

Atef Ltaief

3rd Year Software
Engineering

TD06 TP12



SOA University Project

SOA & Web Services (REST & SOAP) Architecture



O R B I T



Project Context

University information systems are complex and distributed.
Need for scalable and interoperable services.
Service-Oriented Architecture (SOA) is an effective solution.



Main Goal

Design and implement a complete SOA-based university system.

Project Objectives

Design a Service-Oriented Architecture.

Develop RESTful and SOAP web services.

Ensure interoperability between heterogeneous technologies.

Secure services using JWT.

Deploy services using Docker.



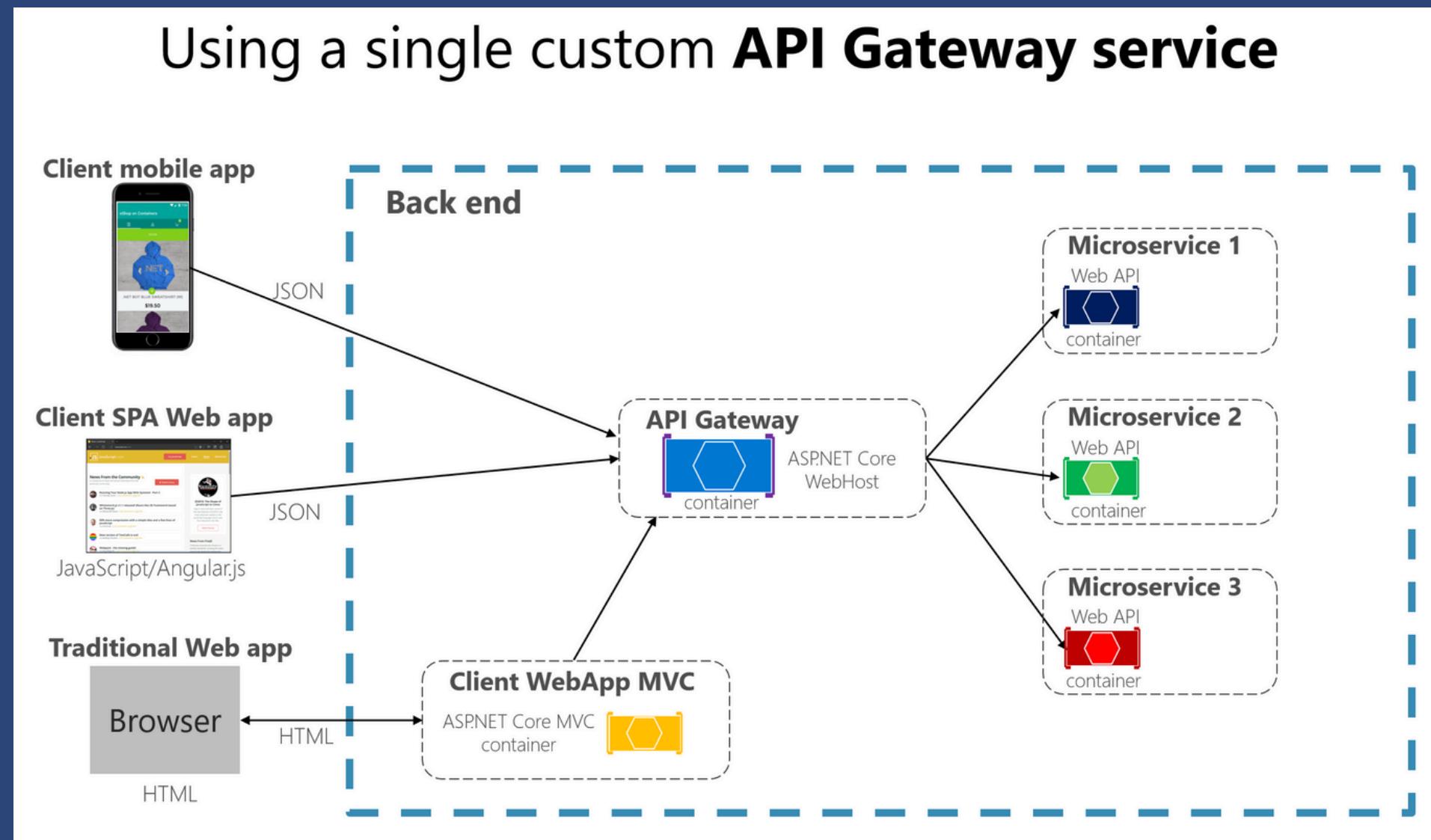
Why SOA Architecture?

