**Quantum Computing Companies Stock Prices Analysis and Forecasting Web Application Project Part 4**

Pushkar D, Sharanbasav S, Vignesh K

#### 

**Tools:**

We are using Fire store Database (A database provided by Google’s Firebase) on the server side. The data is stored in a NoSQL database format. The data has a unique ID, which is derived from two attribute values: Ticker and Date (in MMDDYYYY format). In Fire store database, all the rows are stored as documents, so each document has its own document

In React, the data is managed by state and user interactions. It is then sent to the Firebase server using the database credentials and stored as a document or if it’s a CRUD operation like a Read request, delete request, Update request, the data is managed in similar manner. For generating dashboards, the data is pulled from the Firestore Database using the Ticker value which user will have to give as an input.

Based on the ticker value, the dashboard is generated for that specific stock.

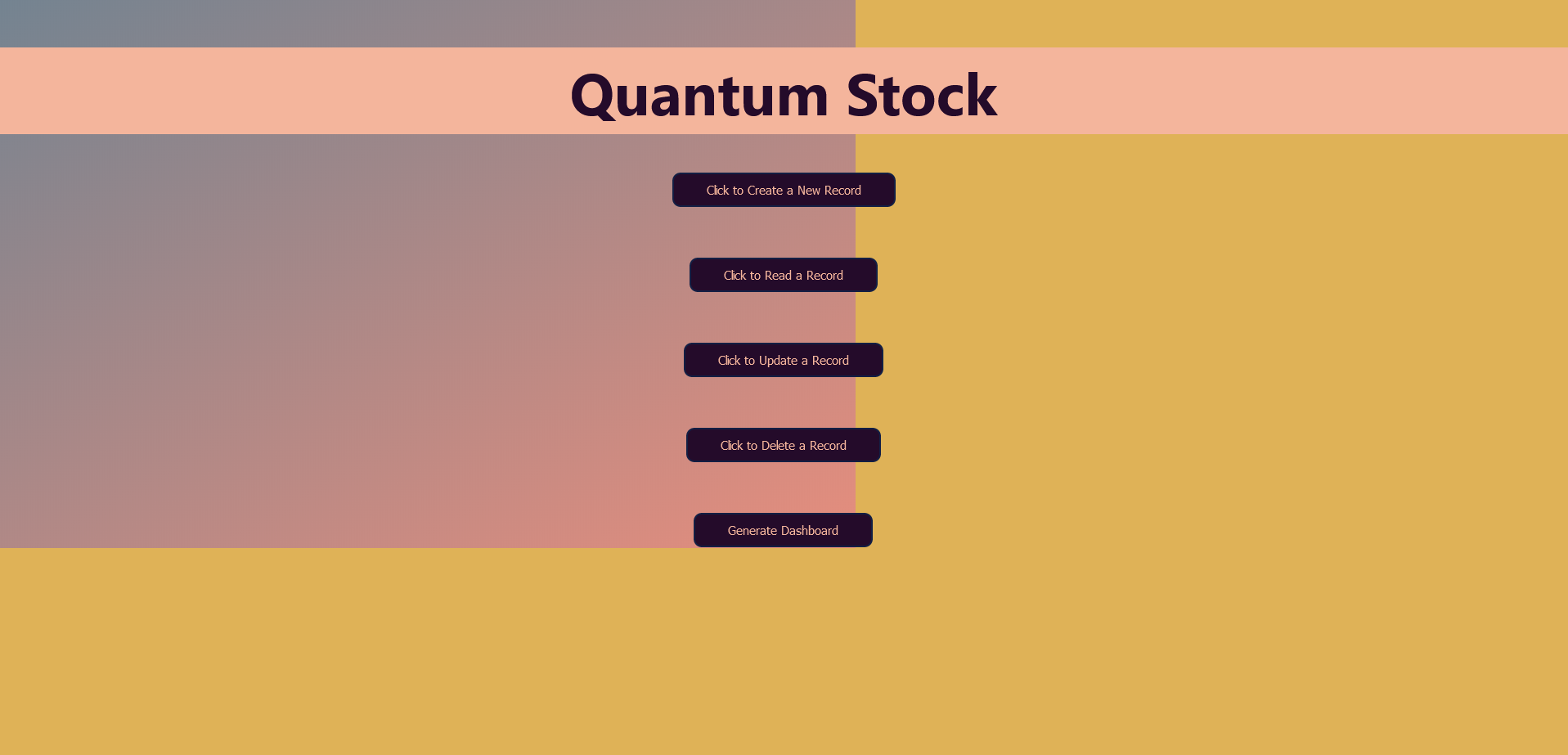
There is a database configuration file in the project repository which takes care of all the credentials: apiKey, authDomain, projectId, storageBucket, messagingSenderId, appId, measurementId. These credentiala are used to make a connection to the Firestore database. There are specifically designed functions like, addDoc(), getDocs(), updateDoc(), deleteDoc() for accessing data from the database. We input the database credentials, document ID or the value from the document that we want to manage.

