******

***Title:General Educational Database Management System***

*Submitted in partial fulfilment of the requirements for the award of degree of*

**Bachelor of Technology**

**in**

**Computer Science & Engineering**

**UE20CS352-OOAD Mini Project Report**

***Submitted by:***

|  |  |
| --- | --- |
| **Vinesh R**  **Vinesh S**  **Vivek Joy** | **PES1UG20CS504**  **PES1UG20CS505**  **PES1UG20CS512** |

*Under the guidance of*

|  |
| --- |
| **Bhargavi Mokashi**  **Assistant Professor,**  **Department of CSE** |

**January - May 2023**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

FACULTY OF ENGINEERING

**PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013)

100ft Ring Road, Bengaluru – 560 085, Karnataka, India

**Problem Statement:**

General Educational Database Management System aims to provide a flexible data management tool that overcomes the shortcomings of commercially available DBMS used in education. It is built on a schema-less database management system, thus granting each user the ability to make the model for their database interactively using the Web app.

The web app also allows the user to edit their datasets directly, and also has features to map columns in csv/xlsx files to their dataset entities.

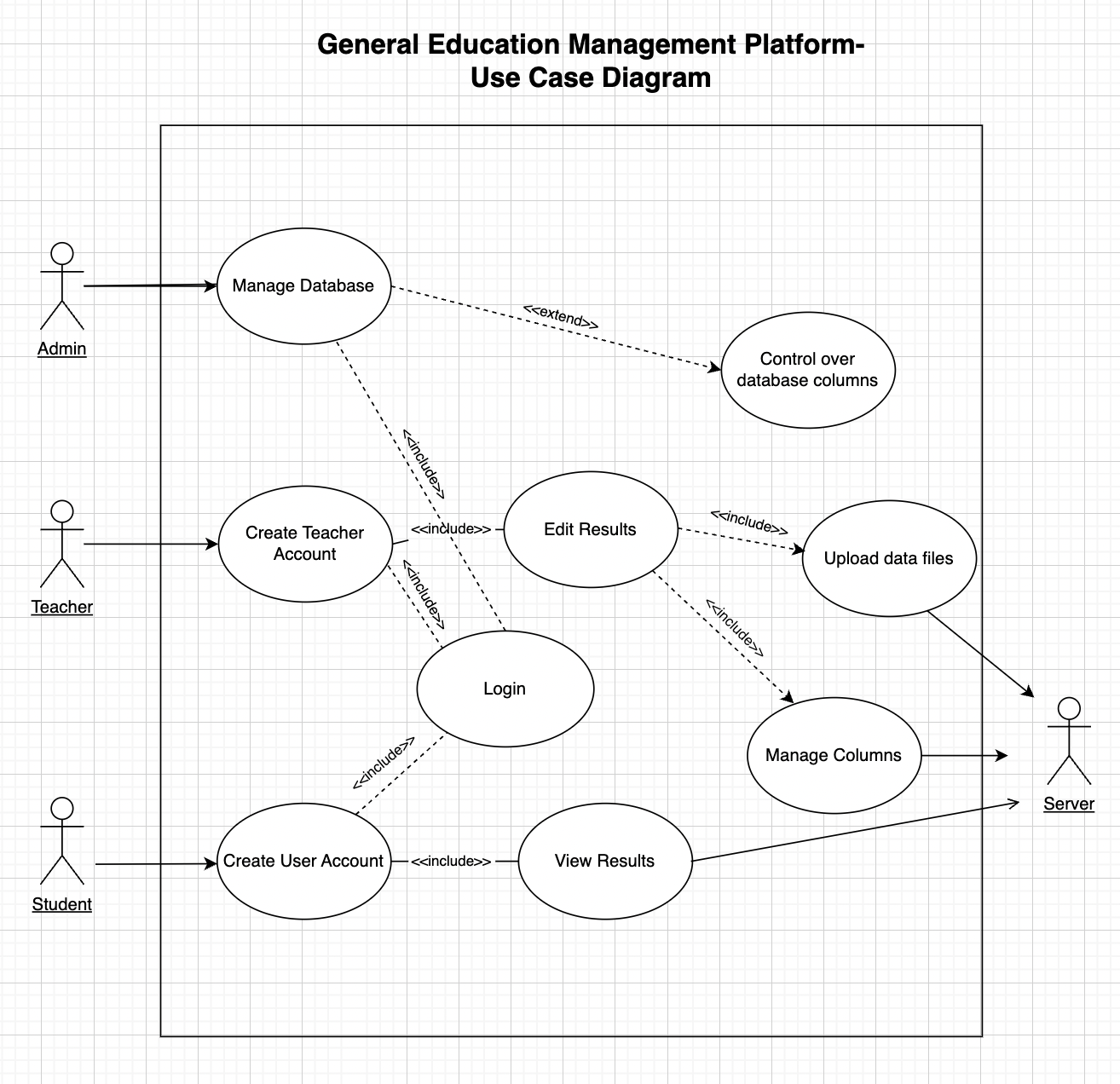
This software will allow users to interactively build databases for their specific purpose.

It also provides some template data models for common entities like student, employee, institution, course, assessment, etc. Each database will be specifically designed for the user thus ensuring that there are no redundant entities/attributes in the database.