Loose coupling between services.
Technology independence.
High scalability and flexibility.
Reusability of services.
Easy maintenance and evolution.



Global System Architecture

The system is composed of independent business services.
An API Gateway acts as a single entry point.
Services communicate via HTTP.

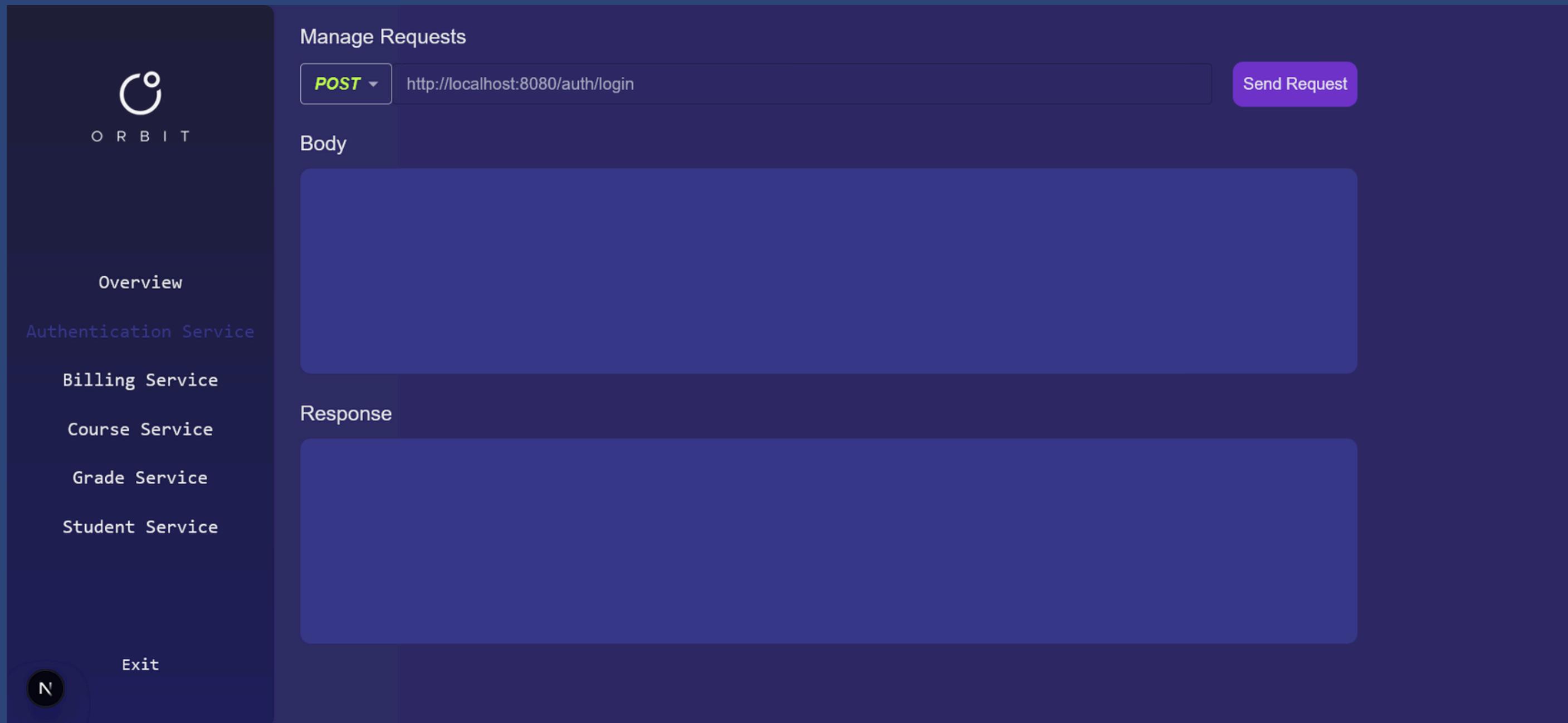


Authentication Service



The Authentication Service is a core component of the system, responsible for managing user access and ensuring secure communication between services. It authenticates users based on their credentials and generates JSON Web Tokens (JWT) used to authorize requests across the platform. By centralizing authentication, this service enhances security, reduces duplication, and enables a scalable and loosely coupled architecture.