We are using React JS in the front end. React is a JavaScript library built for developing web applications. For designing and styling the layout, we are using CSS. It is a component-based library for making interactive and encapsulated UI components that manage their own state. All the components are stateful, meaning when the state changes, it re-invokes the render() and the markup is updated likewise.

The web application is deployed on the firebase server. It creates a build file in the project directory which can be used to deploy and updated the application once it’s deployed. It requires for login to the firebase server and then accordingly the project is deployed by the connecting to the project on the firebase console (server-side).

#### **Web App Layout**

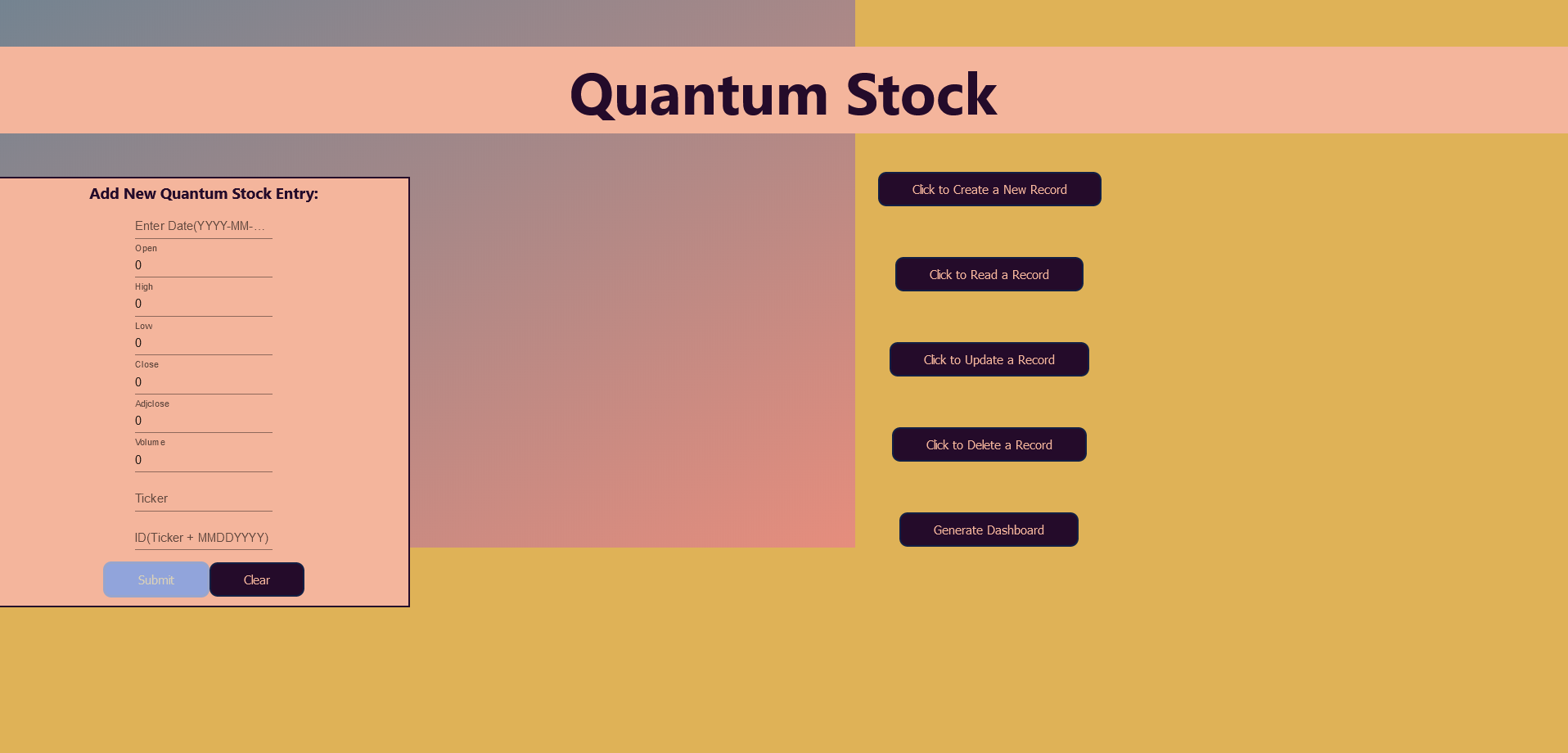
Initial Layout of first page



When the user loads the website the first page itself is the menu page where he has the option to perform Create a new record, delete a new record, Update the new record, and Delete record. There is also an option for generating a dashboard where the user can see the trends for ticker.

