Chapter 1. Introduction

The advancement in technology has brought a revolution in the way people communicate. With the advent of instant messaging and social media, people are now able to connect with each other from all over the world in real-time. However, the increasing use of these platforms has also raised concerns about privacy and security. To address these concerns, many developers have started to create secure and private messaging applications that allow users to communicate without worrying about the security of their messages.

One such application is the Peer-to-Peer Secure Chatting Application, which is a secure and private way for users to communicate with each other. The application is developed using Java Swing for the graphical user interface and MySQL for the database storage. The security of the communication between the peers is ensured through encryption of the messages using secure algorithms.

This project aims to provide a secure and private platform for users to communicate with each other without having to worry about the security of their messages. The user-friendly interface, developed using Java Swing, makes it easy for users to use the application, while the database storage, using MySQL, ensures that the chat history is persistently stored even if the user closes the application.

Furthermore, the application requires users to authenticate themselves before they can start chatting. The authentication is performed by checking the credentials stored in the MySQL database, which is protected using secure authentication methods to prevent unauthorized access.

Chapter 2. Significance of the Study

Addresses privacy and security concerns: The growing use of instant messaging and social media platforms has raised concerns about the privacy and security of online communications. The Peer-to-Peer Secure Chatting Application addresses these concerns by providing a secure and private platform for users to communicate with each other.

Improves communication security: By encrypting the messages being exchanged between the peers, the Peer-to-Peer Secure Chatting Application ensures that the communication between the peers is protected from eavesdropping and other forms of unauthorized access. This improves the security of the communication and provides users with peace of mind.

Enhances user experience: The user-friendly interface, developed using Java Swing, makes the application easy to use, while the database storage, using MySQL, ensures that the chat history is persistently stored even if the user closes the application. These features enhance the user experience and make the application more accessible to users of all skill levels.

Supports research and development: The Peer-to-Peer Secure Chatting Application is a unique project that combines the use of Java Swing, MySQL, and encryption algorithms. This project supports research and development in the field of secure communications and provides valuable insights into the design and implementation of secure messaging applications.

Fosters innovation: By creating a secure and private platform for users to communicate with each other, the Peer-to-Peer Secure Chatting Application fosters innovation in the field of online communications. The application provides a valuable solution for individuals and organizations who value privacy and security in their communications.

Chapter 3. Rationale

The rise of instant messaging and social media platforms has created a growing concern about privacy and security in online communications. With the increasing amount of sensitive information being shared online, it is essential to have a secure and private platform for users to communicate with each other. This is the rationale behind the development of the Peer-to-Peer Secure Chatting Application.

The peer-to-peer model of the application eliminates the risk of a third-party gaining access to the messages being exchanged between the peers. This is a significant advantage over centralized messaging platforms, where a central server could potentially be compromised.

The use of encryption algorithms to encrypt the messages being exchanged between the peers provides an extra layer of security to the application. This ensures that the communication between the peers is protected from eavesdropping and other forms of unauthorized access.

The user authentication requirement ensures that only authorized users can access the chat platform, adding an extra layer of security to the application. The use of a MySQL database to store the user information and chat history provides persistently stored chat history, allowing the users to refer back to their chat history at a later time.

The user-friendly interface, developed using Java Swing, makes the application easy to use, while the database storage, using MySQL, ensures that the chat history is persistently stored even if the user closes the application. These features enhance the user experience and make the application more accessible to users of all skill levels.

Chapter 4. Objective

- To provide a secure and private platform for users to communicate with each other.
- To ensure that the communication between the peers is protected from eavesdropping and other forms of unauthorized access.
- To enhance the user experience by providing a user-friendly interface and persistently stored chat history.
- To support research and development in the field of secure communications.
- To foster innovation in the field of online communications by providing a secure and private platform for users to communicate with each other.
- To implement the application using Java Swing and MySQL database technology.
- To develop and implement encryption algorithms to encrypt the messages being exchanged between the peers.
- To require user authentication to access the chat platform, adding an extra layer of security to the application.
- To evaluate the effectiveness of the application in terms of security, privacy, and user experience.

Chapter 5. Hypothesis

H1: The Peer-to-Peer Secure Chatting Application will provide a secure and private platform for users to communicate with each other.

H2: The Peer-to-Peer Chatting Application need at least one server for connect two computer for chatting.

Chapter 6. System Design and Implementation

❖ SOFTWARE DEVLOPEMENT LIFE CYCLE:

SDLC is a process followed for a software project, within a software organization. It consists of a detailed plan describing how to develop, maintain, replace and alter or enhance specific software. The life cycle defines a methodology for improving the quality of software and the overall development process. The following figure is a graphical representation of the various stages of a typical SDLC.

