**Student Management System - Design Document**

Table of Contents

[1. Overview 2](#_Toc27704822)

[2. Solution Architecture 2](#_Toc27704823)

[3. DB Layer 4](#_Toc27704824)

[4. Swagger API documentation 4](#_Toc27704825)

[5. Attachments 4](#_Toc27704826)

[6. Metadata 5](#_Toc27704827)

# Overview

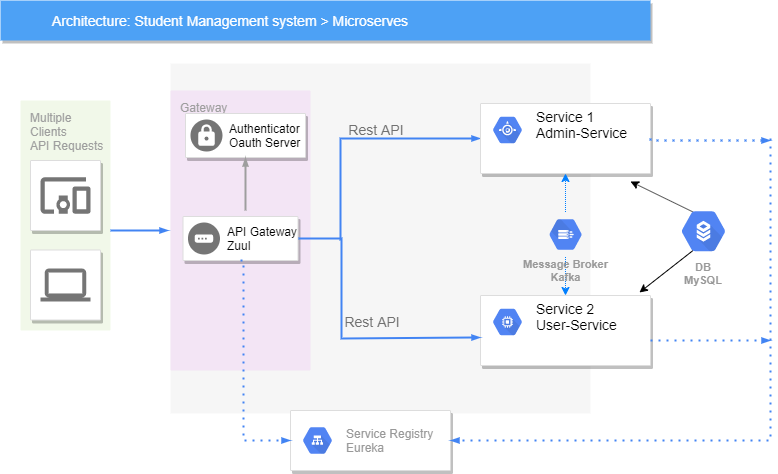
Student management system lets users view student and faculty details as well as manage courses and attendance

Scope

* The aim is to build a system that manages the student details.
* Add/Edit/Delete students and get detailed report for each student
* View teacher schedules based on their Id.
* Update student attendance for different courses

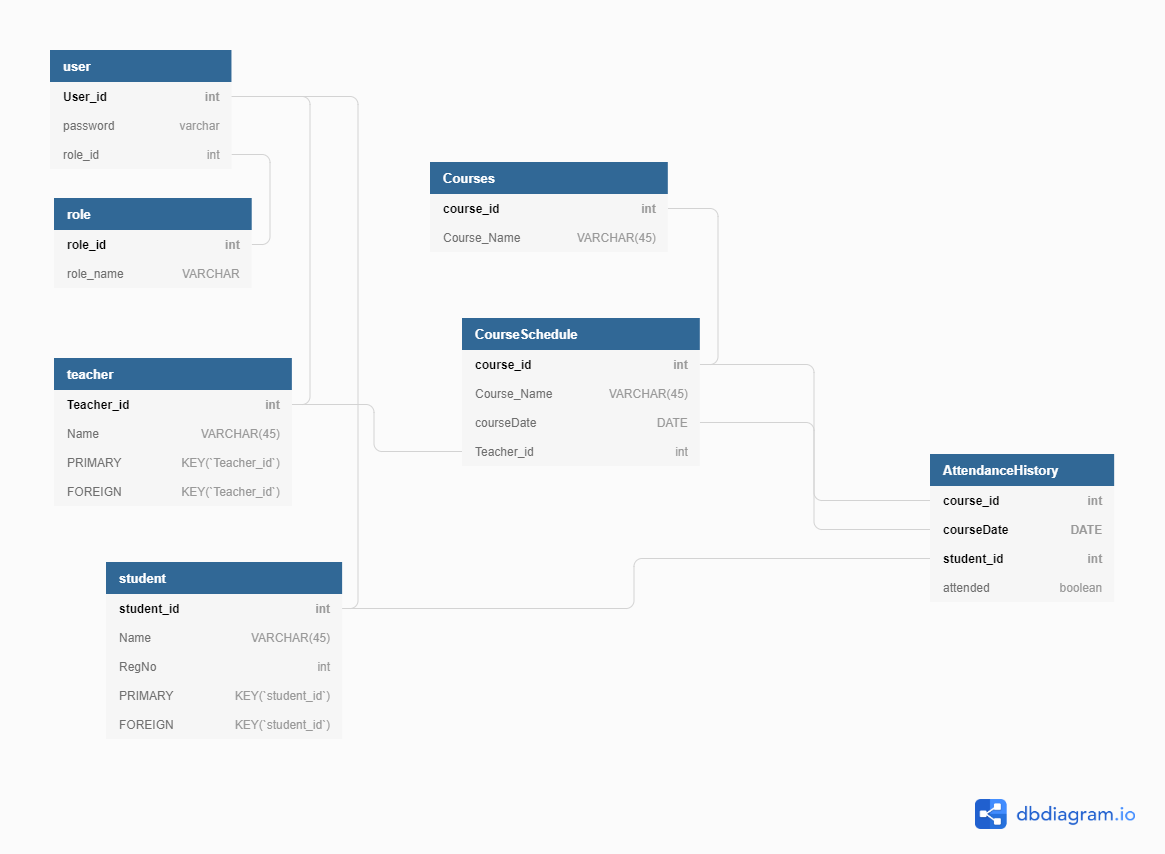
# Solution Architecture

Figure 1: Architecture Diagram



* The Application has been implemented using the **Microservice Architecture**.
* **API Gateway** – Zuul server will be used as the API gateway and it routes all the client’s requests to the individual services.
* **Authorization** – The Authorization will be implemented using OAuth 2.0.
  + The user specific functionalities are restricted using roles (ADMIN and USER).
  + Passwords are encrypted using BCrypt encoder.
  + JDBC Token store is used for saving the token to the database.
  + Grant Type – ‘password’ has been implemented.
  + The OAuth service uses the ‘**user** table of the **StudentManagement** database for authentication
  + The token will be attached with each requests with header ‘***Authorization***’.
* **Registry** – Eureka server has been used as the registry server.
* **Message Broker** 
  + Apache Kafka has been used as the message broker and it is used for the inter-service communications where no response is required back i.e., calls to update some DB Entries (to update status of the books from Member service)
  + For other inter service communication which requires response back (i.e., For User service to get the to update the attendance through Admin service) Rest Template has been used. The token will be taken from the header ‘***Authorization***’ and will be used
* Please refer to [Attachments](#_Attachments) for **Swagger API documentation** of the services
* **Admin Service** – This service handles all the independent activities concerning the Admin**.**
  + Can add, update and monitor teacher and student details
  + Can provide register number for all students, assign each student a course etc.
  + Generate reports like attendance, student’s details.
  + Receives request from User Service to update attendance records.
* **User Service** – This service handles all the activities related to User depending on whether the user is Student or teacher.The following functionalities are handled in this service.
* For Students
  + - view their personal details, course assigned.
    - view their attendance.
* For Teachers
  + - view their schedules.
    - marks attendance.
    - view the student detailed information

# DB Layer

****

* **DB Layer** – MySQL database has been used. PFB the entity diagram for the same.
  + All the primary keys are **Auto Increment columns**

# Swagger API documentation



# 

# Attachments

|  |  |  |
| --- | --- | --- |
| SI. No. | Title | Document |
| 1 | Postman API collection |  |
| 2 | Test Results |  |
| 3 |  |  |

# Metadata

Created by Midhun Raj

As part of training on Micro-service architecture and scope.