Overview of the Authentication Service



POST request executed successfully with valid JSON data



O R B I T

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

N

Manage Requests

POST ▾ http://localhost:8080/auth/login

Send Request

Body

```
{  
  "username": "rayen.khaskhoussi",  
  "password": "T9!xQ2P7"  
}
```

Response OK

```
FAKE-JWT-TOKEN-FOR-rayen.khaskhoussi
```

This screenshot shows the Orbit application interface. On the left, there's a sidebar with service names: Authentication Service, Billing Service, Course Service, Grade Service, Student Service, and an Exit button with a 'N' icon. The main area is titled 'Manage Requests' and shows a POST request to 'http://localhost:8080/auth/login'. The 'Body' section contains a JSON object with 'username' and 'password' fields. Below it, the 'Response' section shows a status of 'OK' and the text 'FAKE-JWT-TOKEN-FOR-rayen.khaskhoussi'.



POST request with an invalid JSON payload



Manage Requests

POST ▾ http://localhost:8080/auth/login Send Request

Body

```
{  
  "username": "rayen.khaskhoussi",  
  "password": "T9!xQS2P7"  
}
```

Response Forbidden

N

Orbit

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

Student Service

The Student Service is responsible for managing student-related data within the system. It provides RESTful endpoints to perform Create, Read, Update, and Delete (CRUD) operations on student records. This service ensures efficient handling of student information and enables seamless integration with other services through standardized HTTP communication.



Student Service Interface

 O R B I T

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

Manage Requests

GET ▾ http://localhost:8080/students

Send Request

Response

Retreiving all Students

Manage Requests

GET ▾ http://localhost:8080/students Send Request

Response OK

```
[{"id":1,"name":"Rayen Khaskhoussi","email":"rayen.khaskhoussi@univ.tn","age":20,"cin":"14587963","country":"Tunisia","address":"Manouba, Tunisia"}, {"id":2,"name":"Ahmed Ben Salah","email":"ahmed.bensalah@univ.tn","age":22,"cin":"11223344","country":"Tunisia","address":"Sfax, Tunisia"}, {"id":3,"name":"Yasmine Trabelsi","email":"yasmine.trabelsi@univ.tn","age":21,"cin":"22334455","country":"Tunisia","address":"Ariana, Tunisia"}, {"id":4,"name":"Mehdi Chaabane","email":"mehdi.chaabane@univ.tn","age":23,"cin":"33445566","country":"Tunisia","address":"Nabeul, Tunisia"}, {"id":5,"name":"Sarra Jaziri","email":"sarра.jaziri@univ.tn","age":20,"cin":"44556677","country":"Tunisia","address":"Bizerte, Tunisia"}, {"id":6,"name":"Anis Khelifi","email":"anis.khelifi@univ.tn","age":24,"cin":"55667788","country":"Tunisia","address":"Gabès, Tunisia"}, {"id":7,"name":"Mariem Gharbi","email":"mariem.gharbi@univ.tn","age":19,"cin":"66778899","country":"Tunisia","address":"Kairouan, Tunisia"}, {"id":8,"name":"Houssem Ayari","email":"houssem.ayari@univ.tn","age":22,"cin":"77889900","country":"Tunisia","address":"Tunis, Tunisia"}, {"id":9,"name":"Nour El Houda Maatoug","email":"nour.maatoug@univ.tn","age":21,"cin":"88990011","country":"Tunisia","address":"Monastir, Tunisia"}, {"id":10,"name":"Oussama Belhadj","email":"oussama.belhadj@univ.tn","age":25,"cin":"99001122","country":"Tunisia","address":"Sousse, Tunisia"}]
```