A typical Software Development Life Cycle consists of the following stages –

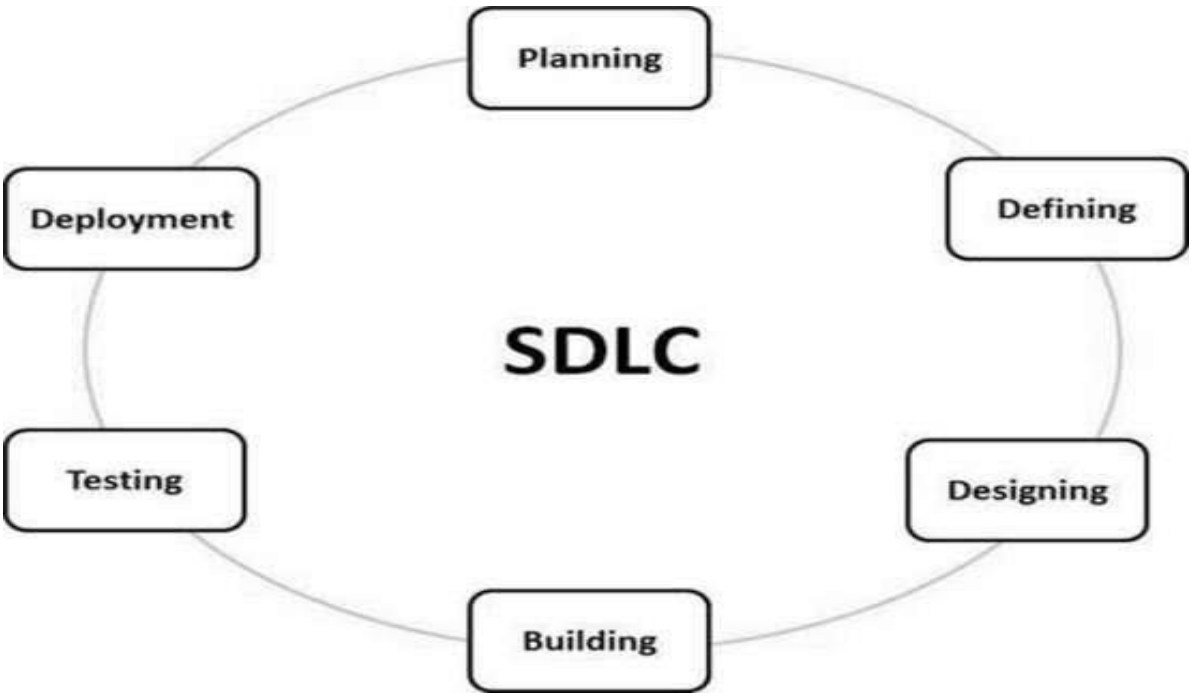


Fig: - software Development life cycle.

1. Planning Phase:

- Define project goals and objectives.
- Gather requirements from stakeholders.
- Identify resources required for the project, including hardware, software, and personnel.
- Create a project plan that outlines the scope, timeline, and budget of the project.

2. Analysis Phase:

- Analyze the requirements gathered from stakeholders.
- Identify any potential risks and constraints.
- Create use cases and user stories to capture the user's requirements.
- Create a functional specification that outlines the application's features and functionality.

3. Design Phase:

- Create a design specification that details the application's architecture, components, and modules.
- Create a user interface design that meets the user's needs and requirements.
- Identify the database design and schema that supports the application's data storage and retrieval requirements.
- Define the encryption and decryption algorithms to be used to secure the application's data.

4. Implementation Phase:

- Develop the application's code based on the design specification.
- Use version control tools to manage changes to the code.
- Test each component of the application to ensure they function as expected.
- Integrate the components to create the final application.

5. Testing Phase:

- Develop test cases and test scripts that verify the application's functionality and performance.
- Conduct unit testing to verify the functionality of each component.
- Conduct integration testing to verify that the components work together.

- Conduct system testing to verify that the application meets the user's requirements.
- Conduct acceptance testing to verify that the application meets the stakeholder's requirements.

6. Deployment Phase:

- Prepare the application for deployment by creating an installer or package.
- Deploy the application to the target environment.
- Verify that the application works as expected in the target environment.
- Train users on how to use the application.

7. Maintenance Phase:

- Monitor the application's performance in the target environment.
- Fix any bugs or issues that arise in the application.
- Make updates and enhancements to the application based on user feedback or changing requirements.
- Upgrade the application as needed to support new hardware or software platforms.

