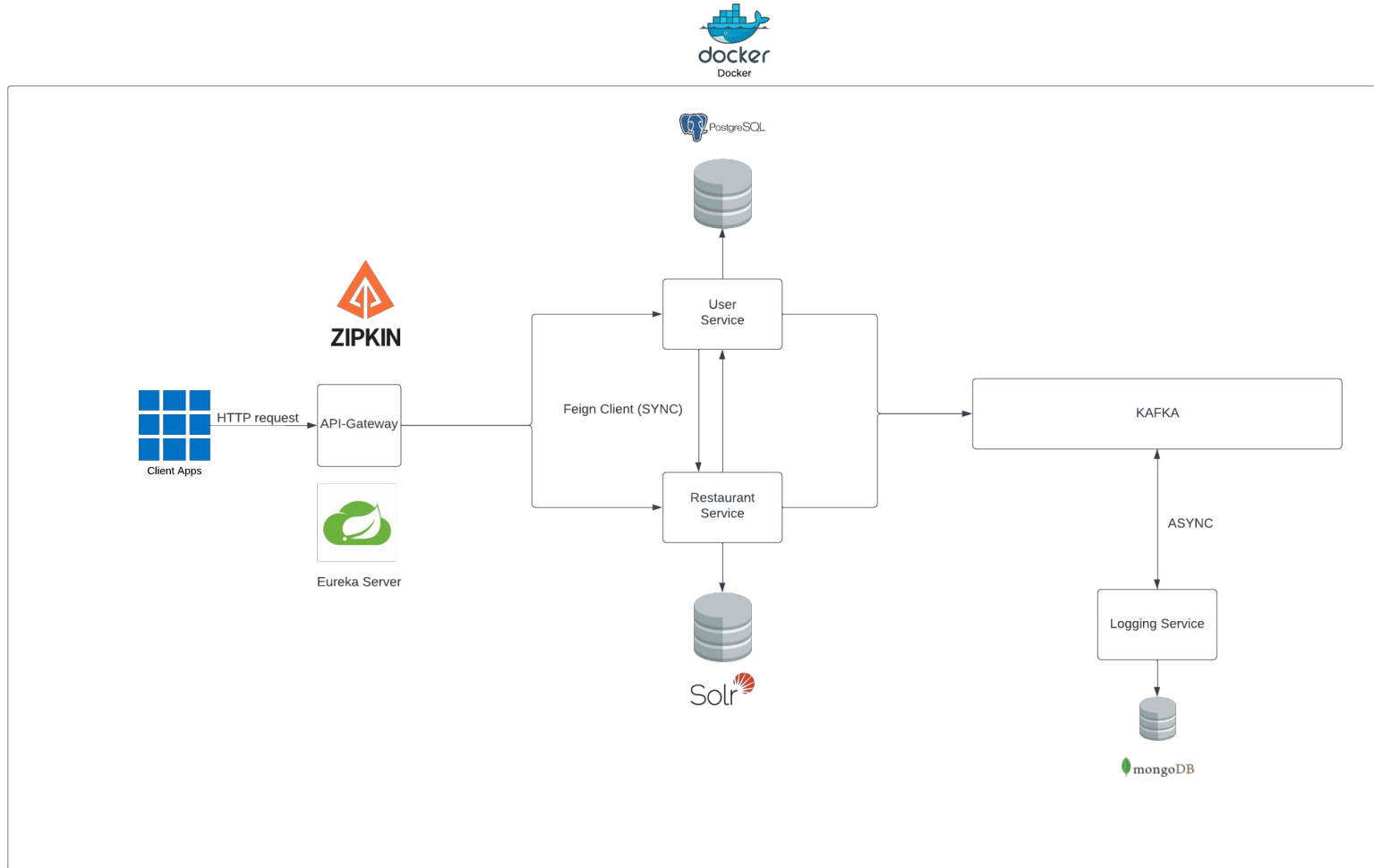



# n11 TalentHub Bootcamp Final Project

- Project Owner : Serhat Acar
- Project Name : Dine Experince Applicaton
- Project Description :Utilizing a microservice architecture to manage users, restaurant, user reviews based on location and ratings. It offers personalized restaurant recommendations, handles user and restaurant data, and performs queries on Apache Solr, ensuring efficient data management and a seamless dining experience.

