Errors Appropriately And Avoid Losing Information During Normal Operations. Suitable Backup And Recovery Mechanisms Should Be Considered For Protecting Important Data.

- **Scalability**
  The System Should Be Designed So That Its Resources And Infrastructure Can Be Expanded As The Number Of Registered Users Increases. The Database Structure And Application Components Should Support Future Enhancements Without Requiring A Complete Redesign Of The System.

- **Portability**
  The Application Should Be Accessible Through Commonly Used Operating Systems And Modern Web Browsers. Users Should Not Be Required To Install A Separate Desktop Application To Access The Basic Web-Based Services.

14. Data Integrity

The Application Should Maintain The Correctness And Consistency Of Stored Information. Duplicate Or Inconsistent Records Should Be Avoided Through Suitable Validation And Database Design. Relationships Between Users, Messages, Posts, Events, And Uploaded Media Should Be Maintained Properly.

- **Message Delivery**
  The Messaging System Should Deliver Messages To The Intended Recipients In The Correct Sequence.
  If A Message Cannot Be Delivered Or Stored Successfully, The Application Should Provide Suitable Feedback To The Sender. The System Should Also Maintain Message Information So That

Users Can Retrieve Their Conversations When Required.

- **Usability**
  The Application Should Provide A Simple And Understandable User Interface. Navigation Between Different Sections, Such As Profiles, Messages, Chat, Events, And Media, Should Be Convenient.
  The System Should Display Meaningful Error Messages And Confirmation Notifications So That Users Can Understand The Result Of Their Actions.
- **Maintainability**
  The Application Should Follow A Modular Design Approach So That Individual Components Can Be Modified, Tested, And Improved Independently.
  A Well-Organized Structure Helps Developers Identify Errors, Implement New Features, And Maintain The System Efficiently.
- **Cost Effectiveness**
  The Application Should Use Development Tools And Infrastructure That Are Economical And Suitable For The Intended Scale Of The Project.
  The System Should Avoid Unnecessary Resource Consumption And Should Be Capable Of Operating Efficiently Within The Available Hardware And Hosting Resources.

15. Software Tools And Technologies

The Proposed Social Networking Application Is Developed Using Modern Web Technologies. These Technologies Are Used To Design The User Interface, Manage Application Logic, Provide Responsive Layouts, And Store Application Data.

The Major Software Tools And Technologies Used In The Project Are Described Below.

- **Html (Hypertext Markup Language)**

Html Stands For Hypertext Markup Language. It Is The Standard Markup Language Used To Create And Structure Web Pages.

Html Provides The Basic Structure Of A Webpage By Defining Elements Such As Headings, Paragraphs, Lists, Tables, Hyperlinks, Images, And Forms. These Elements Help Organize Information In A Meaningful Manner So That Web Browsers Can Display It Properly.

Html Documents Consist Of Different Elements Represented Through Tags. For Example, Heading Tags Are Used To Define Headings, Paragraph Tags Are Used To Display Textual Information, And Image Tags Are Used To Insert Pictures Into A Webpage.

Html Also Supports Interactive Elements Such As Forms, Input Fields, And Buttons. These Elements Are Particularly Useful In Web Applications That Require User Registration, Login, Message Composition, And Information Submission.

Modern Html, Including Html5 Features, Supports Multimedia Content And Provides Additional Elements For Creating Structured And Interactive Web Pages.

In The Proposed Project, Html Is Used As The Foundation For Creating The Structure Of The Application's User Interface.

- **Css (Cascading Style Sheets)**
  Css Stands For Cascading Style Sheets. It Is A Stylesheet Language Used To Control The Presentation And Appearance Of Html Documents. While Html Defines The Structure Of A Webpage, Css Determines How The Webpage Looks. It Provides Facilities For Controlling Colors, Fonts, Spacing, Borders, Backgrounds, Alignment, And Layout.
  Css Is An Essential Technology In Web Development Because It Helps Create Attractive, Consistent, And User-Friendly Interfaces.
  The Major Advantages Of Css Are As Follows:

- **Improved Appearance**
  Css Allows Developers To Create Visually Attractive Webpages By Controlling Colors, Typography, Backgrounds, Borders, And Other Design Elements.

- **Code Reusability**
  A Single Stylesheet Can Be Used By Multiple Webpages. This Reduces Code Duplication And Ensures Consistency Throughout The Application.

- **Easy Maintenance**

Changes Made To A Common Stylesheet can automatically affect all the webpages that use it. This makes website maintenance easier.

- **Faster Development**
  CSS reduces the need to specify presentation-related properties repeatedly in HTML elements. This makes the code more organized and easier to manage.

- **Responsive Design**
  CSS provides media queries and flexible layout techniques that allow webpages to adjust according to different screen sizes and devices.

- **Better User Experience**
  Proper use of CSS improves readability, navigation, and the overall appearance of the application.
  In the proposed project, CSS is used to design the layout, styling, and visual appearance of the social networking application.

16. Advantages Of Css

- It reduces repetitive styling code.
- It allows the same design to be reused across multiple pages.
- It simplifies website maintenance.
- It supports responsive layouts for different devices.
- It provides greater control over webpage presentation.
- It helps maintain a consistent design throughout the application.

7. **JAVASCRIPT**

JavaScript is a widely used programming language for developing interactive and dynamic web applications.

It enables webpages to respond to user actions without requiring the entire page to be reloaded. JavaScript can be used to validate forms, handle button clicks, update webpage content, display notifications, and communicate with backend services. JavaScript works alongside HTML and CSS. HTML provides the structure, CSS manages the appearance, and JavaScript controls the interactive behaviour of the application.

Some important features of JavaScript include:

- Dynamic modification of webpage content.
- Client-side form validation.
- Event handling.
- Asynchronous communication with servers.
- Support for reusable functions and modules.
- Integration with modern web development libraries.

In the proposed application, JavaScript is used as the programming language for implementing interactive functionality and supporting the operation of ReactJS components.

1. **REACTJS**

ReactJS, commonly known as React, is an open-source JavaScript library used for developing interactive user interfaces. It was initially developed at Facebook and released publicly in 2013.

React follows a component-based development approach. In this approach, the user interface is divided into smaller and reusable components. Each component is responsible for displaying a particular part of the application.

For example, a social networking application may contain separate components for:

- Login and registration.
- User profile.
- Navigation menu.
- Friend list.
- Messaging interface.
- Chat window.
- Photo gallery.
- Event management.

These components can be combined to create complex application interfaces. Reusability reduces code duplication and makes application development more manageable.

17. Virtual Dom

One of the important concepts associated with React is the Virtual Document Object Model (Virtual DOM).

The Virtual DOM is an in-memory representation of the actual browser DOM. When application data changes,

React compares the updated Virtual DOM with its previous version and identifies

the changes that need to be reflected in the actual DOM. This approach helps minimize unnecessary DOM updates and improves the efficiency of rendering user interfaces.

18. Component-Based Architecture

React applications are developed using independent and reusable components. Components can receive information through properties and maintain their own state where required.

This approach makes it easier to divide a large application into smaller functional units.

19. Advantages Of Reactjs

- Supports reusable UI components.
- Helps develop interactive user interfaces.
- Provides efficient rendering through the Virtual DOM.
- Improves code organization and maintainability.
- Supports integration with different libraries and services.
- Suitable for developing single-page applications.

In the proposed project, ReactJS is used to develop the frontend interface and organize the different modules of the social networking application.

1. **Material Ui**

Material UI, also known as MUI, is a widely used React component library that provides ready-made interface components based on Google's Material Design principles.

It helps developers create consistent and attractive user interfaces without designing every component from the beginning.

Material UI provides several reusable components, including:

- Buttons.
- Text fields.
- Navigation menus.
- Dialog boxes.
- Cards.
- Tables.
- Icons.
- Form controls.
- Layout and grid components.

These components can be customized according to the requirements of an application.

20. Important Features Of Material Ui

- **Reusable Components:** It provides ready-to-use components that reduce development time.
- **Responsive Design:** Its layout facilities help create interfaces suitable for different screen sizes.
- **Consistent Appearance:** It follows common design principles, helping maintain visual consistency.
- **Customization:** Developers can modify component styles according to application requirements.
- **Accessibility:** Many components provide features that support accessible interface design.
- **Integration with React:** It is designed to work directly with React applications.

In the proposed project, Material UI can be used to develop common interface elements and improve the overall appearance and usability of the application.

1. **Firebase Realtime Database**

Firebase Realtime Database is a cloud-hosted NoSQL database provided by Google Firebase. It allows applications to store and synchronize data between connected clients in real time. Unlike traditional relational databases, Firebase Realtime Database stores information in JSON format. Data is organized as a tree structure containing different nodes and values. Whenever information is modified in the database, connected clients can receive the updated information automatically. This feature makes Firebase Realtime Database suitable for applications that require frequent data synchronization.

IV. MAJOR FEATURES OF FIREBASE REALTIME DATABASE

1. Real-Time Synchronization

The database automatically synchronizes data across connected devices. When a user updates information, other authorized users can receive the latest data without repeatedly refreshing the webpage.

