

**JIMMA INSTITUTE OF TECHNOLOGY**

**FACULTY OF COMPUTING AND INFORMATICS**

**DEPARTMENT OF SOFTWARE ENGINEERING**

WEB SERVICE ASSIGNMENT

GROUP MEMBERS

1. Saron wondwossen………………………………… RU0156/14
2. Fethiya Jeylan ………………………………............ RU0206/14
3. Aymen Aman ……..………………………………... RU 0195/14
4. Dagmawit Endashaw………………………………. RU4711/14
5. Suhayb Abdirahman……………………………….. RU2316/14
6. **Introduction**

This report provides a detailed overview of setting up and testing basic web services using Java, focusing on both RESTful and SOAP approaches. Web services enable communication between different applications over the internet, allowing systems to share data and perform operations remotely. In this project, the goal was to build two simple web services one using REST principles with JAX-RS and another using SOAP with JAX-WS to demonstrate how APIs facilitate structured and reliable communication between clients and servers.

The RESTful service, implemented in the RestaurantService class, provides API endpoints that respond to standard HTTP requests such as GET and POST. These endpoints handle data exchange in JSON format, allowing clients to retrieve or add restaurant information easily. To test the functionality, requests were made using an HTTP client and Postman to ensure the API returned accurate and well-formatted responses.

The SOAP service, implemented in the SoapService class, demonstrates an XML-based approach to web communication.

To add data to the RESTful service, the POST method was used a standard HTTP method for sending data to a server. This practical exercise demonstrates how Java-based web services receive, process, and respond to client requests.

**2. Implementation Steps**

The implementation of this project began with setting up a Java environment and creating two web service classes one for REST and another for SOAP.

For the RESTful web service, the RestaurantService class was developed. The GET method was used to retrieve restaurant data, while the POST method accepted new restaurant information from the client. After writing the code, the service was deployed on a local server, and its endpoints were tested using Postman. The GET requests returned sample data, and the POST requests successfully received input and returned confirmation messages, showing that the REST API was working properly.

Next, the SOAP web service was developed in the SoapService class. To test , SOAP requests were created in XML format and sent through Postman.