# Project architecture and System Design



# User Service API

 **Swagger**  
powered by SMARTBEAR

/v3/api-docs

Explore

## Dine Experience App - User Service API 1.0 OAS 3.0

[/v3/api-docs](#)

OpenAPI documentation for Dine Experience App, a Spring Boot REST API as a graduation project for n11 TalentHub Bootcamp

[Contact Serhat Acar](#)

[Source code on GitHub](#)

Servers

http://localhost:8082 - user-service

Authorize

### user-controller

PUT

/api/v1/users/{debugId}

PUT request to update a user

GET

/api/v1/users

GET request for all users

POST

/api/v1/users

POST request to save a user

GET

/api/v1/users/{id}

GET request for a user by id

DELETE

/api/v1/users/{id}

DELETE request to delete a user

### user-review-controller

PUT

/api/v1/users/reviews

PUT request for user review

POST

/api/v1/users/reviews

POST request to create user review

GET

/api/v1/users/reviews/{id}

GET request for user review by id

DELETE

/api/v1/users/reviews/{id}


DELETE request to delete a user review

GET

/api/v1/users/reviews/user/{userId}

GET request for user reviews by user id

# Restaurant Service API

 **Swagger**  
OpenAPI 3.0

/v3/api-docs

Explore

## Dine Experience App - Restaurant Service API 1.0 OAS3

[/v3/api-docs](#)

OpenAPI documentation for Dine Experience App, a Spring Boot REST API as a graduation project for n11 TalentHub Bootcamp

[Contact Serhat Acar](#)

[Source code on GitHub](#)

Servers

http://localhost:8081 - restaurant-service

Authorize

### restaurant-controller

GET

/api/v1/restaurants

GET request for all restaurants

PUT

/api/v1/restaurants

PUT request to update a restaurant

POST

/api/v1/restaurants

POST request to save a restaurant

GET

/api/v1/restaurants/{id}

GET request for restaurant by id

DELETE

/api/v1/restaurants/{id}

DELETE request to delete a restaurant

### recommendation-controller

GET

/api/v1/restaurants/recommendations/{userId}

GET request for restaurant recommendations by user id

Schemas

RestaurantUpdateRequest

# Swagger Schemas

POST	/api/v1/users/reviews	POST request to create user review	▼
GET	/api/v1/users/reviews/{id}	GET request for user review by id	▼
DELETE	/api/v1/users/reviews/{id}	DELETE request to delete a user review	▼
GET	/api/v1/users/reviews/user/{userId}	GET request for user reviews by user id	▼
GET	/api/v1/users/reviews/restaurant/{restaurantId}	GET request for user reviews by restaurant id	▼
GET	/api/v1/users/reviews/all	GET request for all user reviews	▼

Schemas			^
<div>UserUpdateRequest ▼ {   id &gt; [...]   name &gt; [...]   surname &gt; [...]   birthDate &gt; [...]   email &gt; [...]   gender &gt; [...]   userStatus &gt; [...]   latitude &gt; [...]   longitude &gt; [...] }</div>			
<div>RestResponseUserDTO ▼ {   data UserDTO &gt; {...}   responseDate &gt; [...]   message &gt; [...]   success &gt; [...] }</div>			
<div>UserDTO ▼ {   id &gt; [...]   name &gt; [...]   surname &gt; [...]   birthDate &gt; [...]   email &gt; [...]   ... &gt; [...] }</div>			

# User and User Reviews Tables

The screenshot displays a database management interface with two tables. The top table, 'users', contains 10 rows of user information. The bottom table, 'user\_reviews', contains 10 rows of reviews. The interface includes a search bar, a table selector, and a status bar at the bottom.