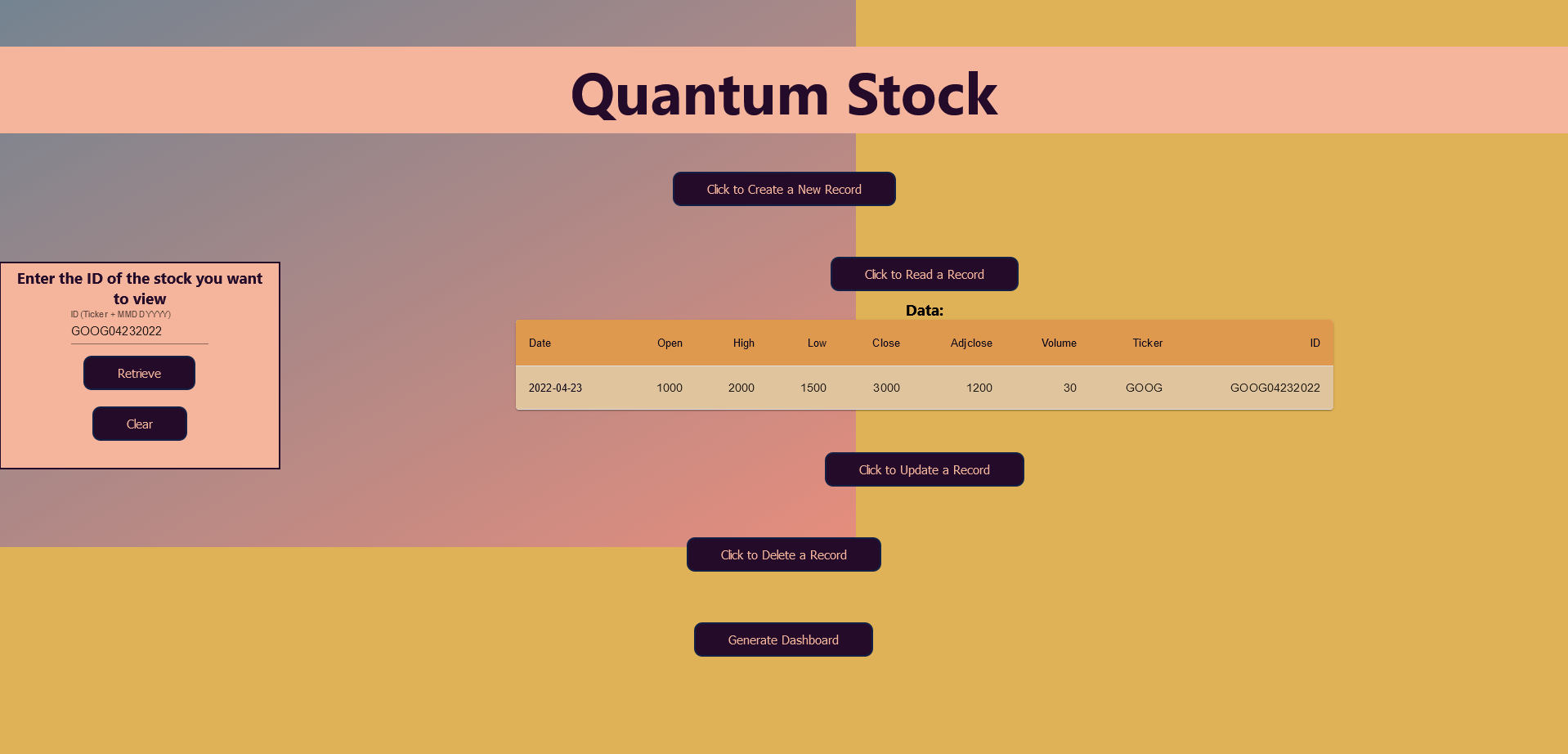
#### **Web App Functionalities**

1. Creating a new record



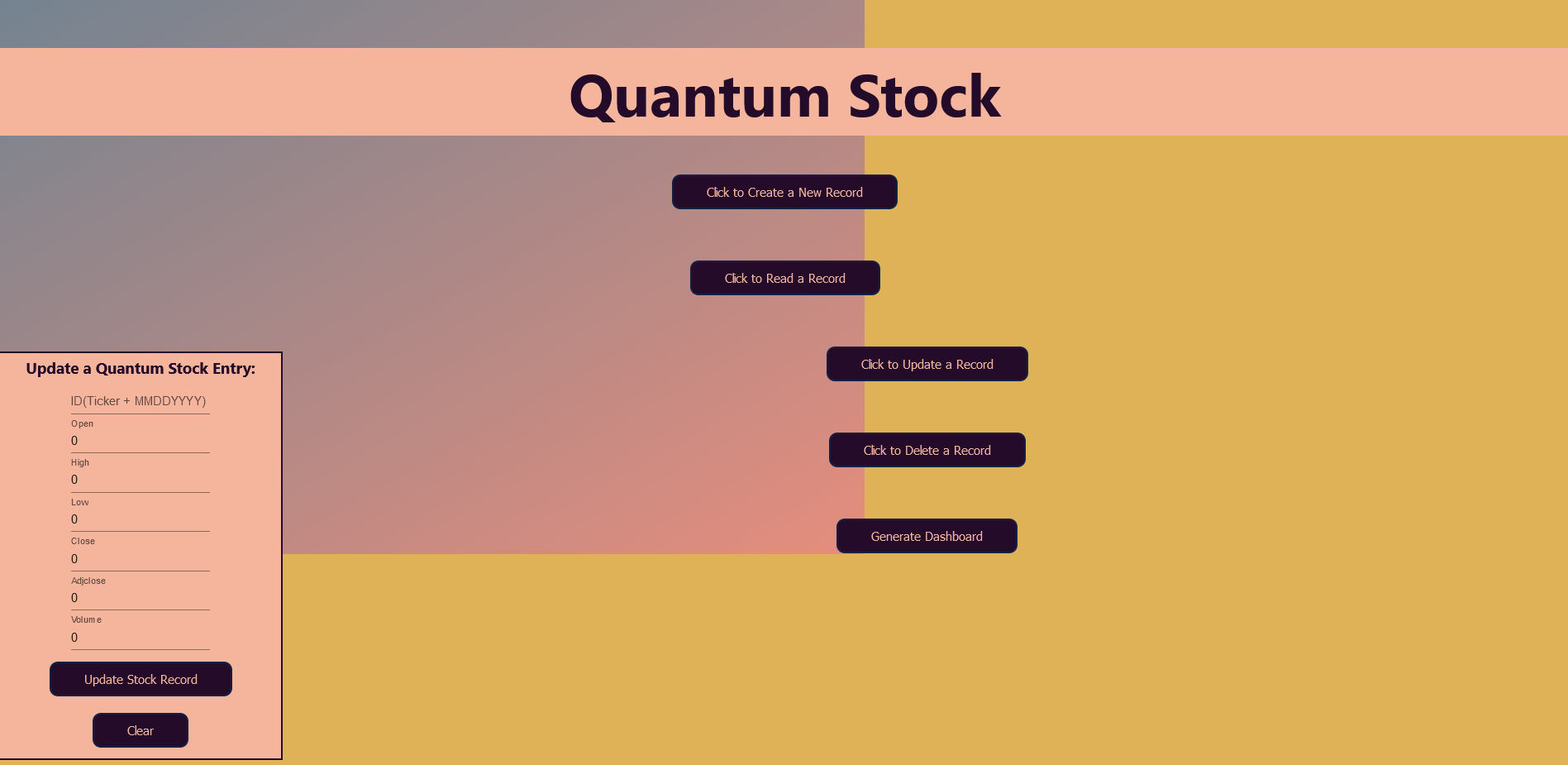
The above image shows the option of creating a new record where user can add into database for particular ticker record

