COEN 346 OPERATING SYSTEMS – FALL 2023

Programming Assignment #2 : A Web Server for Bank Transfers

Due date: Nov 26 2023

In this project, you will be developing a concurrent web server to handle transfers between bank accounts. We are providing a simple Java Server that operates with a single thread. Your task is to enhance this server by making it multithreaded and adding functionality to handle fund transfers between accounts. You will also need to identify and address potential synchronization problems and deadlock scenarios.

Background on Web Servers

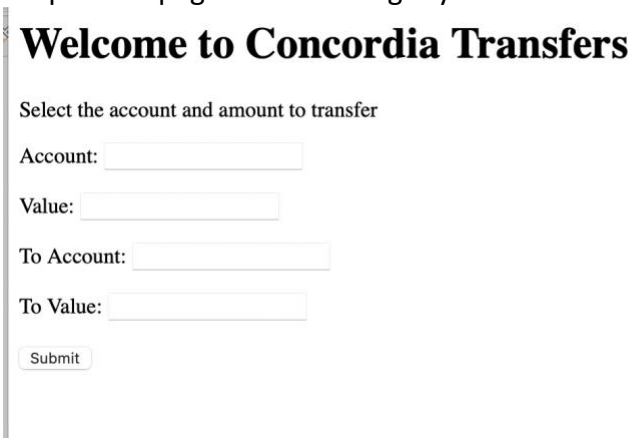
Web servers are software programs that handle requests from clients (such as web browsers) and send back responses containing web pages, files, or other resources. Traditional web browsers and web servers communicate through a text-based protocol known as HTTP, which stands for Hypertext Transfer Protocol. Here's how the interaction typically unfolds: A web browser initiates a connection to a web server and sends an HTTP request for specific content. The web server provides the requested content and terminates the connection. The browser processes and presents the received content on the user's screen.
Web servers are identified by both the machine they run on and the port they use for communication. Ports allow multiple independent network activities to happen simultaneously on a single machine. For example, a web server might handle web traffic on port 80, while a mail server handles emails on port 25.

Although web servers typically use port 80 for HTTP traffic, in some cases, a different port may be used (as in this project).

changed it to 8000

Test it

Run the web server class and go to http://localhost:5000 in your browser. You should see a simple web page. You should get your values back if you click on submit.



COEN 346 OPERATING SYSTEMS – FALL 2023

Your tasks:

Multithreading:

The current implementation of the server is single-threaded. This means that each new client needs to wait until the previous client has finished for its request to be served. To help you see this, we have provided a SimpleWebClient class that waits for one minute before sending a request to the server. Run the server, then run the client and then open the webpage and see what happens.

To solve this problem, implement a multithreaded approach to allow the server to handle multiple client connections concurrently. You will need to detail your strategy (classes used, when a new thread is created, when it is started, what task does each thread handle) in the report.

To evaluate you, we might create thousands of clients, your server should support them.

Fund Transfer Functionality:

Extend the server to handle fund transfers between accounts. Clients should be able to submit the form with details such as source account, source value, destination account, and destination value. The server should accurately process these transfers while maintaining data integrity. For this, once the server is initialized, you should create the accounts provided in simple file in which each line represents an account as in the following example:

```
Account id, balance
123, 4000
321, 5000
432, 2000
```

Synchronization and Deadlock Prevention

As you implement multithreading, it's crucial to be aware of synchronization issues. When multiple threads access shared data concurrently, it can lead to race conditions and data corruption. Identify critical sections in your code where synchronization is required to prevent such issues. Additionally, analyze the code for potential deadlock scenarios and implement strategies to prevent and handle them.
Analyze the code for potential deadlock situations, especially when handling transfers, and implement strategies to prevent and handle such scenarios.

You may use Java's synchronization tools **except the synchronized keyword**.

Deliverables:

All deliverables are submitted via Moodle. You should submit two files:

- a zip file of the code
- a pdf file of the report.

