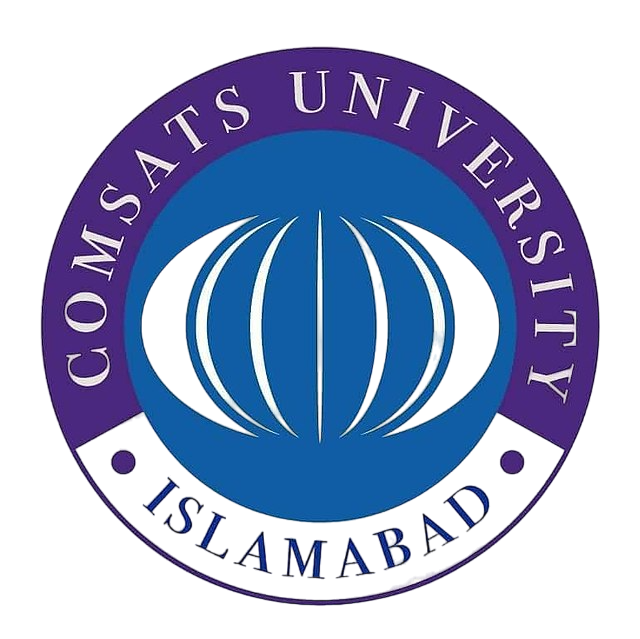
****

**Assignment No 2**

|  |  |
| --- | --- |
| **Course Title** | **Object Oriented Programming** |

**Submitted By**

|  |  |
| --- | --- |
| 1. **Sami Ur Rehman** | **SP24-BSE-086** |
| 1. **Muhammad Eman** | **SP24-BSE-071** |

|  |  |
| --- | --- |
| **Section** | **B** |

**Submitted To**

|  |  |
| --- | --- |
| **Moderator** | **Muhammad Shahid Bhatti** |

**Date: 11/10/2024**

Messaging App Project

**Classes:**

* **Sms:** Represents a single message with content, timestamp, sender, read status, and methods for formatting display.
* **Contact:** Represents a contact in the user's list with name, phone number, and a list of exchanged SMS messages. It includes methods for sorting messages and displaying contact information.
* **BlockList:** Similar to Contact but stores information about blocked contacts.
* **Gui:** This class defines the main application window with its components like buttons, text fields, chat area, etc. It handles user interactions, manages contacts, displays chat history, implements dark mode, and interacts with the server (if in online chat mode).

**Server (Server Class):**

* Starts a server socket on a specified port.
* When a connection is established, it creates a thread to handle communication with that client. (This functionality is not implemented yet in the provided code).
* Handles sending messages to the client (simulated in the Gui class for online chat).

**Gui Functionality:**

* Manages a list of contacts and blocked contacts.
* Allows adding, deleting, viewing, and blocking contacts.
* Offers options for viewing chat history and deleting messages within a specific contact.
* Implements dark mode functionality.
* In online chat mode, acts as a client by starting a server socket and waiting for a connection (simulated communication for demonstration purposes).

**Server Properties:**

* serverSocket: Represents the server socket that listens for incoming connections on a specific port.
* out: A PrintWriter object for sending messages to the client (once connected).
* port: The port number used for communication (set in constructor).
* contactId: Stores the contact ID chosen by the user for online chat (inherited from Gui).
* contact: Reference to the chosen contact object (inherited from Gui).
* contactIndex: Index of the chosen contact in the contact list (inherited from Gui).

**Server Methods:**

* Server(int port): Constructor that initializes the server with a specified port number.
* chatOnline(): Overrides the chatOnline method from the Gui class. It retrieves the contact ID from the user, finds the corresponding contact object, starts a server socket on the defined port, and starts a thread for handling incoming connections.
* run(): Implements the Runnable interface. This method is executed in a separate thread and waits for a client to connect, then establishes communication streams for sending/receiving messages. It updates the chat area with connection status and received messages.
* findContactIndex(String contactId): Finds the index of a contact in the contact list based on the provided ID.
* sendMessage(String message): Sends a message to the connected client using the out object and updates the user's chat window.
* preAddContacts(): (Inherits from Gui) Pre-populates the contact list with some sample contacts (for demonstration purposes).
* login(): (Inherits from Gui) Handles the user login process by prompting for username and password, validating them against predefined values, and displaying success/failure messages.