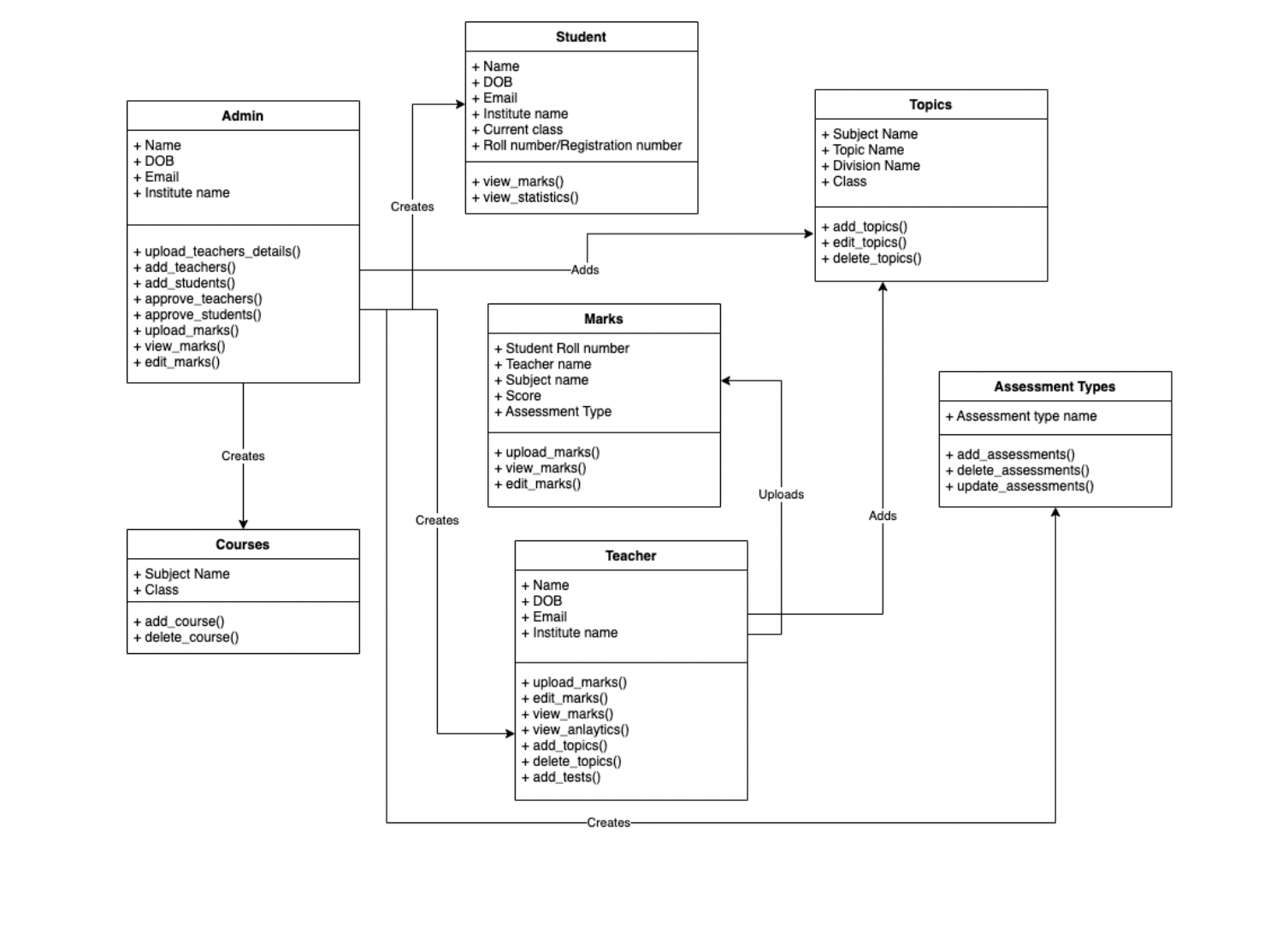
The accompanying Web App will be easy to use and will allow data modeling to be done using simple dropdowns, buttons and input fields, and the backend scripts will do the job of creating the MongoDB database.

**Models:**

*Use Case Diagram*

****

*Class Model Diagram*

**

**Architecture Patterns, Design Principles and Design Patterns:**

*Architecture Pattern-*

MVC: MVC (Model-View-Controller) is a architecture pattern that separates an application into three main interconnected component.MVC provides a clear separation of responsibilities between the Model, View, and Controller components which makes the application easier to develop.MVC provides a lot of flexibility in terms of customizing the application's behavior and appearance since each component is separate.

Client-Server Architecture: The application is separated as client and server. THe client takes care of front end implementation and server manages the backend which is connected to a mongodb database.

*Design Principles-*

Single Responsibility Principle (SRP): This principle ensures that each class in our application only performs a single function.This promotes modularity and makes it easier to modify and maintain code.

Interface Segregation Principle (ISP): Clients should not be forced to depend on interfaces they do not use. This means that a class should not implement unnecessary methods that are not required by its clients.

*Design Pattern-*

Factory Pattern: This pattern is used to create objects without exposing how it's created to the client. It provides a way to create objects of different types without having to specify the exact class of object that will be created.

