RYERSON UNIVERSITY

Faculty of Engineering, Architecture and Science

Department of Electrical and Computer Engineering

Course Number	892
Course Title	Distributed and Cloud Computing
Semester/Year	W2024

Instructor	Dr. Muhammad Jaseemuddin

Lab No.	4-5

Lab Title	FastAPI and Containers

Submission Date	April 7th, 2024
Due Date	April 7th, 2024

Student Name	Student ID	Signature*
Abdulrehman Khan	500968727	A.K.

**By signing above you attest that you have contributed to this written lab report and confirm that all work you have contributed to this lab report is your own work.*

Introduction

The objective of this project was to create a server and an operator interface to simulate communication between an operator and rovers using FastAPI and WebSocket. The server provides endpoints for the operator to interact with, including creating, updating, and deleting mines and rovers, as well as dispatching commands to the rovers. The operator interface, implemented in both Python and HTML/JavaScript, allows users to perform these actions through a user-friendly interface.

Implementation Overview:

The implementation consists of two main parts: the server (implemented in Python using FastAPI) and the operator interface (implemented in Python CLI and HTML/JavaScript).

The server provides several endpoints for interacting with the map, mines, and rovers. It utilizes FastAPI to handle HTTP requests and responses, with data stored in memory using dictionaries. The server handles requests to retrieve and update the map, manage mines (including creation, updating, and deletion), and control rovers (including creation, updating commands, dispatching, and deletion).

The operator interface allows users to interact with the server through either a command-line interface (CLI) implemented in Python or a user-friendly web interface implemented in HTML and JavaScript. Both interfaces provide options to perform actions such as retrieving map data, managing mines, and controlling rovers.

Main Data Structures and Functions:

Server Data Structures (server.py):

- **mines:** A dictionary storing information about mines, with keys as mine serial numbers and values as dictionaries containing mine coordinates.
- **rovers:** A dictionary storing information about rovers, with keys as rover IDs and values as dictionaries containing rover status, commands, and position.

Server Endpoints (server.py):

/map: GET and PUT endpoints to retrieve and update the map.
/mines: GET, POST, DELETE endpoints to manage mines.
/rovers: GET, POST, DELETE endpoints to manage rovers, and PUT endpoint to send commands to a rover and dispatch it.

Operator Interface Functions (op.py):

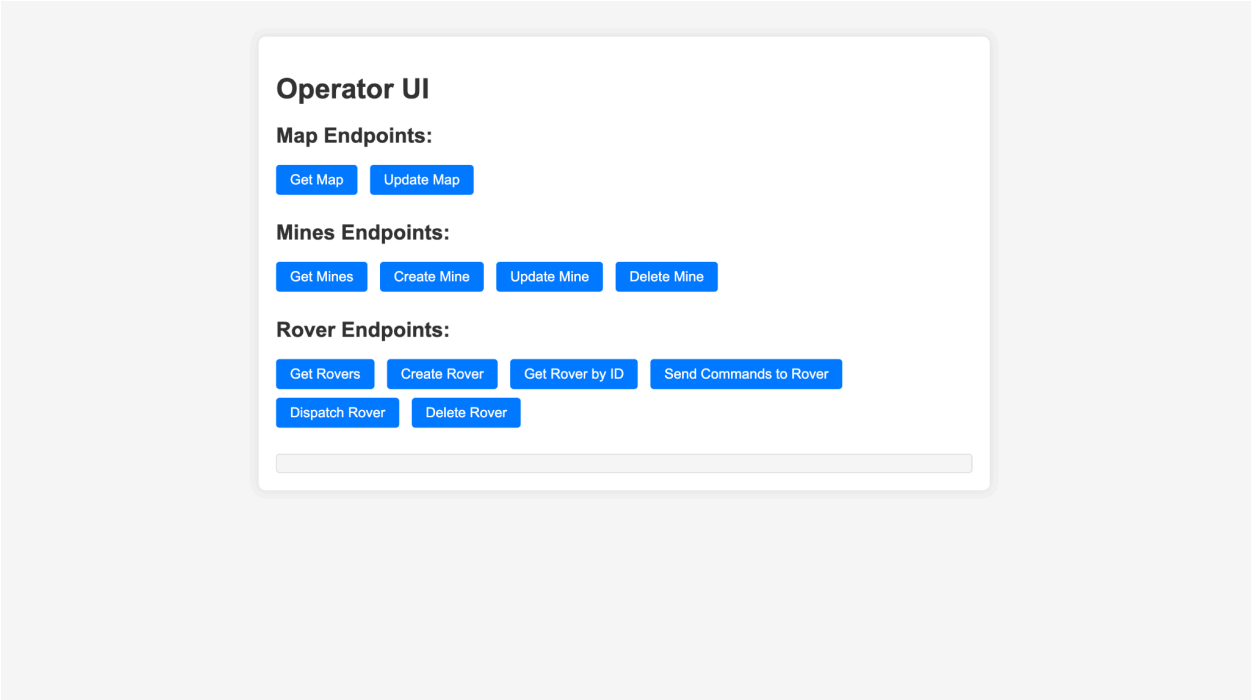
print_menu(): Displays the menu options for the operator in the CLI interface.
Functions to interact with each endpoint, such as **get_map(), create_mine(), get_rovers(), etc.,** which make HTTP requests to the server and display the responses.