1. **Cloud-Based Storage**
The database is hosted on Google's cloud infrastructure. This allows applications to store and access information without maintaining a separate database server.

2. Offline Support
Firebase provides offline capabilities through its supported SDKs. Previously accessed data can be cached locally, and certain changes can be synchronized when the device reconnects to the internet. The exact offline behaviour depends on the platform and SDK configuration.

3. Security Rules
Firebase Realtime Database provides a rule-based security mechanism for controlling access to stored information. Developers can define which users are allowed to read or write particular database records.

When combined with Firebase Authentication, these rules can be used to associate database permissions with authenticated users.

4. Nosql Database Structure
Firebase Realtime Database stores information in JSON format rather than using traditional relational tables. This structure allows developers to organize information according to the data-access requirements of the application.

5. Cross-Platform Support
Firebase provides software development kits for different platforms, including web and mobile applications. This makes it possible to develop applications that share information through a common database.

1. **Scalability**
Firebase provides cloud infrastructure that can support applications with increasing data and user requirements, subject to the selected service plan and database design.

1. **Other Firebase Services**Apart from Firebase Realtime Database, the Firebase platform provides several other services that can support application development.

6. Cloud Firestore
Cloud Firestore is a scalable NoSQL cloud database that stores data in documents organized into collections. It provides querying, synchronization, and offline support for supported platforms.

4.6.1 Firebase Authentication
Firebase Authentication provides facilities for managing user authentication. It supports different authentication methods, including email and password-based login and selected third-party authentication providers.

1. **Firebase Hosting**
Firebase Hosting provides hosting facilities for web applications and static website resources such as HTML, CSS, and JavaScript files.

7. Firebase Remote Config
Firebase Remote Config allows developers to modify selected application settings and behaviour without requiring users to download a new version of the application. The availability and use of these services depend on the actual implementation of the project.

1. ***HARDWARE REQUIREMENTS***
Hardware requirements describe the minimum computer resources required to develop, test, and operate the proposed application.

Since the application is web-based, it does not require highly specialized hardware for basic operation. However, adequate system resources are recommended for smooth development and testing.

The hardware specifications mentioned in the project are as follows:

Hardware Component	Minimum Specification
Processor	Intel Pentium IV, 2.0 GHz or above

Hardware Component	Minimum Specification
RAM	512 MB or above
Hard Disk	80 GB or above
Monitor	CRT or LCD monitor
Keyboard	Standard or multimedia keyboard
Mouse	Compatible pointing device
Network	Internet connectivity

1. ***SOFTWARE REQUIREMENTS***

The software requirements specify the operating system, programming technologies, development tools, and application environment required for developing and executing the proposed system.

The major software requirements are listed below.

Software Component	Technology / Specification
Frontend Library	ReactJS
Markup Language	HTML
Styling Language	CSS
Programming Language	JavaScript
UI Component Library	Material UI
Backend / Database Service	Firebase Realtime Database
Operating System	Windows or another supported operating system
Web Browser	Google Chrome, Microsoft Edge, Mozilla Firefox, or Safari

The application is designed to operate through a web browser. Therefore, users can access the system without installing a separate desktop application, provided that they have a compatible browser and an internet connection.

1. **SUPPORTED WEB BROWSERS**

The web application is intended to support commonly used modern web browsers, including:

- Google Chrome.
- Microsoft Edge.
- Mozilla Firefox.
- Apple Safari.

The application should be tested on the latest stable versions of these browsers to ensure compatibility and consistent performance.

Browser compatibility testing is important because different browsers may interpret certain webpage elements and JavaScript operations differently.

8. Development Environment

An appropriate development environment is required for writing, editing, testing, and debugging the application.Modern web development tools provide features such as code editing, syntax highlighting, error detection, debugging, and project management.

A suitable code editor or Integrated Development Environment (IDE) can be used to develop the ReactJS application. The development environment should also support JavaScript, HTML, CSS, and the required Firebase libraries.

9. Integrated Development Environment (Ide)

An Integrated Development Environment (IDE) is a software application that provides a collection of tools required for software development. In earlier programming environments, developers generally used separate text editors, compilers, and debugging tools. They had to write the source code in a text editor, save the file, execute the compiler, examine error messages, and return to the editor to make corrections. An IDE simplifies this process by combining several development facilities into a common environment.

The major functions of an IDE include:

- Writing and editing source code.
- Managing project files.
- Detecting syntax errors.
- Providing code suggestions and syntax highlighting.
- Running and testing applications.
- Supporting debugging.
- Managing development extensions and libraries.

10. General Working Of An Ide

The development process generally involves the following steps:

- The developer opens or creates a project in the IDE.
- Source code is written or modified using the code editor.
- The project files are saved.
- The development tools check the code for errors.
- The application is executed or built using the appropriate development tools.
- The developer tests the application and identifies any errors.
- Necessary corrections are made and the application is tested again.

For compiled programming languages, the build process may involve preprocessing, compilation, and linking. However, the development workflow for a ReactJS application is different because JavaScript is executed by the browser or a JavaScript runtime, while development tools may bundle and optimize application resources. The IDE therefore plays an important role in improving developer productivity and simplifying application development.

1. ***SUMMARY***

This chapter presented the system requirements and technologies associated with the proposed Social Networking and Instant Messaging Application. The functional specifications described the major components of the system, including user account management, profile details, privacy settings, chat sessions, messaging, wall posts, events, notes, links, and media uploading.

V. CHAPTER 4 DESIGN SPECIFICATIONS

1. Modular Design

Modular design is an important aspect of software development in which the complete application is divided into smaller, independent, and manageable modules. Each module is responsible for performing a specific set of functions. This approach makes the application easier to develop, test, maintain, and upgrade.

The proposed social networking and instant messaging application is divided into different modules to ensure proper organization of functionalities and smooth interaction between users. Each module performs a specific task while communicating with other modules whenever required.

2. Register To Be A Member Module

This module provides functionalities for those people who wants to open an account. Applicants can post their views with personal and professional details. They can also update the profile as frequently as required. The member can also browse through the friends profile available. Members can also get message alerts when their friends message them.

3. Profile Module

The Profile Module is responsible for managing the personal information and profile-related activities of registered members. Every registered user has an individual profile containing relevant personal and professional details. After logging in, users can view their existing profile information. If any information needs to be modified, users can edit and update their details according to their requirements.

This module helps maintain the accuracy and relevance of user information. It also enables members to view other users' profiles, subject to the privacy settings and access permissions provided by the application.

The major functionalities of the Profile Module include:

- Displaying registered users' profile information.
- Editing and updating personal details.
- Managing profile-related information.
- Viewing other members' profiles.
- Supporting profile accessibility according to privacy settings.

VI. SYSTEM DESIGN

1. Data Flow Diagrams

A Data Flow Diagram (DFD) is a graphical representation of the movement of data within an information system. It illustrates how data enters the

system, how it is processed, where it is stored, and how the processed information is delivered to users or other external entities. DFDs are useful during system analysis and design because they provide a simplified representation of the internal working of an application without focusing on programming details.

A Data Flow Diagram mainly represents the following aspects:

- **Processes:** These represent the activities performed by the system to transform input data into meaningful output.
- **Data Flows:** These indicate the movement of information between different processes, data stores, and external entities.
- **Data Stores:** These represent the locations where application data is stored for future use.
- **External Entities:** These represent users or external systems that provide input to the application or receive information from it.

In the proposed application, DFDs illustrate the flow of information during activities such as user registration, login, profile management, friend searching, messaging, and other social networking operations.

Data flow diagrams model the flow of data into, through and out of an information system:

- show the processes that change or transform data
- show the movement of data between processes
- represent a system as a network of processes which transform data flowing between them

The user screen flow shows what a user of the community will see. After successfully logging on, the user will be given various links (such as search users, search boards, view mail, etc.), and be able to select options from there, or go back to their home.

2. Client Screen Flow

The client screen flow represents the sequence of screens and options available to a user while interacting with the application. It describes how users navigate from one screen to another and access different functionalities.

After successfully logging in, the user is directed to the home page, from where different options are available. These options may include searching for users, viewing friend lists, accessing messages, managing profile information, viewing posts, and exploring other available features.

The user can select the required option and navigate to the corresponding screen. The application also provides navigation facilities to return to the home page or move to other sections.

The client screen flow helps in understanding the overall navigation structure of the application and ensures that users can access the required functionalities in an organized manner.

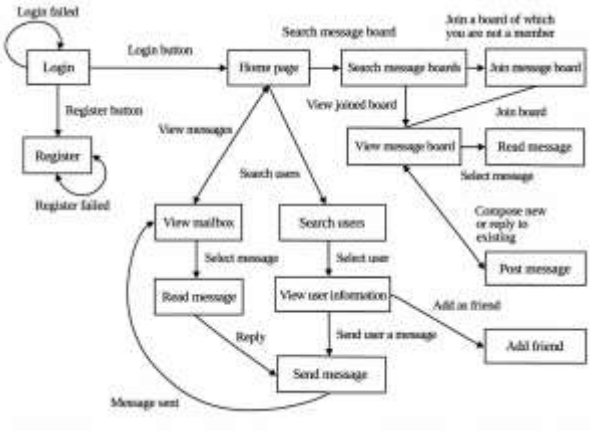


Fig 4.1 Client Screen Flow

Use case diagrams.