**Github Link:**

<https://github.com/PaiN-0769/General-Edu-Managment-System-OOADJ>

**Individual contributions of the team members:**

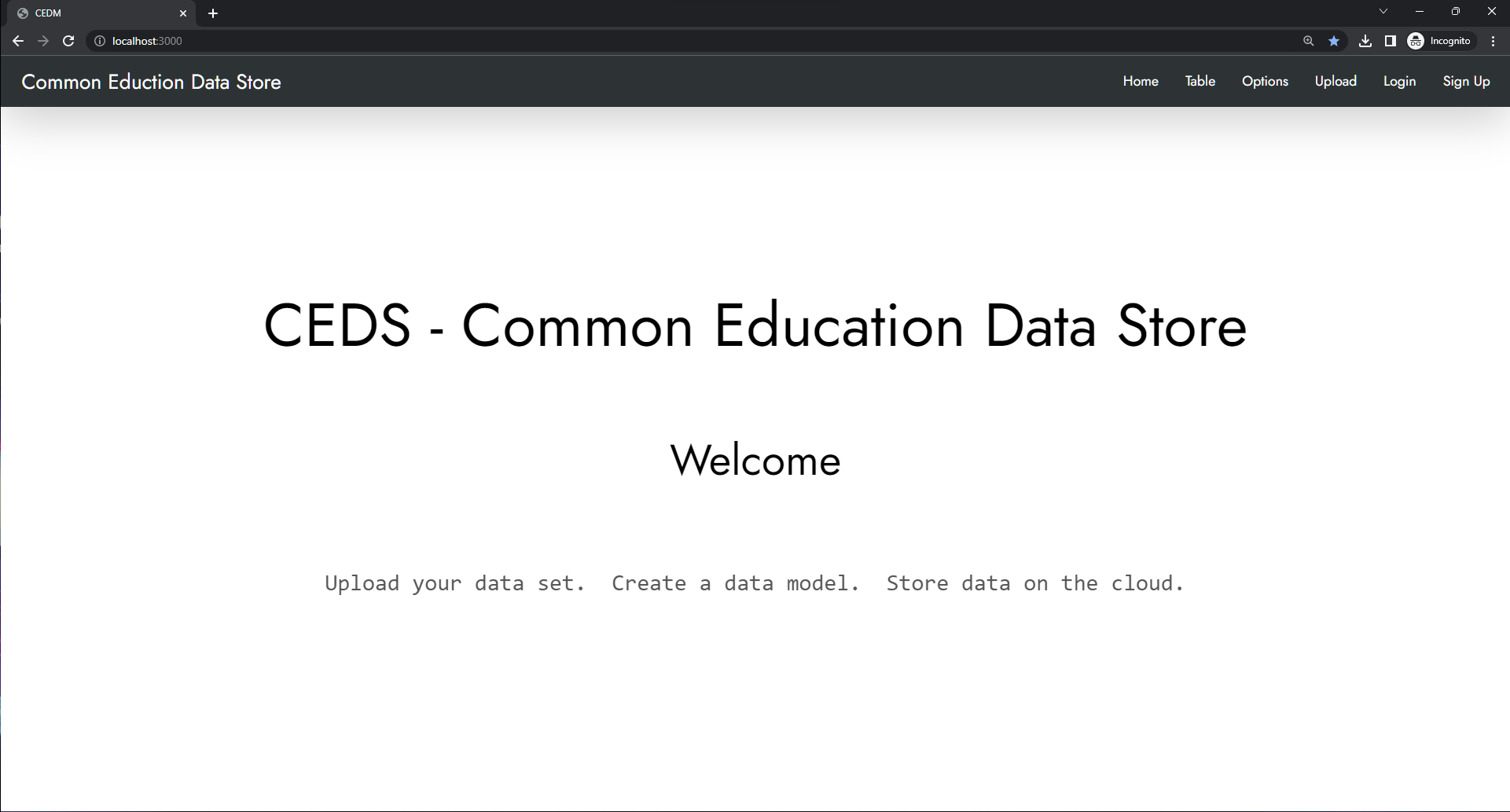