Test Run Screenshots:

Server (server.py):

```
/usr/local/bin/python3 "/Users/abdulrehman/Desktop/892/Abdulrehman khan Lab 4-5 (892)/server.py"

The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Abdulrehmans-MacBook-Pro:Abdulrehman khan Lab 4-5 (892) abdulrehman$ /usr/local/bin/python3 "/Users/abdulrehman/Desktop/892/Abdulrehman k
han Lab 4-5 (892)/server.py"
INFO:     Started server process [54085]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
INFO:     127.0.0.1:50905 - "GET / HTTP/1.1" 200 OK
```

User interface (index.html):

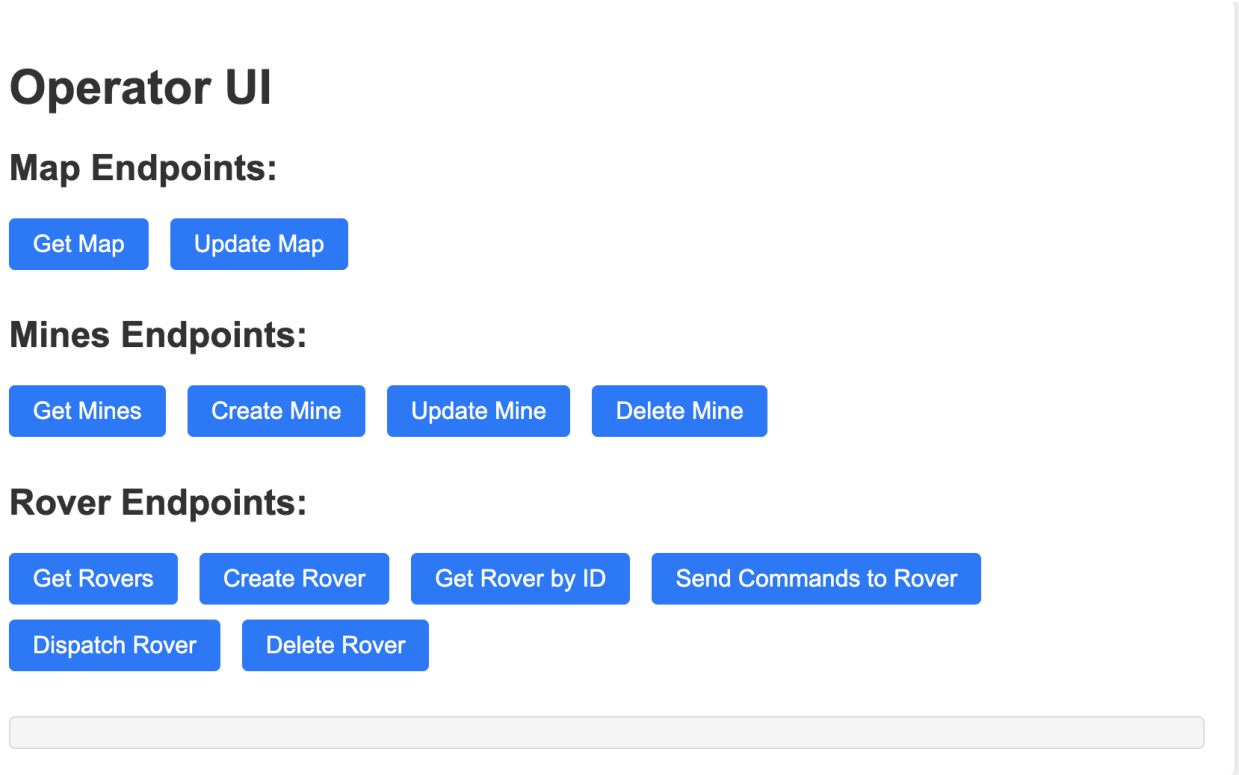


Operator (pp.py):

```
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
Abdulrehmans-MacBook-Pro:Abdulrehman khan Lab 4-5 (892) abdulrehman$ python3 op.py

Map Endpoints:
1. Get Map
2. Update Map

Mines Endpoints:
3. Get Mines
4. Get Mine by ID
5. Create Mine
6. Update Mine
7. Delete Mine

Rover Endpoints:
8. Get Rovers
9. Get Rover by ID
10. Create Rover
11. Send Commands to Rover
12. Dispatch Rover
13. Delete Rover

0. Exit
Enter your choice:
```

Docker Deployment:

	optimistic_c d9fd2faa0444	my-fastapi-app	Running	6.7%	8000:8000	2 minutes ago	



Conclusion:

In conclusion, this project successfully implemented a server and operator interface using FastAPI and WebSocket to simulate communication between an operator and rovers. The server provides endpoints for managing map data, mines, and rovers, while the operator interface allows users to interact with these endpoints through either a CLI or a user-friendly web interface. The project demonstrates the use of FastAPI for building RESTful APIs and provides a foundation for further development.