**ANALYSIS AND DESIGN OF A FASTAPI, MONGODB AND CLOUD APPLICATION**

**Table of Contents**

[Introduction 3](#_Toc165936105)

[Database Design 3](#_Toc165936106)

[1. Teacher Class and Students Manager Database 4](#_Toc165936107)

[2. Teaching Material Manager Database 4](#_Toc165936108)

[3. Assign Assessment to Student & Provide Feedback and Marks Database 4](#_Toc165936109)

[Implementation 5](#_Toc165936115)

[API Design 5](#_Toc165936116)

[Add Student in Teacher Class Service API 6](#_Toc165936117)

[Get Students Assigned to Teacher Class Service API 6](#_Toc165936118)

[Remove Student from Teacher Class Service API 6](#_Toc165936119)

[Upload Teaching Material Service API 6](#_Toc165936120)

[Get All Uploaded Teaching Materials Service API 6](#_Toc165936121)

Assign Student Assessment & Provide Feedback and Marks to Students [Service API 7](#_Toc165936122)

Get List of students assigned with document, Feedback and Marks  [Service API 7](#_Toc165936123)

[BackEnd Development 7](#_Toc165936125)

[Database Integration 8](#_Toc165936126)

[Testing & Integration 8](#_Toc165936127)

[Unit Testing 8](#_Toc165936128)

[Integration Testing 8](#_Toc165936129)

[End-to-End Testing 9](#_Toc165936130)

[Performance Testing 9](#_Toc165936131)

[Conclusion 9](#_Toc165936132)

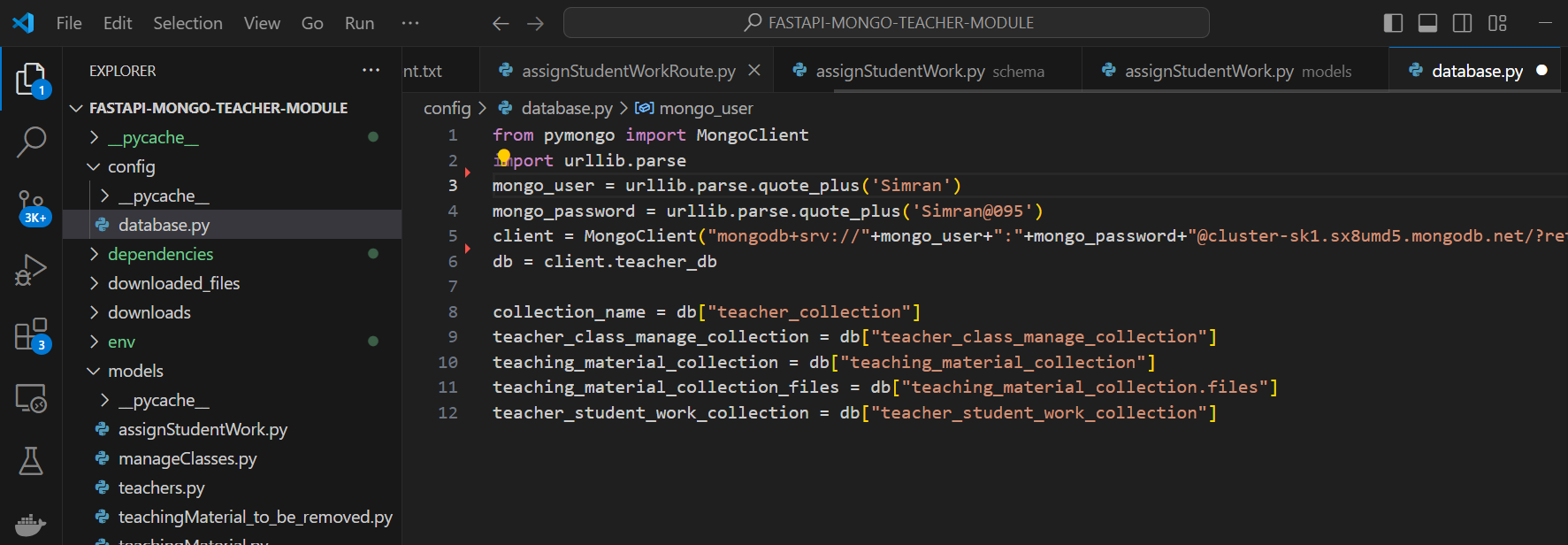
[References 10](#_Toc165936133)

**Introduction**

The intricate characteristics and architecture of an Teacher’s module application created with the use of contemporary technologies are examined in this study. This project makes use of Python Language and Its latest Framework i.e FastAPI and MongoDB for a robust database design. The goal of this project is to showcase a thorough understanding of FastAPI, cloud computing, and modern web development methods. By merging these components, the project illustrates how to combine efficiency, scalability, and user-centric API’s design in a dynamic digital world. Through extensive investigation and analysis, the current work clarifies the strategic choices utilized in implementation, the reasoning behind database design decisions, the stringent testing procedures used, and the simple integration of many aspects. The knowledge acquired from this research contributes to the greater conversation on how technology may be used to make significant changes in the data-driven world of today.