A use case diagram is a graphic depiction of the interactions among the elements of a system. A use case is a methodology used in system analysis to identify, clarify, and organize system requirements. In this context, the term "system" refers to something being developed or operated, such as a mail-order product sales and service Web site. Use case diagrams are employed in UML (Unified Modeling Language), a standard notation for the modeling of real-world objects and systems.

System objectives can include planning overall requirements, validating a hardware design, testing and debug ging a software product under development, creating an online help reference, or performing a

consumer-service-oriented task. For example, use cases in a product sales environment would include item ordering, catalogue updating, payment processing, and customer relations. A use case diagram contains four components. The boundary, which defines the system of interest in relation to the world around it.

- The actors, usually individuals involved with the system defined according to their roles.
- The use cases, which the specific roles are played by the actors within and around the system.
- The relationships between and among the actors and the use cases.

Login/Registration

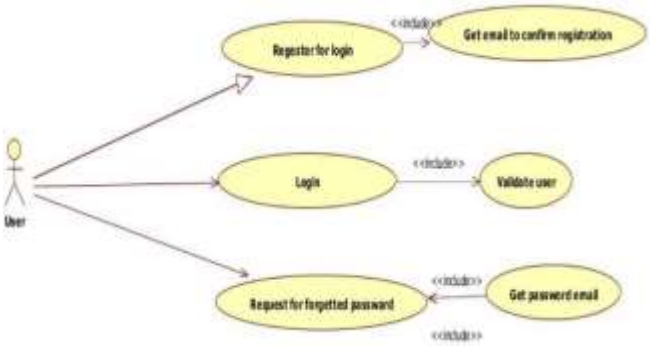


Fig 4.2 Login/Registration

Wall Page

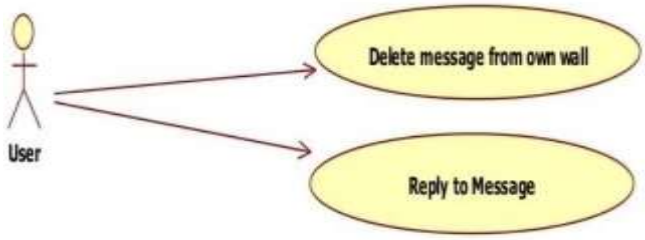


Fig 4.3 Wall Page

Profile page

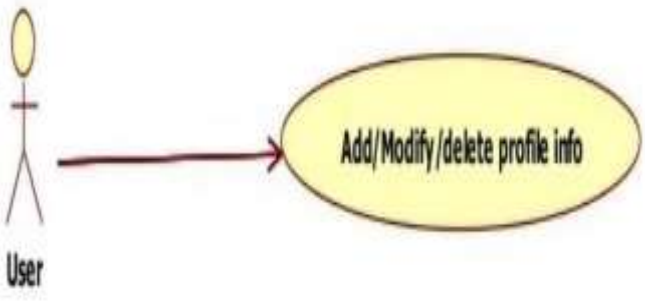


Fig 4.4 Profile Pag

Search friend page

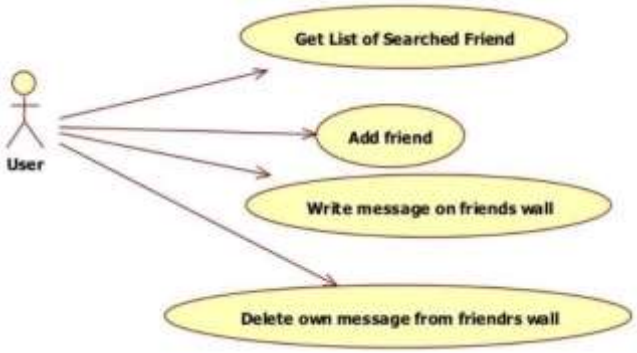


Fig 4.5 Search friend page

Friend list page

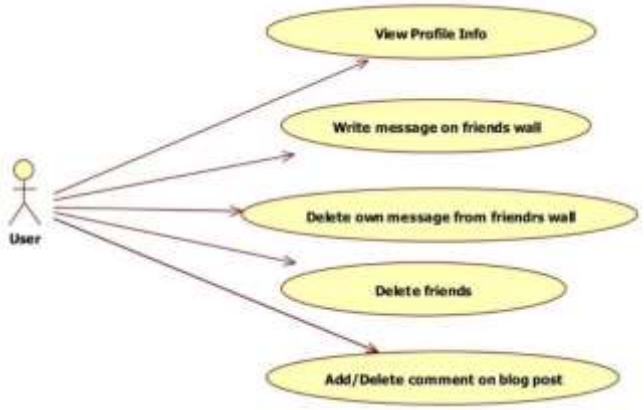


Fig 4.6 Friend list page

Photo Page

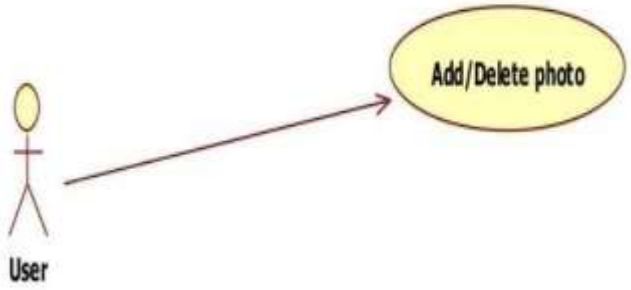


Fig 4.7 Photo Page

Home Page



Fig 4.8 Home Page

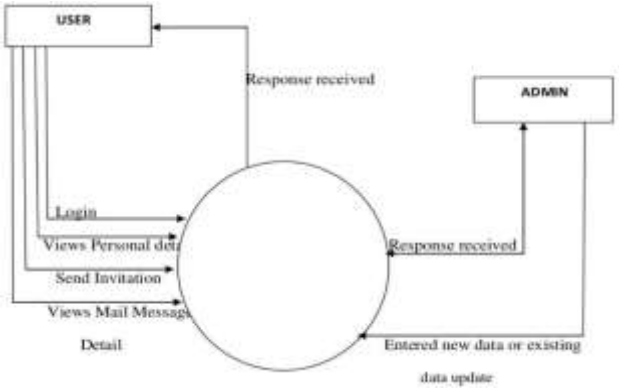
3. Context Diagram

The highest level data flow diagram is the context diagram.

- The context diagram shows the interaction of the system with its environment in terms of data flows
- The context diagram defines the boundary of the system (the scope of the system)
- Only the data flows which leave the system and the data flows which come from outside the system are shown.

Level DFD

A level 0 DED, also called a fundamental system model or context diagram represents the entire software element as a single bubble with input and output data indicated by incoming and outgoing arrows, respectively.



Level DFD:

Fig. 4.9 Level 0 DFD

This level of DFD provide more detailed structure. It provides a detailed a view of requirements and flow of data from 1 bubble to another.

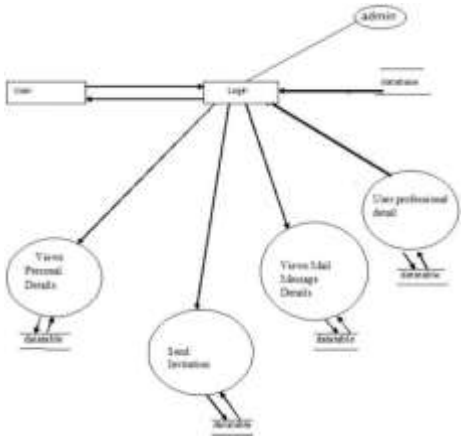


Fig. 4.10 Level 1 DFD

4. Entity Relationship Diagram

The entity relationship model is a high level data model. It is based on a perception of a real world that consists of a collection of basic objects, called entities, and of relationship among these objects. It was developed to facilitate database design by allowing specification of an enterprise schema, which represent the overall logical structure of a database.

Entity: An entity is an object that has its existence in the real world. It includes all those "things" about which data is collected. An entity may be a tangible object such as a student, a place or a part. It may also be non-

tangible such as an event, a job title or a customer account. For example, if we say that a customer buys goods, it means customer and goods are entities.Diagrammatically, entities are represented in rectangles.

An Entity Set: It is a set of entities of the same type that share the same properties, or attributes. The set of all persons who are customers at a given bank, example, can be defined as the entity set customer.

Attributes: Attributes are units that describe the characteristics or properties of entities. In a database, entities are represented by tables and attributes by columns. For example, a customer entity might have numerous attributes such as code, name and addresses. Similarly, the goods entity may have attributes like code and price. They are drawn in elliptical shapes along with the entity rectangles.