❖ SOFTWARE DEVLOPEMENT MODEL

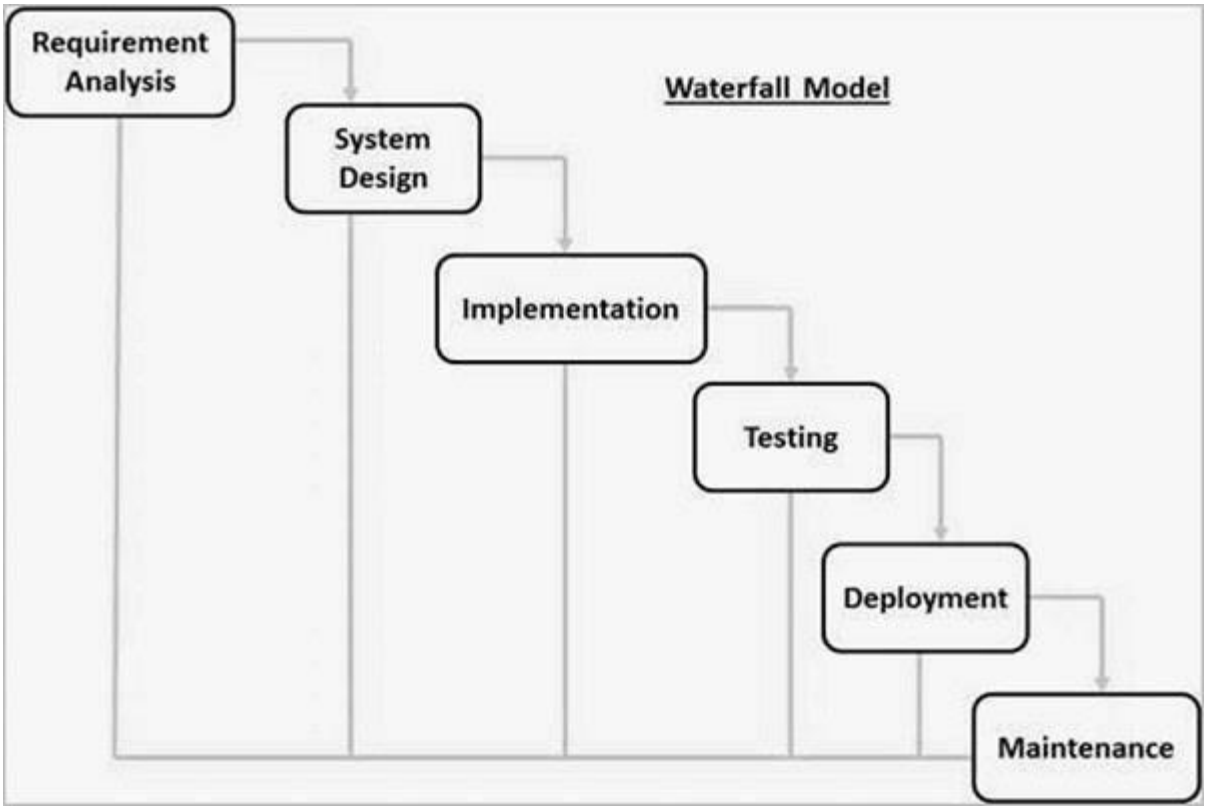
WATERFALL MODEL:

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a software development life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.The Waterfall model is the earliest SDLC approach that was used for software development.The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

WATERFALL MODEL

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the

whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially.The following illustration is a representation of the different phases of the Waterfall



The sequential phases in Waterfall model are −

• Requirement Gathering and analysis − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.

• System Design − the requirement specifications from first phase are studied in

this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall

system architecture.

• Implementation − with inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.

• Integration and Testing − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

• Deployment of system − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

• Maintenance − There are some issues which come up in the client environment. To fix those issues, patches are released. Also, to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

All these phases are cascaded to each other in which progress is seen as flowing steadily downwards (like a waterfall) through the phases. The next phase is started only after the defined set of goals are achieved for previous phase and is signed off, so the name "Waterfall Model". In this model, phases do not overlap.