# Database Design

The report manages several attributes that are important in the field of education using an advanced database structure. The database architecture contained a number of essential elements that are appropriate for particular system management requirements.



**Figure 1: Database Model Design**

(Source: self-created)

## 1. Teacher Class and Students Manager Database

This relational database manages teacher’s classes and students assigned to these classes by their teachers. There are fields called Teacher ID, Student ID and Class ID.

## 2. Teaching Material Manager Database

This database is part of the effort to save teaching material uploaded by teacher for their students. It includes two tables such as Teaching Material Collection Files and Teaching Material Collection Chunks. Fields like Teacher ID, File Name, File ID, Chunk Size, File Length, File Uploaded Date, and File Data in Binary Base64 Format.

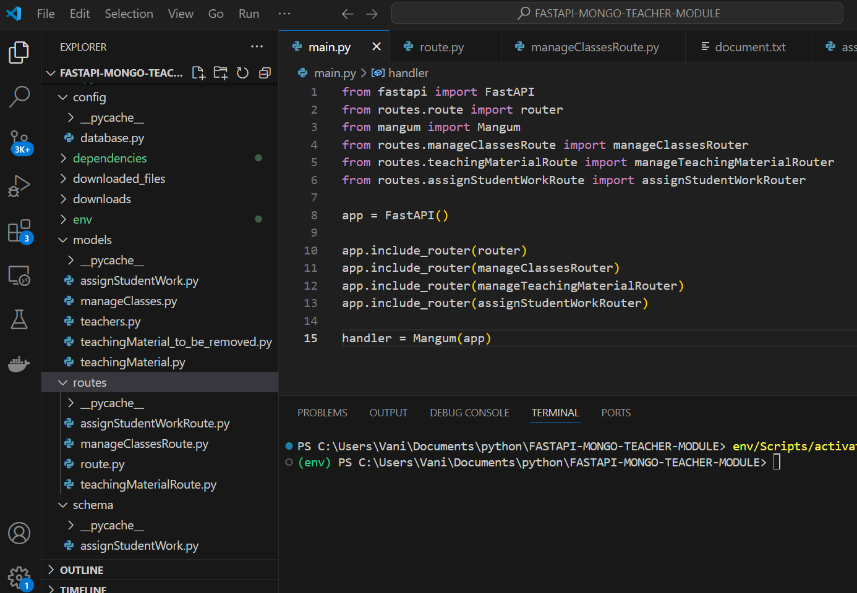
## 3. Assign Assessment to Student & Provide Feedback and Marks Database

This database collection purpose is to store information like which teacher’s assignment assigned to which student along with student’s marks and feedback provided by teacher, fields like Teacher ID, Student ID, Assignment ID, Teacher Feedback and Marks Obtained are included.

# Implementation

## API Design

The project's implementation includes using Python FastAPI framework to build safe APIs that enable smooth communication between the frontend, backend, and MongoDB database. Each microservice is mapped to a single endpoint so that CRUD operations and business logic may be handled efficiently.



**Figure 2: API Segmentation**

(Source: self-created)

## Add Student in Teacher Class Service API

The Add Student API offers endpoints for assigning student to teacher’s class for assigning student into teacher class. These endpoints accept inputs such as Teacher ID, Student ID and Class ID.

## Get Students Assigned to Teacher Class Service API

The Get Teacher Class Students API provides end points for providing list of students those are assigned to selected teacher’s class. Parameters like Teacher ID used by API to provide JSON format data like Teacher ID Student ID, Class ID and unique Record ID(unique ID of student and teacher class mapping).

## Remove Student from Teacher Class Service API

Endpoints for removing student from teacher’s class are offered by the Remove Student from Class API. It requires unique ID of student and teacher class mapping as an input parameter to execute its function, this parameter user will fetch from previous API.

## Upload Teaching Material Service API

Uploading teaching material of any type format of file such as .doc, .ppt, .txt, .jpeg, .png, .xml, .xlsx etc by teacher is managed by the Upload Document by Teacher API. With inputs like Teacher ID, and Assignment File in Binary Multipart/Form-data.

## Get All Uploaded Teaching Materials Service API

Getting List of assignment material uploaded by teacher is handled by the Get All Teacher Documents API. With the input parameter Teacher ID, it provides JSON data such as Teacher ID, File Name, and File Uploaded date.

