Distributed Computing Project 1

Ben Lenihan – T00200674

Contents

[1. Introduction 2](#_Toc99590308)

[2. Objectives 2](#_Toc99590309)

[3. Protocol Design 2](#_Toc99590310)

[4. Project Demonstration 2](#_Toc99590311)

[5. UML 2](#_Toc99590312)

[6. Conclusion 2](#_Toc99590313)

# Introduction

This project demonstrates the process of a client-server messaging protocol. This client utilises these functions:

* Login to server
* Send Message
* Download All Messages
* Logout of server

# Objectives

The objectives of the project are as follows:

* Design the protocols

The protocol design consists of the four previously mentioned functions: login, send message, download all messages and logout. First the design will describe the type of message. The messages will then be written out in pseudo code to show how it may work in the code itself.

* Document the protocol

The protocol will be documented through description of the workings of each protocol and how they are used. Sequence diagrams will be used to detail the interactions and communications between the client and server.

* Implement the protocol in the software

This software will allow users to login to the server with a username and password, send a message to the server, download all messages from the server and logout from the server. A graphical user interface will be utilized for the client-side interactions with server. The server will allow for concurrent access, will use SSL to secure all communication and will handle possible errors created from user input.

# Protocol Design

## Login to server

Message: Login

Description: The user logs in to the server using a username and password

Code: 100

Parameters: string uName, string pWord

Responses:

* Code: 101
  + Text: Logged in Successfully
  + Reason: Username was found, user was logged in.
* Code: 102
  + Text: Logged in Successfully, User Added
  + Reason: Username was not found and was added to the existing usernames, user was logged in.
* Code: 103
  + Text: Error. Check if Username or Password was blank
  + Reason: Username field or Password field was left empty.

Login is implemented through the login() function.

Steps:

* User enters username and password.
* User clicks login button.
* If the username does not already exist, it is added to the existing list of usernames and passwords.
* If the list does not exist, it is created and the username and password are then added.
* System returns appropriate response

Psuedo Code:

If username not in usernames

If usernames does not exist

Create usernames

Else

Add username to usernames

If username is in usernames

Return code 101

If username is not in usernames

Return code 102

Catch errors

Return code 103

## Login Sequence Diagram

Table

Description automatically generated

Figure 3.1 – Sequence diagram of Login

## Send Message

Message: Send Message

Description: The user can enter a message to the text field, the user can then click the send button to send the message to the server.

Code: 200

Parameters: string messageText.

Responses:

* Code: 201
  + Text: Message Sent.
  + Reason: Message sent successfully.
* Code 202
  + Text: Message Field Empty.
  + Reason: Message field blank.

Send Message is implemented through the sendMessage() function.

Steps:

* User enters message text.
* User clicks send message button
* System returns appropriate response
* Message is stored on server

Psuedo Code:

Input message

Try

Send message

Save message

Return code 201

Catch errors

Return code 202

Send Message Sequence Message

Graphical user interface, text, application, email

Description automatically generated

Figure 3.2 – Sequence diagram of send message

## Download All Messages

Message: Download All Messages

Description: The user clicks the download message button, all messages stored will be downloaded.

Code: 300

Parameters: ArrayList<strings> messages.

Responses:

* Code: 301
  + Text: Messages Downloaded.
  + Reason: Messages Downloaded from server

Send Message is implemented through the downloadMessages() function.

Steps:

* User clicks download messages button
* Messages are retrieved from server
* Messages are written to files

Pseudo Code:

If button is pressed

Return stored messages

Return code 301

Download All Messages Sequence Message

Graphical user interface, text, application, email

Description automatically generated

Figure 3.3 – Sequence diagram of Download All Messages

## Logout

Message: Logout

Description: The user clicks the logout button; user is returned to login window.

Code: 400

Parameters: N/A.

Responses:

* Code: 401
  + Text: Logged Out.
  + Reason: User successfully logged out.

Send Message is implemented through the sendMessage() function.

Steps:

* User clicks the logout button
* User is logged out.
* User is returned to login window

Pseudo Code:

If logout is clicked

Current user logged out

Login window opens

Logout Sequence Message

Graphical user interface, text, application, email

Description automatically generated

Figure 3.4 – Sequence Diagram of Logout

# Project Demonstration

Screenshots demonstrating the working parts of the project. Due to an error I am unable to show some of the components working.

Graphical user interface, text, application, email

Description automatically generated

Figure 4.1

Text

Description automatically generated

Figure 4.2

Graphical user interface, application

Description automatically generated

Figure 4.3

# UML

Graphical user interface

Description automatically generated

Figure 5

# Conclusion

In summary, this project creates a secure, concurrent, client-server, twitter-like messaging protocol using TCP. A GUI consisting of a login screen and a main screen are utilised. The login screen allows the user to enter a username and password. The main screen allows the user to send a message, download all messages and logout returning to the main screen. This document records the development of the project including the goals, design, a demonstration through screenshots of working components and a UML class diagram of the project