Vinesh R- Frontend-backend Integration

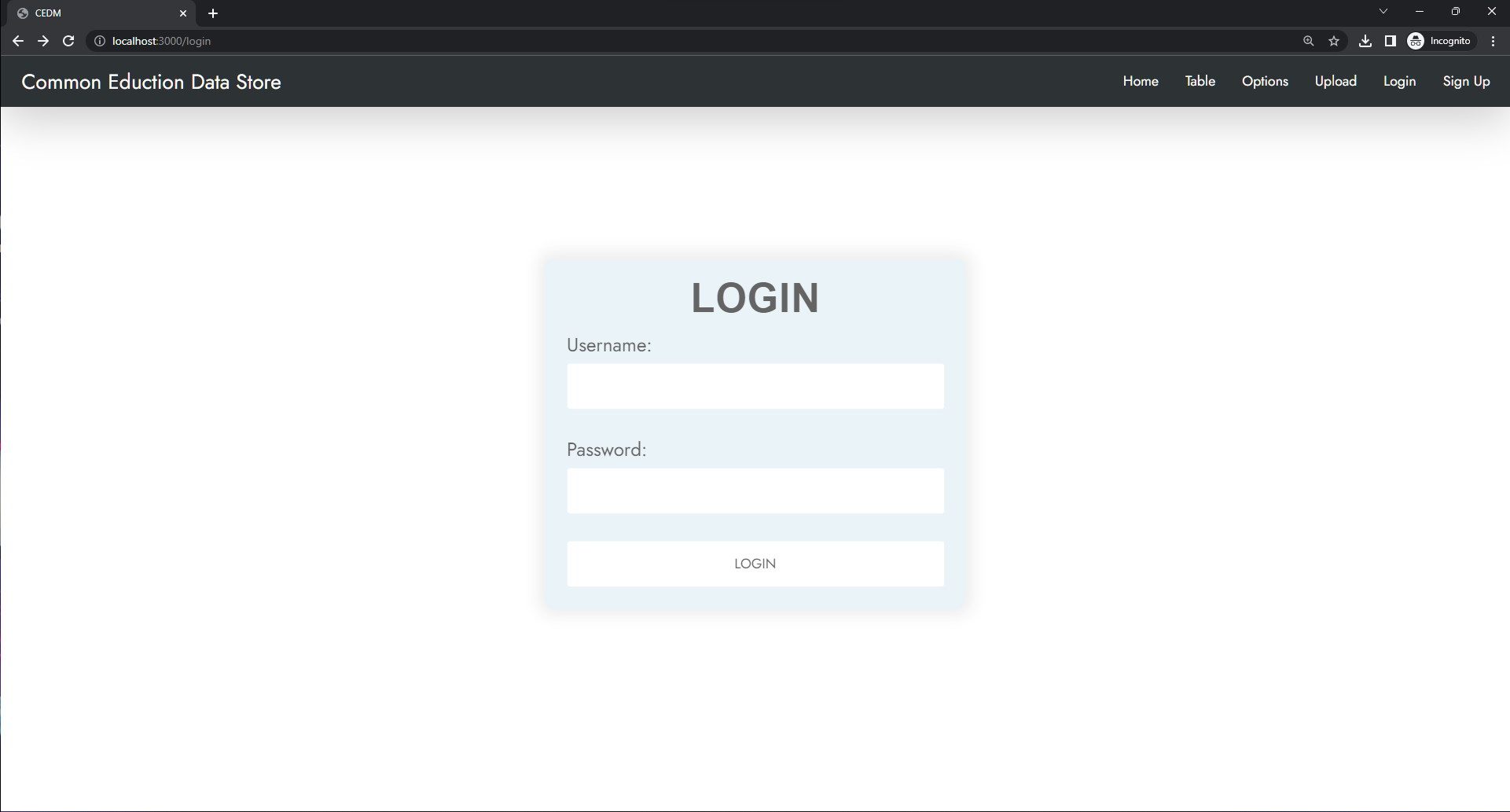
Vinesh S- Frontend (React) Code

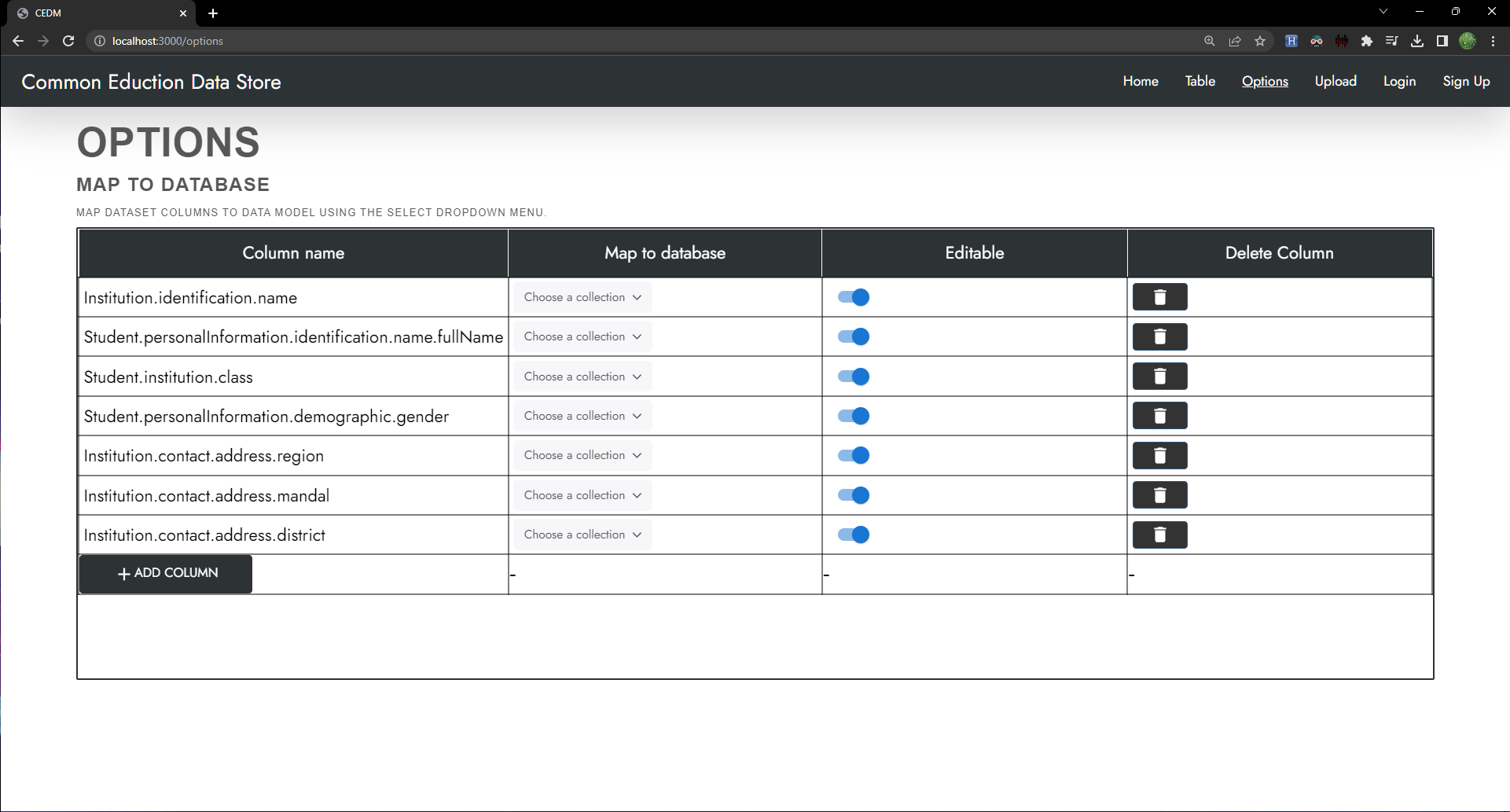
