CONCORDIA UNIVERSITY

DEPARTMENT OF COMPUTER SCIENCE AND SOFTWARE ENGINEERING

SOEN 6441

Advanced Programming Practices

STUDENT MANAGEMENT SYSTEM

Instructor - Dr. C. Constantinides

Group Member Names:

- Srikar Hasthi - 40230004
- Harshavardhana Mudduluru - 40231250

Purpose of Student Management System:

The system is an application used for searching information of students and teachers of an institution. The application contains a filter for searching information from database for types (teachers and students). The application allows the user to **view, add, modify, delete** the information from the database.

Scope of the Application:

Since the application is not hosted on any medium, it runs only on localhost of the user. But the system can be used as the Office of Institution information seeks to protect the privacy of students and teachers to safeguard the information. The application can be further developed into full-stacked application and hosted in real time for an institution with well-developed UI and database with security authentication and few other advance search techniques.

Technology Stack Used:

Front-End development:

- Html, CSS
- JavaScript
- Embedded JavaScript

Database development:

- SQLite

Middleware Layer used:

- NodeJS – Used to run the application on a local server.

Data Retrieval Methodology Used:

- ExpressJs is used to retrieve data from the local-server. (**Queries are not Automated**)

Testing Tool:

- Postman

Design Patterns:

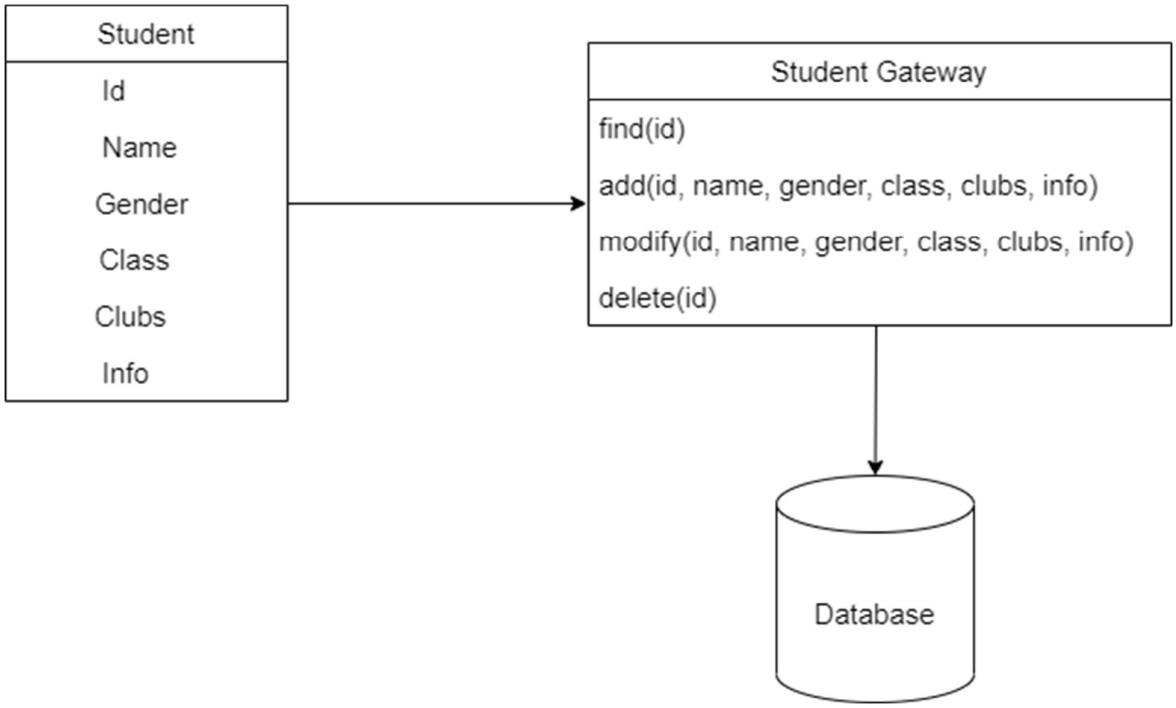
Object-Relational Mapping (ORM):

Object relational mapping — or ORM, is a design pattern for converting (wrapping) that data stored within a relational database into an object that can be used within an object-oriented language.

In the application, when we try to insert new data into the database, first the data is mapped to the Student class in Student.js file. Then, the variables present in class is used to add data into the database.

Similarly, the same mapping goes for all the methods (modify, delete, search).

Table Gateway Diagram (Data-Source Architecture):



Re-Factoring Strategies:

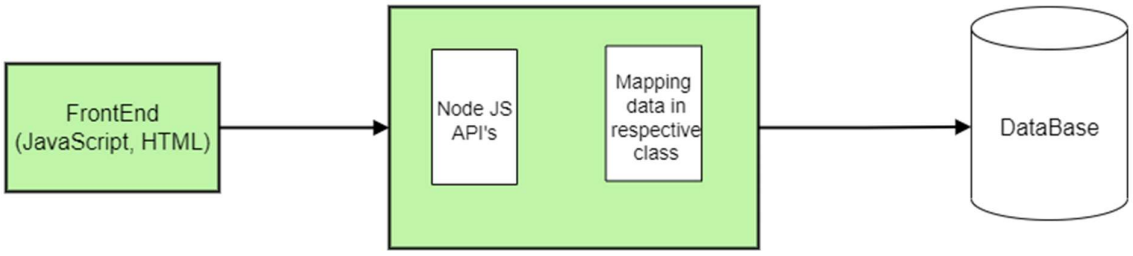
- In the application, the utils.js file, we have defined a few functions which can be exported and used in various files of the application.
- So, later, if we would like to make any changes to the definition of the function in application, we could just change in the utils.js file to reflect everywhere it is used.
- This helps in **code Readability** and **Re-usability** and **time saving**.