1. Read record



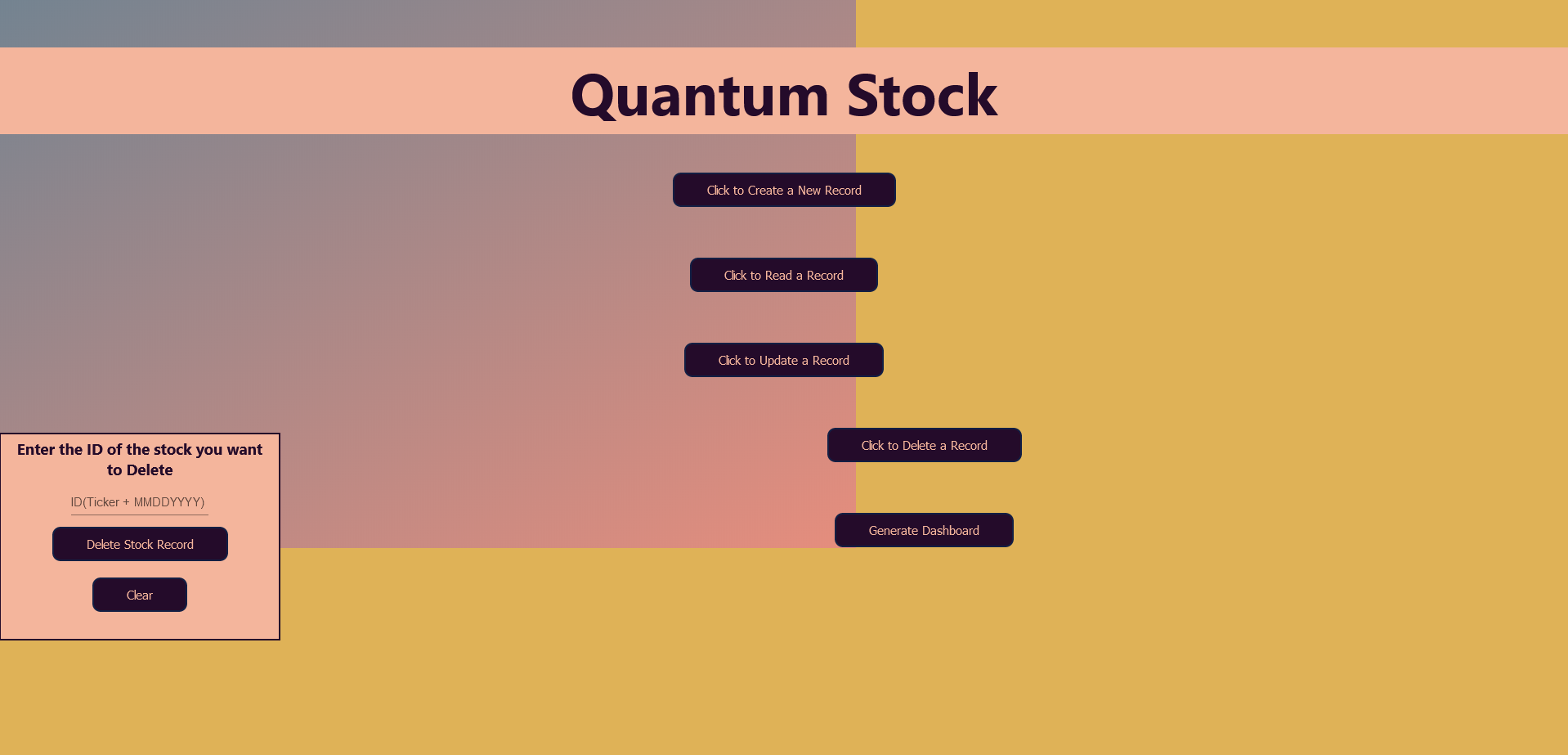
The user gives an input of ticker ID which is in format ticker+MMDDYYY which gives unique ID for all tickers and easy to fetch records for the particular ticker record at a given date