Vivek Joy- Backend (Springboot) Code

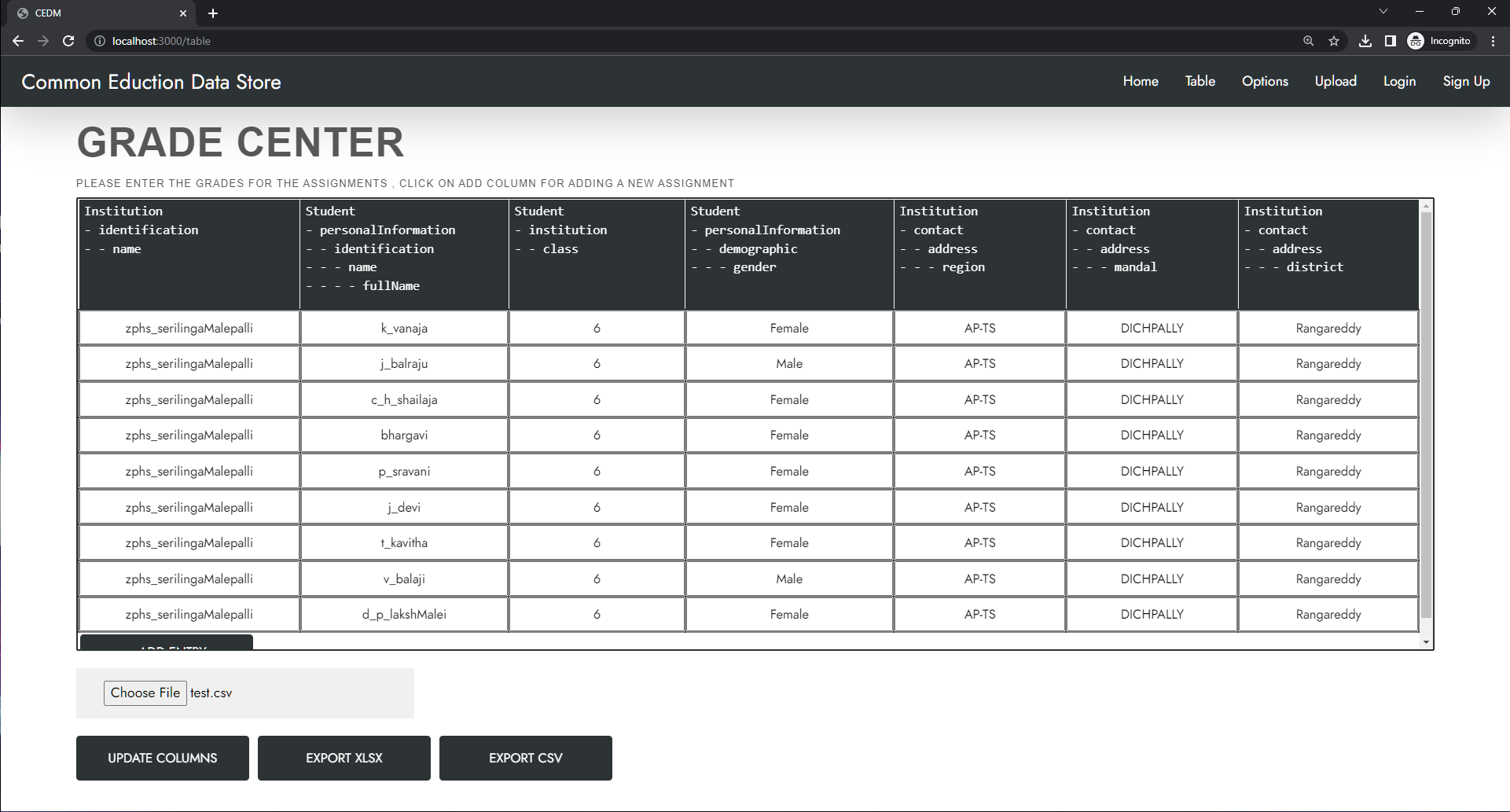
**Screenshots with input values populated and output shown:**