Overview
Authentication Service
Billing Service
Course Service
Grade Service
Student Service

Exit N

Adding new Students

O R B I T

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

N

Manage Requests

POST ▾ <http://localhost:8080/students>

Send Request

Body

```
{
  "id": 1,
  "name": "Atef Ltaief",
  "email": "atefltaif@univ.tn",
  "age": 21,
  "cin": "14055277",
  "country": "Tunisia",
  "address": "Monastir, Tunisia"
}
```

Response Created

```
{"id":11,"name":"Atef Ltaief","email":"atefltaif@univ.tn","age":21,"cin":"14055277","country":"Tunisia","address":"Monastir, Tunisia"}
```

Updating Student information

The screenshot shows the Orbit application interface. On the left sidebar, there are several service icons: Authentication Service (key icon), Billing Service (credit card icon), Course Service (book icon), Grade Service (percentage icon), and Student Service (student icon). The Student Service icon is highlighted with a blue border. At the bottom of the sidebar is an 'Exit' button with a circular arrow icon.

The main area is titled "Manage Requests". It shows a "PUT" method selected and a URL "http://localhost:8080/students/1". A "Send Request" button is located to the right of the URL input.

The "Body" section contains the following JSON payload:

```
{  
  "name": "atef.ltaief"  
}
```

The "Response" section shows a successful "OK" status with the following JSON response:

```
{"id": 1, "name": "atef.ltaief", "email": "rayen.khaskhoussi@univ.tn", "age": 20, "cin": "14587963", "country": "Tunisia", "address": "Manouba, Tunisia"}
```

Updating Student information (case student does not exist)

The screenshot shows the Orbit application interface. On the left sidebar, there are several service links: Overview, Authentication Service, Billing Service, Course Service, Grade Service, and Student Service. The Student Service link is highlighted in blue, indicating it is currently selected. The main content area is titled "Manage Requests". A "PUT" dropdown menu is set to "http://localhost:8080/students/1". A "Send Request" button is located to the right of the URL input. Below the URL input, the "Body" section contains the following JSON payload:

```
{  
  "name": "Atef Ltaief modified",  
}
```

Under the "Response" section, a red "Bad Request" message is displayed, indicating that the request failed because the student with ID 1 does not exist.

Deleting Student (case not found)

Manage Requests

DELETE ▾ http://localhost:8080/students/1

Send Request

Response Not Found

```
{"message": "Student not found"}
```

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

N

Deleting Student (case not found)

Manage Requests

DELETE ▾ <http://localhost:8080/students/1>

Send Request

Response **OK**

```
{"id":1,"name":"atef.ltaief","email":"rayen.khaskhoussi@univ.tn","age":20,"cin":"14587963","country":"Tunisia","address":"Manouba, Tunisia"}
```

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

N

Course Service

The Course Service is responsible for managing academic courses and schedules within the university system. Implemented as a SOAP-based service, it provides well-defined operations through a WSDL contract, ensuring reliable and standardized communication. This service enables structured data exchange and interoperability with other system components.



Retreiving All Courses

Manage Requests

POST ▾ http://localhost:8080/courses/CourseService?wsdl

Send Request

Function & Arguments

```
{  
    "operation": "getAllCourses",  
    "args": {}  
}
```

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

N

Response OK

```
{  
    "return": [  
        {  
            "code": "SI201",  
            "credits": 3,  
            "schedule": "Mar 14:00-16:00",  
            "teacher": "Pr. X",  
            "title": "Systèmes d'Information"  
        }  
    ]  
}
```

Retreiving All Courses

Manage Requests

POST ▾ http://localhost:8080/courses/CourseService?wsdl

Send Request

Function & Arguments

```
{  
    "operation": "getCourseByCode",  
    "args": { "code": "GL101" }  
}
```

Response **OK**

```
{  
    "return": {  
        "code": "GL101",  
        "credits": 7,  
        "schedule": "Mon 10:00-12:00",  
        "teacher": "Dr. Smith",  
        "title": "Algorithms"  
    }  
}
```

