UNIVERSITY OF WOLVERHAMPTON

# Assignment Brief

| | |
|---|---|
| **Module code and title** | 5CS022 Distributed and Cloud Systems Programming |
| **Module leader** | Deepson Shrestha |
| **Assignment name** | Portfolio |
| **Assignment type** | Portfolio |
| **Assignment weighting** | 100% |
| **Assignment size** | |
| **Submission date** | Week 12 |
| **Submission method** | Canvas |
| **Assignment requirements** | |
| **Assessment criteria** | LO1 Adapt serial algorithms to operate in a distributed way.<br>LO2 Design, implement and test programs that operate in a distributed way.<br>LO3 Modify programs to run on and deploy on cloud infrastructure.<br>LO4 Evaluate cloud-based solutions with respect to performance, security and other pertinent factors. |
| **Additional instructions** | Submit your entire portfolio as a Zip file to Canvas. **Do not** submit the individual files separately |
| **Professional Body requirements** | |
| **University Regulations** | University's Academic Regulations: https://www.wlv.ac.uk/about-us/governance/legal-information/policies-and-regulations/academic-regulations/<br>Academic Integrity Policy: https://www.wlv.ac.uk/media/departments/office-of-the-vice-chancellor/documents/Academic-Integrity-Policy-from-2019-20.pdf<br>Performance descriptors: https://www.wlv.ac.uk/media/departments/office-of-the-vice-chancellor/documents/University-Performance-Descriptors.pdf |
| **Support** | Student Support and Wellbeing: https://www.wlv.ac.uk/current-students/student-support/student-support-and-wellbeing-ssw/<br>Study Guides: https://www.wlv.ac.uk/lib/skills-for-learning/study-guides/<br>Student Handbook: https://www.wlv.ac.uk/current-students/student-handbook/<br>Assessment information: http://www.wlv.ac.uk/assessment<br>You should also refer to your Course and Module Guides |

UNIVERSITY OF WOLVERHAMPTON

| | |
|---|---|
| **Date by which feedback will be provided** | 4 weeks after close of submission |
| **Method by which feedback will be provided** | Via Canvas. |

**Assessment**

This assessment is a Portfolio for 5CS022 Distributed and Cloud Systems Programming, which accounts for 100% of the module marks.

**Deadline**

A zip archive of all of your work (source code, binary executable, supporting documents) must be submitted, as one zip file, via Canvas on or before **Week 12**.

**Details**

**Coursework**

The coursework will consist of a number of questions that you will have to answer by writing a short research-based report and a number of tasks which you will have to carry out by creating a number of specified programs.

**Assignment Brief**

1. Write a research-based report on the following question: "How do Cloud systems and solutions compare with traditionally hosted systems (either in-house or out-sourced), in terms of performance, security, cost and reliability?" The report should be around 1000 words, and all sources should be correctly referenced with the Harvard referencing style. This report will contribute 20% of the marks to the Portfolio.

2. Create an MPI program that will calculate 4 statistical measures for a file of numbers. These are the total, the average, the largest positive number and the largest negative number. The MPI program must read the numbers in from the file in the Rank 0 process and send the appropriate share of numbers to the processes of other Ranks. These processes will then calculate the 4 statistical measures for the number that they have and then send those back to the Rank 0 process, which will consolidate all the values from all the Ranks into the final 4 values, and print them out. Your program must be able to work with a minimum of 2 processes and a maximum of 10 processes. The Rank 0 process should share the numbers between all the other Ranks as equally as possible. This task will contribute 20% of the marks to the Portfolio.

3. Using the Akka Actor framework, create a simulation of a bank account with multiple concurrent deposits and withdrawals, over a specified number of transactions.

   You should do the following:

- Create a BankAccount class as the Actor to hold the account balance, and to respond to deposit and withdrawal messages.
- Create a Deposit class to serve as the message sent to the BankAccount class to deposit money into the account.
- Create a Withdrawal class to serve as the message sent to the BankAccount class to withdraw money from the account.
- Create a Main class to serve as the simulation running program.
- When the Main class starts running, it should create an Actor of the BankAccount class.
- The BankAccount actor should initialise its balance to £100 on startup and print this balance to the console output.
- The Main program should then create 10 random values between -1000 to 1000.
- If a value is greater than zero, the Main program should create a Deposit message with that amount, and send it to the BankAccount actor, which would then add it to its current balance, and then print out the new balance on the console output.
- If a value is less than zero, the Main program should create a Withdrawal message with that amount, and send it to the BankAccount actor, which would then subtract it from its current balance, and then print out the new balance on the console output.
- After all 10 transactions have been processed, the program should terminate.

This task will contribute 20% of the marks to the Portfolio.

4. In a similar approach to Workshop on web hosting, create a static website on , consisting of 3 HTML web pages, and one image. The 3 web pages should be named “index.html”, “page1.html”, and “page2.html”. The image file should be named “image1.png” and should contain either a picture of a kitten or flowers. The “index.html” page should display the image in a <img> tag, similar to the workshop exercise. In addition, it should also contain a link to “page1.html” and a link to “page2.html”. The two files “page1.html” and “page2.html” should both contain a link back to “index.html”, and to each other. The image file “image1.png”, as well as “page1.html” and “page2.html”, should then be deployed to the CloudFront service, and all the links updated. The only non-CloudFront link should be only to “index.html”. All other links should point to the CloudFront distribution. Your submission should be a plain text file containing your name, student ID and a link to your static website on S3. This task will contribute 20% of the marks to the Portfolio.

5. In a similar approach to Workshop on Google Firestore, create a “movies” review HTML/JavaScript web app, based on a Google Firestore NoSQL database. The web app should access the movie review data store in a Google Firestore NoSQL database via JavaScript and present the information via HTML on the web browser. The web app should have the ability to add the movie name, a rating score from 0 to 5 (integers only), the director’s name and the release date. It should have the ability to be sorted in order on any of the fields. It should have the ability to edit and delete individual reviews. There is no requirement for any user authentication for the web app. This task will contribute 20% of the marks to the Portfolio.