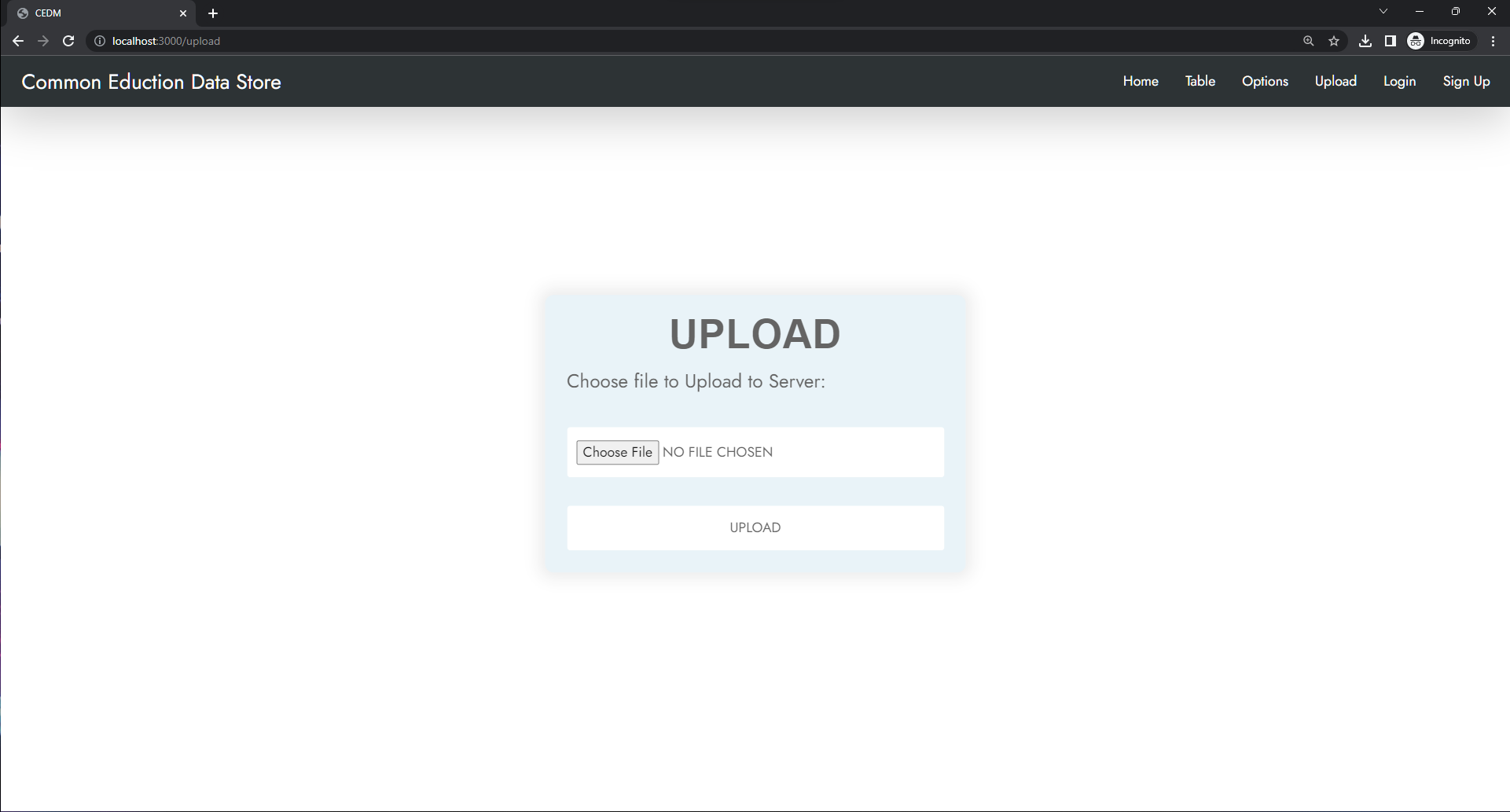
**Frontend:**

****

****

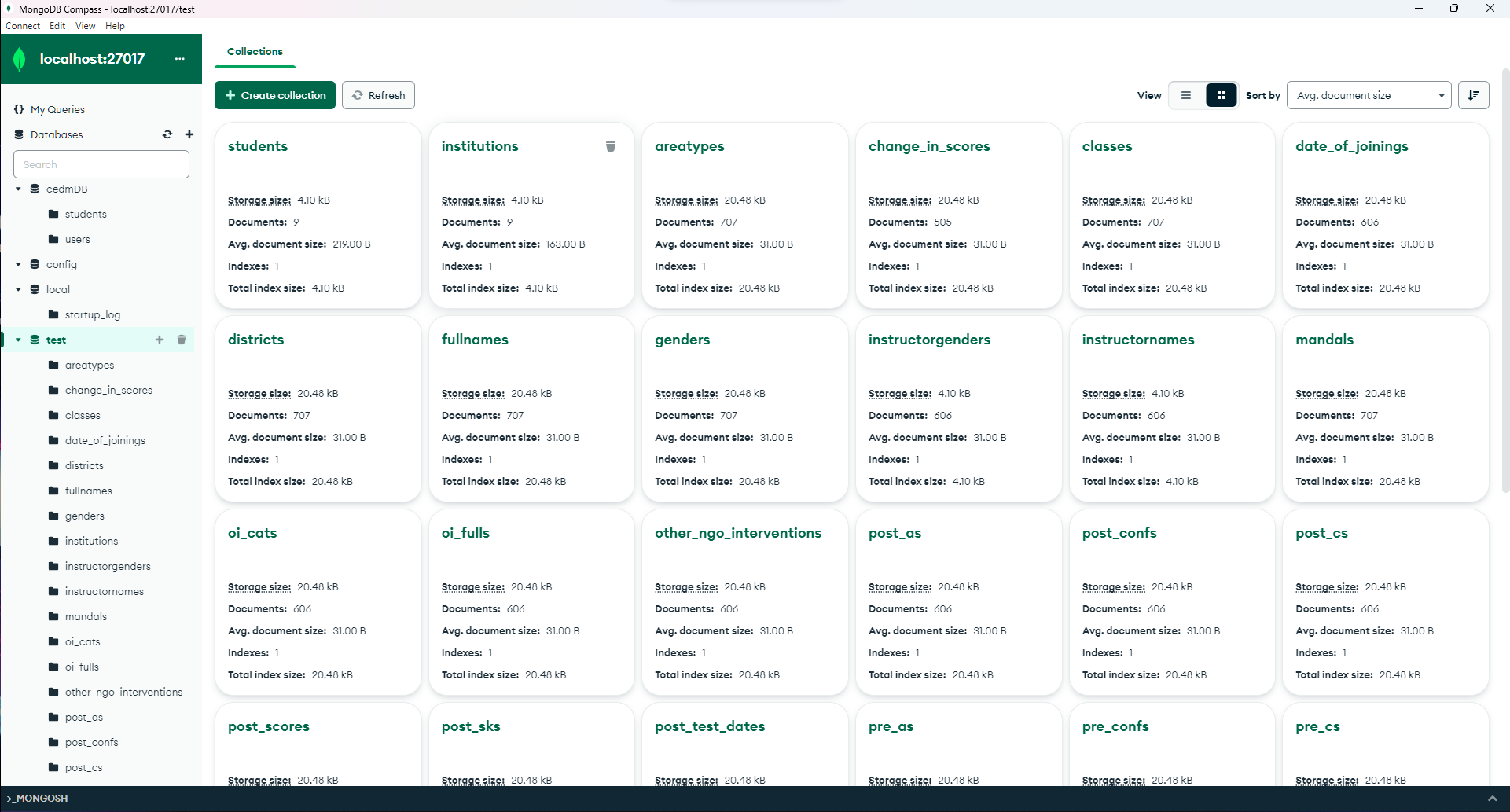
****

****

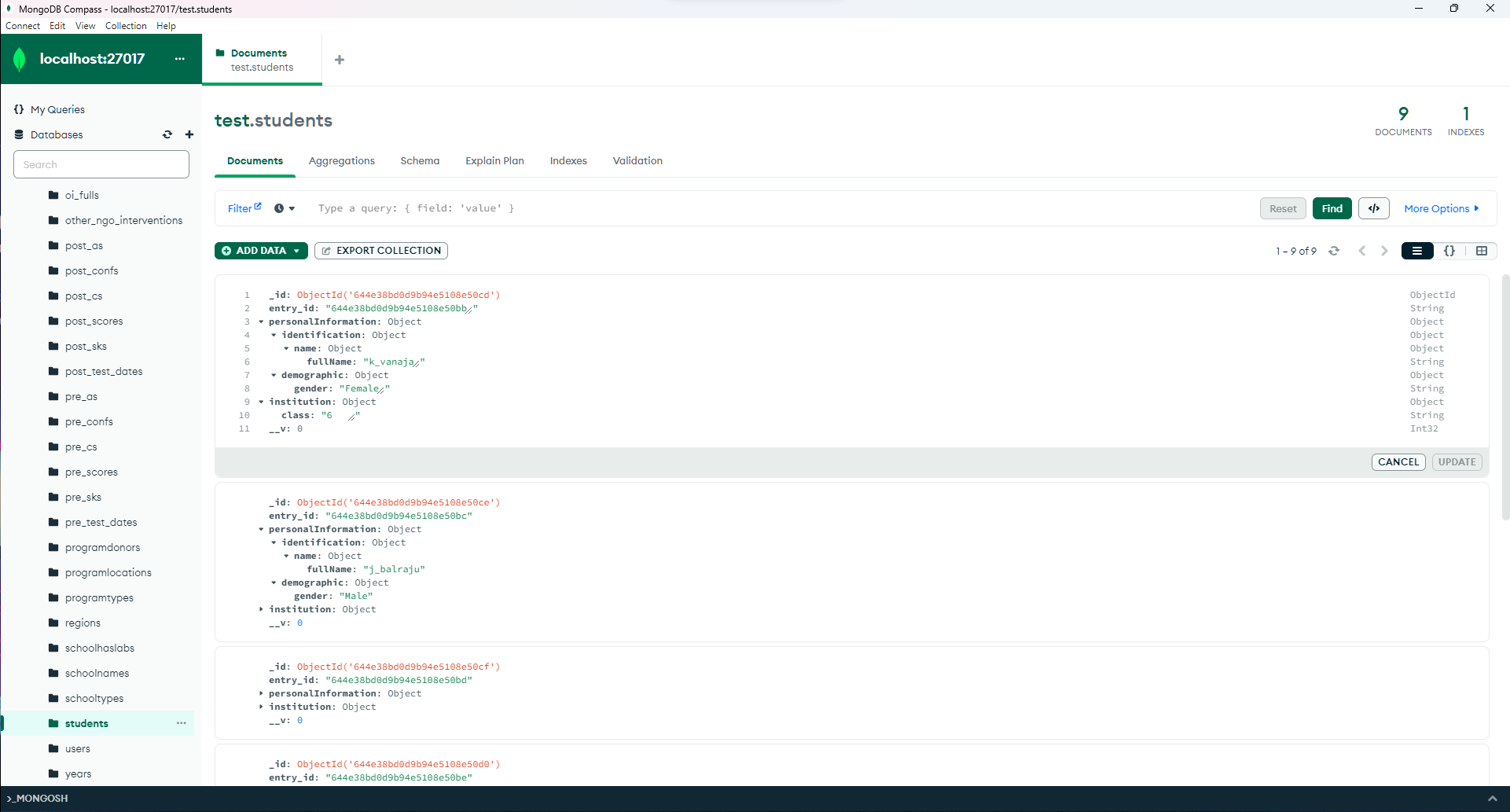
****

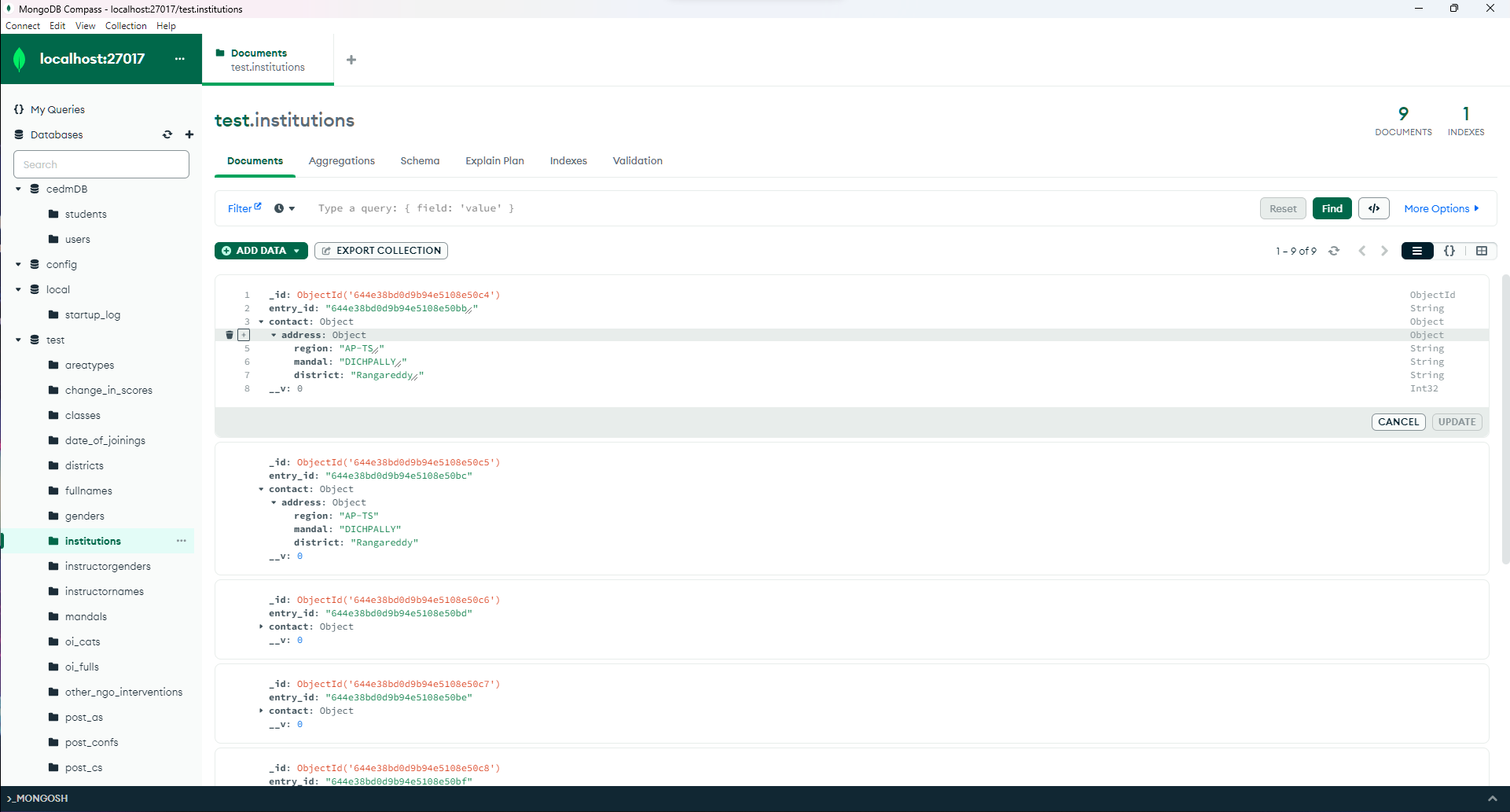
**Backend:**

**MongoDB database:**

****

**Few Collections:**

****

****