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

Update Specific Course by code

Manage Requests

POST ▾ <http://localhost:8080/courses/CourseService?wsdl>

Send Request

Function & Arguments

```
{  
    "operation": "updateCourse",  
    "args": {  
        "course": {  
            "code": "GL101",  
            "title": "Algorithms",  
            "credits": 7,  
            "teacher": "Dr. Smith",  
            "schedule": "Mon 10:00-12:00"  
        }  
    }  
}
```

Response **OK**

```
{  
    "return": true  
}
```

Exit



O R B I T

- Overview
- Authentication Service
- Billing Service
- Course Service
- Grade Service
- Student Service

N

Delete Course by code

Manage Requests

POST ▾ http://localhost:8080/courses/CourseService?wsdl

Send Request

Function & Arguments

```
{  
    "operation": "deleteCourse",  
    "args": { "code": "GL101" }  
}
```

Response OK

```
{  
    "return": true  
}
```

Exit

O R B I T

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

N

Handling unkown function error

The screenshot shows the ORBIT application interface. On the left sidebar, there are several service icons: Overview, Authentication Service, Billing Service, Course Service, Grade Service, Student Service, and Exit. The main area is titled "Manage Requests". It shows a POST request to "http://localhost:8080/courses/CourseService?wsdl" with the following JSON payload:

```
{  
  "operation": "unkownFunction",  
  "args": { "code": "GL101" }  
}
```

The response is a "Bad Request" with the following JSON message:

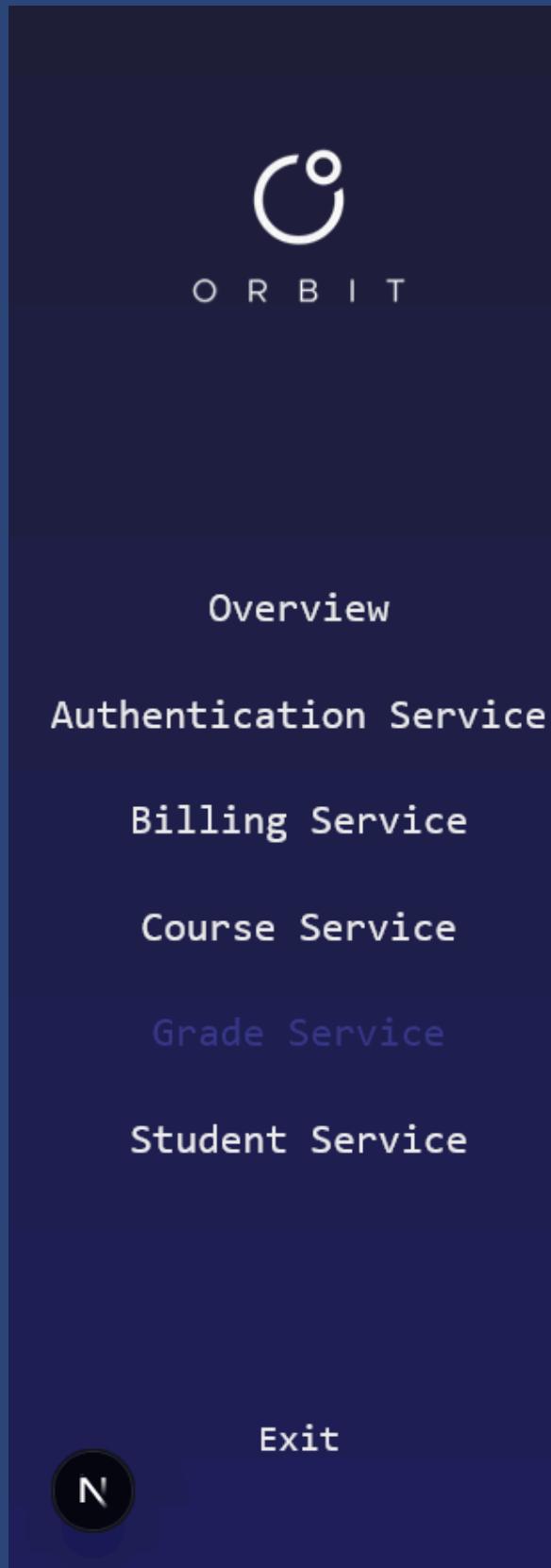
```
{  
  "message": "Invalid SOAP operation",  
  "allowed": [  
    "getAllCourses",  
    "getCourseByCode",  
    "addCourse",  
    "updateCourse",  
    "deleteCourse"  
  ]  
}
```