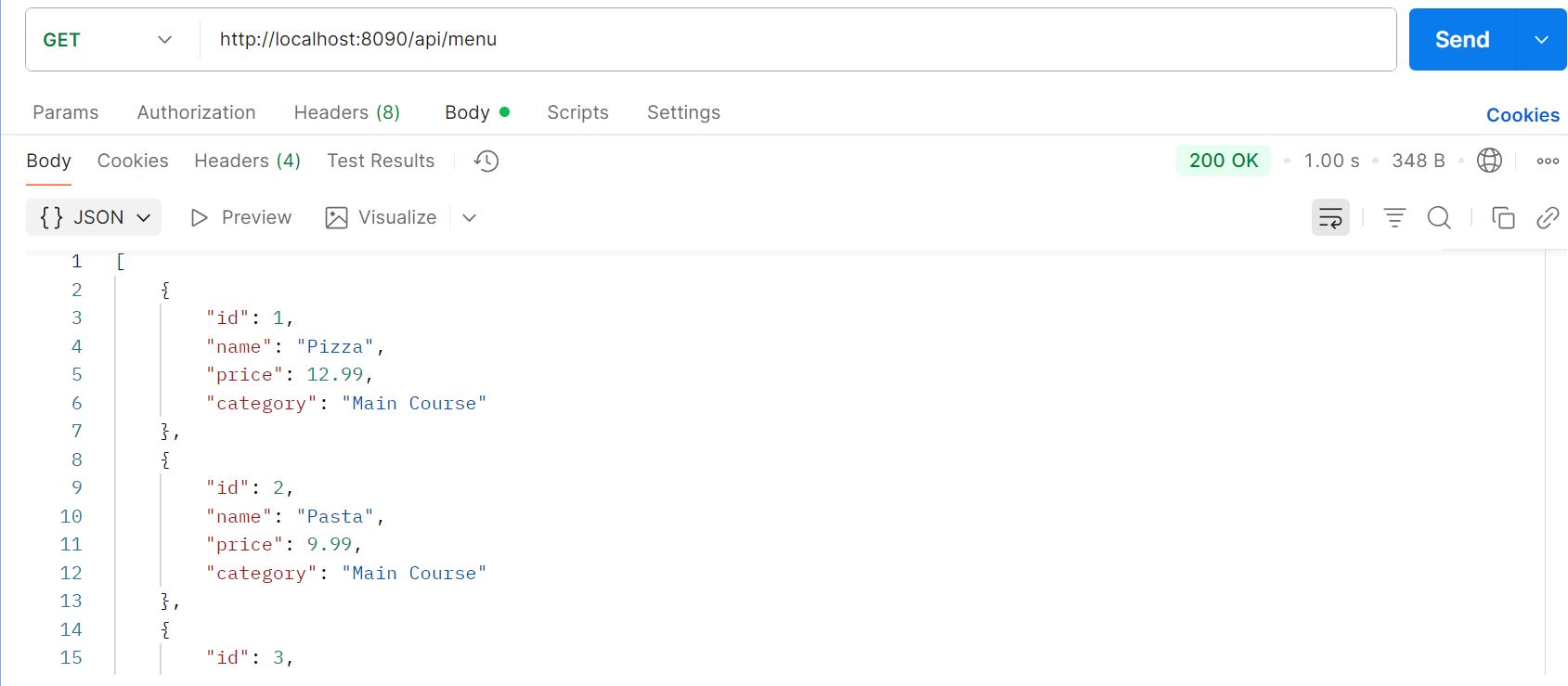
Through these steps, both web services were successfully implemented and tested. The RESTful service demonstrated the simplicity and flexibility of using JSON-based communication, while the SOAP service showed the structured and standardized nature of XML-based communication. Together, they provided a complete understanding of how different types of web services operate in Java.

1. **Tools and Technologies Used**

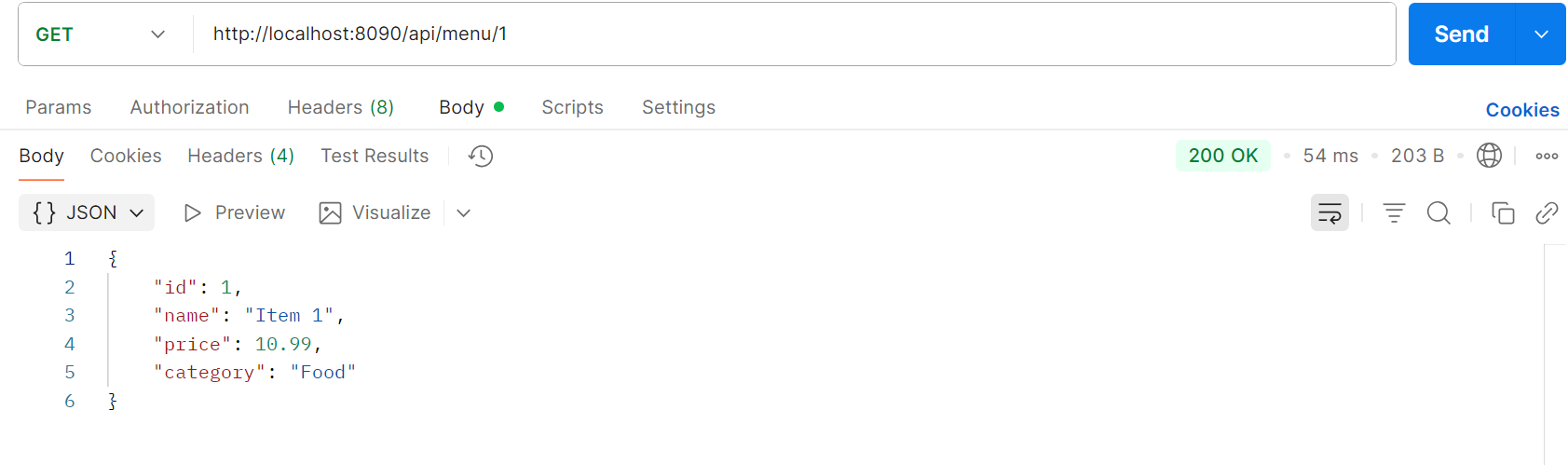
Several tools and technologies were used to develop and test the web services in this project. The main programming language used was **Java**, as it provides strong support for both REST and SOAP web service development. Testing was done using **Postman**. These tools helped send different types of requests (GET, POST, and SOAP XML) and view the corresponding responses from the server. Finally, the project used **JSON** for REST communication and **XML** for SOAP communication. These data formats are essential for web services because they define how information is structured and exchanged between the client and server.