Entity Relationship Diagram

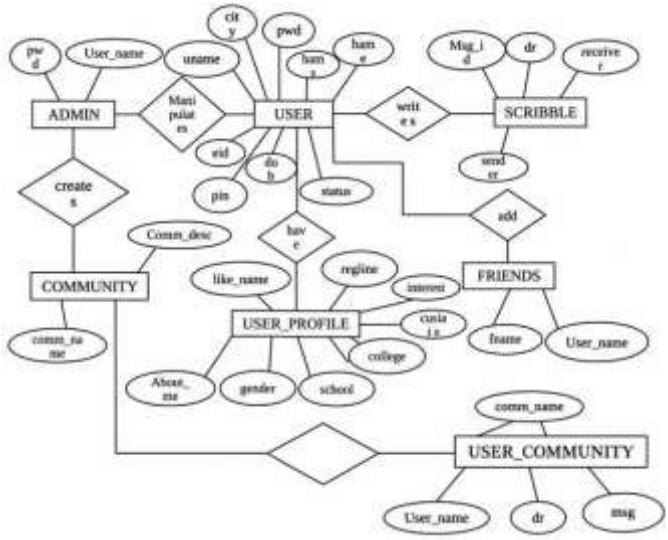


Fig 4.11 ER Diagram

VII. IMPLEMENTATION AND PSEUDOCODE

Login/Signup

```
import React, { useState } from "react"; import "./Signup.css";

import Headersignin from "../Headers/Headersignin"; import { auth, database } from "../../firebase";

import { useHistory } from "react-router-dom";

function Signup() {

const [username, setUsername] = useState(""); const [email, setEmail] = useState("");

const [password, setPassword] = useState(""); const [repassword, setRepassword] = useState(""); const [error, seterror] = useState("");

const history = useHistory();

// const {user} = useContext(AuthContext);

const createUser = (event) => {

event.preventDefault(); // this will prevent screen from refreshing if (username === "" || email === "") {

seterror("* All fields are mandatory to fill."); return;

}

if (password === "" || password !== repassword) { seterror("* Password doesn't match.");

return;

}

auth

.createUserWithEmailAndPassword(email, password)

.then((userObj) => { if (userObj) {

database.users.doc(userObj.user.uid).set({ username: username,

email: email, password: password,

})

.then(() => { history.push("/home");

});

}

})

.catch((e) => { alert(e.message);

});

};

return (

<div className="login-outer-div">

<Headersignin />
```

```
<div className="login">
<div className="login_data">
{/* {user && <h4>{user.uid}</h4>} */}
<h1>Alumini Connect</h1>
<h2>GBPIET Pauri Garhwal</h2>
<p>
A platform to connect, network & leverage your private alumni network. AlmaConnect gets up news & updates of your alums and help you interact with alums ...
</p>
</div>
<div className="login_container">
<h1>Sign-UP</h1>
<form>
<input type="text"
placeholder="Username" value={username}
onChange={(event) => setUsername(event.target.value)}
/>
<input type="text"
placeholder="E-mail" value={email}
onChange={(event) => setEmail(event.target.value)}
/>
<input type="password"
placeholder="password" value={password}
onChange={(event) => setPassword(event.target.value)}
/>
<input type="password"
placeholder="Retype Password"
value={repassword}
onChange={(event) => setRepassword(event.target.value)}
/>
{error && <p style={{ color: "#D8000C" }}>{error}</p>}
<button type="submit"
className="login-signup" onClick={createUser}
>
Sign Up
</button>
</form>
</div>
</div>
</div>
);
}
export default Signup;
```

Home Page

```
import React, { useContext, useState, useEffect } from "react"; import { AuthContext } from "../../Context/AuthContext"; import Navbar from "../Headers/Navbar";
import "./HomePage.css";
import ProfileDetail from "./ProfileDetail"; import CollegePost from "./CollegePost"; import Feeds from "./Feeds";
import NetworkList from "./NetworkList"; import { database } from "../../firebase";
function HomePage() {
const { user } = useContext(AuthContext); const [userData, setUserData] = useState("");
useEffect(() => {
const unsub = database.users.doc(user.uid).onSnapshot((snapshot) => { setUserData(snapshot.data());
});
return () => { unsub();
};
}, [user]); return (
<div className="homepage">
```

```
<Navbar userData={userData} />
{/* {user && <h1>{user.uid}</h1>} */}
<div className="homepage_container">
<div className="homepage_container_div leftdiv">
<ProfileDetail userData={userData} />
{/* <CollegePost /> */}
</div>
<div className="homepage_container_div middiv">
<Feeds userData={userData} />
</div>
<div className="homepage_container_div rightdiv">
<NetworkList />
</div>
</div>
</div>
);
}
export default HomePage;
```

Connections

```
import React, { useState, useEffect, useContext } from "react"; import "./Connections.css";
import MyConnections from "./MyConnections"; import AddConnections from "./AddConnections"; import Requests from "./Requests";
import Sent from "./Sent";
import { AuthContext } from "../../Context/AuthContext"; import { database } from "../../firebase";
function Connections() {
const [active, setActive] = useState("myconnection"); const { user } = useContext(AuthContext);
const [userData, setUserData] = useState("");
useEffect(() => {
const unsub = database.users.doc(user.uid).onSnapshot((snapshot) => { setUserData(snapshot.data());
});
return () => { unsub();
};
}, [user]);
const handleChange = (value) => { setActive(value);
};
return (
<div className="connection">
<div className="container_header">
<h1>Connections</h1>
</div>
<div className="connection_container">
<div className="menu">
{active === "myconnection" ? (
<div
className="menu_items myconnection" onClick={() => handleChange("myconnection")}
>
My Connections
</div>
) : (
<div className="menu_items"
onClick={() => handleChange("myconnection")}
>
My Connections
</div>
)}
{active === "addconnection" ? (
<div
className="menu_items addconnection" onClick={() => handleChange("addconnection")}
>
Add Connections
</div>
```

```
) : (
<div className="menu_items"
onClick={() => handleChange("addconnection")}
>
Add Connections
</div>
)}
{active === "requests" ? (
<div
className="menu_items requests" onClick={() => handleChange("requests")}
>
Requests
</div>
) : (
<div className="menu_items"
onClick={() => handleChange("requests")}
>
Requests
</div>
)}
{active === "Sent" ? (
<div
className="menu_items Sent" onClick={() => handleChange("Sent")}
>
Sent
</div>
) : (
<div className="menu_items"
onClick={() => handleChange("Sent")}
>
Sent
</div>
)}
</div>
<div className="result">
{active === "myconnection" && (
<MyConnections userData={userData} />
)}
{active === "addconnection" && (
<AddConnections userData={userData} />
)}
{active === "requests" && <Requests userData={userData} />}
{active === "Sent" && <Sent userData={userData} />}
</div>
</div>
</div>
);
}
export default Connections;
```

Profile Page

```
import React, { useContext, useState, useEffect } from 'react'; import PersonalDetailRight from './PersonalDetailRight'; import PersonalDetails from './PersonalDetails';
import Navbar from '../Headers/Navbar';
import { AuthContext } from "../../Context/AuthContext"; import { database } from "../../firebase";
import "./Profile.css"; function Profile() {
const { user } = useContext(AuthContext);const [userData, setUserData] = useState("");
useEffect(() => {
const unsub = database.users.doc(user.uid).onSnapshot((snapshot) => { setUserData(snapshot.data());
});
return () => { unsub();
```

```
};
}, [user]);
return (
<div>
<Navbar userData={userData} />
<div className="profile">
<PersonalDetails user1={user} userData={userData}/>
<PersonalDetailRight user={user}/>
</div>
</div>
)
}
export default Profile;
```

Headers

1. Headers Signin

```
import React, { useState } from "react"; import "./Headersignin.css";
import { auth } from "../../firebase";
import { useHistory } from "react-router-dom"; function Headersignin() {
const [username, setUserName] = useState(""); const [password, setPassword]
= useState(""); const history = useHistory(); const signin = (event) => {
event.preventDefault(); // prevent from refreshing if (username === "" || password === "") { alert("Username or Password is
missing"); return;} auth
.signInWithEmailAndPassword(username, password)
.then((auth) => { history.push("/home"); })
.catch((error) => { alert(error.message);
});
};
return (<div className="header">
<div className="header-input">
<input type="text" placeholder="E-mail" value={username}
onChange={(event) => setUserName(event.target.value)}
/>
<input type="password" placeholder="Passwor d" value={password}
onChange={(event) => setPassword(event.target.value)}
/>
<button onClick={signin}>Sign In</button>
</div>
</di v>
);}
export default Headersignin;
```

