Computer Science 2B

Practical Assignment 06

2024-09-10

Deadline: 2024-09-17 12h00 Marks: 100

This practical assignment must be uploaded to eve.uj.ac.za **before** 2024-09-17 12h00. Late or incorrect submissions **will not be accepted**, and will therefore not be marked. You are **not allowed to collaborate** with any other student.

Make use of proper coding conventions and documentation. Marks will be deducted if these are not present. Your submission should include a batch file.

The reminder page includes details for submission and queries. Please ensure that **ALL** submissions follow the guidelines. The reminder page can be found on the last page of this practical - read the reminder page carefully.

This practical will focus on semester test revision

1. See below for a past practical semester test. [100]
   Please note:
   - The "ClientFrame" class element was relevant for Swing GUIs that were used in 2018. Instead of a "ClientFrame" you will create a "ClientPane".
   - You are required to create a **Java FX application** for the GUI component - your practical will be marked accordingly.
   - Make use of the **source files and additional files** provided to construct your project.

FACULTY OF SCIENCE

ACADEMY OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING	
MODULE	**COMPUTER SCIENCE 2B** CSC2B10
CAMPUS	AUCKLAND PARK CAMPUS (APK)
PRACTICAL TEST	A

DATE: 2018-10-02 **SESSION:** 14h00 - 17h00

LECTURER(S): **PROF. DT. VAN DER HAAR**

MODERATOR: **MR. A. MAGANLAL**

DURATION: 180 MINUTES **MARKS:** 100

Instructions

- Work on the T: drive.
- Save every 5 minutes.
- Submit every 10 minutes.
- The folder on the T: drive as well as the submitted zip file must be named as follows:

  `SURNAME_INITIALS_STUDENTNUMBER_CSC2B10_2018_PTA`

Make sure to save and submit your work regularly.

The University of Johannesburg has tasked you to develop a networked client application and server application. The application is a basic pdf file downloader which makes use of the BUKA protocol. The BUKA protocol requires clients to login in to the server before other commands can be processed. The server keeps track of the available pdf files by storing each pdf file and corresponding ID in a text file. BUKA runs on port 2018.

The following request commands are available in BUKA :

- **`AUTH <Name> <Password>`**
  Provide a name and password for client login, e.g.

  `AUTH` Drizzy p455w0rd. The server should validate the provided name and password. If the provided credentials are not valid, an error message must be returned. However if the login is successful, the server should allow the client to make other requests.

- **`LIST`**
  Return a list of available pdf files

  The server retrieves the available pdf files from the text file and return the list to the client.

- **`PDFRET <ID>`**
  Return the requested pdf file to the client

  The server should validate the ID and return an error message if the ID is not valid. If the ID is valid, a confirmation message should be sent including the file size. The File should then be read in and transmitted to the client

- **`LOGOUT`**
  Log a client off.

The following responses are used in BUKA :

- `200 <Message>` Successful command with `<Message>` providing a helpful message from the server.
- `500 <Message>` Unsuccessful command with `<Message>` providing a reason that the command did not work.

Complete the **`BUKAServer`** class. This class is responsible for binding to the BUKA port to listen for clients. The server must be able to handle multiple clients. Any clients which connect are handled by the **`BUKAHandler`**.

Complete the **`BUKAHandler`** class. This class is responsible for handling commands which are received from a client. This class is also responsible for handling client login. Registered users are stored in a text file called `users.txt`.

Complete the **BUKAClient** and **BUKAClientFrame** classes. The **BUKAClient** class will processes any communication using BUKA . The **BUKAClientFrame** class will have buttons for each command. The pdf files that are downloaded should be saved to disk.

Any errors from the server must be displayed to the client.

Reminder

Your submission must follow the naming convention.

`SURNAME_INITIALS_STUDENTNUMBER_CSC2B10_2018_PT`

Your submission must include the following folders:

- `bin` - Should be empty at submission but will contain runnable binaries when your submission is compiled.
- `docs` - Generated JavaDoc and any additional documentation files.
- `src` - Contains all relevant source code. Source code must be places in relevant sub-packages!
- `data` - Contains sub-folders for client and server where transferred files are saved.

Student #										**PC #**			
Surname										**Initials**			
ID Number													

Before final submission, read through and tick in the last column.		
1.	The full and final solution that I intend to submit has been uploaded to the correct locations as specified by the invigilators.	
2.	A zip file containing the full and final solution listed in point 1 has been uploaded to Eve.	
3.	I have **personally confirmed** the version of the full and final solution that has been saved to the backup media is the **correct copy** of the solution in point 1.	
Signature		

Official use only					
EVE		**CD**		**USB**	
Assistant signature		**Assistant initials**			

Official use only					
Marker signature			**Marker initials**		
Test total	100	**Mark**		**Moderation**	

See mark sheet on next page.

NB

Submissions which **do not compile** will be capped at 40%

Execution marks are awarded for a correctly functioning application and not for having some related code.

Failure to save the solution to the correct locations will mean that the Academy will not be able to mark the submission and you will forfeit marks as a result.

Mark sheet

1. **BUKAServer**
   (a) Create **ServerSocket**. ____ out of [02]
   (b) Accept client and pass to **BUKAHandler**. ____ out of [03]
   (c) Multi-threaded client handling. ____ out of [05]
2. **BUKAHandler**
   (a) Handle **AUTH** . ____ out of [05]
   (b) Handle **LIST** . ____ out of [05]
   (c) Handle **PDFRET**
      i. Process parameters. ____ out of [03]
      ii. Returning file size. ____ out of [05]
      iii. Returning pdf file . ____ out of [05]
   (d) Handle LOGOFF. ____ out of [02]
3. **BUKAClient**
   (a) Connect to **BUKAServer**. ____ out of [02]
   (b) Setup streams. ____ out of [02]
   (c) Send commands. ____ out of [02]
   (d) Process responses. ____ out of [04]
4. **BUKAClientFrame**
   (a) GUI Layout. ____ out of [02]
   (b) LOGIN button and listener. ____ out of [02]
   (c) LIST button and listener. ____ out of [02]
   (d) DOWNLOAD button and listener. ____ out of [02]
   (e) LOGOFF button and listener. ____ out of [02]
5. Coding convention (structure, layout, OO design) ____ out of [05]
6. Commenting (normal and JavaDoc commenting). ____ out of [05]
7. Correct execution
   (a) Show **BUKAClientFrame**. ____ out of [05]
   (b) Sending requests and responses. ____ out of [10]
   (c) Successfully sending and saving pdf files . ____ out of [10]
   (d) Handle exceptions. ____ out of [05]
   (e) Connection cleanup. ____ out of [05]