

**ASCENDION**

# Capstone Project



## Personal BACKGROUND

(Name, Past Experience, Qualification, Career Summary)

---

**Name : CH Syamala Pranavi**

**Past Experience : Fresher**

**Qualification : BTech CSE**

**Career Summary :**

**Graduated with a degree in Computer Science and commenced my professional journey with Ascendion Engineering Private Limited.**

ASCENDION



## Key Takeaways/Learnings from the Program (HTD)

---

- Gained hands-on experience with Spring Boot and Microservices architecture.
- Developed and deployed microservices using Spring Cloud and AWS, including EC2 instances.
- Learned to integrate and manage REST APIs with Postman.
- Mastered database management with MySQL and ORM using Spring Data JPA.
- Ensured code quality through Sonarlint checks, unit testing, and JIRA for project management.
- Gained experience in working with others, understanding team dynamics, and contributing to a collective goal.





# ASCENDION

## Problem Statement of the Capstone Project

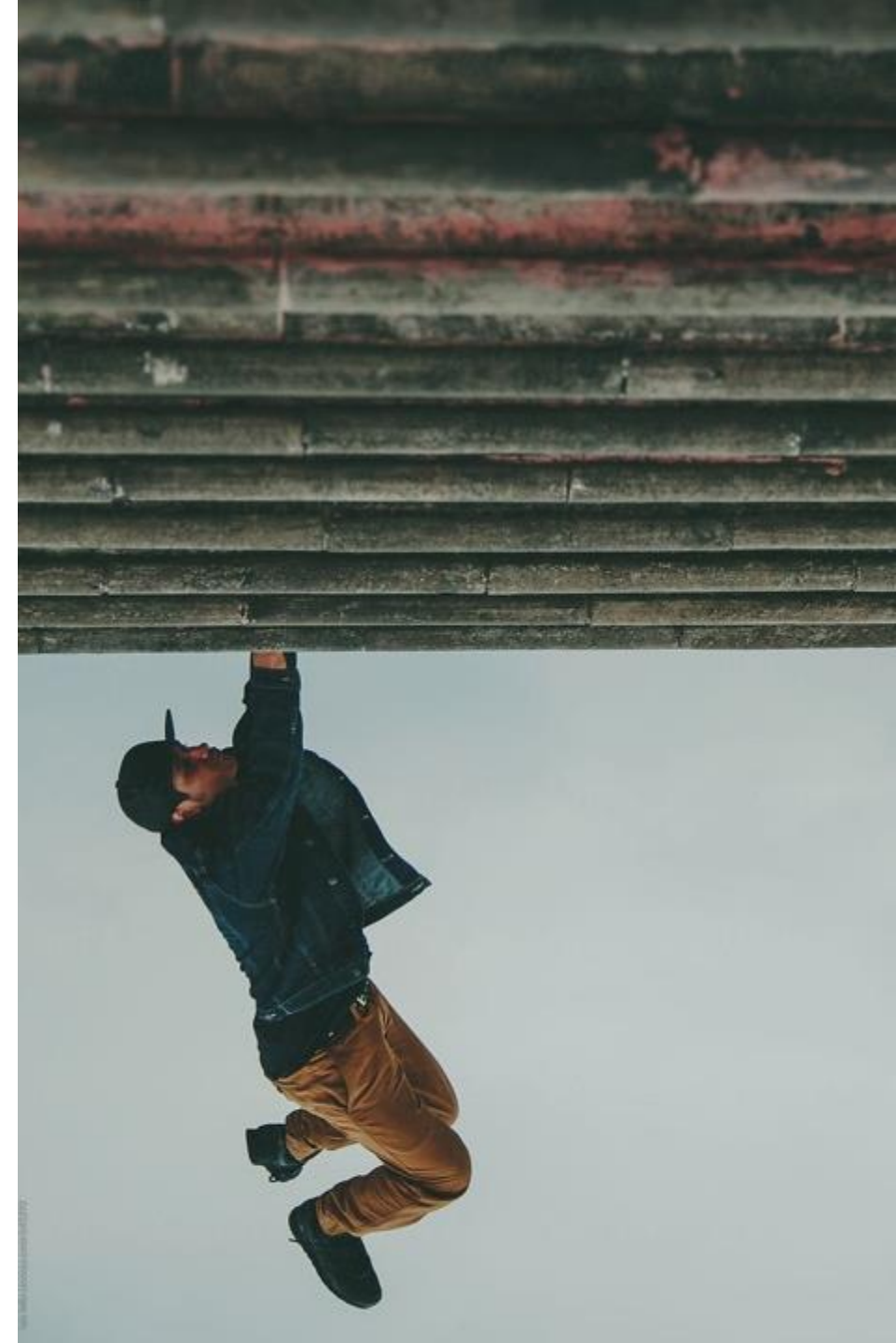
---

### Air Transport System (ATS)

#### Objective:

Develop an **Air Transport System (ATS)** for administrators to manage:

- **Bookings**
- **Airports**
- **Planes**



# Microservices Implemented

---

## Login Microservice

### Functionality:

- Handles user authentication.
- Validates login credentials.
- Manages failed login attempts and account locking.
- Provides error messages for incorrect credentials or locked accounts.

### Features:

- Locks accounts after three failed login attempts.
- Resets failed attempts and unlocks accounts on successful login.
- Integrates with the Booking microservice to fetch booking details upon successful login.

## Booking Microservice

### Functionality:

- Manages booking information and CRUD operations.
- Provides endpoints to fetch all bookings.

### Features:

- Implements a BookingDto model for booking details.
- Supports operations like creating, reading, updating, and deleting bookings.
- Provides an API endpoint to retrieve all bookings, which is consumed by the Login microservice after a successful login.

## Signup Microservice

### Functionality:

- Manages user registration.
- Handles signup requests and validates input.

### Features:

- Checks if the email already exists in the system.
- Provides appropriate messages if the user exists or if registration is successful.
- Stores new user details in the login\_details table.

## Plane Microservice

### Functionality:

- Manages plane-related information.
- Handles CRUD operations for planes.

### Features:

- Contains fields such as ID, name, and capacity.
- Implements all necessary CRUD operations for plane management.

## Airport Microservice

### Functionality:

- Manages airport-related information.
- Handles CRUD operations for airports.

### Features:

- Contains fields such as ID, name, city, and country.
- Implements all necessary CRUD operations for airport management.

