MICROSERVICES ASSIGNMENT GUIDE

1. RUNNING THE APPLICATION:

Java-17 is required. In-memory H2 database is being used.

Build JAR of each application using Maven.

Once you are ready with JARs, please follow below steps.

- start the 'config-server' application. It will run on port 8888.
- start the 'service-discovery-server' application. It will run on port 8761.
- start the 'customer-management-service' application. It will run on port 8081.
- start the 'account-management-service' application. It will run on port 8082.
- start the 'api-gateway' application. It will run on port 9090.

2. APIs TO WORK WITH

1. CREATE CUSTOMER

URL: http://localhost:9090/api/customer/create

METHOD: POST

SAMPLE REQUEST BODY:

```
{
   "name":"Nikhil Bhardwaj",
   "email":"nikhil@gmail.com",
   "password":"nikhil",
   "dob":"1999-09-10",
   "nationality":"India"
}
```

SAMPLE RESPONSE:

```
STATUS 201 CREATED
{
   "data": {
      "customerId": 1,
      "name": "Nikhil Bhardwaj",
      "email": "nikhil@gmail.com",
      "dob": "1999-09-10",
      "nationality": "India"
   }
}
```

2. GET ALL CUSTOMERS

URL: http://localhost:9090/api/customer/get

METHOD: GET

SAMPLE RESPONSE:

```
STATUS 200 OK
{
    "data": [
        {
            "customerId": 1,
            "name": "Nikhil Bhardwaj",
            "email": "nikhil@gmail.com",
            "dob": "1999-09-10",
            "nationality": "India"
        },
        {
            "customerId": 3,
            "name": "Nishant Sharma",
            "email": "nishant@gmail.com",
            "dob": "1999-08-08",
            "nationality": "India"
        }
    ]
}
```

3. GET A PARTICULAR CUSTOMER

URL: http://localhost:9090/api/customer/get/{customerId}

METHOD: GET

SAMPLE RESPONSE

STATUS 200 OK

```
{
    "data": {
        "customerId": 1,
        "name": "Nikhil Bhardwaj",
        "email": "nikhil@gmail.com",
        "dob": "1999-09-10",
        "nationality": "India"
    }
}
```

4. UPDATE DETAILS OF A CUSTOMER

URL: http://localhost:9090/api/customer/update

METHOD: PUT

SAMPLE REQUEST BODY:

```
{
    "customerId":1,
    "name":"N Bhardwaj",
    "email":"nikhil@gmail.com",
    "password":"newpassword",
    "dob":"2002-08-08",
    "nationality":"India"
}
```

SAMPLE RESPONSE:

STATUS 200 OK

```
{
    "data": {
        "customerId": 1,
        "name": "N Bhardwaj",
        "email": "nikhil@gmail.com",
        "dob": "2002-08-08",
        "nationality": "India"
    }
}
```

5. DELETE A CUSTOMER

URL: http://localhost:9090/api/customer/delete/{customerId}

METHOD: DELETE

<u>SAMPLE RESPONSE:</u>

STATUS 204 NO CONTENT

6. CREATE ACCOUNT

URL: http://localhost:9090/api/account/create

METHOD: POST

<u>SAMPLE REQUEST BODY:</u>

```
{
    "type":"saving",
    "customerId":1,
    "balance":1000,
    "status":"active"
}
```

<u>SAMPLE RESPONSE:</u>

STATUS 201 CREATED

```
{
    "data": {
        "id": 1,
        "type": "saving",
        "customerId": 1,
        "balance": 1000.0,
        "status": "active"
    }
}
```

7. GET DETAILS OF A PARTICULAR ACCOUNT

URL: http://localhost:9090/api/account/get/{accountId}

METHOD: GET

<u>SAMPLE RESPONSE:</u>

STATUS 200 OK

```
{
    "data": {
        "accountId": 1,
        "type": "saving",
        "customerId": 1,
        "name": "N Bhardwaj",
        "email": "nikhil@gmail.com",
        "dob": "2002-08-08",
        "nationality": "India",
        "balance": 1000.0,
        "status": "active"
    }
}
```

8. DEPOSIT INTO ACCOUNT

URL: http://localhost:9090/api/account/deposit

METHOD: POST

SAMPLE REQUEST

```
{
    "customerId":1,
    "accountId":1,
    "amount":500
}
```

SAMPLE RESPONSE:

STATUS 200 OK

```
{
    "data": {
        "status": "success",
        "balance": 1500.0
    }
}
```

9. WITHDRAW FROM AN ACCOUNT

URL: http://localhost:9090/api/account/withdraw

METHOD: POST

SAMPLE REQUEST:

```
{
    "customerId":1,
    "accountId":1,
    "amount":300
}
```

SAMPLE RESPONSE:

```
STATUS 200 OK

{
   "data": {
      "status": "success",
      "balance": 1200.0
   }
}
```

10. DELETE AN ACCOUNT

URL: http://localhost:9090/api/account/delete/{accountId}

METHOD: DELETE

<u>SAMPLE RESPONSE:</u>

```
STATUS 204 NO CONTENT
```

3. ASSUMPTIONS

- One customer can have multiple accounts. But only 1 account of a particular type.
- Ideally different end points should be provided for password update, email update etc. But here a single endpoint is provided, you need to fill all required fields again even if you do not want to change them.
- When getting details of a user, you get all details except the password.

4. ERRORS AND EXCEPTIONS:

For certain exceptions, you get custom error in the below format:

```
{
   "errorCode": 409,
   "message": "Email already exists!"
}
```

Some cases considered:

- Customer email must be unique.
- Password must be at least 4 character long.
- A customer cannot have more than one account of same type.
- Deposit/withdraw in/from own accounts only.