Testing Tool:

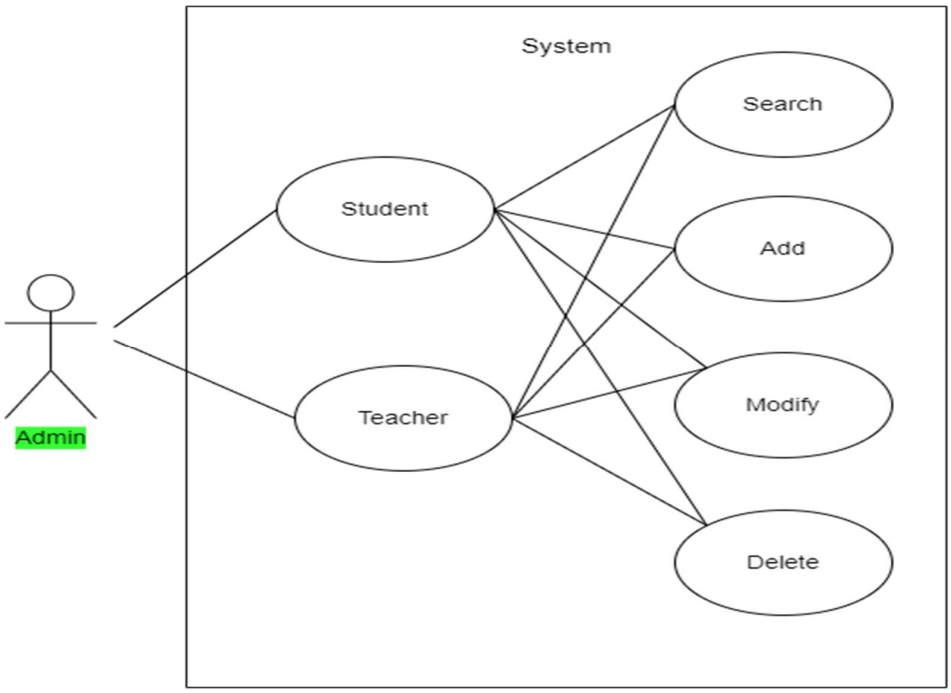
POSTMAN:

- Postman is used to check whether the data is being added or modified or deleted from the database correctly.
- It also checks for the correct datatype being used when using the functions of the application.

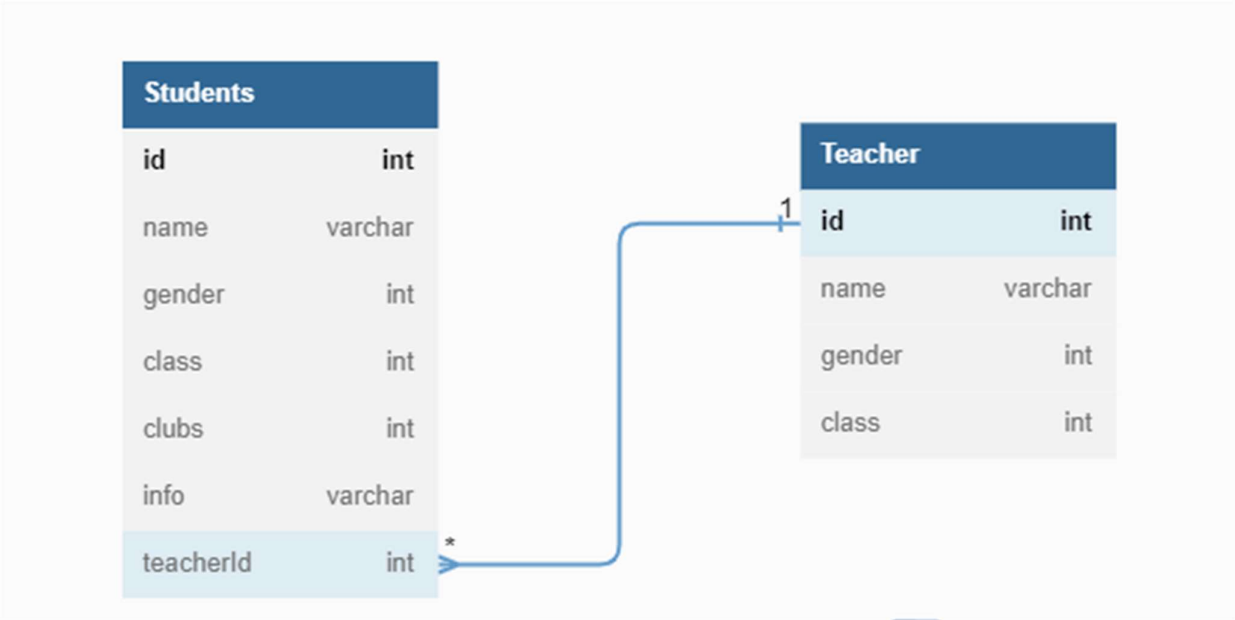
Architecture Diagram



Use Case Diagram



Entity Relational Diagram:



GITHUB REPOSITORY LINK:

https://github.com/SrikarHasthi/APP-Project

REFERENCES:

Student Data Source:

https://jsonformatter.org/json-editor/a2ec9f