**Observer Pattern**

**Observer.java**

package chat.util;

public interface Observer {

void update(String message);

}

**Subject.java**

package chat.util;

import java.util.ArrayList;

import java.util.List;

public class Subject {

private List<Observer> observers = new ArrayList<>();

public void addObserver(Observer observer) {

observers.add(observer);

}

public void removeObserver(Observer observer) {

observers.remove(observer);

}

protected void notifyObservers(String message) {

for (Observer observer : observers) {

observer.update(message);

}

}

}

**Singleton Pattern**

**ChatRoomSingleton.java**

package chat.util;

import chat.model.ChatRoom;

import java.util.HashMap;

import java.util.Map;

public class ChatRoomSingleton {

private static ChatRoomSingleton instance;

private Map<String, ChatRoom> chatRooms;

private ChatRoomSingleton() {

chatRooms = new HashMap<>();

}

public static synchronized ChatRoomSingleton getInstance() {

if (instance == null) {

instance = new ChatRoomSingleton();

}

return instance;

}

public ChatRoom getChatRoom(String roomId) {

return chatRooms.get(roomId);

}

public void addChatRoom(ChatRoom chatRoom) {

chatRooms.put(chatRoom.getId(), chatRoom);

}

}

**Adapter Pattern**

**ChatRoomAdapter.java**

package chat.util;

import chat.model.ChatRoom;

public interface ChatRoomAdapter {

void sendMessage(ChatRoom chatRoom, String message);

}

**Model Classes**

**ChatRoom.java**

package chat.model;

import chat.util.Subject;

import java.util.ArrayList;

import java.util.List;

public class ChatRoom extends Subject {

private String id;

private List<String> messages;

private List<User> activeUsers;

public ChatRoom(String id) {

this.id = id;

this.messages = new ArrayList<>();

this.activeUsers = new ArrayList<>();

}

public String getId() {

return id;

}

public List<String> getMessages() {

return messages;

}

public void addMessage(String message) {

messages.add(message);

notifyObservers(message);

}

public List<User> getActiveUsers() {

return activeUsers;

}

public void addUser(User user) {

activeUsers.add(user);

}

public void removeUser(User user) {

activeUsers.remove(user);

}

}

**Message.java**

package chat.model;

public class Message {

private String sender;

private String content;

public Message(String sender, String content) {

this.sender = sender;

this.content = content;

}

public String getSender() {

return sender;

}

public String getContent() {

return content;

}

}

**User.java**

package chat.model;

import chat.util.Observer;

public class User implements Observer {

private String name;

public User(String name) {

this.name = name;

}

public String getName() {

return name;

}

@Override

public void update(String message) {

System.out.println(name + " received: " + message);

}

}

**Services**

**ChatRoomService.java**

package chat.service;

import chat.model.ChatRoom;

import chat.model.User;

import chat.util.ChatRoomSingleton;

public class ChatRoomService {

private ChatRoomSingleton chatRoomSingleton = ChatRoomSingleton.getInstance();

public ChatRoom createChatRoom(String roomId) {

ChatRoom chatRoom = new ChatRoom(roomId);

chatRoomSingleton.addChatRoom(chatRoom);

return chatRoom;

}

public ChatRoom joinChatRoom(String roomId, User user) {

ChatRoom chatRoom = chatRoomSingleton.getChatRoom(roomId);

if (chatRoom != null) {

chatRoom.addUser(user);

}

return chatRoom;

}

public void sendMessage(ChatRoom chatRoom, User user, String message) {

String formattedMessage = user.getName() + ": " + message;

chatRoom.addMessage(formattedMessage);

}

public void leaveChatRoom(ChatRoom chatRoom, User user) {

chatRoom.removeUser(user);

}

}

**Main Application**

**Main.java**

package chat;

import chat.model.ChatRoom;

import chat.model.User;

import chat.service.ChatRoomService;

import java.util.Scanner;

public class Main {

public static void main(String[] args) {

ChatRoomService chatRoomService = new ChatRoomService();

Scanner scanner = new Scanner(System.in);

System.out.println("Enter your username:");

String username = scanner.nextLine();

User user = new User(username);

System.out.println("Enter chat room ID to create/join:");

String roomId = scanner.nextLine();

ChatRoom chatRoom = chatRoomService.createChatRoom(roomId);

chatRoom.addObserver(user);

chatRoomService.joinChatRoom(roomId, user);

while (true) {

System.out.println("Enter message (type 'exit' to leave):");

String message = scanner.nextLine();

if (message.equalsIgnoreCase("exit")) {

chatRoomService.leaveChatRoom(chatRoom, user);

break;

}

chatRoomService.sendMessage(chatRoom, user, message);

}

scanner.close();

}

}