**Technical Documentation for Restaurant Reservation System**

**Description**: This project implements a restaurant Reservation System. It allows the following functions:

1. Create New Reservation
2. Cancel A Reservation
3. View A Reservation
4. Update A Reservation

**Scope and Assumptions**:

1. This system will be deployed on a local server. In this case, it can run on a machine for this exam.
2. No frontend implementation and deliverable is a backend project that exposes web services that will perform the functions mentioned in the description.
3. Postman will be used to test how the system will perform.
4. Formatting for Localdatetime should already be provided by the request.
5. Security is not in scope.
6. Validation is not in scope.
7. Searching for Update and Delete is using ID only.
8. No actual sending of email or mobile phone is required but a response body saying the message has been sent.

Audience: The system will be used by Restaurant Owners and their employees. A frontend system may access the web services but for this implementation, we will use postman to check for the service.

**Tech Stack:**

List the technologies used in the project (e.g., Spring Boot, Spring MVC, Spring Data JPA).

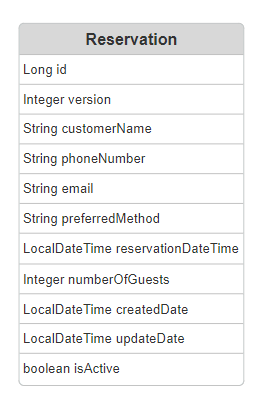
1. Spring Framework
2. Spring Boot
3. H2 Database
4. Maven
5. Lombok

**Component Breakdown**

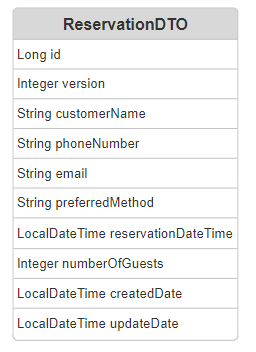
* ReservationController
* ReservationService
* ReservationRepository
* BootstrapData - used to create initial records

**Data Model**

System will use one table - Reservation



DTO Model



**API Reference:**

1. Create New Reservation Service

Description: Allows user to create new reservation by providing name, mobile number and/or email, preferred method and number of guests

Location: /api/v1/reservation

HTTP Method: Get

Sample Json Request:

{

"customerName": "Tara Nyembo",

"email": "tara@yahoo.com",

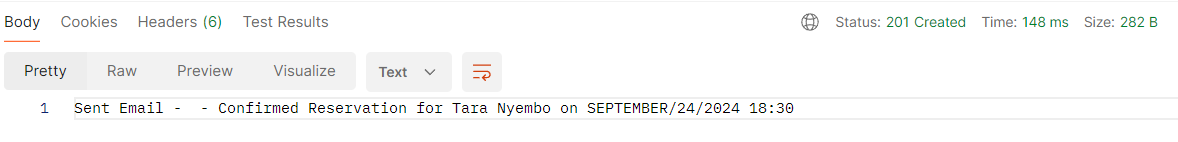
"preferredMethod": "E",

"reservationDateTime": "2024-09-24T17:30:00",

"numberOfGuests": 5

}

Positive Response Sample



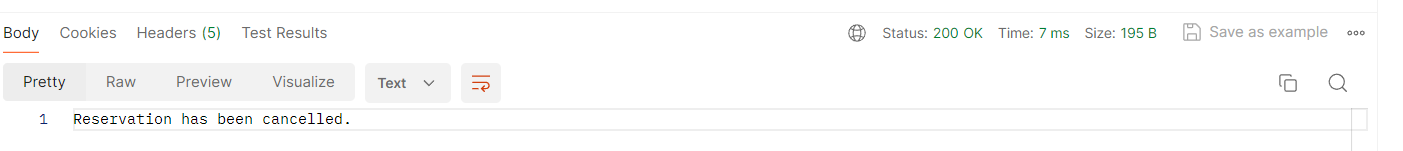
1. Cancel Reservation

Description: Allows user to cancel a reservation using reservationId

Location: /api/v1/reservation/{reservationId}

HTTP Method: Delete

Positive Result:





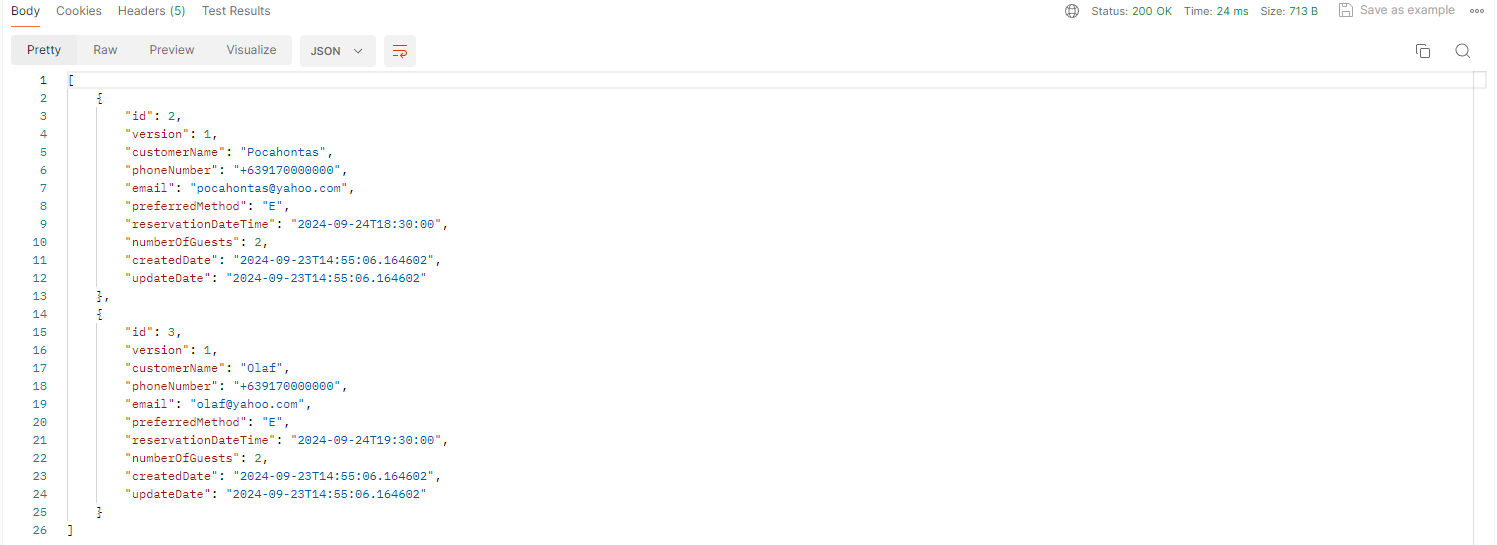
View my Reservations

Description: Allows user to view all of the upcoming reservations

Location: /api/v1/reservation/

HTTP Method: Get

Positive Response:



Update my reservation

Description: User can update their reservation by providing updated reservation date and time &/or number of Guests using reservationId

Location: /api/v1/reservation/{reservationId}

HTTP Method: Put

Sample Json Requests:

{

"reservationDateTime": "2024-09-24T19:30:00",

"numberOfGuests": 6

}

{

"reservationDateTime": "2024-09-24T20:30:00"

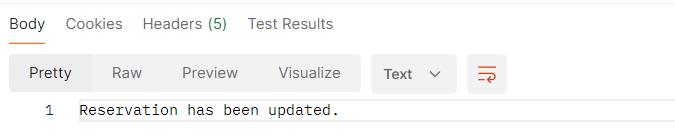
}

{

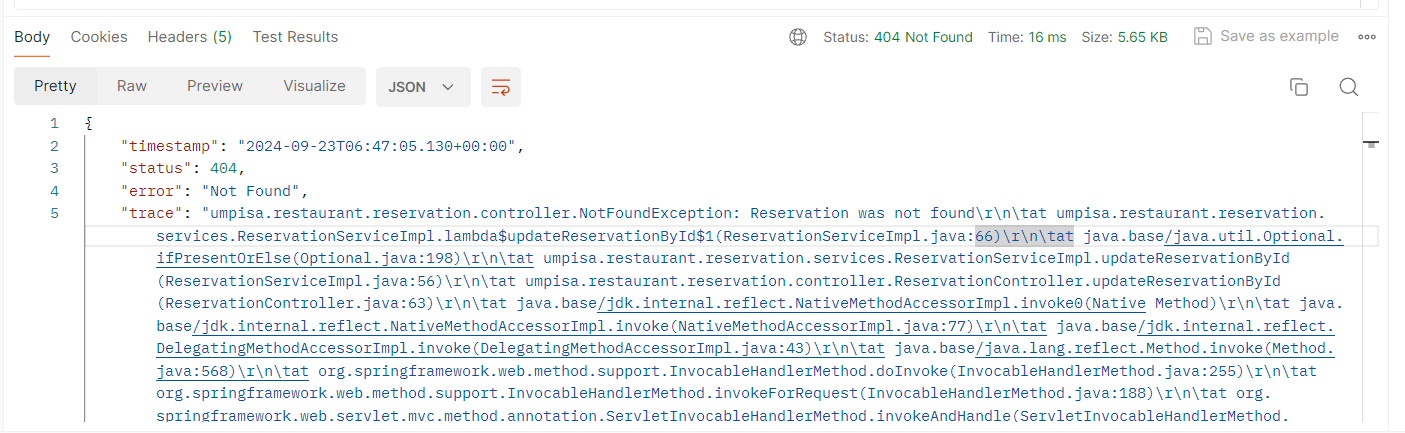
"numberOfGuests": 4

}

Positive Response



Negative Response



**Error Handling:**

1. Reservation is not found, NotFoundException will be thrown