## Key Implementations

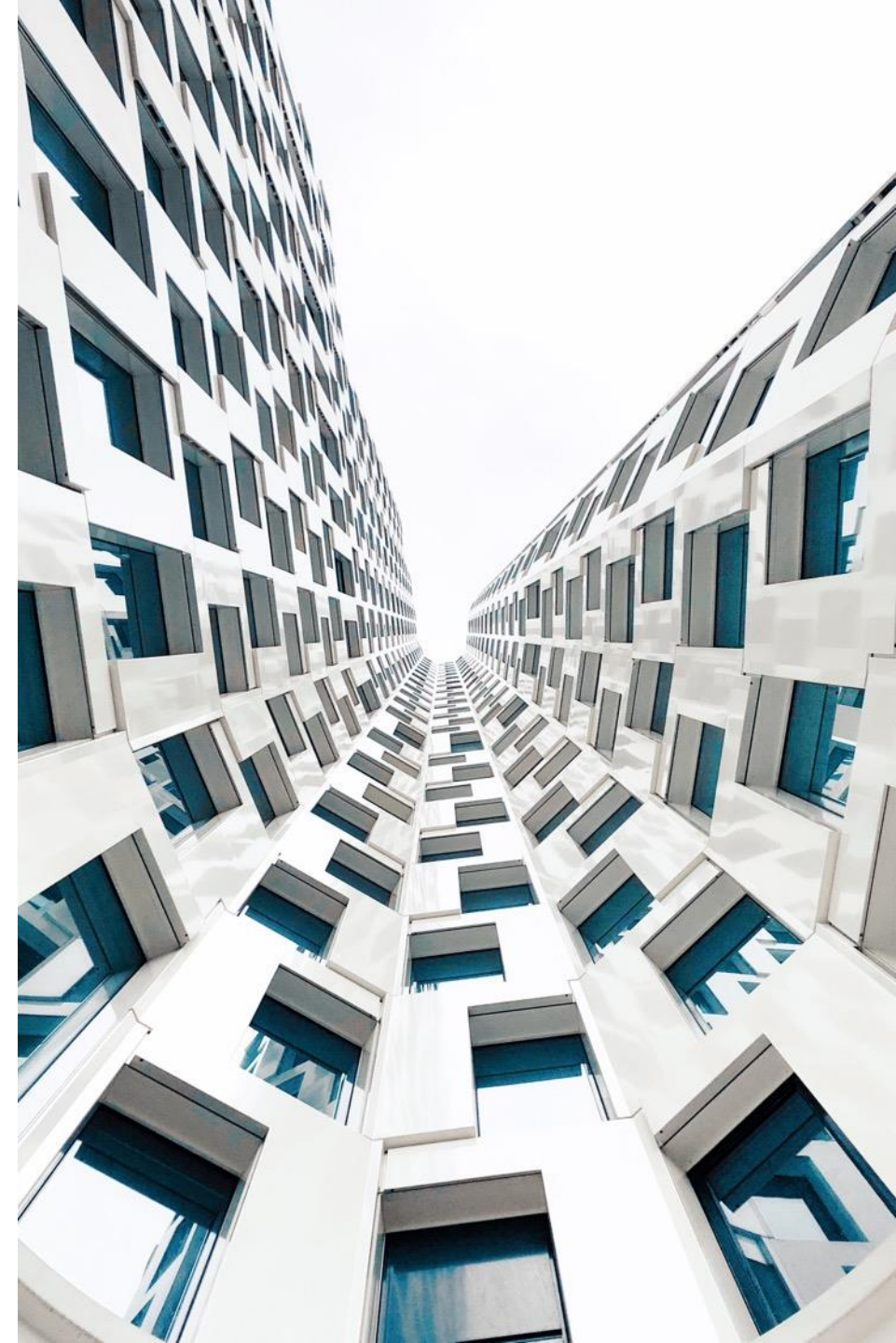
---

### Service Integration:

- **FeignClient:** Used for communication between microservices. For example, the Login microservice uses BookingFeignClient to fetch booking details from the Booking microservice.

### Error Handling and Validation:

- **Login Microservice:** Implements password validation, failed login attempt tracking, account locking, and error messaging.
- **Signup Microservice:** Ensures unique email addresses and handles user registration.





## Microservices Architecture:

- **Eureka Server:** Used for service discovery.
- **Spring Boot & Spring Cloud:** Used for developing and managing microservices.