Navbar

```
import React, { useState } from "react"; import "./Navbar.css";
import { Avatar } from "@material-ui/core";
import ChaletIcon from "@mui/icons-material/Chalet";
import NotificationsActiveIcon from "@material-ui/icons/NotificationsActive"; import MessageIcon from "@mui/icons-material/Message";
import PeopleAltIcon from "@mui/icons-material/PeopleAlt"; import Menu from "@mui/material/Menu";
import MenuItem from "@mui/material/MenuItem"; import { useHistory } from "react-router-dom"; import { auth } from "../../firebase";
function Navbar({ userData }) {
const [anchorEl, setAnchorEl] = useState(null); const open = Boolean(anchorEl);
const history = useHistory();
const handleClick = (event) => { setAnchorEl(event.currentTarget);
};
```

```
const handleClose = () => { setAnchorEl(null);
};
const HomePage = () => { history.push("/home");
};
const Signout = () => { auth.signOut(); history.push("/");
};
const handleConnection = () => { history.push("/connections");
};
const handleProfile = () => { history.push("/profile"); handleClose();
};
const handleMyPost = () => { history.push("/myPost"); handleClose();
};
const handleChat = () => { history.push("/Chat");
};
return (
<div className="navbar">
<div className="navbar-header" onClick={HomePage}>
<h3 id="collegename1">Govind Ballabh Pant Engineering College</h3>
<h3 id="collegename2">G B P I E T</h3>
</div>
<div className="navbar-header">
<div className="navbar-container">
<ChaletIcon className="icon" onClick={HomePage} />
{/* <span>Home</span> */}
</div>
{/* <div className="navbar-container">
<NotificationsActiveIcon className="icon" />
<span>Notification</span>
</div> */}
<div className="navbar-container">
<MessageIcon className="icon" onClick={handleChat} />
{/* <span>Chat</span> */}
</div>
<div className="navbar-container">
<PeopleAltIcon className="icon" onClick={handleConnection} />
{/* <span>Connections</span> */}
</div>
<div className="navbar-container">
<Avatar alt={userData.username} src="/static/images/avatar/1.jpg" id="basic-button"
aria-controls={open ? "basic-menu" : undefined} aria-haspopup="true"
aria-expanded={open ? "true" : undefined} onClick={handleClick}
/>
<Menu
id="basic-menu" anchorEl={anchorEl} open={open} onClose={handleClose} MenuListProps={{
"aria-labelledby": "basic-button",
}}
>
<MenuItem onClick={handleProfile}>Profile</MenuItem>
<MenuItem onClick={handleMyPost}>My Post</MenuItem>
<MenuItem onClick={Signout}>Sign Out</MenuItem>
</Menu>
</div> </div> </div> ); } export default Navbar;
My Post
import React, { useContext, useState, useEffect } from "react"; import Navbar from "../Headers/Navbar";
import CloseIcon from "@material-ui/icons/Close"; import { AuthContext } from
```

"../../Context/AuthContext"; import { database } from "../../firebase";

import "./MyPost.css";

const MyPost = () => {

const { user } = useContext(AuthContext); const [userData, setUserData] = useState(""); const [posts, setPostData] = useState([]);

useEffect(() => {

const unsub = database.users.doc(user.uid).onSnapshot((snapshot) => { setUserData(snapshot.data());

});

return () => { unsub();

};

}, [user]);

useEffect(() => { database.posts.onSnapshot((snapshot) => { setPostData(

snapshot.docs.map((post) => ({ id: post.id,

post: post.data(),

}))

);

});

}, [user]); console.log(posts);

const handleClick = (postId) => {

if (window.confirm("Do you want to delete this post")) {

// Save it! database.posts.doc(postId).delete();

}

};

return (

<Navbar userData={userData} />

{posts.map(({ id, post }) => (

<CloseIcon onClick={() => handleClick(id)} />

{post.photo && }

{post.video && <video src={post.video} controls></video>}

{!post.photo && !post.video && {post.text} }

{post.video && post.text && {post.text} }

{post.photo && post.text && {post.text} }

))}

);

};

export default MyPost;

Chat

import React, { useContext, useState, useEffect } from "react"; import Navbar from "../Headers/Navbar";

import { AuthContext } from "../../Context/AuthContext"; import { Avatar } from "@material-ui/core";

import { database } from "../../firebase"; import firebase from "firebase/compat/app"; import "./Chat.css";

function Chat() {

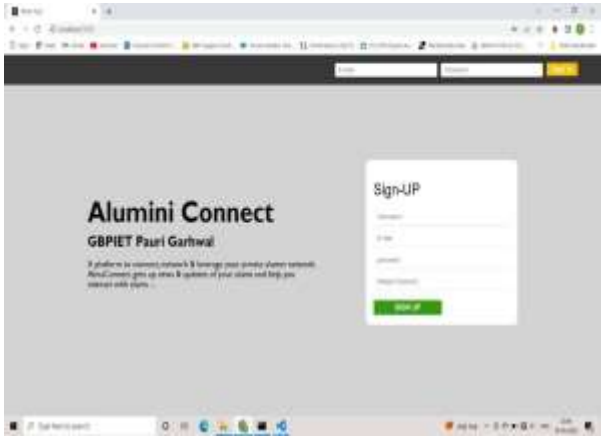
const { user } = useContext(AuthContext); const [userData, setUserData] = useState(""); const [userConn, setUserConn] = useState([]); const [messg, setMessg] = useState("");

```
const [usermessg, setUserMessg] = useState([]); const [curruser, setCurrentUser] = useState("");
useEffect(() => {
const unsub = database.users.doc(user.uid).onSnapshot((snapshot) => { setUserData(snapshot.data());
});
return () => { unsub();
};
}, [user]);
useEffect(() => { database.users
.doc(user.uid)
.collection("ConnectionUserIds")
.onSnapshot((snapshot) => {
// console.log(snapshot) setUserConn(snapshot.docs.map((doc) => doc.data()));
});
}, []);
const fetchChat = (data) => {
// console.log(data); setCurrentUser(data); database.users
.doc(user.uid)
.collection("ChatRoom")
.doc(data.id)
.collection("messages")
.orderBy("createdAt")
.onSnapshot((snapshot) => {
// console.log(snapshot) setUserMessg(snapshot.docs.map((doc) => doc.data()));
});
console.log(usermessg);
};
const sendMessg = (event) => { event.preventDefault();
// console.log("messg send");
// console.log(user.uid);
// console.log(curruser.id); database.users
.doc(user.uid)
.collection("ChatRoom")
.doc(curruser.id)
.collection("messages")
.add({
createdAt: firebase.firestore.FieldValue.serverTimestamp(), messg: messg,
user: user.uid,
});
database.users
.doc(curruser.id)
.collection("ChatRoom")
.doc(user.uid)
.collection("messages")
.add({
createdAt: firebase.firestore.FieldValue.serverTimestamp(), messg: messg,
user: user.uid,
});
setMessg("");
};
console.log(user.uid);
console.log(curruser.id);
return (
<div className="chat">
<Navbar userData={userData} />
<div className="chat_container">
<div className="chat_connection_container">
{userConn.map((data) => (
<div className="chat_conn"
onClick={() => fetchChat(data)} key={data.id}
>
```

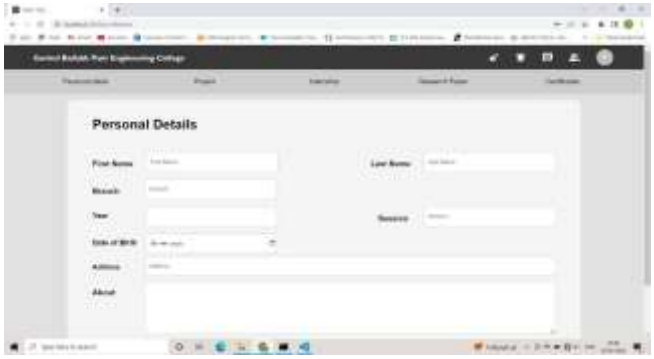
```
<div className="chat_conn_avatar">
<Avatar alt="shubham" src="" />
</div>
<h4
className="chat_conn_name">{data.username}</h4>
</div>
))}
</div>
<div className="chat_messg_container">
<div className="chat_messg">
{usermessg.map((data) => (
<div>
{data.user == user.uid ? (
<div className="my_messg">{data.messg}</div>
) : (
<div className="other_messg">{data.messg}</div>
)}
</div>
))}
</div>
<div className="chat_messg_input">
<form className="chat_messg_form">
<input
className="chat_input"
type="text"
placeholder="Type Message"
value={messg}
onChange={(e) => setMessg(e.target.value)}
/>
<button
className="chat_button"
disabled={!messg && !curruser}
type="submit"
onClick={sendMessg}
>
Send
</button>
</form>
</div>
</div>
</div>
);
}
export default Chat;
```

