**Task Backend Developer**

**GIT HUB : https://github.com/mohamhedshoaib/shoaib-glintsbackend.git**

**ETL** :

1. Saved Restaurant & user JSON data as JSON files in src/main/resources/restaurants.json and users.json

2. Used H2 as relational database which runs as in-memory , so it restarts whenever the application is restarted

3. On Spring boot application On Load , the users & restaurants json data is saved in H2 Database with proper one-to-many mapping

**Application**:

The application is built on microservcie architecture, hence it implies the spring boot annotations and features.It also

contains Respective controllers,its services, model beans based on DAO provided and also its repositories.

**Docker**:

The Docker file is located in the Project directory

To run the image on your local, kindly follow the steps given below:

1. GO to the project Directory

2. Run the following command in CMD to build the image: docker build -f DockerFileDev -t glints .

3.Check the image is build by running command: docker images

4. To run the docker image: docker run -d -p 8400:8400 glints

5. To Access **H2 Database**, kindly run the url Url: <http://localhost:8400/h2-console> and JDBC URL: jdbc:h2:mem:testdd

And hit Connect

6. Once Done run the below tasks with given URL

**Run**:

To run the application, you can follow the below given url or can access the Docker File Located in project directory named as **DockerFileDev**.

It runs on port **8400** and request name or application name is "**glints**"

**Task 1 :List all restaurants that are open at a certain datetime :**

Default: /glints

Uri: /searchOpenHours

**Complete url**: http://localhost:8400/glints/searchOpenHours?day=Fri&startTime=3:30 pm&endTime=8 pm

1. Once user provides these 3 values then it fetches the database for like match and return the user with required result

**Task 2 : List top y restaurants that have more or less than x number of dishes within a price range:**

Default: /glints

Uri: /listTopRestaurnats

**Complete url:** http://localhost:8400/glints/listTopRestaurnats?price=20.00&topSize=5

**Business Logic:**

1. It checks whether the Restaurant database has any menu item less than or equal to the given price

2. If matched, then it fetches all the restaurnats with the price range and groups them

3. The topSize field from the request params is passed to the grouped data and helps to displays users only with the restaurants

whose menu items are more than provided topSize(menu list more than topSize).

**Task 3. Search for restaurants or dishes by name, ranked by relevance to search term**

Default: /glints

Uri: /searchRestaurants

**Complete url:** http://localhost:8400/glints/searchRestaurants?restaurantName=247 Craven&dishName=Fried Soft Rock Crabs

**Business Logic:**

1. It checks whether the database has the given restaurnat or the dish name , if so then it return the restaurant info and its menu items to the user

**Task 4. Process a user purchasing a dish from a restaurant, handling all relevant data changes in an atomic transaction**

Default: /glints

Uri: /purcahseDish

**Complete url:** http://localhost:8400/glints/purcahseDish?restaurantName=247 Craven&dishName=Fried Soft Rock Crabs&userId=1

**Business Logic:**

1. It checks whether the available restaurnat has the given dish , if so then it orders for the provided user id

2. Once ordered, it successfully increases the merchant cash balance and reduces the user cash balance in Database

3. Successful message is delivered to the user