**users**

	id	name	surname	gender	user_status	birth_date	email	latitude	longitude	creator_id	created_at	updated_id	updated_at
1	1	John	Doe	MALE	ACTIVE	1998-01-15	john.doe@example.com	40.74	-74.1	1	2022-01-01 00:00:00.000000	1	2022-01-01 00:00:00.000000
2	2	Jane	Smith	FEMALE	ACTIVE	1985-05-20	jane.smith@example.com	40.7	-73.93	2	2022-01-02 00:00:00.000000	2	2022-01-02 00:00:00.000000
3	3	Bob	Jones	MALE	ACTIVE	1998-08-10	bob.jones@example.com	40.69	-74.07	1	2022-01-03 00:00:00.000000	1	2022-01-03 00:00:00.000000
4	4	Alice	Miller	FEMALE	INACTIVE	1980-03-25	alice.miller@example.com	40.69	-73.99	3	2022-01-04 00:00:00.000000	3	2022-01-04 00:00:00.000000
5	5	David	Wilson	MALE	ACTIVE	1995-12-03	david.wilson@example.com	40.72	-74.05	2	2022-01-05 00:00:00.000000	2	2022-01-05 00:00:00.000000
6	6	Emily	Brown	FEMALE	INACTIVE	1987-09-18	emily.brown@example.com	40.72	-73.95	3	2022-01-06 00:00:00.000000	3	2022-01-06 00:00:00.000000
7	7	Charlie	Anderson	MALE	ACTIVE	1993-07-12	charlie.anderson@example.com	40.68	-73.92	1	2022-01-07 00:00:00.000000	1	2022-01-07 00:00:00.000000
8	8	Olivia	White	FEMALE	INACTIVE	1983-11-30	olivia.white@example.com	40.75	-73.97	2	2022-01-08 00:00:00.000000	2	2022-01-08 00:00:00.000000
9	9	Samuel	Martin	MALE	ACTIVE	1991-04-22	samuel.martin@example.com	40.72	-73.96	3	2022-01-09 00:00:00.000000	3	2022-01-09 00:00:00.000000
10	10	Ava	Thompson	FEMALE	ACTIVE	1989-06-05	ava.thompson@example.com	40.73	-74.05	1	2022-01-10 00:00:00.000000	1	2022-01-10 00:00:00.000000


**user\_reviews**

	id	restaurant_id	user_id	comment	user_rate	creator_id	created_at	updated_at	updated_id
1	1	1		1 Great food!	1.00	3	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	1
2	2	2		2 Good service	2.00	2	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	2
3	3	3		3 Excellent atmosphere	3.00	4	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	3
4	4	4		4 Could be better	4.00	1	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	4
5	5	5		5 Nice place	5.00	3	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	5
6	6	6		6 Average food	1.00	2	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	6
7	7	7		7 Loved it!	2.00	4	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	7
8	8	8		8 Not my favorite	3.00	1	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	8
9	9	9		9 Good overall	4.00	3	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	9
10	10	10		10 Okay experience	5.00	2	2024-03-11 17:57:10.354737	2024-03-11 17:57:10.354737	10

Database > docker\_db > user\_service\_db > public > tables > user\_reviews

SUM: Not enough values

# Solr Restaurant Documents



Dashboard

Logging

Security

Core Admin

Java Properties

Thread Dump

restaurants

Overview

Analysis

Documents

Params

Files

Ping

Plugins / Stats

Query

Replication

Schema

Segments info

Request-Handler (qt)

/select

common

q

\*:\*

q.op

OR

fq

sort

id asc

start.rows

010

fl

df

paramset(s)

Select paramset(s)...

wt

-----

☒ indent on

☐ debugQuery

defType

-----

☐ hl

☐ facet

☐ spatial

☐ spellcheck

Raw Query Parameters

JSON Query

Execute Query

http://localhost:8983/solr/restaurants/select?indent=true&q.op=OR&q=\*&rows=10&sort=id%20asc&useParams=

```
{
  "responseHeader": {
    "status": 0,
    "QTime": 13,
    "params": {
      "q": ":*:*",
      "indent": "true",
      "q.op": "OR",
      "sort": "id asc",
      "rows": "10",
      "useParams": "",
      "_": "1710650858659"
    }
  },
  "response": {
    "numFound": 11,
    "start": 0,
    "numFoundExact": true,
    "docs": [
      {
        "id": "1",
        "name": ["Test Restaurant 1"],
        "address": ["Test Address 1"],
        "phone": "1234567890",
        "email": ["test1@test.com"],
        "description": ["Test Description1"],
        "website": ["www.testrestaurant1.com"],
        "workingHours": ["09:00-18:00"],
        "Latitude": [40.71],
        "Longitude": [-74.03],
        "restaurantRate": [1],
        "status": ["ACTIVE"],
        "version": "179363969979158016"
      },
      {
        "id": "10",
        "name": ["Test Restaurant 10"],
        "address": ["Test Address 10"],
        "phone": "1234567890",
        "email": ["test10@test.com"],
        "description": ["Test Description"],
        "website": ["www.testrestaurant10.com"],
        "workingHours": ["09:00-18:00"],
        "Latitude": [41.71],
        "Longitude": [-71.2],
        "restaurantRate": [5],
        "status": ["ACTIVE"],
        "version": "1793639700303446016"
      },
      {
        "id": "2",
        "name": ["Test Restaurant 2"],
        "address": ["Test Address 2"],
        "phone": "1234567890",
        "email": ["test2@test.com"]
      }
    ]
  }
}
```