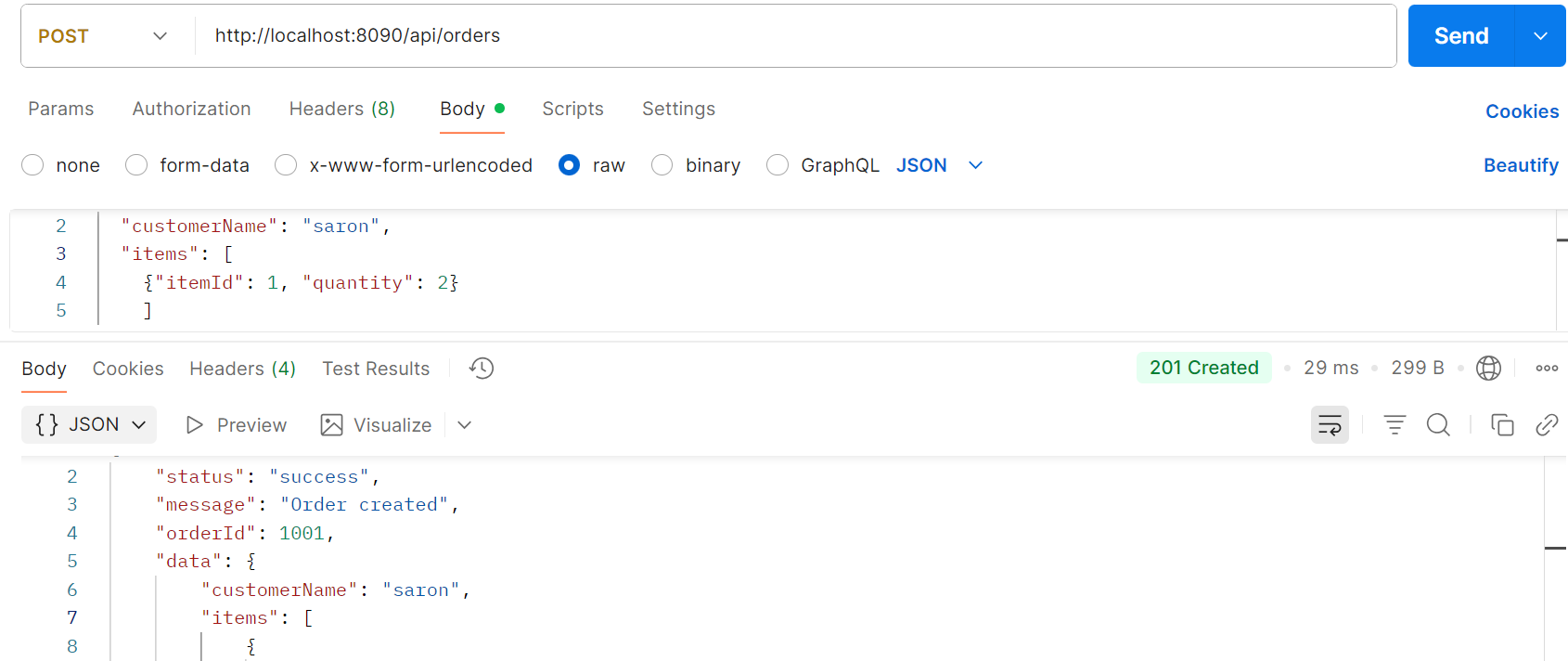
1. **Results and Observations**

The first endpoint, GET /api/menu, returns a JSON array containing a list of menu items.