Class Diagram

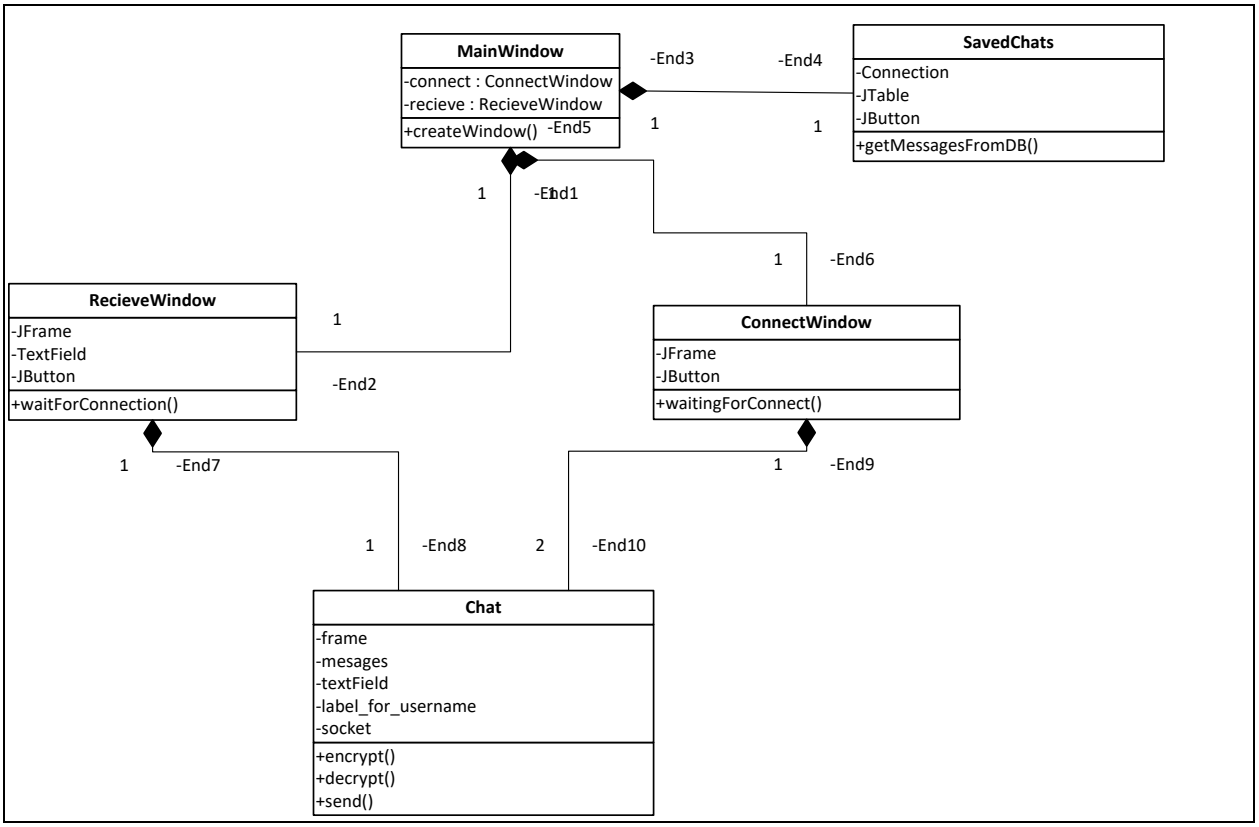


Figure: - Class Diagram

DFD Diagrams

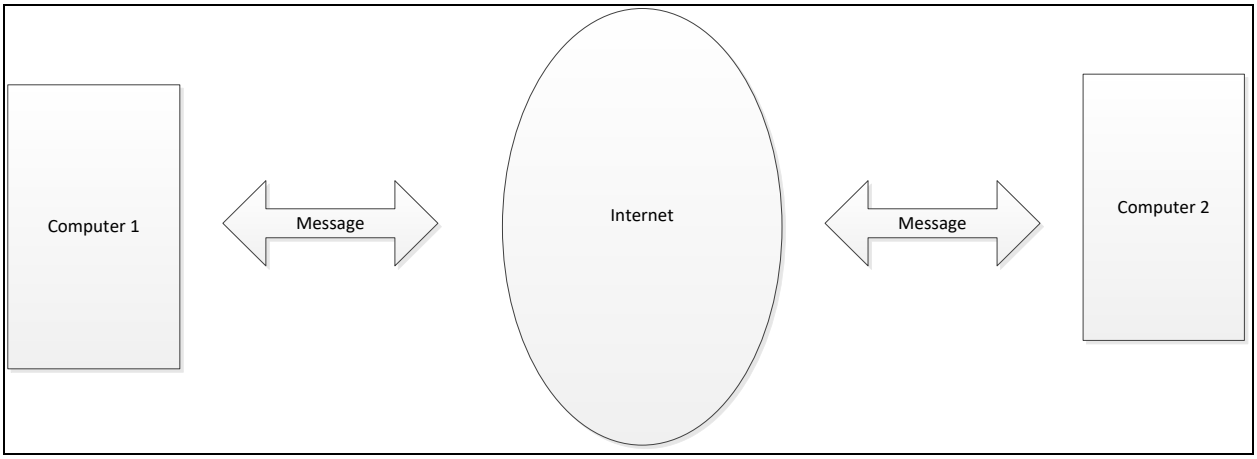


Figure: - 0 level DFD Diagram

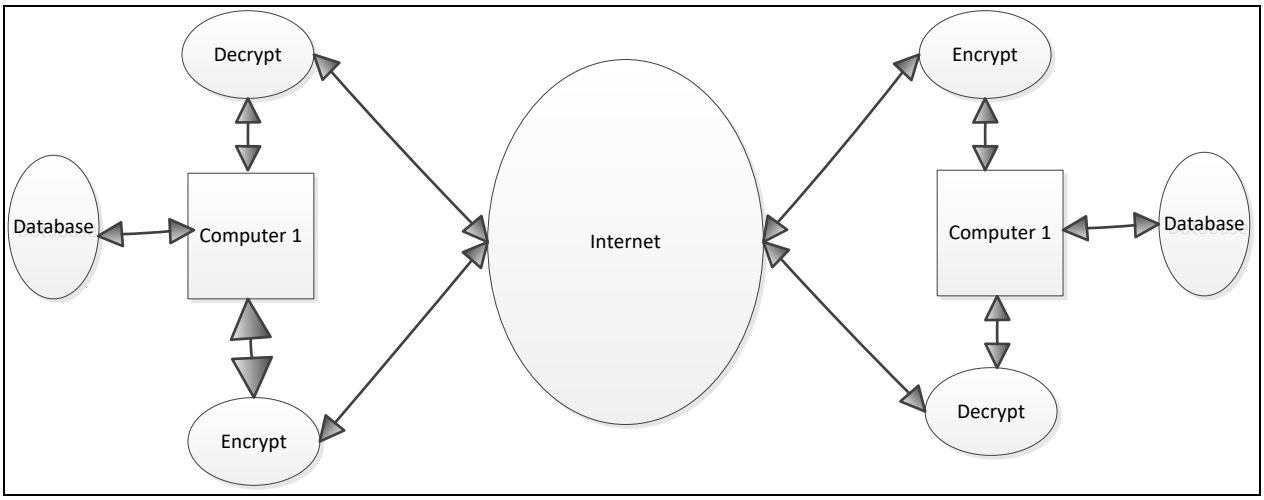


Figure: - 1 level DFD Diagram

Chapter 7. Software Requirement and Specification

❖ **Software Requirements:**

- Operating System: - Windows XP or newer version of Windows (Windows 7, 8, 10, 11), Linux (Need Java Installed)
- Front End: - Java Swing.
- Back End- Xampp, MY-SQL