Grade Service



The Grade Service is responsible for managing students' grades and calculating academic averages. Implemented as a RESTful service using FastAPI, it provides efficient endpoints for adding grades and retrieving computed averages. This service ensures high performance, simplicity, and seamless integration with other system components.

Adding a grade



Manage Requests

POST ▾ <http://localhost:8080/grades>

Send Request

Body

```
{  
  "cin": "12445678",  
  "course_code": "GL10122",  
  "value": 16  
}
```

Response

```
{  
  "cin": "12445678",  
  "course_code": "GL10122",  
  "value": 16  
}
```

Retreiving all grades

Manage Requests

GET ▾ http://localhost:8080/grades

Send Request

Response OK

```
[  
  {  
    "cin": "14587963",  
    "course_code": "GL101",  
    "value": 15.5,  
    "id": 1  
  },  
  {  
    "cin": "14587963",  
    "course_code": "BD202",  
    "value": 13,  
    "id": 2  
  },  
  {  
    "cin": "11223344",  
    "course_code": "NET301",  
    "value": 16,  
    "id": 3  
  },  
  {  
    "cin": "11223344",  
    "course_code": "NET301",  
    "value": 16,  
    "id": 4  
  }]
```

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

N

Retreiving a Specific Grade

Manage Requests

PUT ▾ http://localhost:8080/grades/1

Send Request

Body

```
{  
    "cin": "111111111",  
    "course_code": "GL101",  
    "value": 18.0  
}
```

Response OK

```
{  
    "cin": "111111111",  
    "course_code": "GL101",  
    "value": 18,  
    "id": 1  
}
```

Exit

O R B I T

- Overview
- Authentication Service
- Billing Service
- Course Service
- Grade Service
- Student Service

N

Deleting a Specific Grade

Manage Requests

DELETE ▾ <http://localhost:8080/grades/1>

Send Request

Response No Content

The screenshot shows the Orbit application's main interface. On the left, there is a sidebar with several service names listed: Overview, Authentication Service, Billing Service, Course Service, Grade Service, Student Service, and Exit. The Grade Service item is highlighted with a blue background. The main area has a dark blue header with the text "Manage Requests". Below the header is a form with a "DELETE" button, a dropdown menu, and a URL input field containing "http://localhost:8080/grades/1". To the right of the URL is a purple "Send Request" button. Underneath the form, the word "Response" is followed by a green button labeled "No Content". The rest of the main area is a large, semi-transparent blue rectangle.

Handling Unknown Grade ID

The screenshot shows the Orbit application interface. On the left, there is a sidebar with the Orbit logo and several service links: Overview, Authentication Service, Billing Service, Course Service, Grade Service, Student Service, and Exit. A small circular icon with the letter 'N' is at the bottom left.

The main area is titled "Manage Requests". It shows a "POST" method selected and a URL "http://localhost:8080/grades/1234". There is a "Send Request" button. Below this, there is a large blue "Body" section which is currently empty.

At the bottom, there is a "Response" section showing a "Bad Request" status. The response body contains the following JSON:

```
{  
    "detail": "Method Not Allowed"  
}
```

Billing Service

The Billing Service is responsible for managing university fees and financial transactions related to students. Implemented as a SOAP-based service, it provides standardized operations for retrieving fee details and generating invoices. This service ensures reliable, secure, and contract-driven communication within the SOA architecture.