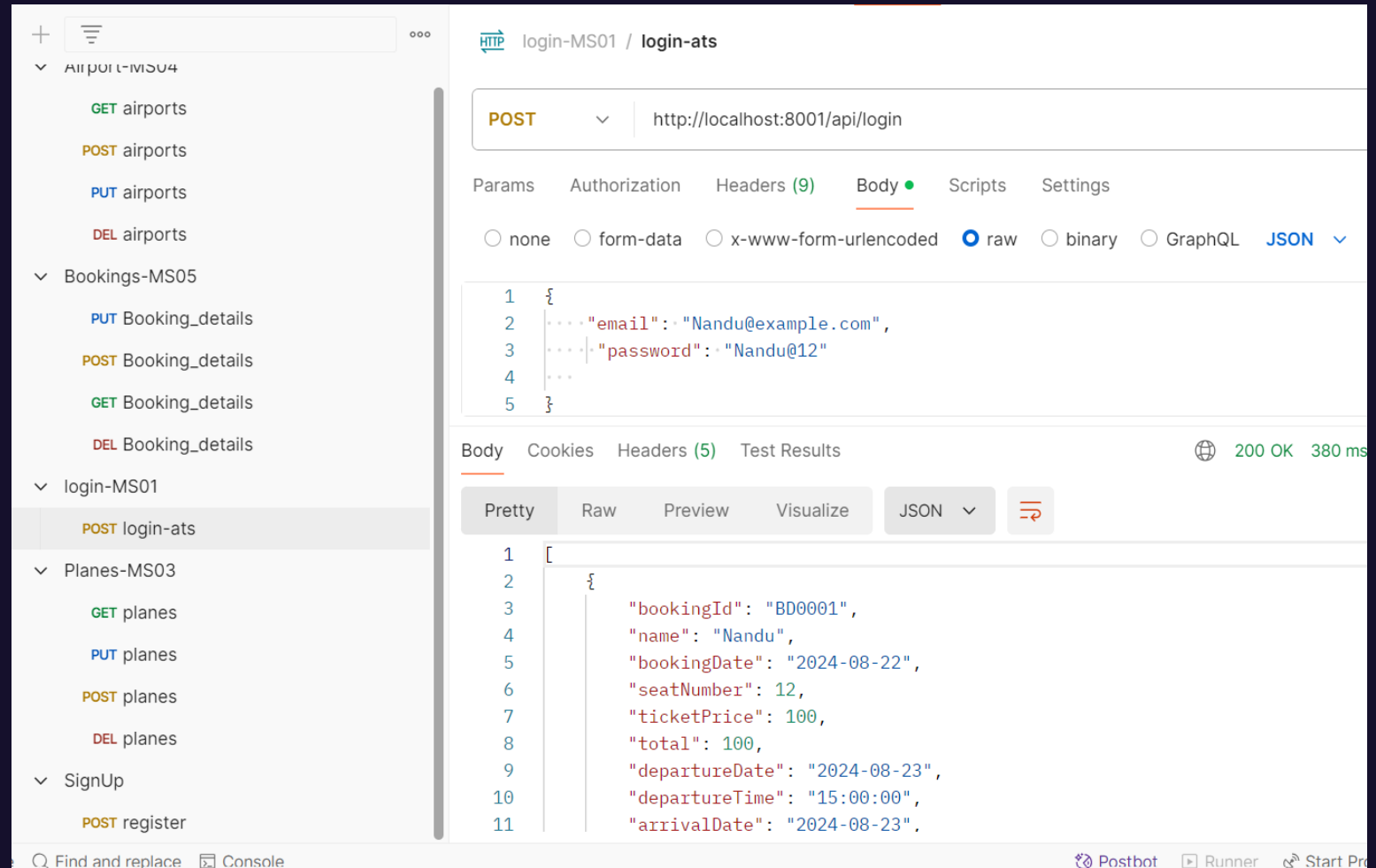
# ASCENDION

## ER Diagram



## Important areas of the Project with screenshots

Upon successful login, the system automatically retrieves and displays the user's booking details



## Important areas of the Project with screenshots

### Validating password Field

POST http://localhost:8001/api/login

Params Authorization Headers (9) **Body** Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON**

```
1 {
2   ... "email": "Nandu@example.com",
3   ... "password": "Nandu@2"
4   ...
5 }
```

Body Cookies Headers (5) Test Results **401 Unauthorized**

Pretty Raw Preview Visualize Text

```
1 Invalid password. Attempt 1 of 3.
```

POST http://localhost:8001/api/login

Params Authorization Headers (9) **Body** Scripts Settings

☐ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL **JSON**

```
1 {
2   ... "email": "Nandu@example.com",
3   ... "password": "Nandu@2"
4   ...
5 }
```