**Client Properties:**

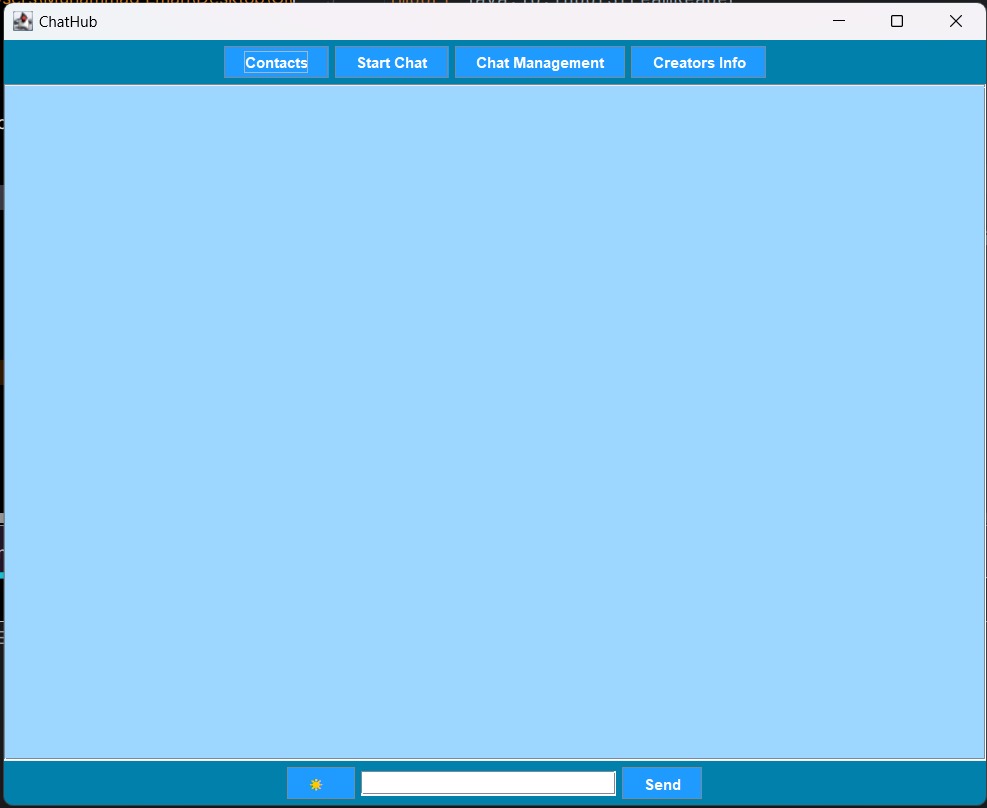
* socket: Represents the socket connection to the server.
* out: A PrintWriter object for sending messages to the server.
* serverAddress: Stores the server IP address (user input).
* port: The port number used for communication (set in constructor).
* contactId: Stores the contact ID chosen by the user for online chat.
* contact: Reference to the chosen contact object.
* contactIndex: Index of the chosen contact in the contact list.