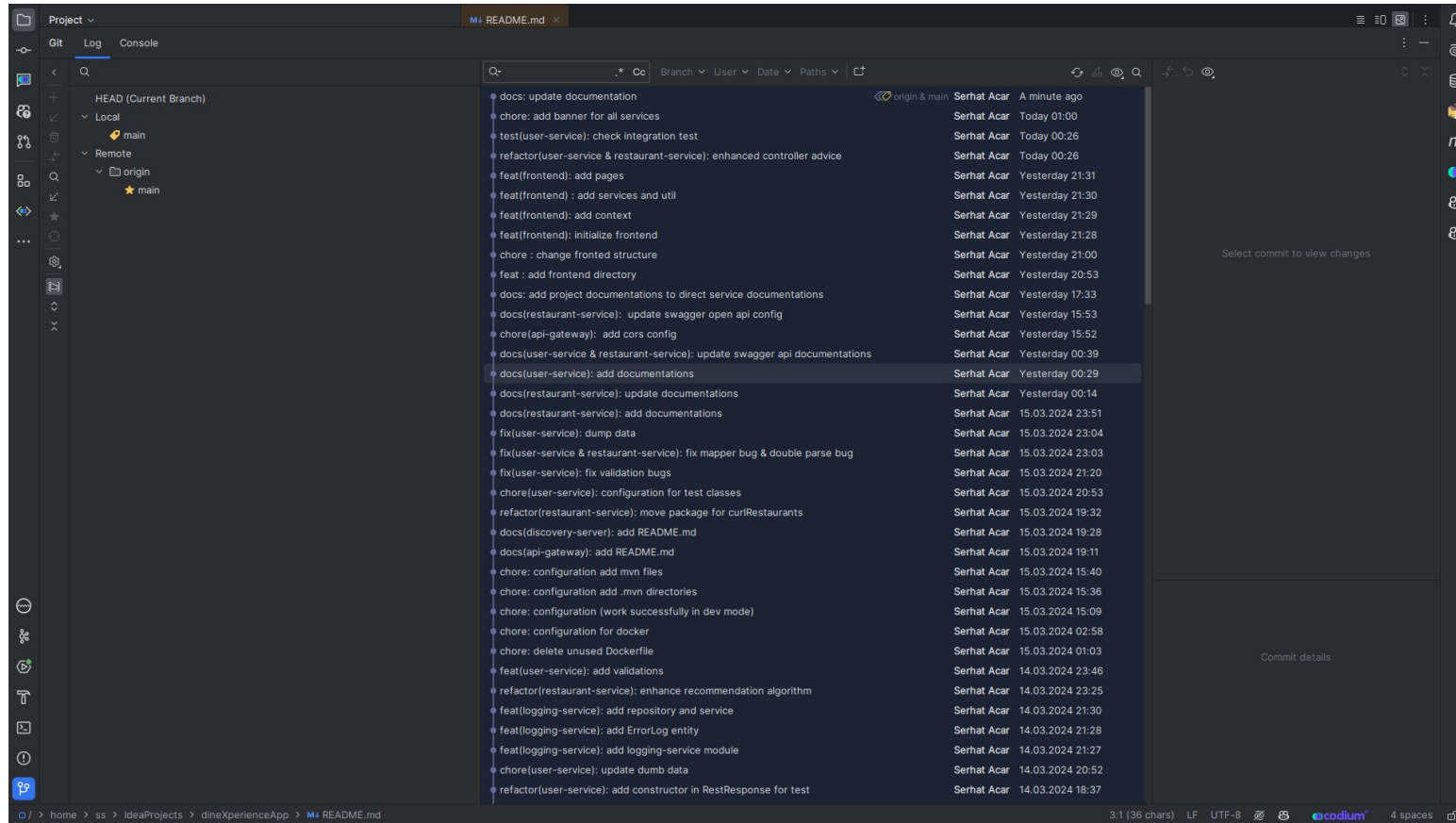
# Docker-Compose File

```
1 version: '3'
2
3 services:
4   db:
5     image: postgres
6     container_name: postgres_db
7     environment:
8       POSTGRES_USER: postgres
9       POSTGRES_PASSWORD: 123
10      POSTGRES_DB: user_service_db
11      ports:
12        - "5432:5432"
13      networks:
14        - back-tier
15      volumes:
16        - db_data:/var/lib/postgresql/data
17
18   solr:
19     image: solr
20     container_name: solr
21     ports:
22       - "8983:8983"
23     volumes:
24       - data:/var/solr
25     networks:
26       - back-tier
27     command:
28       - solr-precreate
29       - restaurants
30
31   zipkin:
32     image: openzipkin/zipkin
33     container_name: zipkin
34     networks:
35       - back-tier
36     ports:
37       - "9411:9411"
38
39   discovery-server:
40     build:
41       context: ./discovery-server
42       dockerfile: Dockerfile
43     container_name: discovery-server
```

