PRESENTATION ON
3-body Runge-Kutta Solver Infrastructure

Submitted by:

1.Mohammad Tohin Bapari ID:1836950

2. Agyapong Prince ID: 1737596

3. Anselem Okeke ID: 1943585

Supervised by:

1. Torsten Harenberg

2. Marisa Sandhoff

Content

- Introduction
- Containers
  - Solver-api
  - Frontend
  - Traefik
- Compose

Introduction

The aim of this project is to create a 3-body Runge-Kutta solver infrastructure using multi container docker system. In our project we create three containers.

Container Name	Task
Solver-api	To solve the 3-body Runge-Kutta problem and store output data in a database
Frontend	To visualize the result graphically
Traefik	To create routing between the containers.

Solver

- Import necessary libraries like asyncio, numpy,websockets,json,socket,sqlite3,pandas etc.
- Create two function def f(t,y,m) and def ODE45(f, tspan, h0, e, y0, s,m) and get output value.
- Create database and we create to function for sending and recieving data.

```
async def save_to_db(data):
    connect_db=sqlite3.connect("Solver_DB.db")
    post=connect_db.cursor()
```

```
async def backend_service(websocket, path):
    initial = await websocket.recv()
    print(initial)
    print("Backend service")
    data = json.loads(initial)
    print(data)
    Id=await save_to_db(data)
```

Frontend

- Connection to the client:

```
document.querySelector('#play-sim').onclick = function(e) {
    if (!websocket_conn) {
        //websocket_conn = new WebSocket('wss://localhost:8001');
websocket_conn = new WebSocket('ws://' + window.location.hostname+ ':8001');
    }
```

- Printing and Simulating data from server using client:

```
websocket_conn.addEventListener('message', function (event) {
    data = event.data;

    recv = JSON.parse(data)
```

Compose

Solver_api

```
build: ../backend/server
   image: backend-server:latest
   container_name: api
   deploy:
     mode: replicated
     replicas: 1
   restart: always
   volumes:
     - ../backend/server:/server
   labels:
     - "traefik.enable=true"
     - "traefik.port=8001"
     -"traefik.http.routers.mywebsocket.
entrypoints=server1"
     -"traefik.http.routers.mywebsocket .tls  =
false"
     - "traefik.http.routers.mywebsocket.rule
    =Host(`${HOST}`)"
```

Frontend

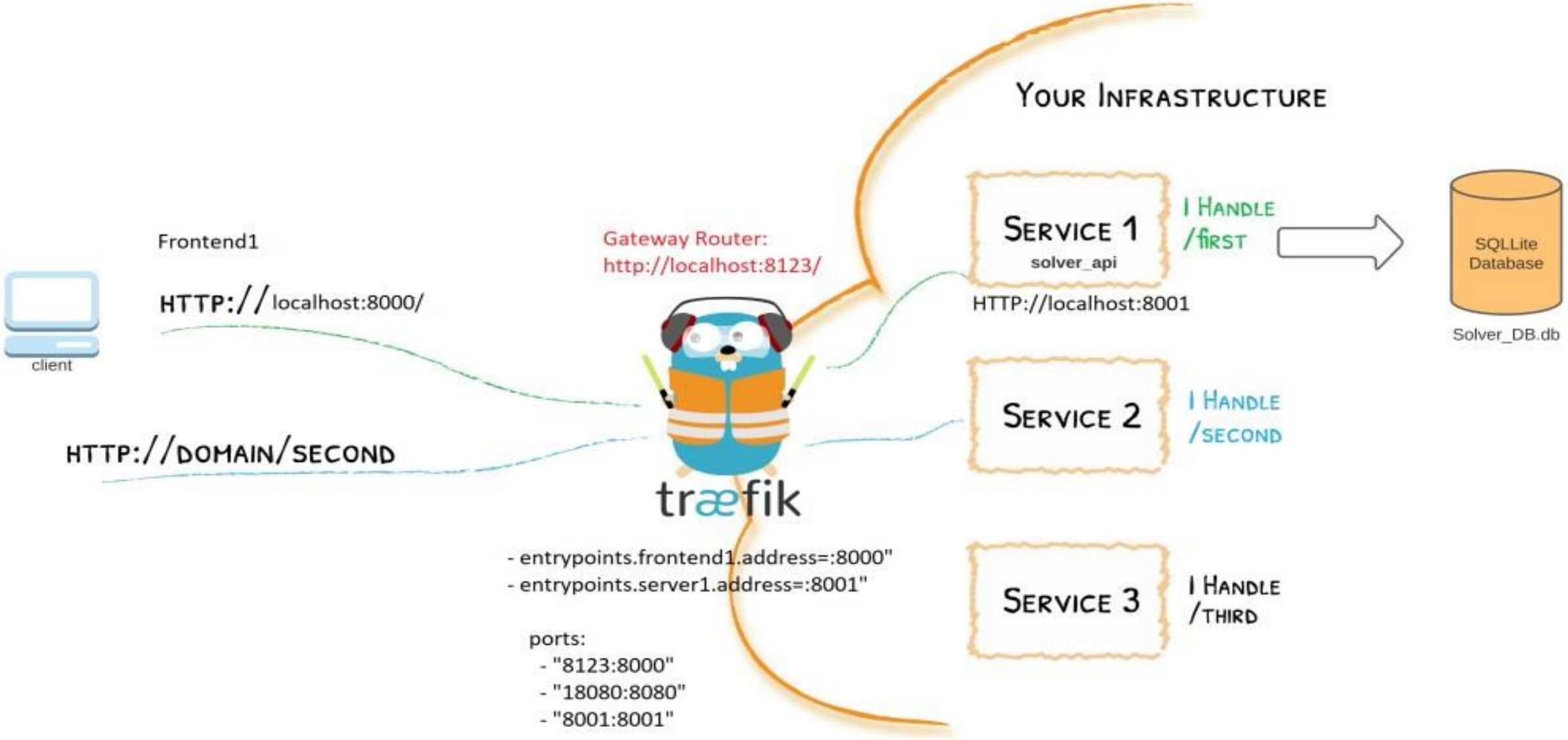
```
build: ../build/frontend_container
   image: frontend:latest
   restart: always
   container_name: frontend
   ports:
    - "8000:8000"

   labels:
     - "traefik.enable=true"
     - "traefik.port=8000"
     - "traefik.http.routers.myrouter.entrypoints
        =frontend1"
     - "traefik.http.routers.myrouter.tls=true"
     - "traefik.http.routers.myrouter.rule
        =Host(`${HOST}`)"
```

Traefik

```
image: traefik:2.3
   container_name: traefik_router
   volumes:
     "/var/run/docker.sock:/var/run/docker.soc
k:ro"
   ports:
     - "8123:8000"
     - "18080:8080"
     - "8001:8001"
   command:
     - "--api.insecure=true"
     - "--entrypoints.frontend1.address=:8000"
     - "--entrypoints.server1.address=:8001"
     - "--providers.docker=true"
     - "--providers.docker.exposedbydefault
        =false"
```

Traefik


YOUR INFRASTRUCTURE
Frontend1
HTTP://localhost:8000/
client
Gateway Router:
http://localhost:8123/
HTTP://DOMAIN/SECOND
træfik
- entrypoints.frontend1.address=:8000"
- entrypoints.server1.address=:8001"
ports:
- "8123:8000"
- "18080:8080"
- "8001:8001"
SERVICE 1
solver_api
I HANDLE
/first
HTTP://localhost:8001
SQLLite
Database
Solver_DB.db
SERVICE 2
I HANDLE
/SECOND
SERVICE 3
I HANDLE
/THIRD