❖ **Hardware Requirements:**

Minimum: 1.6GHZ CPU or more

1GB Ram or more

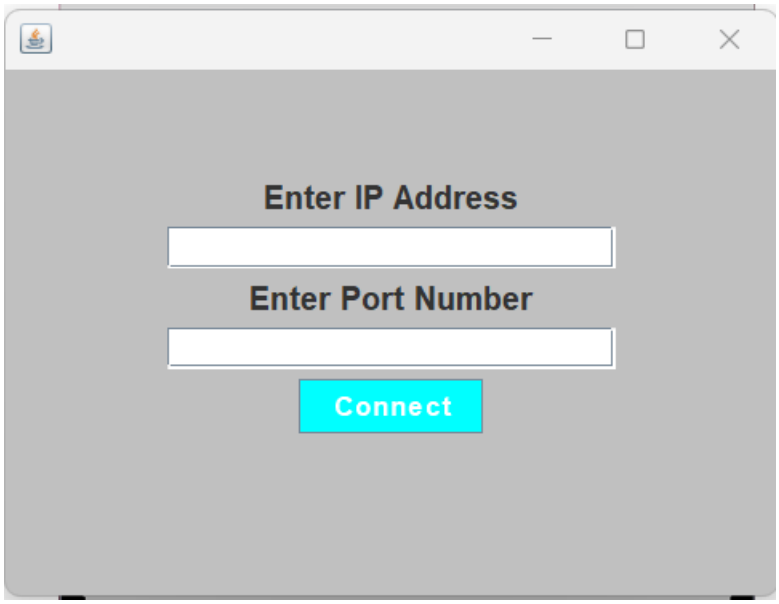
Monitor: 1024*768 Display

Hard Disk: 2Gb free space

