Assignment 3 (Hugbo) – Construction 1

Team 16 - brj46, hfd2, jjo1, ros30

October 2025

Link to GitHub Repository

`https://github.com/Reynirjr/Hugbo1-team16`

UC4 – View Menu

GET `http://localhost:8080/api/menus/1`

In this case, we are getting the first menu we created (a dummy menu) that returns a JSON:


```
{
  "id": 1,
  "name": "Main Menu",
  "currency": "ISK",
  "sections": [
    {
      "id": 1,
      "name": "Burgers",
      "displayOrder": 1,
      "items": [
        {
          "id": 2,
          "name": "Veggie Burger",
          "description": "Black bean patty",
          "priceIsk": 1790,
          "available": true,
          "tags": ["veg"],
          "imageData": null
        }
      ]
    }
  ]
}
```


UC1 – Place Order

When placing an order, we use the **POST** method:

`POST http://localhost:8080/api/baskets`

to create a basket with a unique basketId:


```
{
  "basketId": "8dbd0028-9c2c-4040-a590-4903212e70a0",
  "items": [],
  "createdAt": "2025-10-19T17:02:39.937Z"
}
```


UC5 – Sign In

When logging in, we use:

```
POST http://localhost:8080/api/auth/login
```

Example request:


```
{
  "username": "admin",
  "password": "admin123"
}
```

Response:

```
{
  "token": "eyJhbGciOiJIUzI1NiJ9...",
  "username": "admin"
}
```

UC10 – Set Order Status (Staff/Admin)

After signing in as a staff or admin user, you can read and update order status.

Read order:

```
GET http://localhost:8080/api/orders/12
```

Example response:

```
{
  "orderId": 12,
  "createdAt": "2025-10-15T17:25:44.345Z",
  "status": "RECEIVED",
  "totalIsk": 2600,
  "estimatedPickupAt": "2025-10-15T17:45:44.345Z",
  "customerPhone": "555-1234",
  "items": [ ... ]
}
```

Set order status:

```
PUT http://localhost:8080/api/orders/12/status?value=PREPARING
PUT http://localhost:8080/api/orders/12/status?value=READY
PUT http://localhost:8080/api/orders/12/status?value=PICKED_UP
```

Response example:

```
{
  "id": 12,
  "status": "PREPARING",
  "updatedAt": "2025-10-15T17:45:10.123Z"
}
```

UC2 – Manage Menu

You must authenticate as an admin using the bearer token from UC5.

To create a new item:

```
POST http://localhost:8080/api/menus/1/items
```

Body:

```
{
  "name": "Classic Burger",
  "description": "Beef, cheese, lettuce",
  "priceIsk": 1990,
  "available": true,
  "tags": ["beef", "popular"],
  "sectionId": 1
}
```

UC3 – Set Queue Time

To get the current queue time (for staff and admin):

```
GET http://localhost:8080/api/settings/queue-time
```

Response:

```
{
    "minutes": 20
}
```

To update the queue time:

```
PUT http://localhost:8080/api/settings/queue-time?minutes=20
```

Response:

```
{
    "minutes": 20,
    "updatedBy": "system",
    "updatedAt": "2025-10-15T16:59:02.858027Z"
}
```

Example of error handling if the input is out of range:

```
PUT http://localhost:8080/api/settings/queue-time?minutes=9999
```

Error response:

```
{
    "timestamp": "2025-10-19T22:16:31.394+00:00",
    "status": 500,
    "error": "Internal Server Error",
    "message": "Queue time must be between 0 and 180 minutes.",
    "path": "/api/settings/queue-time"
}
```

UC7 – Adjust Opening Hours & Exceptions

To view opening hours:

```
GET http://localhost:8080/api/hours
```

Example body:

```
{
  "weekday": "MONDAY",
  "openTime": "08:00",
  "closeTime": "17:00"
}
```

UC9 – View Order Status

```
GET http://localhost:8080/api/orders/2
```

Response:


```
{
  "id": 2,
  "createdAt": "2025-10-15T17:16:54.806Z",
  "basketId": "fabcee45-204d-434e-a42a-3a685a2fb6c3",
  "customerPhone": "5551234",
  "status": "RECEIVED",
  "totalIsk": 5200
}
```


Brief Notes

The project is nearing completion, with most major use cases implemented and verified through local testing. Core features such as authentication, menu management, basket operations, and order handling are functional and stable. The backend runs locally, and deployment to Render with PostgreSQL is currently being prepared.

Progress

- Completed authentication and role-based access control.
- Menu management (UC2), order placement (UC1), and queue time updates (UC3) are fully functional.
- Opening hours and exceptions (UC7) implemented and tested.
- UC9 (View Order Status) verified via Postman.
- UC10 (Set Order Status) successfully tested with PUT requests.

Outstanding Work

- UC6 (Sales Report): Next to develop, including totals and item popularity.
- Image Uploads: To be tested and integrated with frontend.
- Database Deployment: PostgreSQL migration to Render in progress.

Reason for Delay (UC6)

UC6 (Sales Report) was considered a secondary feature compared to the main operational use cases. The team focused first on delivering the essential end-to-end functionality for customers and staff—such as ordering, menu management, and queue handling—before adding analytics. This ensures the system is stable before expanding functionality.

Next Steps

- Deploy and connect backend to Render PostgreSQL.
- Implement UC6 (Sales Report).
- Add image upload endpoints and test via Postman.
- Conduct final integration testing and prepare for demo.