Retreiving all Invoices

Manage Requests

POST ▾ <http://localhost:8080/billing/FacturationService.svc>

Send Request

Function & Arguments

```
{  
  "operation": "GetAllInvoices",  
  "args": {}  
}
```

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

N

Response OK

```
{  
  "GetAllInvoicesResult": {  
    "Invoice": [  
      {  
        "Amount": 450,  
        "CreatedAt": "2025-12-15T08:48:52.499Z",  
        "Description": "Tuition fees",  
        "Id": 1,  
        "IsPaid": false  
      },  
      {  
        "Amount": 300,  
        "CreatedAt": "2025-12-15T08:48:52.499Z",  
        "Description": "Books fees",  
        "Id": 2,  
        "IsPaid": true  
      }  
    ]  
  }  
}
```

Create an Invoice for payment

Manage Requests

POST ▾ <http://localhost:8080/billing/FacturationService.svc>

Send Request

Function & Arguments

```
{  
  "operation": "CreateInvoice",  
  "args": {  
    "cin": "14587963",  
    "amount": 450.0,  
    "description": "Tuition fees"  
  }  
}
```

Response **OK**

```
{  
  "CreateInvoiceResult": {  
    "Amount": 450,  
    "CreatedAt": "2025-12-15T08:48:52.499Z",  
    "Description": "Tuition fees",  
    "Id": 1,  
    "IsPaid": false  
  }  
}
```

Exit

N

Retrieving Invoice by Student CIN

Manage Requests

POST ▾ http://localhost:8080/billing/FacturationService.svc

Send Request

Function & Arguments

```
{  
  "operation": "GetInvoicesByStudent",  
  "args": {  
    "cin": "14587963"  
  }  
}
```

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

N

Response OK

```
{  
  "GetInvoicesByStudentResult": {  
    "Invoice": [  
      {  
        "Amount": 450,  
        "CreatedAt": "2025-12-15T08:48:52.499Z",  
        "Description": "Tuition fees",  
        "Id": 1,  
        "IsPaid": false  
      },  
      {  
        "Amount": 300,  
        "CreatedAt": "2025-12-15T08:48:52.499Z",  
        "Description": "Activity fees",  
        "Id": 2,  
        "IsPaid": false  
      }  
    ]  
  }  
}
```

Paying Invoice

Manage Requests

POST ▾ <http://localhost:8080/billing/FacturationService.svc>

Send Request

Function & Arguments

```
{  
  "operation": "PayInvoice",  
  "args": {  
    "invoiceld": 1  
  }  
}
```

Response **OK**

```
{  
  "PayInvoiceResult": true  
}
```

Overview

Authentication Service

Billing Service

Course Service

Grade Service

Student Service

Exit

N

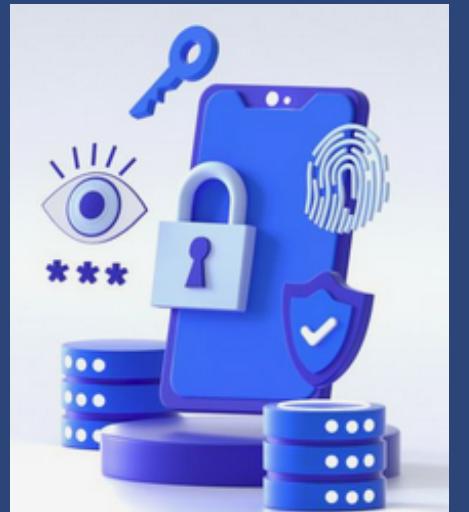
API Gateway

The API Gateway acts as a single entry point for all client requests, routing them to the appropriate backend services. It centralizes authentication, request forwarding, and load balancing, ensuring secure and efficient communication between clients and the distributed services in the SOA architecture.



Security

The Security module ensures that only authorized users can access the system services. It relies on JWT (JSON Web Tokens) for authentication, validating tokens at the API Gateway level, and securing internal communication between services. This approach provides centralized, scalable, and robust protection for the SOA architecture.



Deployment & Containerization

The system is fully containerized using Docker, and all services are orchestrated via Docker Compose. Each service runs in an isolated container with its own environment and network configuration, enabling easy deployment, scalability, and maintenance. This approach ensures seamless communication between services while simplifying the setup of the entire SOA architecture.