Body Cookies Headers (5) Test Results **403 Forbidden**

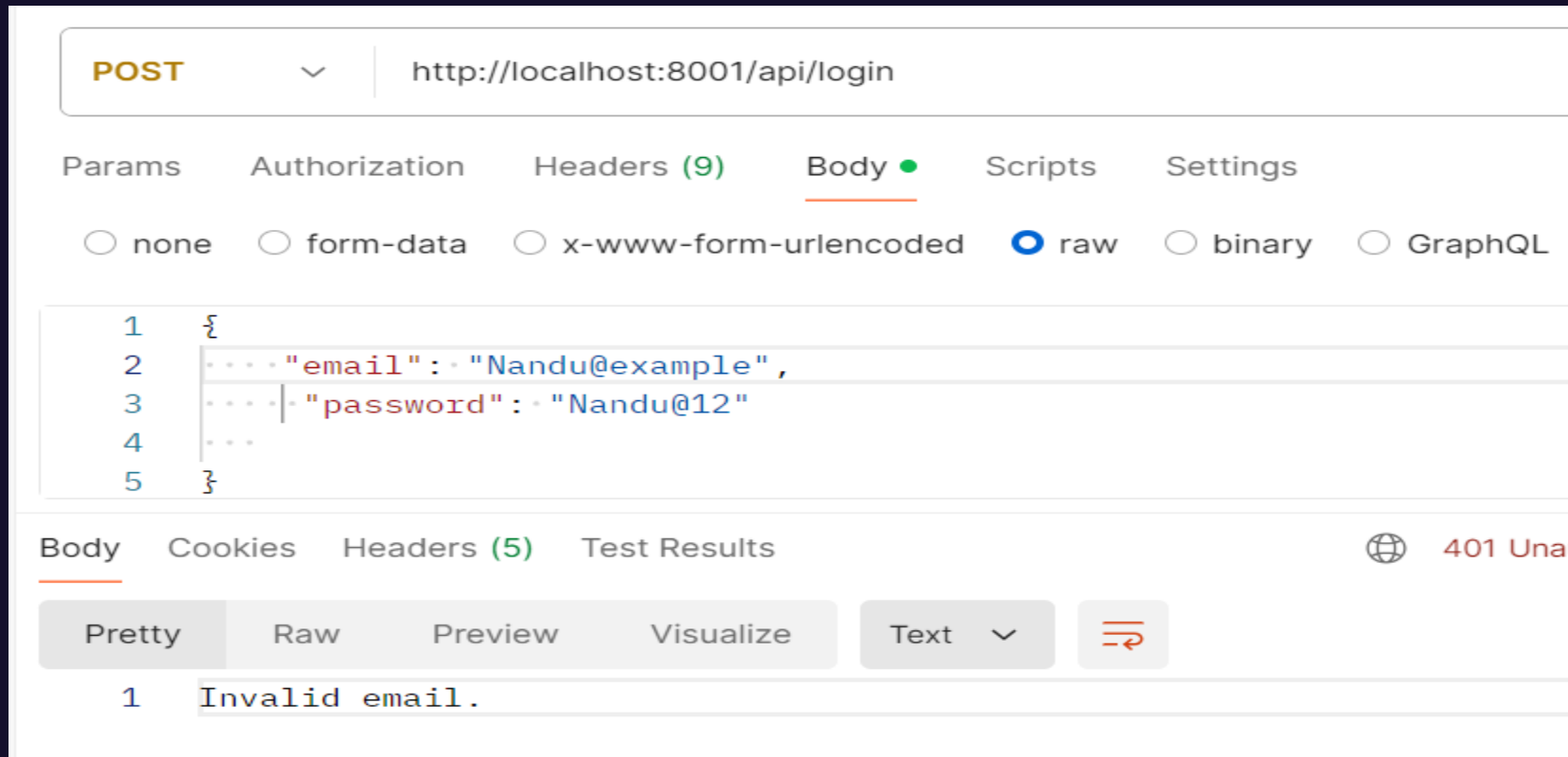
Pretty Raw Preview Visualize Text

```
1 Invalid password. Account locked for 30 minutes.
```



