2023-10-07

应用系统体系架构 — 作业3

学号：521030990006

姓名：VAHAGN GHAZARYAN

## **Programming plan: WebSocket message format design, client filtering method design**

## **Code snippets:**

## Back

## 

## 

## 

## I have used WebSocketServer and WebSocketConfig classes from WebSocketTransfer classwork, which is enough for backend to work. My Result listner calles WebSocketServer’s sendMessageToUser method giving user id of order and message that it’s successful.

## Front

## For every new received message I use antd’s message success to show user that order was successful.

## 

## When the login is successful I create w webSocket session and save it in userContext.

## 

## After logout I clean the user context and close the session.

## **DEMO**

## After login

## 

## 

## 

## 

## **Why choose a thread-safe collection type to maintain the client Session, and why the type I have chosen is threaded safe?**

## WebSockets handle multiple connections ***simultaneously***. This means multiple threads could be accessing and modifying the collection of Sessions ***concurrently***. Concurrency can lead to unpredictable behavior if not handled appropriately.

## For instance:

## While one thread is iterating through the map, another thread might modify it, leading to ConcurrentModificationException.

## Multiple threads might attempt to put a Session for the same user simultaneously.

## Such operations can result in data inconsistency, lost updates, or runtime exceptions if the collection being used isn't thread-safe.

## For that reason I used this Thread safe data structures.

## 

## **AtomicInteger** uses hardware-level features to ensure that its value is changed safely by only one thread at a time. **ConcurrentHashMap**, on the other hand, splits its data into segments, allowing multiple threads to read and write simultaneously without interfering with each other. Both are designed to handle concurrent operations without the need for extensive locking, making them thread-safe.