File names should be in the format <SID1>_<SID2>_assignment2_<type> Where SID corresponds to the student id of each member of the group and type is either code of report.

COEN 346 OPERATING SYSTEMS – FALL 2023

Multithreaded Web Server Code: Provide a Java codebase for the enhanced web server with multithreading and fund transfer handling. Submit a zip file. Make sure to remove any compiled code.

Design Report: Offer a report detailing your multithreading strategy, your synchronization strategies implemented and how they address potential issues. Include any testing or scenarios you used to validate the correctness of your implementation.

You must explain the following 3 aspects of your proposed design. Create sections for each of these aspects.

- **Data structures and functions** – Describe any struct/class definitions, global (or static) variables, typedefs, or enumerations that you will be adding or modifying (if it already exists). You should write this using UML diagram. Include a brief explanation the purpose of each modification. Your explanations should be as concise as possible.
- **Algorithms** – This is where you explain how your code works. Your description should be at a level below the high-level description of requirements given in the assignment. We have read the project spec too, so it is unnecessary to repeat or rephrase what is stated here. On the other hand, your description should be at a level above the code itself. Don't give a line-by-line run-down of what code you plan to write. Instead, you should try to convince us that your design satisfies all the requirements, including any uncommon edge cases.
- The length of this section depends on the complexity of your design. Simple explanations are preferred, but if your explanation is vague or does not provide enough details, you will be penalized. We recommend that you split the explanation into parts. Describe your algorithm for each part in a separate section. Start with the simplest component and build up your design, one piece at a time.
- **Rationale** – Tell us why your design is better than the alternatives that you considered, or point out any shortcomings it may have. You should think about whether your design is easy to conceptualize, how much coding it will require, the time/space complexity of your algorithms, and how easy/difficult it would be to extend your design to accommodate additional features.

Project Evaluation:

Your project will be evaluated based on the following criteria:

- Correct implementation of multithreading in the web server.
- Accurate handling of fund transfers without data corruption or race conditions.
- Effective synchronization strategies to prevent concurrency issues.
- Adequate measures to identify and prevent potential deadlocks.
- Quality of code base (comments, variable names, structure, function names). Comments in the code should explain sections, synchronization points, and deadlock prevention strategies.