## Important areas of the Project with screenshots

### Validate Email field



The screenshot displays a REST client interface for a POST request to `http://localhost:8001/api/login`. The request body is a JSON object with the following structure:

```
1 {  
2   ... "email": "Nandu@example",  
3   ... "password": "Nandu@12"  
4   ...  
5 }
```

The response status is **401 Unauthorized**. The response body, viewed in the 'Pretty' format, contains the message:

```
1 Invalid email.
```

# Important areas of the Project with screenshots

## Login\_details table with Last\_login Implementation

SELECT \* FROM login\_details LIMIT 100

🔍 Search results

⚙️

📧 1

🔄

+

+

🗑️

↺

↻

↓ Export

▶️

👁️

Cost: 50ms

< 1 >

Total 8

* name varchar(255)	* email_id varchar(255)	* password varchar(255)	* phone_number varchar(20)	* failed_attempts int	* is_account_locked tinyint(1)	lock_time datetime	last_login datetime
Filter	Filter	Filter	Filter	Filter	Filter	Filter	Filter
Chintu	chintu@example.com	Chintu123!	9666111333	0	0	(NULL)	2024-08-23 10:22:27
Nandu	Nandu@example.com	Nandu@12	9106111533	3	1	2024-08-26 10:27:44	2024-08-25 20:59:01
Bindu	Bindudu@example.com	Binduu!65	9000002533	0	0	(NULL)	(NULL)
Chandu	Chandu@example.com	Chan@125	9002302533	0	0	(NULL)	(NULL)
Raju ch	raju@example.com	Rajuu@15	9000543210	0	0	(NULL)	(NULL)
Manvi	manu@example.com	Manvi@10	9002306233	0	0	(NULL)	(NULL)
Manvi	manuuu@example.com	Manvi@10	9023306233	0	0	(NULL)	(NULL)
Tanvi	Tanvii@example.com	Tanvi@110	9023306033	0	0	(NULL)	(NULL)

## Important areas of the Project with screenshots

Services registered in Eureka Server

localhost			
Instances currently registered with Eureka			
Application	AMIs	Availability Zones	Status
AIRPORT-MS04-ATS	n/a (1)	(1)	UP (1) - <a href="#">ASCINLAP60800:Airport-MS04-ATS:8031</a>
BOOKINGS-MS05-ATS	n/a (1)	(1)	UP (1) - <a href="#">ASCINLAP60800:Bookings-MS05-ATS:8041</a>
LOGIN-MS01-ATS	n/a (1)	(1)	UP (1) - <a href="#">ASCINLAP60800:login-MS01-ATS:8001</a>
PLANES-MS03-ATS	n/a (1)	(1)	UP (1) - <a href="#">ASCINLAP60800:planes-MS03-ATS:8021</a>
SIGNUP-MS02-ATS	n/a (1)	(1)	UP (1) - <a href="#">ASCINLAP60800:SignUp-MS02-ATS:8011</a>
General Info			
Name	Value		
total-avail-memory	94mb		
num-of-cpus	12		
current-memory-usage	60mb (63%)		

## Conclusion

---

This project demonstrates the successful implementation of a microservices architecture for an Air Transport System, providing a scalable and maintainable solution for managing various aspects such as airports, planes, schedules, bookings, and users. The system effectively integrates multiple microservices, each responsible for specific functionalities, ensuring modularity and ease of development. With robust features like user authentication, secure login, and seamless interaction between services using Feign clients, the project showcases the practical application of Spring Boot, Spring Cloud, and other technologies to build a cohesive and efficient application. The solution not only meets the current requirements but is also designed to accommodate future enhancements and scalability needs.





**Thank You**

**ASCENDION**