VIII. OUTPUT

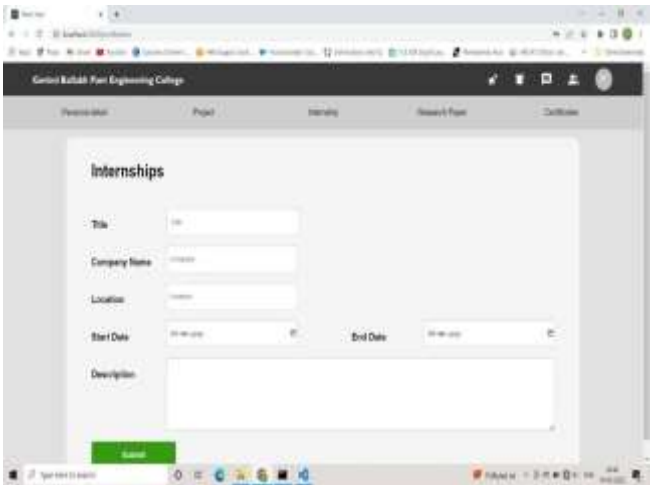
Sign Up and Login Page



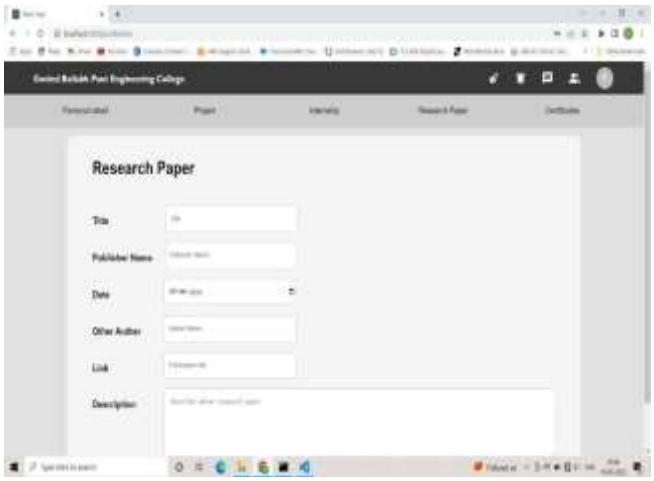
Personal Details Page



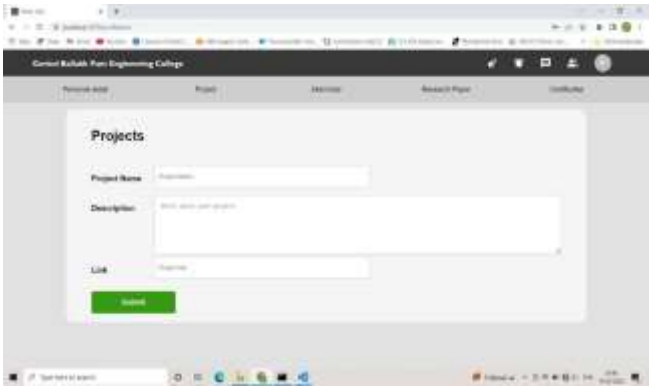
Internships Page



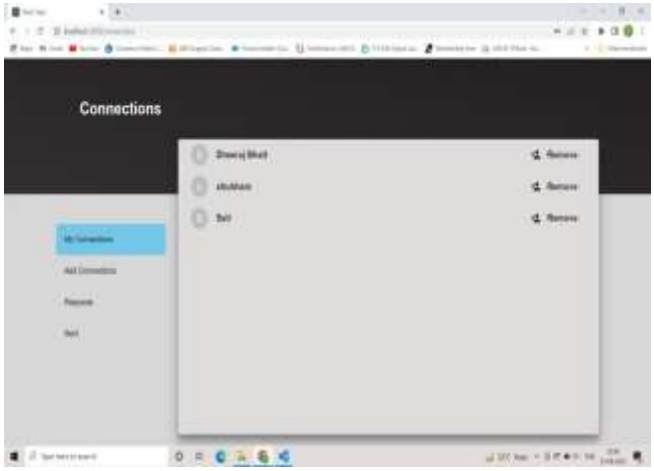
Projects Page



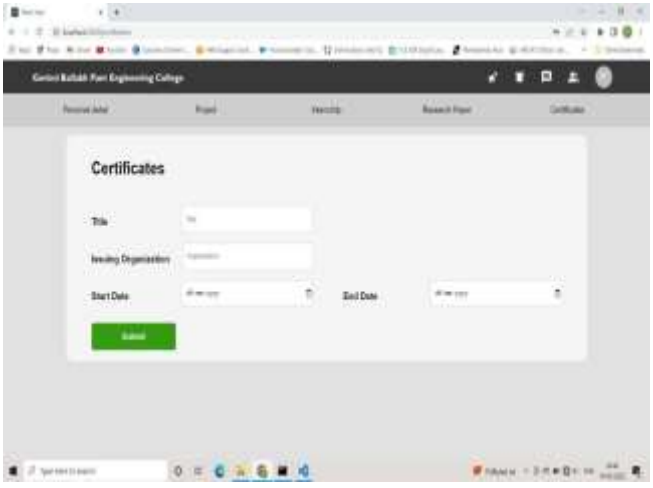
My Connections Page



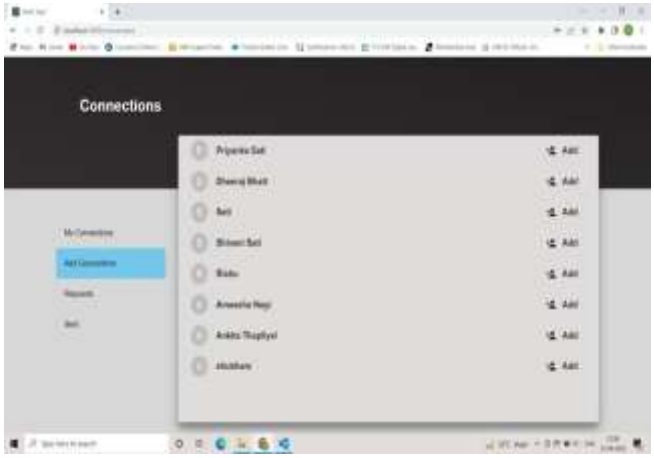
Certificates Page



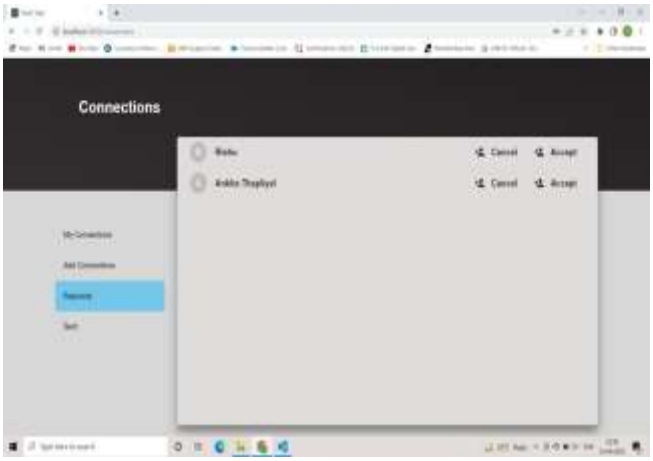
Add Connections Page



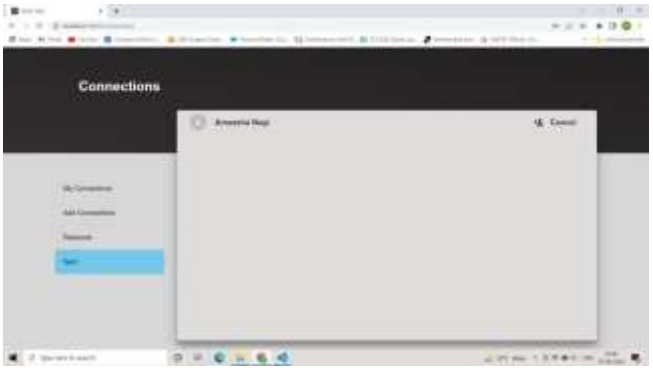
Research Paper Page



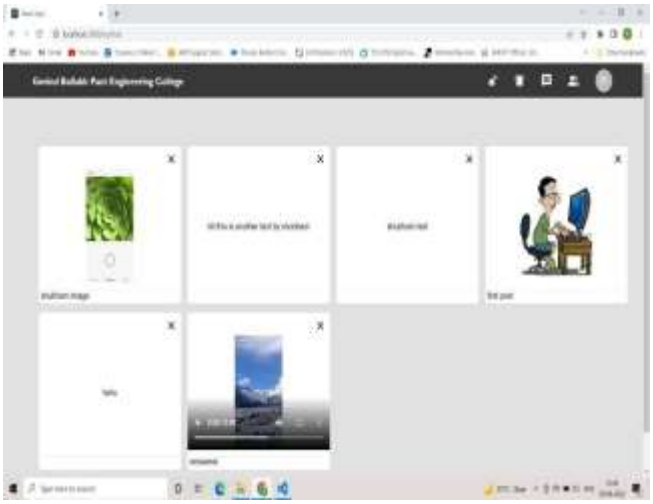
Received Requests Page



Sent Requuest Page



My Profile Page



IX. SOFTWARE TESTING

1. System Testing

System testing is an essential phase of the software development life cycle (SDLC) and plays a significant role in ensuring the quality, reliability, and functionality of the developed application. It involves a comprehensive evaluation of the complete software system to verify whether it meets the specified functional and non-functional requirements.

The primary objective of system testing is to identify errors, defects, and inconsistencies in the software and ensure that they are corrected before the application is deployed for actual use. It evaluates the overall performance of the system, including its individual modules, user interfaces, database operations, security mechanisms, and interactions between different components.

System testing is performed after the integration of all individual modules. It verifies whether the complete application functions according to the objectives and requirements defined during the system analysis and design phases. Both manual and automated testing techniques may be employed to examine the behaviour of the application under different conditions.