Project Organization

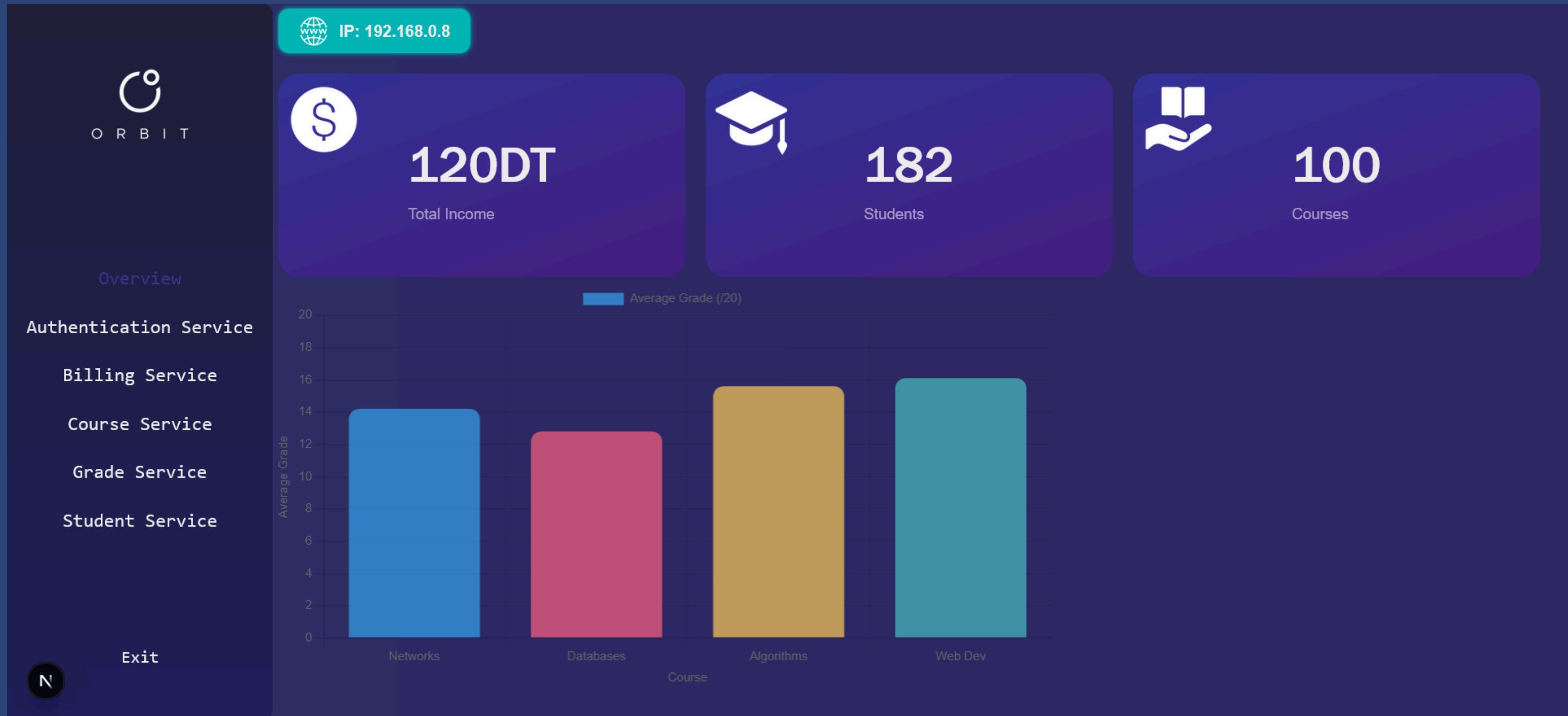
The project is structured to ensure clarity, maintainability, and effective teamwork. Services are separated into individual folders, documentation is centralized, and deployment configurations are organized under a dedicated directory. This organization supports collaborative development, agile practices, and simplifies navigation and future maintenance of the SOA system.

Conclusion

In conclusion, the SOA-based University Information System demonstrates a fully functional architecture integrating REST and SOAP services across multiple technologies. The system ensures interoperability, scalability, security, and maintainability, while Docker-based deployment simplifies orchestration. Future improvements may include enhanced security, real database integration, and monitoring for better performance and reliability.



Final Overview



Thank You

For listening