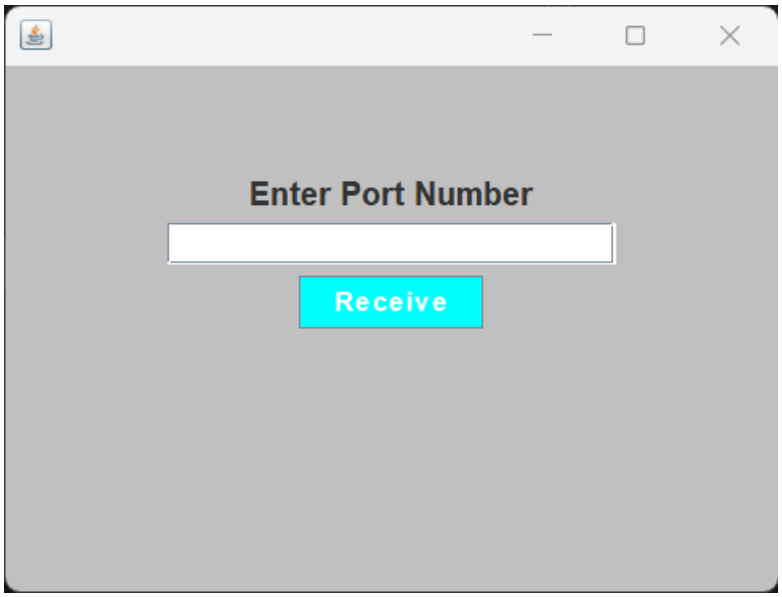
Chapter 8. Snapshots

1. Main Window:

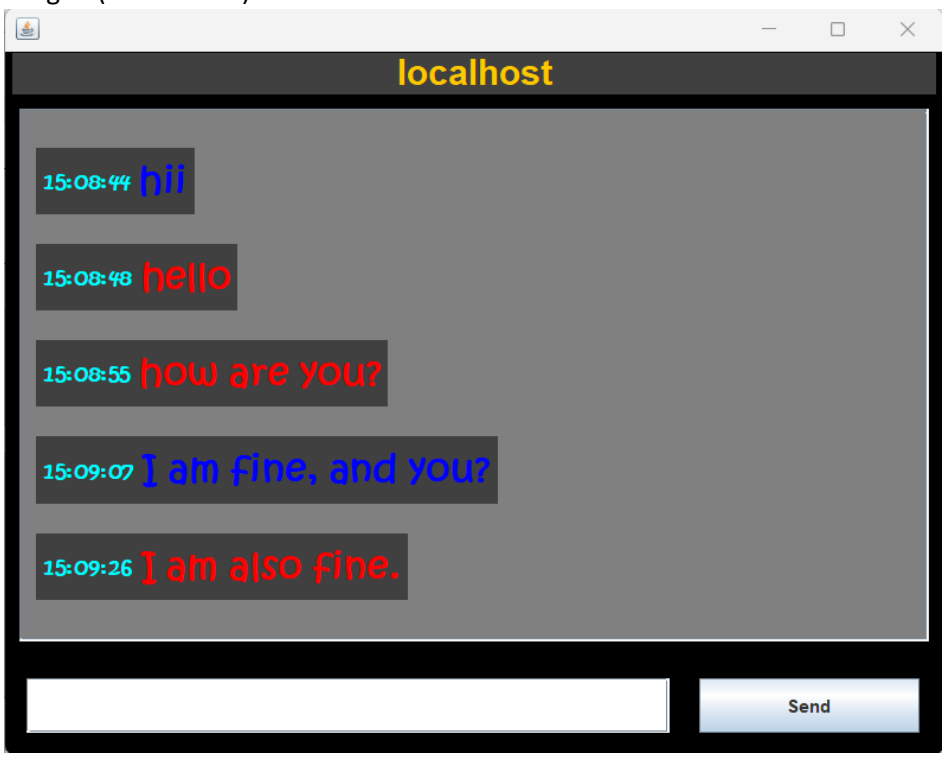
2. Connect:



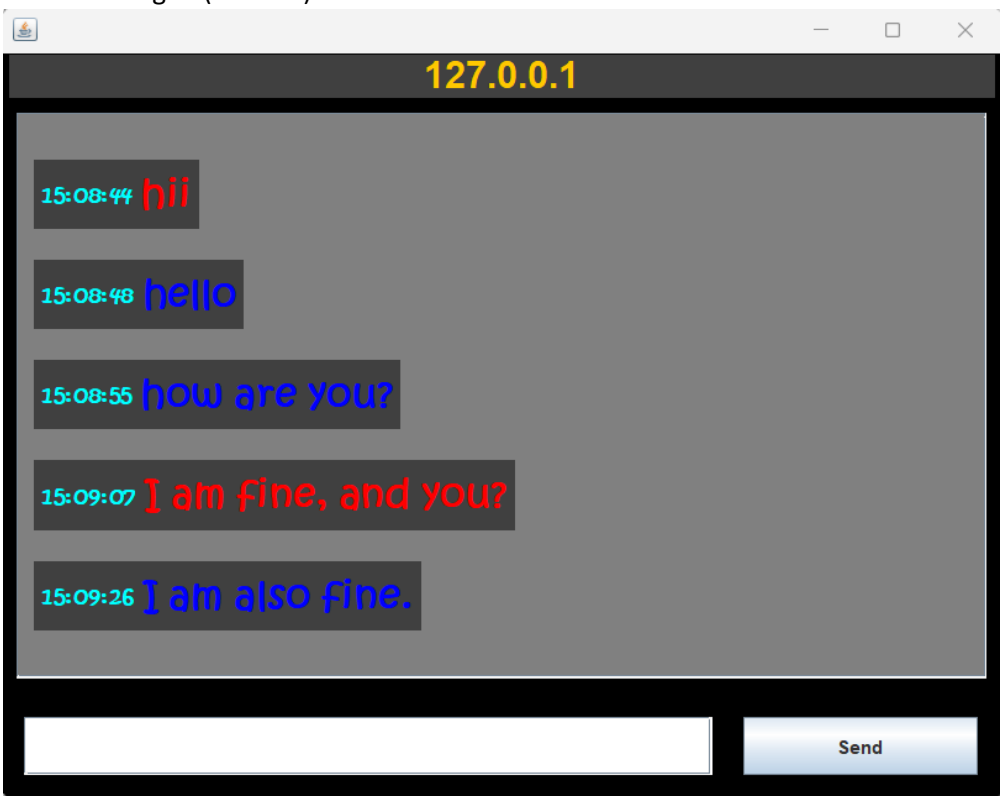
3. Receive:



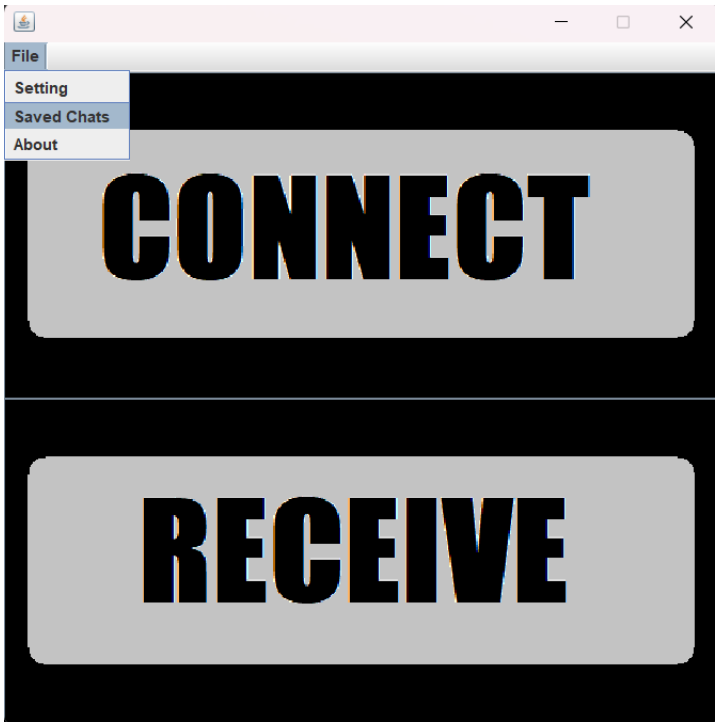
4. Start Chatting at (Connection):



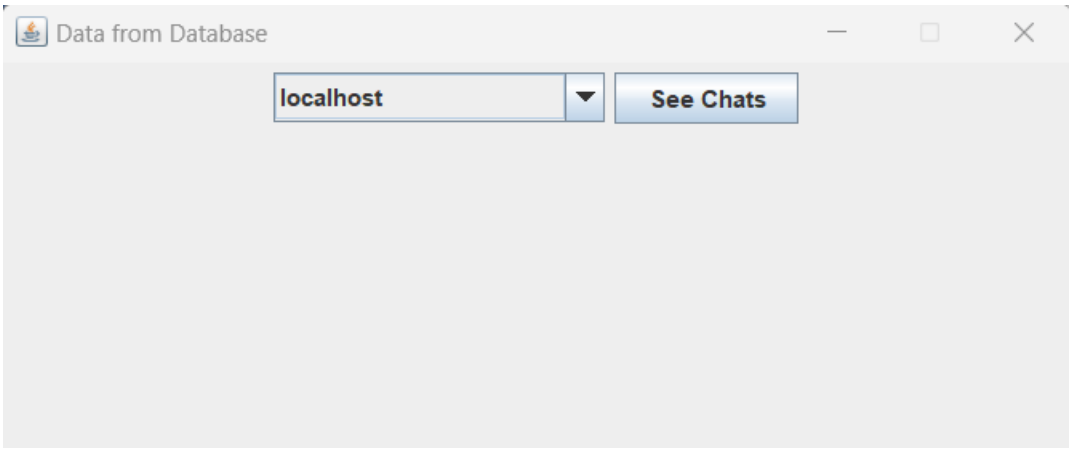
5. Start Chatting at (Receive):



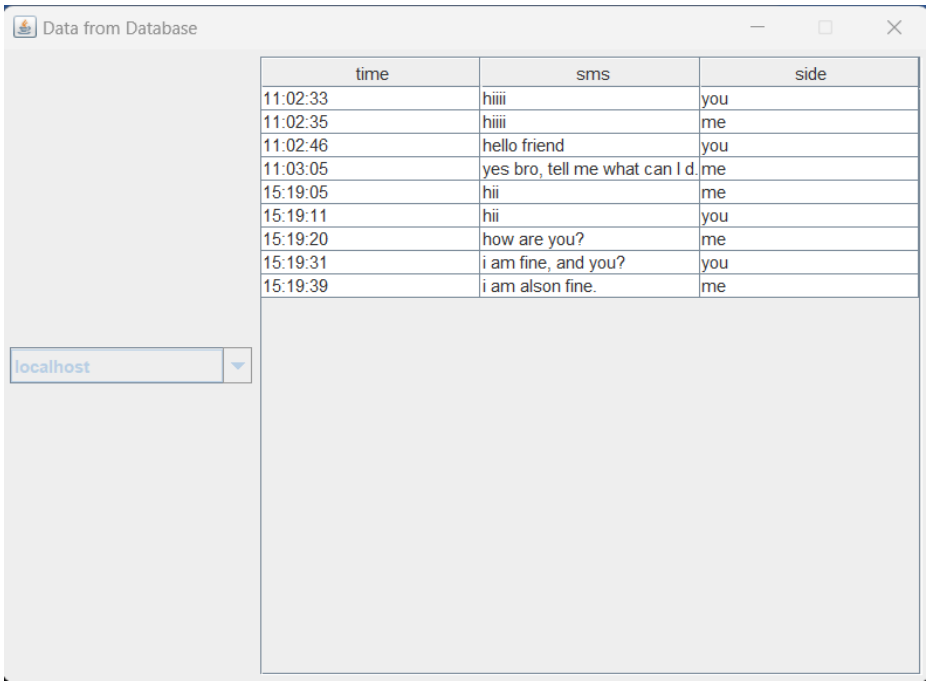
6. Menu Options:



7. Saved Chat Window:



8. Saved Chats in database:



Chapter 9. Test Cases

Sr. No	Case Name	Input	Expected Output	Result
1	Sending message	User types a message in the text field, selects a recipient from the list of available users, and clicks on the "Send" button.	The message is encrypted and sent to the recipient, and the recipient can decrypt and read the message	**Pass**
2	Receiving message	A message is sent from another user to the current user	The message is received and decrypted successfully, and the user can read the message	**Pass**
3	Network Disconnection	The network connection is lost while the user is sending or receiving a message	The application displays an error message indicating that the connection is lost, and the user cannot send or receive messages until the connection is restored	**Pass**
4	Database Corruption	The database is corrupted due to an error or malicious attack	The application detects the corruption and displays an error message, and the user cannot access their messages until the database is restored	**Pass**

Chapter 10. Advantages

1. Improved Security: The application offers improved security by using encryption algorithms to secure the communication between peers. This ensures that the messages exchanged are protected from eavesdropping and unauthorized access.

2. Data Privacy: The application ensures that users' data is kept private and confidential, reducing the risk of data breaches and cyber-attacks.

3. User-Friendly Interface: The user-friendly interface of the application makes it easy for users of all skill levels to use and access the features of the application.

4. Accessibility: The application can be accessed from any location, as long as there is an internet connection, enabling users to communicate with peers from anywhere in the world.

5. Real-time Communication: The application allows for real-time communication, making it ideal for users who require immediate responses to their messages.

6. Database Storage: The application stores chat history in a database, enabling users to access and refer to previous conversations easily.

7. Environmentally Friendly: The application is environmentally friendly, as it reduces the need for paper-based communication, such as letters and faxes.

Chapter 11. Disadvantages

1. Dependence on Internet Connection: The application requires a stable internet connection for communication to take place. If the internet connection is poor or interrupted, the communication between peers can be affected.

2. Limited Functionality: The application only offers basic chat functionalities, limiting its use to only text-based communication. This can be a disadvantage for users who require more advanced features such as file sharing and video calling.

3. User Limitations: The application is limited to only users who have created an account on the platform. This limitation can be a disadvantage for users who want to communicate with people who are not on the platform.

4. Database Management: The application's database requires careful management to ensure that it does not become cluttered with old and irrelevant chat messages. Failure to manage the database can affect the application's performance and result in slow response times.

5. Security Risks: While the application offers improved security, there is still a risk of cyber-attacks and data breaches. The application's security depends on the strength of the encryption algorithm and the user's password. If the encryption algorithm or password is compromised, the application's security can be compromised as well.

6. User Adoption: The success of the application depends on user adoption. If users are not willing to use the application, it can be challenging to convince them to switch to the platform.

Chapter 12. Application

The Peer-to-Peer Secure Chatting Application is a Java-based desktop application that allows users to send and receive encrypted messages securely. The application provides end-to-end encryption, which means that only the sender and the recipient can read the messages.

The application uses a client-server architecture, where each user acts as both a client and a server. The users can connect to each other directly, without the need for a centralized server. This architecture ensures that there is no single point of failure, and that the communication is more secure, since there is no middleman involved.

The application includes a user registration system, which allows users to create an account and log in to the application. Each user has a unique ID, username, password, and public-private key pair. The public key is shared with other users to encrypt messages, while the private key is kept secret and used to decrypt messages.

Once the users are logged in, they can start sending messages to each other. The messages are encrypted using the recipient's public key, and can only be decrypted using the recipient's private key. The application also uses a session key, which is generated randomly for each session, to encrypt and decrypt the messages. This ensures that even if someone intercepts the encrypted messages, they cannot decipher them without the session key.

The application also includes a database system, which stores user data and messages. The database is secured using encryption and access control mechanisms to ensure that only authorized users can access it.

Chapter 13. Bibliography

Following websites are referring to create this project reports.

- https://www.stackoverflow.com
- https://www.codeproject.com
- https://www.w3school.com
- https://www.youtube.com
- https://www.codeguru.com
- https://www.dreamincode.net
- https://www.javatpoint.com