1. Update a record



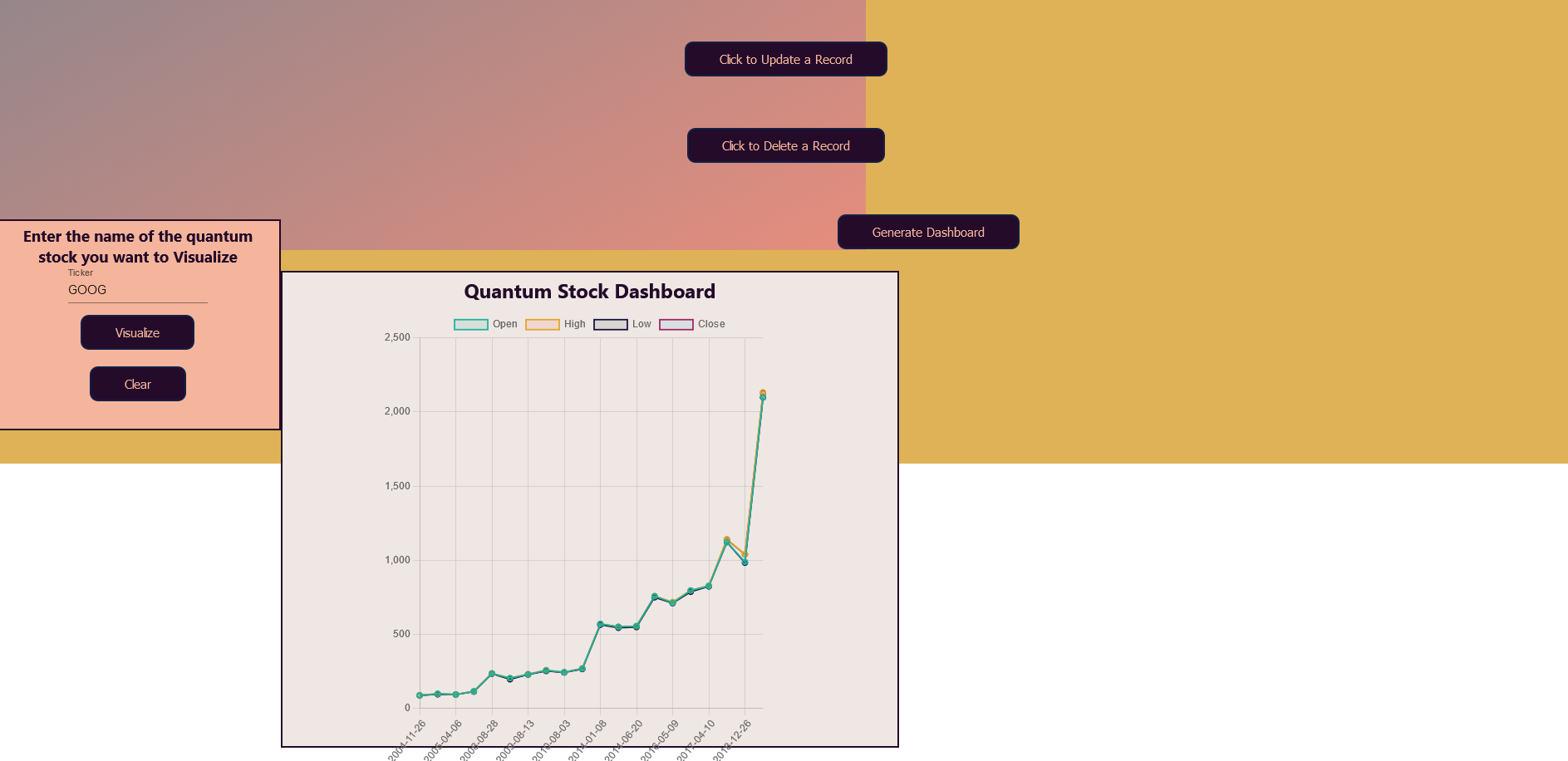
The user gives an input of tickerID for which the user wants to update the record.

1. Delete Record



Similar to read record in delete records the user gives an input of TickerID for the particular ticker date the user wants to delete the record for.

1. Generate Dashboard



Generate dashboard gives user an option to view the graphical representation of the ticker to get an idea of overall trend and performance of the ticker.

The color scheme used for creating front end are rgb(223, 178, 87), rgb(255, 255, 255), rgb(244, 181, 156), rgb(36, 11, 42). There are no tabs used in this project. The user will be able to see the output on the same page as shown in the screenshot above. For this project we require only single page to perform all the required functionalities.

TEAMWORK:

Webapp Backend: Vighnesh

Webapp Frontend and Design: Pushkar

Database and connections: Sharanbsav

Documentation: All team members contributed.