Document 1/1 | services: db: volumes:



# Git Commits



The screenshot displays the IntelliJ IDEA interface with the Git Log window open. The window shows a list of commits for the README.md file, sorted by date. The commits are listed in a table with columns for the commit message, the author (Serhat Acar), and the date. The commit messages include various updates to documentation, tests, and code refactoring. The date column shows the commit time in UTC-8.

Commit Message	Author	Date
docs: update documentation	Serhat Acar	A minute ago
chore: add banner for all services	Serhat Acar	Today 01:00
test(user-service): check integration test	Serhat Acar	Today 00:26
refactor(user-service & restaurant-service): enhanced controller advice	Serhat Acar	Today 00:26
feat(frontend): add pages	Serhat Acar	Yesterday 21:31
feat(frontend) : add services and util	Serhat Acar	Yesterday 21:30
feat(frontend): add context	Serhat Acar	Yesterday 21:29
feat(frontend): initialize frontend	Serhat Acar	Yesterday 21:28
chore : change frontend structure	Serhat Acar	Yesterday 21:00
feat : add frontend directory	Serhat Acar	Yesterday 20:53
docs: add project documentations to direct service documentations	Serhat Acar	Yesterday 17:33
docs(restaurant-service): update swagger open api config	Serhat Acar	Yesterday 15:53
chore(api-gateway): add cors config	Serhat Acar	Yesterday 15:52
docs(user-service & restaurant-service): update swagger api documentations	Serhat Acar	Yesterday 00:39
docs(user-service): add documentations	Serhat Acar	Yesterday 00:29
docs(restaurant-service): update documentations	Serhat Acar	Yesterday 00:14
docs(restaurant-service): add documentations	Serhat Acar	15.03.2024 23:51
fix(user-service): dump data	Serhat Acar	15.03.2024 23:04
fix(user-service & restaurant-service): fix mapper bug & double parse bug	Serhat Acar	15.03.2024 23:03
fix(user-service): fix validation bugs	Serhat Acar	15.03.2024 21:20
chore(user-service): configuration for test classes	Serhat Acar	15.03.2024 20:53
refactor(restaurant-service): move package for curlRestaurants	Serhat Acar	15.03.2024 19:32
docs(discovery-server): add README.md	Serhat Acar	15.03.2024 19:28
docs(api-gateway): add README.md	Serhat Acar	15.03.2024 19:11
chore: configuration add mvn files	Serhat Acar	15.03.2024 15:40
chore: configuration add .mvn directories	Serhat Acar	15.03.2024 15:36
chore: configuration (work successfully in dev mode)	Serhat Acar	15.03.2024 15:09
chore: configuration for docker	Serhat Acar	15.03.2024 02:58
chore: delete unused Dockerfile	Serhat Acar	15.03.2024 01:03
feat(user-service): add validations	Serhat Acar	14.03.2024 23:46
refactor(restaurant-service): enhance recommendation algorithm	Serhat Acar	14.03.2024 23:25
feat(logging-service): add repository and service	Serhat Acar	14.03.2024 21:30
feat(logging-service): add ErrorLog entity	Serhat Acar	14.03.2024 21:28
feat(logging-service): add logging-service module	Serhat Acar	14.03.2024 21:27
chore(user-service): update dumb data	Serhat Acar	14.03.2024 20:52
refactor(user-service): add constructor in RestResponse for test	Serhat Acar	14.03.2024 18:37