## Assign Student Assessment & Provide Feedback and Marks to Students Service API

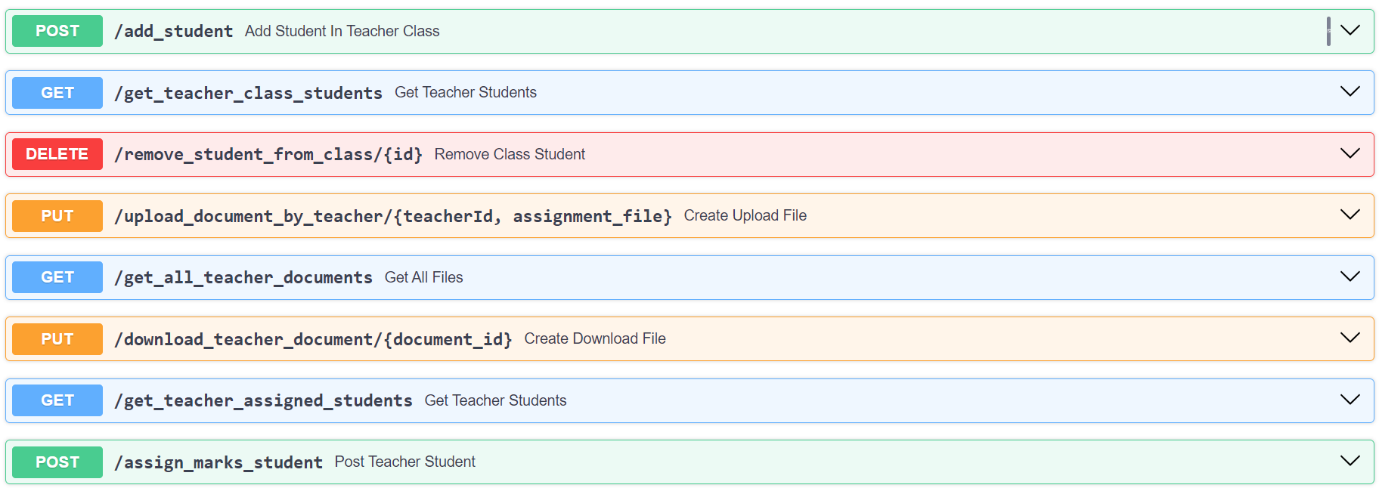
Assigning teacher assessment to teacher along with functionality to provide feedback and give marks to student’s assessment by teacher is handled by the Assign Marks Student API. Input parameters are Teacher ID, Student ID, Document ID, Marks Obtained and Teacher Feedback.

## Get List of students assigned with document, Feedback and Marks Service API

Getting list of students and their assigned documents along with feedback and marks assigned by teacher is handled by the Get Teacher Assigned Students API. Input parameters is Teacher ID and JSON response output contains data such as Teacher ID, Student ID, Document ID, Marks Obtained and Teacher Feedback.

# Backend Development

In backend development, python FastAPI framework and its libraries along with our python code are used to communicate with the MongoDB subsystem to store data. In order to facilitate the management of tasks related to the university teachers, teaching material, teacher class students association and to provide feedback and marks few API endpoints are created to help Frontend developer to save and access data.



**Figure 3: Homepage**

(Source: self-created)

## Database Integration

FastAPI is connected with the MongoDB database to facilitate efficient data retrieval and storage (Dorasamy, 2021). By interacting with the MongoDB collections that are pertinent to its functions, every microservice makes sure that the data is scalable, consistent, and reliable. The microservices architecture and database design work together to efficiently meet the data management needs of the application.

# Testing & Integration

The project goes through extensive testing and integration procedures to guarantee the system's functionality, scalability, and dependability.

## Unit Testing

Every microservice built using FastAPI undergoes unit testing to verify its distinct functionality and endpoints. Several scenarios are simulated using mock data to make that every API endpoint reacts appropriately to varied inputs and gracefully handles errors.

## Integration Testing

The interface between several microservices and the MongoDB database is tested through integration testing (Serbout *et al.* 2021). This testing phase confirms that data flows across services appropriately, APIs interact successfully, and all components operate together fluidly.

## End-to-End Testing

The entire application workflow, from user interactions on the back end to data processing and storage in the back end, is validated through end-to-end testing. Through testing, the application's predicted functionality in real-world scenarios such as authentication, data retrieval, and teachers tasks is confirmed.

## Performance Testing

Performance testing is done to evaluate how well the application responds to different loads, scales, and uses resources. To find any bottlenecks, enhance system performance, and provide the best possible user experience, scalability, load, and stress tests are carried out.

# Conclusion

The project shows how to use FastAPI, MongoDB, Python to successfully construct a complete Teacher Module application. With careful planning, strong API development, thorough testing, and smooth integration, the project demonstrates how contemporary technologies may be combined to successfully handle challenging teacher role’s requirements. The project's accomplishments demonstrate how crucial it is to employ cloud computing, microservices architecture, and MongoDB to develop scalable, effective, and user-centered educational solutions.

# References

<https://fastapi.tiangolo.com/> for learning Fast API formation and its functions libraries to be used such as upload documents, making MongoDB connections and so on.

https://github.com/tiangolo/fastapi.

<https://stackoverflow.com/> For resolving issues encountered during project development.

<https://www.mongodb.com/> For accessing MongoDB Atlas online platform and connection string to be used in FastAPI.

<https://aws.amazon.com/> For creating Cloud platform and uploading my project on cloud.