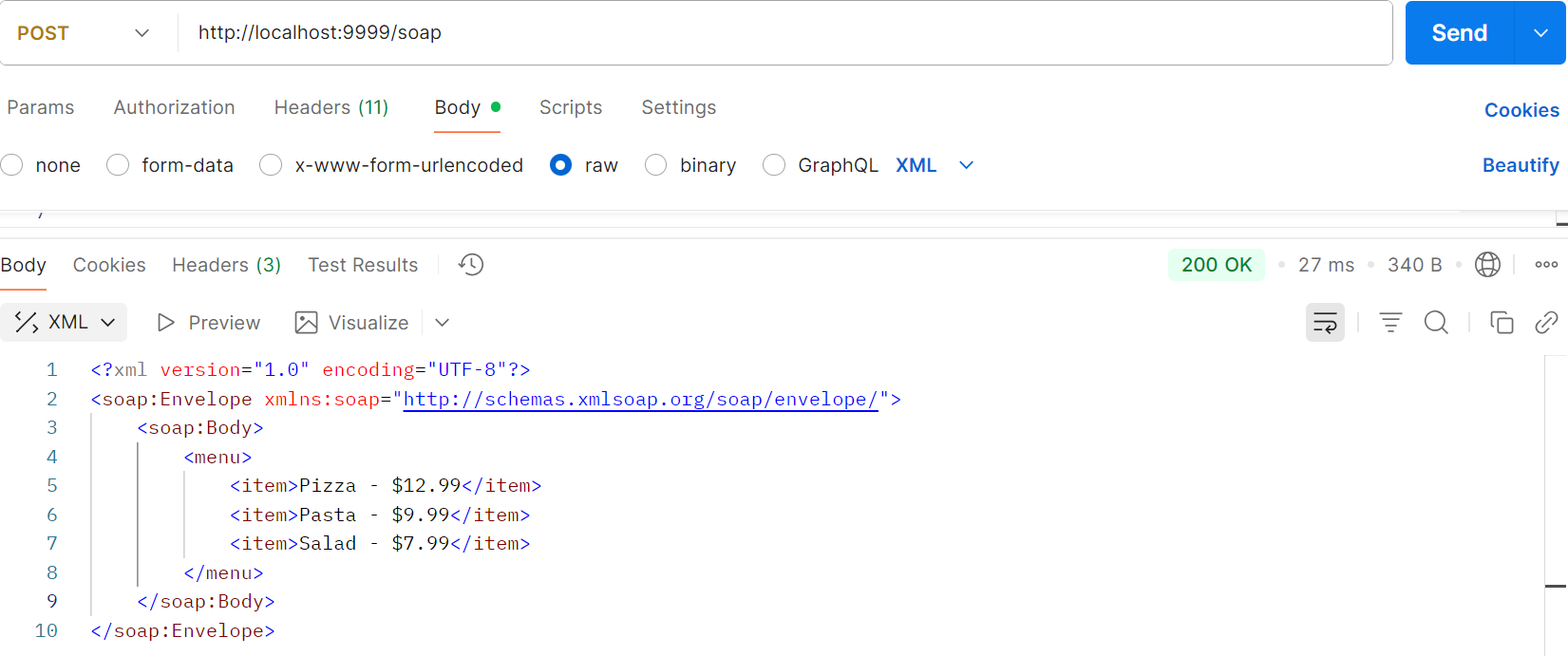


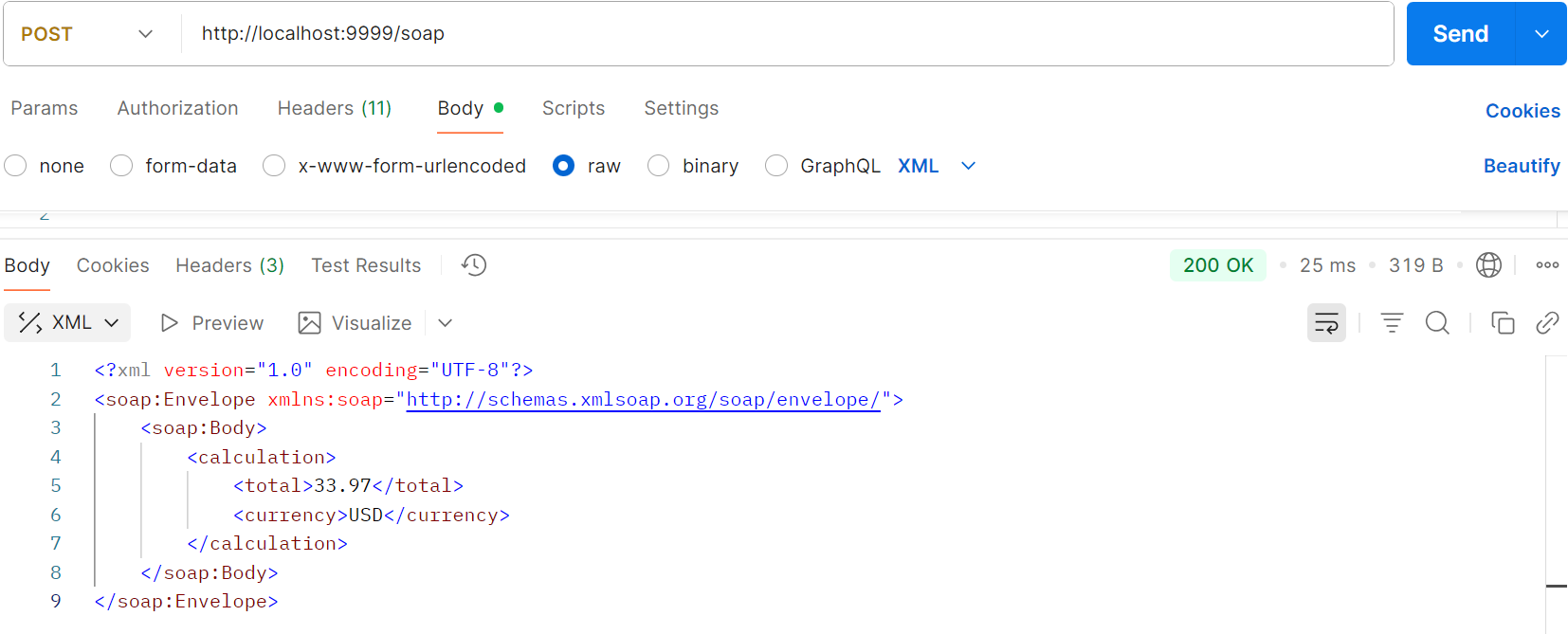
The second endpoint, GET /api/menu/{id}, retrieves a specific menu item based on its unique ID.



The third endpoint, POST /api/orders, accepts order details in JSON format and sends back a confirmation response.

The second part involved implementing the SOAP web service. This service was set up using an HttpServer running on port 9999 and included a single /soap endpoint that handled POST requests. Some of these requests and responses are shown in the image below. Since SOAP enables communication between different applications, this setup allows clients to send requests, such as **getting the menu** or **calculating order totals**, and receive structured responses in a standardized format.



****