In the development of the proposed application, testing was carried out to verify the proper functioning of various modules, including user registration, login authentication, profile management, connections, messaging, community management, and post management.

The testing process involved preparing test cases with predefined inputs and expected outputs. The actual outputs generated by the application were then compared with the expected results to identify discrepancies. Various testing techniques, including Black Box Testing and White Box Testing, were considered to examine the functionality and internal logic of the application.

White Box Testing was used to examine the internal programming logic, control structures, conditional statements, and execution paths. Black Box Testing focused on verifying the application's functionality without considering its internal implementation.

The major objectives of system testing are:

- To verify that all functional requirements of the application are properly implemented.
- To identify and eliminate errors and defects in the software.
- To ensure proper communication and data exchange between different modules.
- To verify the correctness of database operations, including insertion, retrieval, updating, and deletion of records.
- To evaluate the performance and responsiveness of the application.
- To ensure that the application provides a consistent and user-friendly interface.
- To verify that the system behaves correctly under normal and exceptional conditions.

System testing helps improve the overall quality of the application and increases confidence in its reliability before deployment.

2. Test Data Implementation And Their Results

The quality and reliability of a software application depend largely on the effectiveness of its testing procedures and the extent to which it satisfies the predefined requirements. Testing is performed using different test cases, input values, and testing methodologies to evaluate the correctness and performance of the developed application.

Software testing involves examining the application in both standalone and integrated environments. Individual modules are initially tested independently, followed by testing of the complete application after integrating all the modules.

In the proposed application, different test cases were designed to verify the functionality of various features. These test cases included valid and invalid inputs, boundary conditions, authentication scenarios, database operations, and interactions between different modules.

The following testing methodologies were considered during the testing process:

- Flow Graph Testing
- Condition Testing
- Loop Testing
- Black Box Testing
- White Box Testing
- Unit Testing
- Integration Testing
- Validation Testing

3. Flow Graph Methodology

The Flow Graph methodology was used to represent the logical flow of the program during testing. A flow graph is a graphical representation of the control flow of a program. It consists of nodes and directed edges that represent the sequence of execution of program statements and the possible paths of execution.

Each node represents one or more procedural statements, while the directed edges indicate the flow of control between different nodes. Decision points are represented through branching paths.

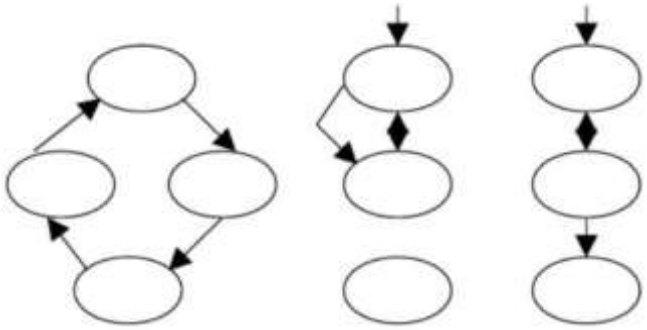


Fig. 7.1: Flow Graph

The flow graph helps identify different execution paths and provides a systematic approach to testing the program logic. It is also useful for identifying independent paths and determining the test cases required to achieve adequate code coverage.

4. Condition Testing

Condition Testing was performed to examine the logical conditions used in the application. It verifies whether individual conditions and decision statements produce the expected results under different input scenarios. Both true and false outcomes of conditional expressions were considered to ensure that the application responds correctly to different situations.

Loop Testing

Loop Testing was used to verify the correctness of iterative statements and ensure that loops execute properly under different conditions.

The following types of loops were considered:

1. **Simple Loops:** These loops were tested to verify their execution under minimum, normal, and maximum iteration conditions.

2. **Nested Loops:** Loops containing one or more loops within them were examined to verify their execution and termination.

3. **Concatenated Loops:** Sequential loops were tested to ensure that the execution of one loop did not adversely affect the execution of another.

4. **Unstructured Loops:** Irregular loop structures were examined to identify possible errors in program execution and termination.

3. Characteristics Of A Good Test Case

A good test case should possess the following characteristics:

- **High Probability of Detecting Errors:** A good test case should have a high probability of identifying errors or defects in the application.
- **Non-Redundancy:** Test cases should avoid unnecessary repetition and should examine different aspects of the application's functionality.
- **Best of Breed:** Each test case should be designed to identify a specific category of errors effectively.
- **Appropriate Complexity:** A test case should neither be too simple nor unnecessarily complicated. It should be designed to test the intended functionality effectively.

BLACK BOX TESTING

Black Box Testing is a software testing technique in which the functionality of an application is examined without considering its internal programming structure or implementation details. The testing process focuses primarily on the inputs provided to the system and the corresponding outputs generated.

The main objective of Black Box Testing is to verify whether the application performs according to the specified functional requirements. It helps identify missing functionality, incorrect outputs, interface-related errors, and improper handling of user inputs.

In this technique, test cases are designed based on the requirements and expected behaviour of the application. The tester provides different inputs and compares the actual output with the expected output to determine whether the system is functioning correctly.

The major characteristics of Black Box Testing are:

- Black Box Testing focuses on testing the software through its user interface and external functionality.
- It examines the fundamental features of the application without requiring knowledge of its internal code structure.
- Test cases are designed using functional requirements and expected system behaviour.
- Different input conditions are used to verify the correctness of the generated outputs.
- It helps identify missing functions, incorrect results, and errors in the interaction between the user and the application.

4. Black Box Testing Of The Proposed Application

Black Box Testing was considered for verifying the following aspects of the proposed application:

- User Authentication: Verification of login functionality using valid and invalid credentials.
- User Registration: Checking the registration process with valid and incomplete user information.
- Profile Management: Verification of profile creation, viewing, and updating functionality.
- Connections Management: Testing the functionality of sending, receiving, accepting, and rejecting connection requests.
- Messaging Functionality: Verification of sending and receiving messages between connected users.
- Post Management: Checking the functionality of creating, viewing, and deleting posts.
- Community Management: Verification of community creation, searching, joining, and viewing community-related information.
- Input Validation: Checking the handling of empty, invalid, and unexpected input values.

- Database Operations: Verification of data storage, retrieval, modification, and deletion.
- Navigation and Interface: Checking whether the application navigates correctly between different pages and modules.

Black Box Testing helps ensure that the application behaves as expected from the user's perspective and satisfies the specified functional requirements.

1. **White Box Testing**

White Box Testing, also known as Glass Box Testing or Structural Testing, is a software testing technique in which the internal structure, programming logic, and implementation of the application are examined.

Unlike Black Box Testing, White Box Testing requires knowledge of the source code and internal design of the software. It focuses on verifying the correctness of program statements, conditional expressions, loops, control flow, and data handling mechanisms.

The primary objective of White Box Testing is to ensure that the internal code structure works correctly and that different execution paths are properly tested.

The following objectives are considered in White Box Testing:

- To ensure that all independent execution paths within a module are exercised at least once.
- To verify that all logical decisions are tested for both true and false conditions.
- To execute loops at their boundaries and within their permissible execution limits.
- To examine internal data structures and ensure that data is processed correctly.
- To identify logical errors, incorrect conditions, and unnecessary code segments.
- To verify the proper handling of variables, parameters, and function arguments.

5. White Box Testing Of The Proposed Application

During White Box Testing, the internal implementation of different modules was examined with particular attention to the following aspects:

- Function Parameters and Arguments: Verification that the number, data types, and attributes of function parameters and arguments are consistent.
- Variable Declaration: Checking that variables are declared and initialized correctly before their use.
- Global Variables: Verification that global variables are defined and accessed consistently across different modules.
- Conditional Statements: Examination of conditional expressions to ensure that all possible decision outcomes are handled correctly.
- Loop Structures: Verification of loop initialization, termination conditions, and execution boundaries.
- Database Operations: Checking the correctness of database queries and operations used for storing, retrieving, updating, and deleting information.
- Data Validation: Ensuring that the application handles valid and invalid data appropriately.
- Input and Output Operations: Verification that input values are processed correctly and that the generated outputs match the expected results.
- Error Handling: Checking whether exceptions and unexpected conditions are handled appropriately.
- Code Execution: Ensuring that the program follows the intended execution sequence without logical errors.

White Box Testing helps improve the internal quality of the software by identifying programming errors and verifying the correctness of its implementation.

6. Unit Testing

Unit Testing is the process of testing individual components or modules of an application independently to verify that each unit performs its intended functionality correctly.It is generally performed during the coding phase of software development. Each module is tested separately to identify errors before it is integrated with other modules.In the proposed application, individual modules such as user registration, login authentication, profile management, connection management, messaging, and post management were examined independently.

The primary objectives of Unit Testing are:

- To verify the correctness of individual functions and components.
- To identify errors at an early stage of development.
- To ensure that each module produces the expected output for the given input.
- To simplify debugging and reduce the cost of error correction.
- To ensure that individual modules are ready for integration.

Unit Testing helps establish the correctness of individual components and provides a strong foundation for subsequent testing activities.

7. Integration Testing

Integration Testing is performed after the individual modules have been tested successfully. In this phase, different modules are combined and tested together to verify whether they interact and exchange information correctly.