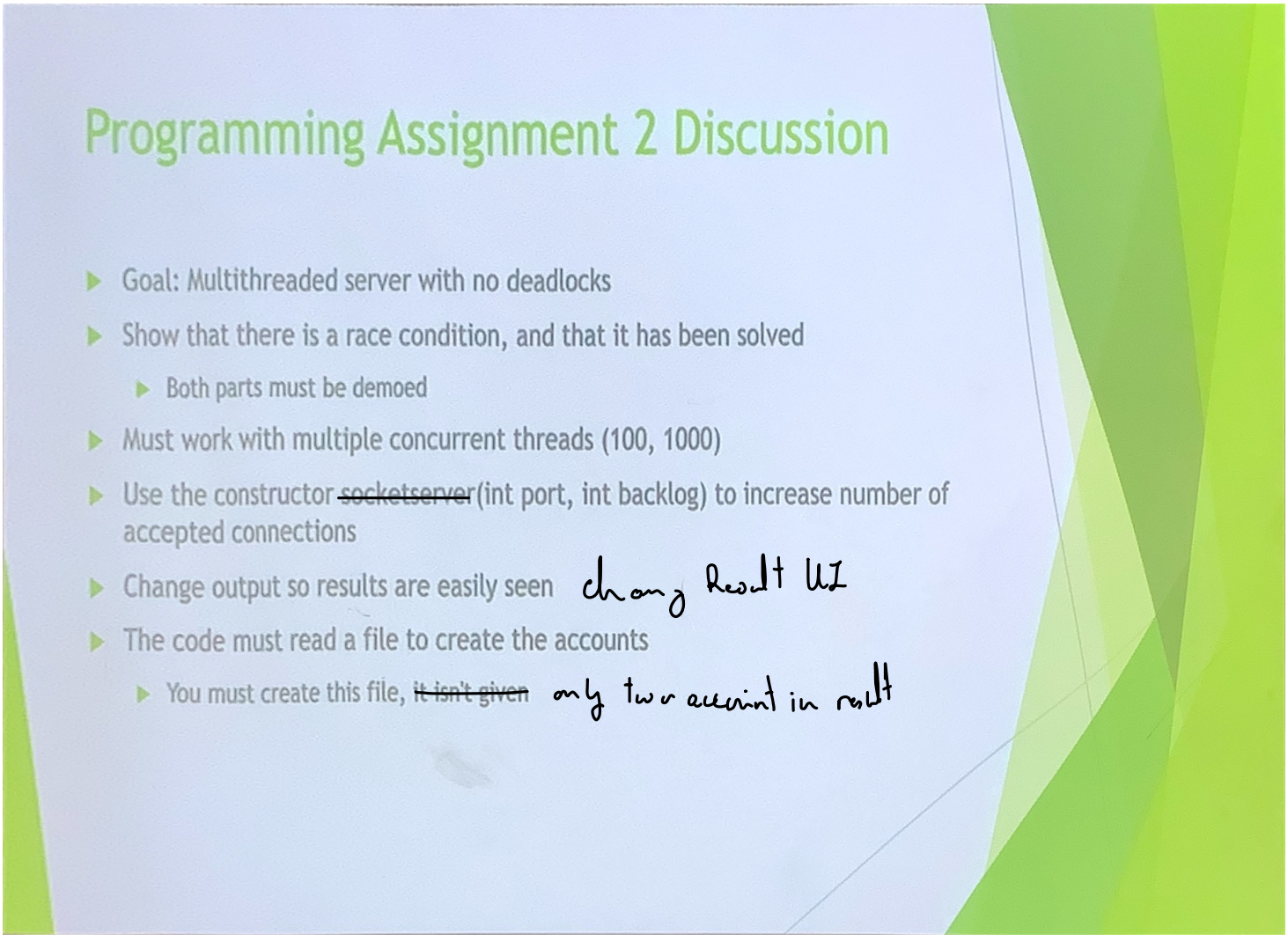
- Demo and answers to TA

Additional resources

Thread documentation
https://download.java.net/java/early_access/panama/docs/api/java.base/java/lang/Thread.html

Concurrency tutorial (for java version 8, note java is currently version 21). Main concepts remain https://docs.oracle.com/javase/tutorial/essential/concurrency/

Virtual threads
https://docs.oracle.com/en/java/javase/21/core/virtual-threads.html#GUID-DC4306FC-D6C1-4BCC-AECE-48C32C1A8DAA



Design reports

You must explain the following 3 aspects of your proposed design. Create sections for each of these aspects.

- Data structures and functions – Describe any struct/class definitions, global (or static) variables, typedefs, or enumerations that you will be adding or modifying (if it already exists). You should write this using UML diagram. Include a brief explanation the purpose of each modification. Your explanations should be as concise as possible.
- Algorithms – This is where you explain how your code works. Your description should be at a level below the high-level description of requirements given in the assignment. We have read the project spec too, so it is unnecessary to repeat or rephrase what is stated here. On the other hand, your description should be at a level above the code itself. Don't give a line-by-line run-down of what code you plan to write. Instead, you should try to convince us that your design satisfies all the requirements, including any uncommon edge cases.
- The length of this section depends on the complexity of your design. Simple explanations are preferred, but if your explanation is vague or does not provide enough details, you will be penalized. We recommend that you split the explanation into parts. Describe your algorithm for each part in a separate section. Start with the simplest component and build up your design, one piece at a time.
- Rationale – Tell us why your design is better than the alternatives that you considered, or point out any shortcomings it may have. You should think about whether your design is easy to conceptualize, how much coding it will require, the time/space complexity of your algorithms, and how easy/difficult it would be to extend your design to accommodate additional features.