Assignment 1

Course Title: Distributed Systems and Applications

Course Code: DSA612S

Assessment: First Assignment

Released on: 30/08/2023.

Due date: 22/09/ 2023 at midnight

Total Marks: 100

Task:
We need to develop an Api, using open Api.
Why?
To managing staff effectively

Question 1: Restful APIs

The problem centers on developing a Restful API for effectively managing staff, their offices, and allocated courses within the Faculty of Computing and Informatics. These offices can accommodate multiple lecturers. A lecturer is characterized by essential attributes, including a staff number, office number, staff name, title, and a list of the courses they are teaching. Additionally, a course is characterized by specific details such as the course name, course code, and NQF (National Qualifications Framework) level.

The API should include the following functionalities:

- Add a new lecturer
- Retrieve a list of all lecturers within the faculty.
- Update an existing lecturer's information.
- Retrieve the details of a specific lecturer by their staff number.
- Delete a lecturer's record by their staff number.
- Retrieve all the lecturers that teach a certain course.
- Retrieve all the lecturers that sit in the same office.

Note that the staff number should serve as a unique identifier for a lecturer.

Your task is to define the API following the **OpenAPI** standard and implement the client and service in the Ballerina language.

Deliverables:

- OpenAPI specification in YAML or JSON format.(15 marks)
- Client Implementation in a Ballerina that effectively interacts with the designed API.(10 marks)
- Service Implementation (25 Marks)

Question 2: Remote invocation

Your task is to design and implement a library system using gRPC that allows two types of users—a student and a librarian—to interact with the system. The system should provide essential functionalities for managing books, borrowing them, and returning them. The student should be able to get a list of available books, borrow a book, search for a book, locate it, and return a book. On the other hand, a librarian should be able to add a book, update a book, remove a book, and list all the borrowed books.

In short, we have the following operations:

- **add_book,** where the librarian creates a book. The books should have the following fields: title, author_1, optional author_2, location of the book in library, ISBN (International Standard Book Number), and status, if the book is available or not. This operation should return the ISBN for the added book.
- **create_users,** where several users, each with a specific profile, are created. The users are streamed to the server, and the response is returned once the operation completes.
- **update-book**, where the librarian alters the details of the given book.
- **remove-book**, where the librarian removes a book from the collection of library books. The function should return the new list of books after the book have been removed.
- **List_avaialable_books,** where a student gets a list of all the available books.
- **locate_book,** where a student searches for a book based on their ISBN. If the book is available, the function should return the location of the book; otherwise, tell the student that the book is not available.
- **borrow-book,** where a student borrows a book by providing their user ID and the book's ISBN.

Your task is to define a protocol buffer contract with the remote functions and implement both the client and the server in the Ballerina Language.

Server Implementation:

Implement the server logic using the Ballerina Language and gRPC. Your server should handle incoming requests from clients and perform appropriate actions based on the requested operation.

Client Implementation:

The clients should be able to use the generated gRPC client code to connect to the server and perform operations as implemented in the service. Clients should be able to handle user input and display relevant information to the user.

Please be aware that you have the freedom to include additional fields in your records if you believe they would enhance the performance and overall quality of your system.

Deliverables:

We will follow the criteria below to assess this problem:

- Definition of the remote interface in Protocol Buffer. (15 marks)
- Implementation of the gRPC client in the Ballerina language. (10 marks)
- Implementation of the gRPC server and server-side logic in response to the remote invocations in the Ballerina Language. [25 marks]

Submission Instructions

- This assignment is to be completed by groups of four students each.
- For each group, a repository should be created on Gitlab or Github. The repository should have all group members set up as contributors.
- All assignments must be uploaded to a GitHub or GitLab repository. Students who haven't **pushed** any codes to the repository will not be given the opportunity **to present and defend the assignment.** More particularly, if a student's username does not appear in the commit log of the group repository, that student will be assumed not to have contributed to the project and thus be awarded the mark 0.
- The assignment will be group work, but individual marks will be allocated based on each student's contribution to the assignment.
- Marks for the assignment will only be allocated to students who have presented the assignment.
- It's the responsibility of all group members to make sure that they are available for the assignment presentation. An assignment cannot be presented more than once.
- The submission **date is Sunday, September 22, 2023, at midnight.** Please note that commits after that deadline will not be accepted. Therefore, a submission will be assessed based on the clone of the repository at the deadline.
- Any group that fails to submit on time will be awarded the mark 0. Late Submiss
- There should be no assumption about the execution environment of your code. It could be run using a specific framework or on the command line.

- In the case of plagiarism (groups copying from each other or submissions copied from the Internet), all submissions involved will be awarded the mark 0, and each student will receive a warning.