The primary objective of Integration Testing is to identify errors associated with the interfaces and communication between different modules.

In the proposed application, integration testing focused on verifying the interaction between the following modules:

- **Login and Registration:** Verification that registered users can successfully authenticate and access the application.
- **User Profile and Database:** Checking whether user information is correctly retrieved and displayed from the database.
- **Connections and User Management:** Verification that connection requests are properly processed and updated.
- **Messaging and Connections:** Checking whether connected users can exchange messages successfully.
- **Posts and Database:** Verification that posts are correctly stored, retrieved, displayed, and deleted.
- **Community and User Management:** Checking whether users can search, join, and access community-related information.

Integration Testing ensures that the individual components work together as a complete system and that information is transferred correctly between different modules.

8. Validation Testing

Validation Testing is performed to ensure that the developed software satisfies the specified functional and user requirements. It determines whether the application performs the intended tasks and meets the expectations defined during the requirement analysis phase.

The main objective of Validation Testing is to verify that the right software has been developed according to the actual needs of the users.

In the proposed application, validation testing was carried out to examine whether the implemented features fulfilled the requirements of an alumni networking platform.

The following aspects were considered during Validation Testing:

- Verification that users can register and log in successfully.
- Verification that users can create and manage their profiles.
- Checking whether users can search for and establish connections with other users.
- Verification that connection requests can be sent, received, accepted, and rejected.
- Checking whether users can send and receive messages.
- Verification that users can create, view, and manage posts.
- Checking whether users can create, search, and join communities.
- Verification that the application provides proper navigation and a user-friendly interface.

The application was also subjected to functional evaluation to identify discrepancies between the expected and actual behaviour.

Validation Testing helps ensure that the developed application meets its intended objectives and provides the required functionality to its users.

Overall, the different testing techniques discussed in this chapter help evaluate the correctness, reliability, and functionality of the proposed application. They also help identify defects at different stages of development and improve the overall quality of the software before its deployment.

X. CHAPTER 8 CONCLUSION AND FUTURE SCOPE

Conclusion

The Development Of The Proposed Online Social Networking System (Alumni Connect) Has Been Undertaken With The Objective Of Providing An Effective And User-Friendly Platform For Connecting Alumni, Students, And Other Members Of The Institution. The Project Has Been Developed By Utilizing Appropriate Software Development Tools, Technologies, Programming Techniques, And Available Resources To Create A Functional Online Networking Application. The Primary Objective Of The System Is To Establish A Common Platform Where Users Can Interact, Share Information, Exchange Ideas, And Maintain Professional And Social Relationships. The Application Provides Various Features Such As User Registration And Login, Profile Management, Connection Management, Messaging, Post Sharing, And Community Interaction. These Features Help Users Remain Connected And Facilitate The Exchange Of Information Within The Alumni Network. During The Development Process, Special Attention Was Given To The Design And Implementation Of A Simple, Attractive, And User-Friendly Interface.

The Application Has Been Designed To Make Its Different Functionalities Easily Accessible So That Users Can Interact With The System Without Requiring Extensive Technical Knowledge. The System Also Provides Facilities For Managing Personal Profiles, Establishing Connections With Other Registered Users, Communicating Through Messages, And Sharing Posts. These Functionalities Contribute Towards Creating An Interactive Online Environment That Supports Communication, Collaboration, And Networking Among Members Of The Institution. The Project Has Provided Practical Experience In Software Development, Database Management, User Interface Design, Application Integration, And Software Testing.

The Development Process Also Helped In Understanding The Importance Of Requirement Analysis, System Design, Implementation, And Testing In The Successful Completion Of A Software Project. Although Considerable Efforts Have Been Made To Develop A Reliable And Functional Application, Certain Limitations Remain. Some Additional Features Could Not Be Implemented Due To Time Constraints And The Limited Scope Of The Project. However, The Developed System Provides A Foundation For Further Improvements And Enhancements. Overall, The Proposed Alumni Connect Application Successfully Demonstrates How An Online Social Networking Platform Can Be Utilized To Strengthen Communication And Interaction Among Students And Alumni. It Can Serve As A Useful Medium For Maintaining Institutional Relationships, Sharing Information, And Developing A Connected Alumni Community.

Limitations of the System

Despite the successful implementation of the major functionalities, the proposed system has certain limitations that can be addressed in future versions. The major limitations are as follows:

1. Administrative Module: A separate administration panel for managing users, monitoring activities, handling complaints, and controlling the overall functioning of the platform has not been fully implemented. Limited Functionalities: Some advanced social networking features, such as event management, online discussion forums, and notification customization, are not included in the current version. Security Enhancements: Additional security mechanisms, such as advanced user verification, multi-factor authentication, and enhanced access control, can be incorporated to improve system security. Mobile Application: The current system is primarily designed as a web-based application. A dedicated mobile application for Android and iOS platforms can further improve accessibility. Scalability: Further optimization may be required to support a significantly larger number of users, posts, messages, and concurrent activities. Advanced Search: The search functionality can

be enhanced to provide more refined search options based on graduation year, department, professional designation, location, and areas of interest. FUTURE SCOPE

The proposed Alumni Connect application has considerable potential for further development. Several additional features can be incorporated to make the system more comprehensive, secure, and efficient. The following enhancements may be considered in future versions:

2. Administration Panel: A separate administrative dashboard can be developed to enable authorized administrators to manage user accounts, verify registrations, monitor activities, manage communities, and maintain the overall system. Job and Placement Portal: A dedicated section can be introduced where alumni and organizations can share job opportunities, internships, and placement-related information with students and other alumni. Event Management: The system can be enhanced with an event management facility through which institutions and alumni associations can organize reunions, seminars, workshops, webinars, and other institutional activities. Advanced Notification System: Real-time notifications can be incorporated to inform users about new connection requests, messages, posts, community updates, and important announcements. Mobile Application Development: A dedicated mobile application can be developed to provide convenient access to the platform through smartphones and tablets. Enhanced Security: Advanced authentication mechanisms, email verification, password recovery, role-based access control, and other security features can be implemented to protect user information and prevent unauthorized access. Alumni Directory: A comprehensive alumni directory can be developed to enable users to search for former students based on their batch, department, specialization, organization, and geographical location. Discussion Forums: Online discussion forums can be introduced to encourage knowledge sharing, academic discussions, technical guidance, and professional collaboration among students and alumni. Performance Optimization: The application can be further optimized to improve loading speed, database performance, responsiveness, and its ability to handle a large number of users.

With these enhancements, the proposed application can evolve into a comprehensive institutional networking platform that strengthens the relationship between the institution, its students, and its alumni. It can also contribute towards professional networking, knowledge sharing, career development, and long-term institutional engagement.

REFERENCES

1. The following books, research papers, articles, and online resources were referred to during the study and development of the project.
2. Mishra, Prof. (2014). "Social Networking Sites Effect." The Times of India, January 14, 2014. Available at: http://articles.timesofindia.indiatimes.com/2013-06-23/india/40146190_1.html
3. Singh, Upender. (2014). "Popularity of SNSs among Indian Non-IT Students." Available at: http://ajeit.com/vol%202/Issue%207/HEIT14122013 01_32.pdf
4. Jain, Indira. "Social Networking Trends." The Times of India, New Delhi.
5. Matas, Alina. "SNSs Become an Online Opportunity for Employers and Students." The Washington Post, November 7, 1993, p. H2.
6. "Building a Career Path." The Washington Post, January 19, 1998.
7. Wikipedia. "Social Networking Service." Available at: http://en.wikipedia.org/wiki/SNSs_website (Accessed January 16, 2014).
8. MediaNama. "SNSs in India: The Make of Finding a Dream." Available at: http://www.medianama.com/2010/07/223-SNSs-in-india-the-make-of-finding-a-dream/ (Accessed February 21, 2014).
9. Fromm, E. (1969). Escape from Freedom. New York: Avon Books.
10. Joly, K. (2006). "Flickr, a Giant Real-Time Viewbook of College Life?" Available at: http://collegewebeditor.com/blog/index.php/archives/2006/09/08/flickr-a-giant-real-time-viewbook-of-college-life/ (Accessed April 6, 2010).
11. McKenna, K. Y. A., & Bargh, J. A. (2000). "Plan 9 from Cyberspace: The Implications of the Internet for

Personality and Social Psychology." Personality and Social Psychology Review, 4(1), 57–75.
12. Gao, H., Hu, C., & Jiang, T. (2015). "An Exploratory Study of Paper Sharing in Mendeley's Public Groups." In Proceedings of iConference 2015, March 24–27, Newport Beach, California.
13. Abdulhayoglu, M. A., & Thijs, B. (2017). "Use of ResearchGate and Google CSE for Author Name Disambiguation." Scientometrics, 111, 1965–1985. https://doi.org/10.1007/s11192-017-2341-y

14. Alheyasat, O. (2015). "Examination Expertise Sharing in Academic Social Networks Using Graphs: The Case of ResearchGate." Contemporary Engineering Sciences, 8, 137– https://doi.org/10.12988/ces.2015.515