```java
import org.junit.jupiter.api.Test;
import java.util.ArrayList;
import java.util.List;


import static org.junit.jupiter.api.Assertions.*;

class ChatServerTest {

    @Test
    void getChatServerInstance() {
        ChatServer server1 = ChatServer.getChatServerInstance();
        ChatServer server2 = ChatServer.getChatServerInstance();

        assertEquals(server1, server2);
    }

    @Test
    void registerUser() {
        ChatServer chatServer = new ChatServer();

        User newUser = chatServer.registerUser("testUser");

        assertNotNull(newUser);

        assertEquals("testUser", newUser.getUsername());

        assertTrue(chatServer.getListOfUsers().contains(newUser));
    }

    @Test
    void addBlockedUser() {
        ChatServer chatServer = new ChatServer();
        User userToBlock = chatServer.registerUser("userToBlock");
        chatServer.addBlockedUser(userToBlock);
        assertTrue(chatServer.getListOfBlockedUsers().contains(userToBlock));
    }

    @Test
    void isSenderBlocked() {
        ChatServer chatServer = new ChatServer();
        User blockedUser = chatServer.registerUser("blockedUser");
        User senderUser = chatServer.registerUser("senderUser");

        chatServer.addBlockedUser(blockedUser);

        Message blockedMessage = new Message("blockedUser", "Hello", new User[]{senderUser}, "123");
        Message unblockedMessage = new Message("unblockedUser", "Hello", new User[]{senderUser}, "456");

        assertTrue(chatServer.isSenderBlocked(blockedMessage));
        assertFalse(chatServer.isSenderBlocked(unblockedMessage));
    }
    @Test
    void sendUndoRequest() {
        ChatServer chatServer = new ChatServer();
        User user1 = chatServer.registerUser("user1");
        User user2 = chatServer.registerUser("user2");
        User userRequesting = chatServer.registerUser("userRequesting");

        List<Message> messagesUser1 = new ArrayList<>();
        messagesUser1.add(new Message("user1", "Message 1", new User[]{user2}, "123"));
        messagesUser1.add(new Message("user1", "Message 2", new User[]{user2}, "124"));

        List<Message> messagesUser2 = new ArrayList<>();
        messagesUser2.add(new Message("user2", "Message 3", new User[]{user1}, "125"));
        messagesUser2.add(new Message("user2", "Message 4", new User[]{user1}, "126"));

        // Add some messages for each user
        for (Message message : messagesUser1) {
            user1.updatePercievedChat(message);
        }

        for (Message message : messagesUser2) {
            user2.updatePercievedChat(message);
        }

        // Send undo request for userRequesting
        chatServer.sendUndoRequest(userRequesting);

        // Verify that each user processed undo request for userRequesting
        assertFalse(user1.getPercievedChat().isEmpty());
        assertFalse(user2.getPercievedChat().isEmpty());
    }

}
```