**Client Methods:**

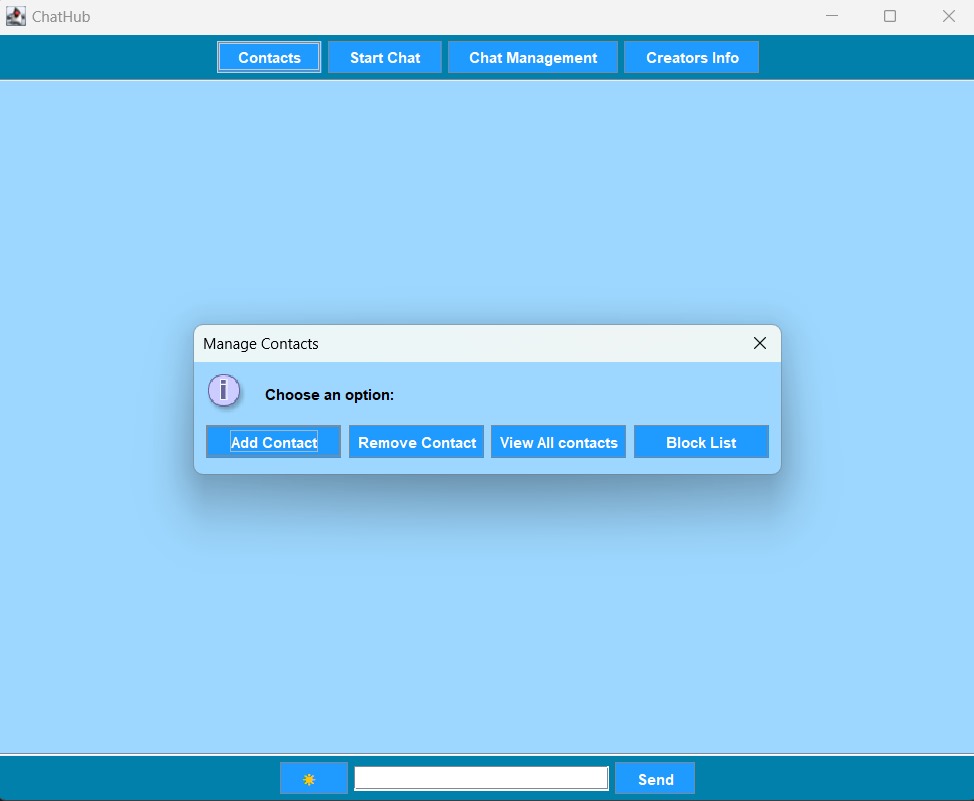
* Client(int port): Constructor that initializes the client with a specified port number.
* chatOnline(): Overrides the chatOnline method from the Gui class. It retrieves the contact ID from the user, finds the corresponding contact object, prompts for the server IP address, connects to the server, and starts a thread for receiving messages.
* run(): Implements the Runnable interface. This method is executed in a separate thread and continuously reads messages received from the server and updates the chat history.
* sendMessage(String message): Sends a message to the server using the out object and updates the user's chat window.
* preAddContacts(): (Inherits from Gui) Pre-populates the contact list with some sample contacts (for demonstration purposes).
* login(): (Inherits from Gui) Handles the user login process by prompting for username and password, validating them against predefined values, and displaying success/failure messages.

Outputs:

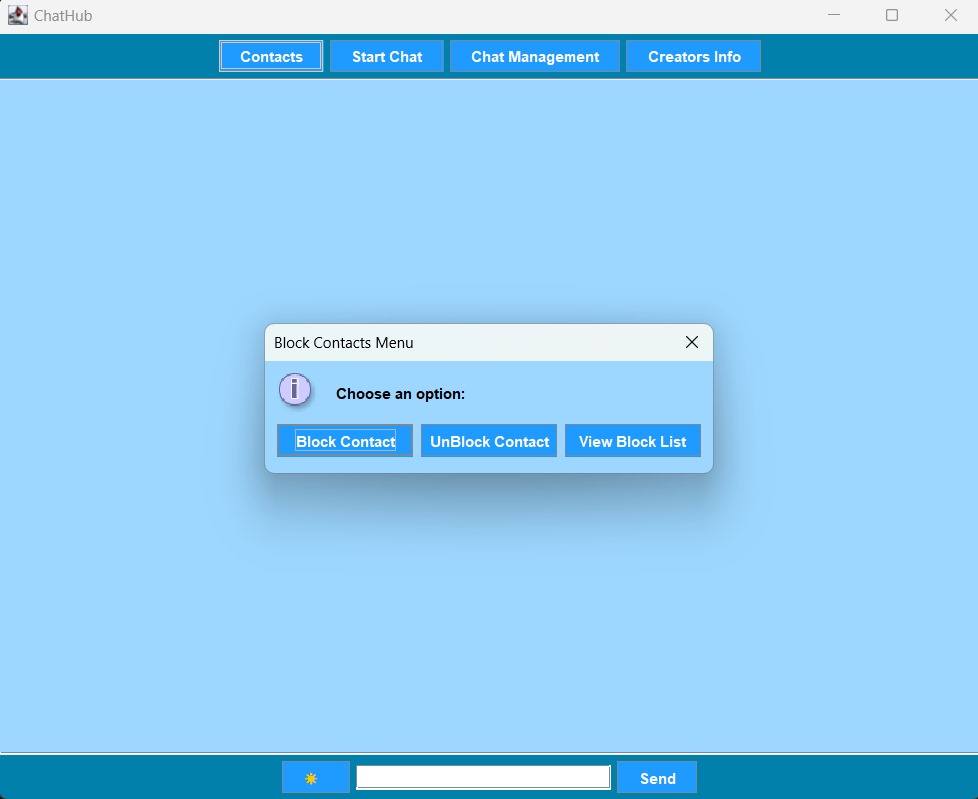
Interface Of Application:



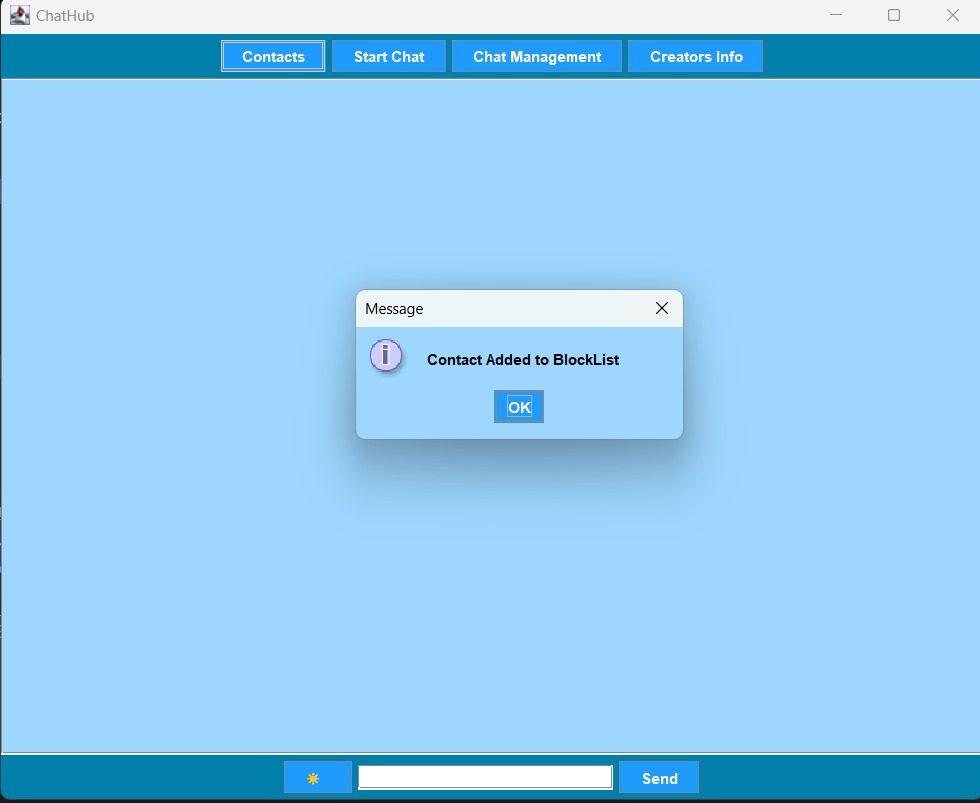
Manage Contacts:



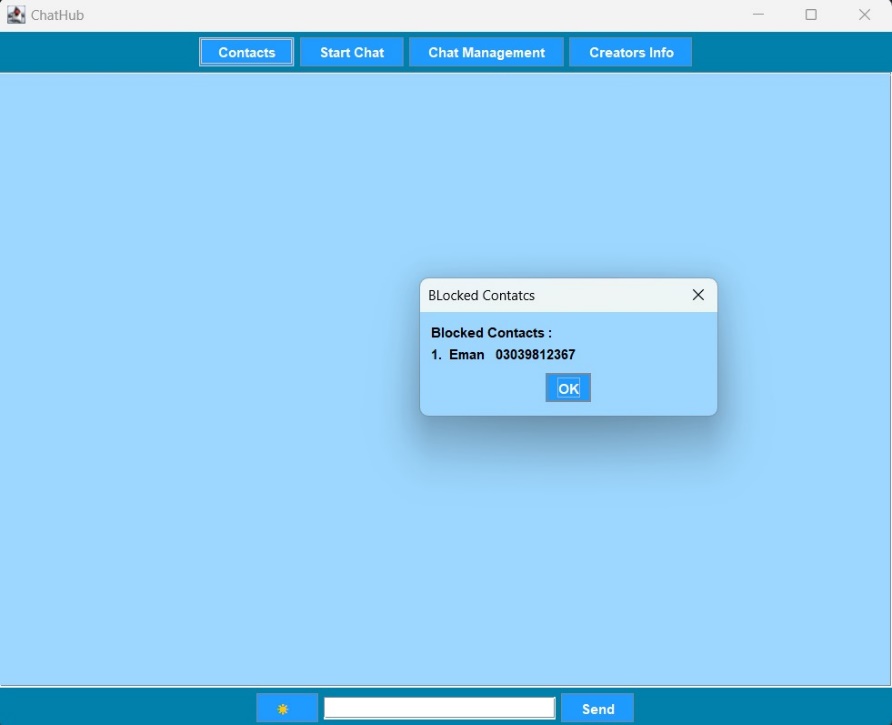
Block Menu:



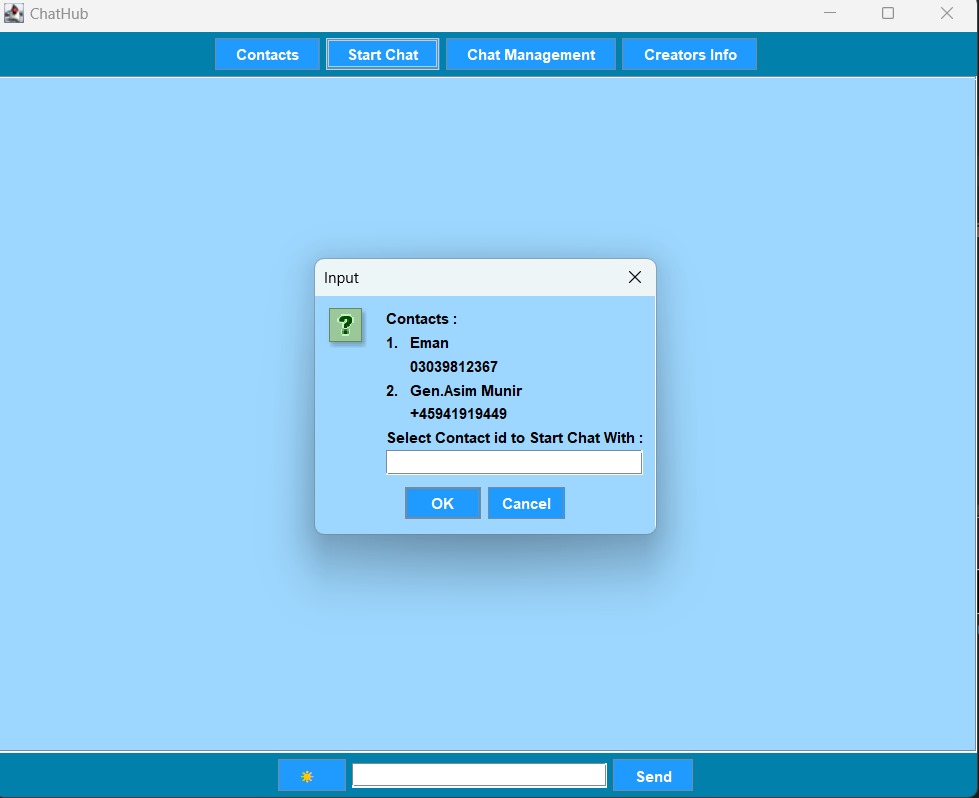
Add To BlockList:



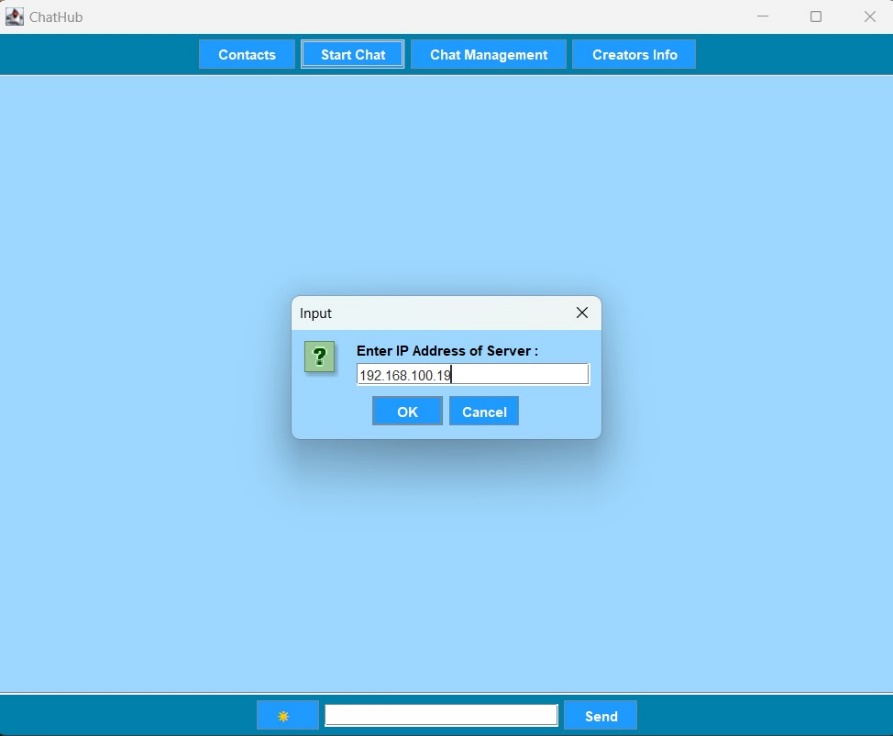
View BlockList:



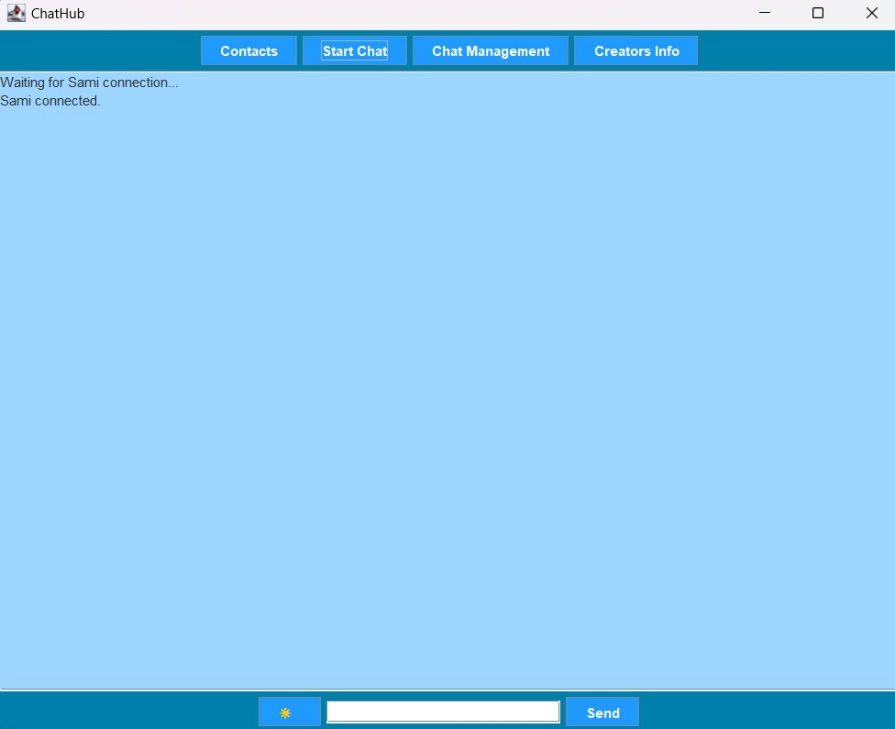
Start Chat:



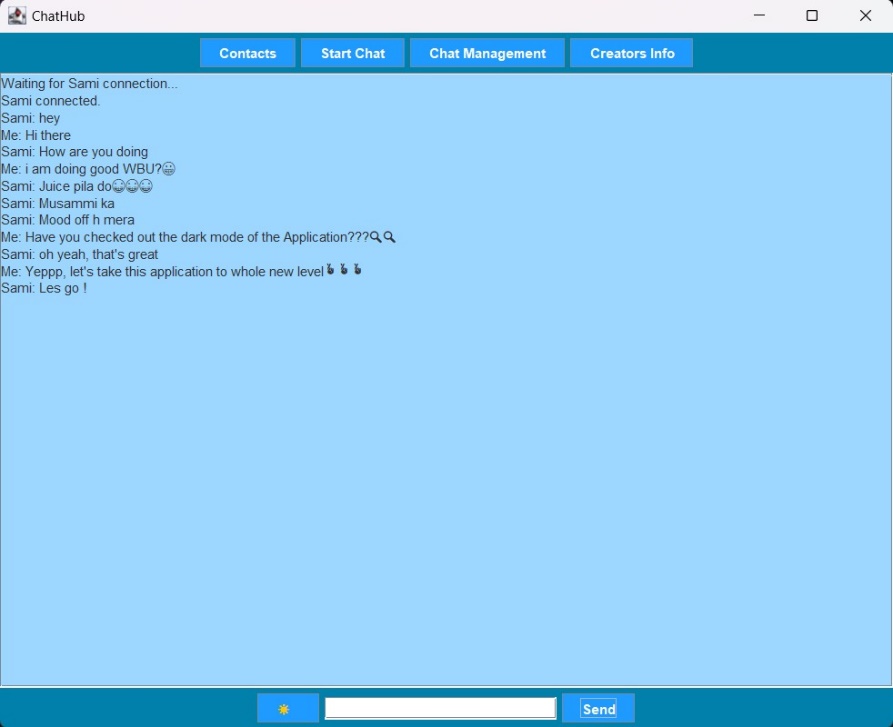
Entering IP Address to Start Chat:



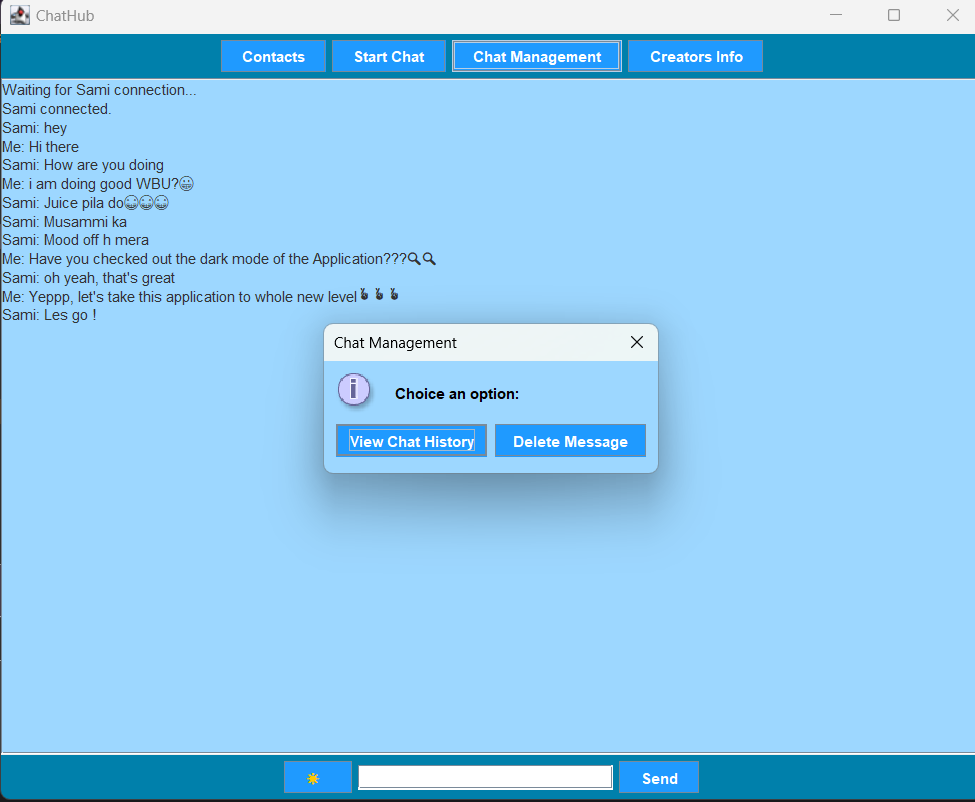
Building Connection:



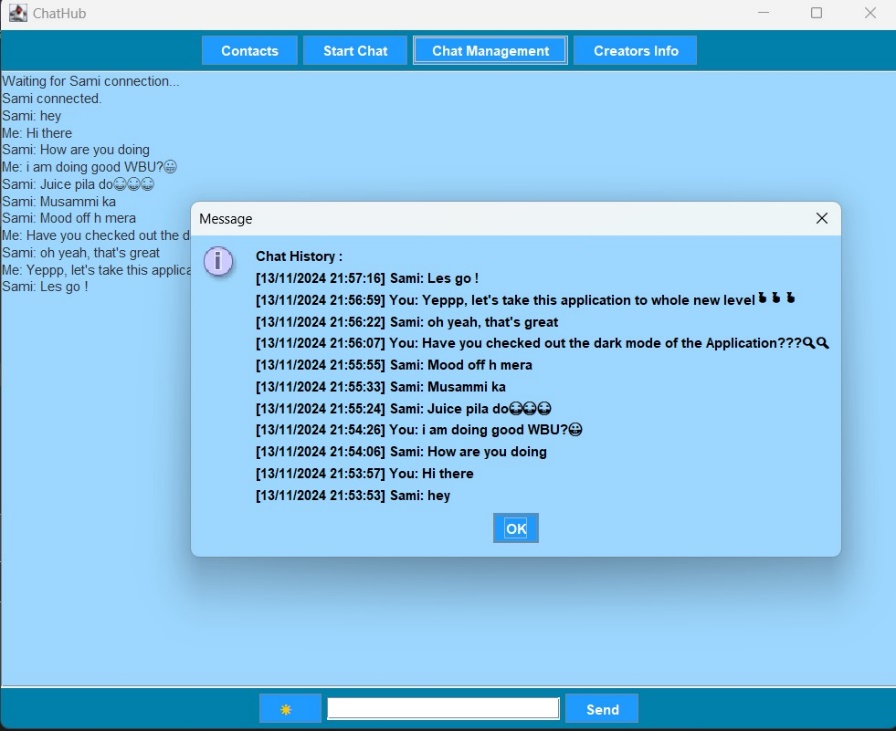
Chatting:



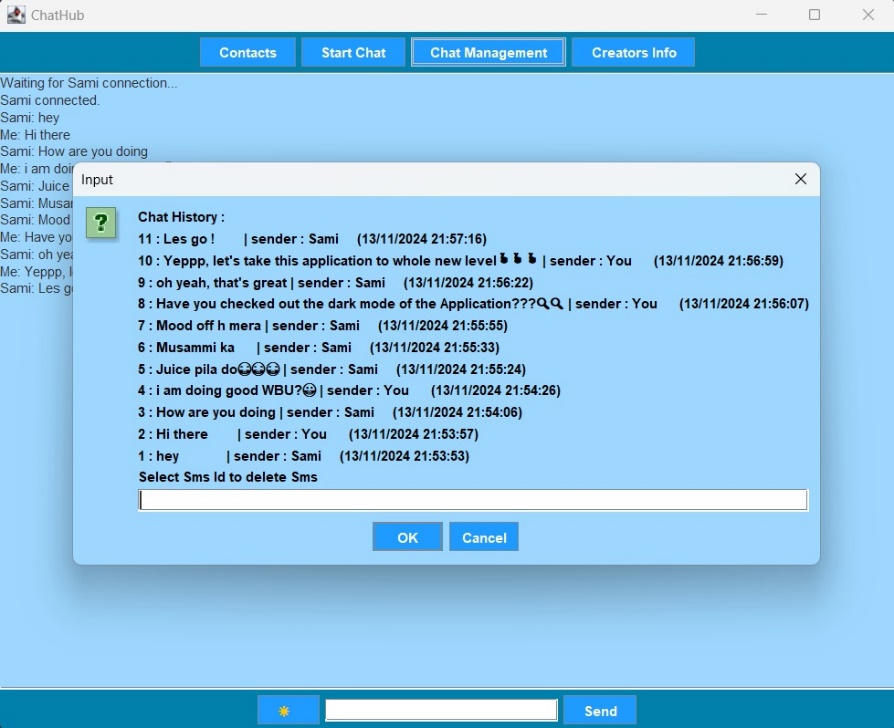
Chat Management:



View Chat History:



Delete Message:



Dark Mode:

