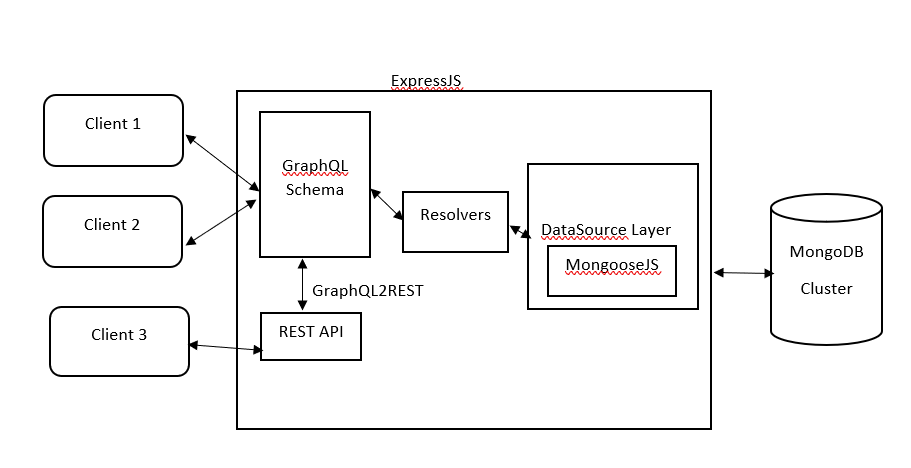
**Overall Architecture of system :-**

****

**Front end:-**

**Ui library used :- React**

React is a front-end library that has gradually become the go-to framework for modern web development within the JavaScript community.In React, you develop your applications by creating reusable components that you can think of as independent Lego blocks. These components are individual pieces of a final interface, which, when assembled, form the application’s entire user interface.

## **LIbrary to manage state : Local, Global, Server, Url**

## **-Manage Local State in React**

useState useReducer

## **-Manage Global State in React**

= Redux is also great, but make sure that you get started using Redux Toolkit. The benefit of a library like Zustand is that it is small, makes your entire global state a custom hook, and to read or update state, you just call this hook in your components. To use Zustand, run npm install zustand. After that, make a dedicated store file or folder and create your store:

**-Manage server state**

=**SWR** and **React Query**.

They not only give us a convenient hook to both get and change data from an API, but they keep track of all the necessary states and cache the data for us.

## **-Manage URL State in React**

If you are using Next.js, almost everything can be accessed directly from calling useRouter.

**- UI components library**

Material-UI : A wide range of helpful components are available, like app bars, auto complete, badges, buttons, cards, dialog boxes, icons, menus, sliders and more. Helpfully, MUI also offers React themes and templates, so you can have a custom colour theme for your app.

**Backend:**

**We are use both GraphQL & Rest API:**

**-GraphQL**

In GraphQL, the client requests data with queries.

GraphQL is an excellent solution to a unique problem around building and consuming APIs.

Use cases:

Data fetching control

Using multiple data sources

Alleviating bandwidth concerns

Rapid prototyping

**-Rest API**

In a REST architecture, the client makes an HTTP request and data is sent as an HTTP response.

REST API might be easier to design because you can establish multiple endpoints for specific needs, and you can fine-tune specific queries to retrieve the data in an efficient way.

**We are use expressJS framework :**

Express.js or simply Express, is a back-end web application framework for building REST full APIs with Node.js.

It is designed for building web applications and APIs.

It has been called the de facto standard server framework for Node.js.

Express was created to make APIs and web applications with ease.

**We are use mongoDB Database:**

MongoDB is a source-available cross-platform document-oriented database program.

Classified as a NoSQL database program, MongoDB uses JSON-like documents with optional schemas.

This scheme of storage improves speed and performance.

**We use ODM:**

ODM is **Object Document Mapping**. It is like an ORM for non-relational databases or distributed databases such as MongoDB, i.e., mapping an object model and NoSQL database (document databases, graph database, etc.).

**Why use ODM:**

**Depends on the type of data,** If your data needs flexible space, a non-relational database would be a better option for you to achieve your goals.

**The bigger the data set**, the more likely a non-relational database is a better fit. Non-relational databases can store unlimited sets of data of any type and have the flexibility to change the data type