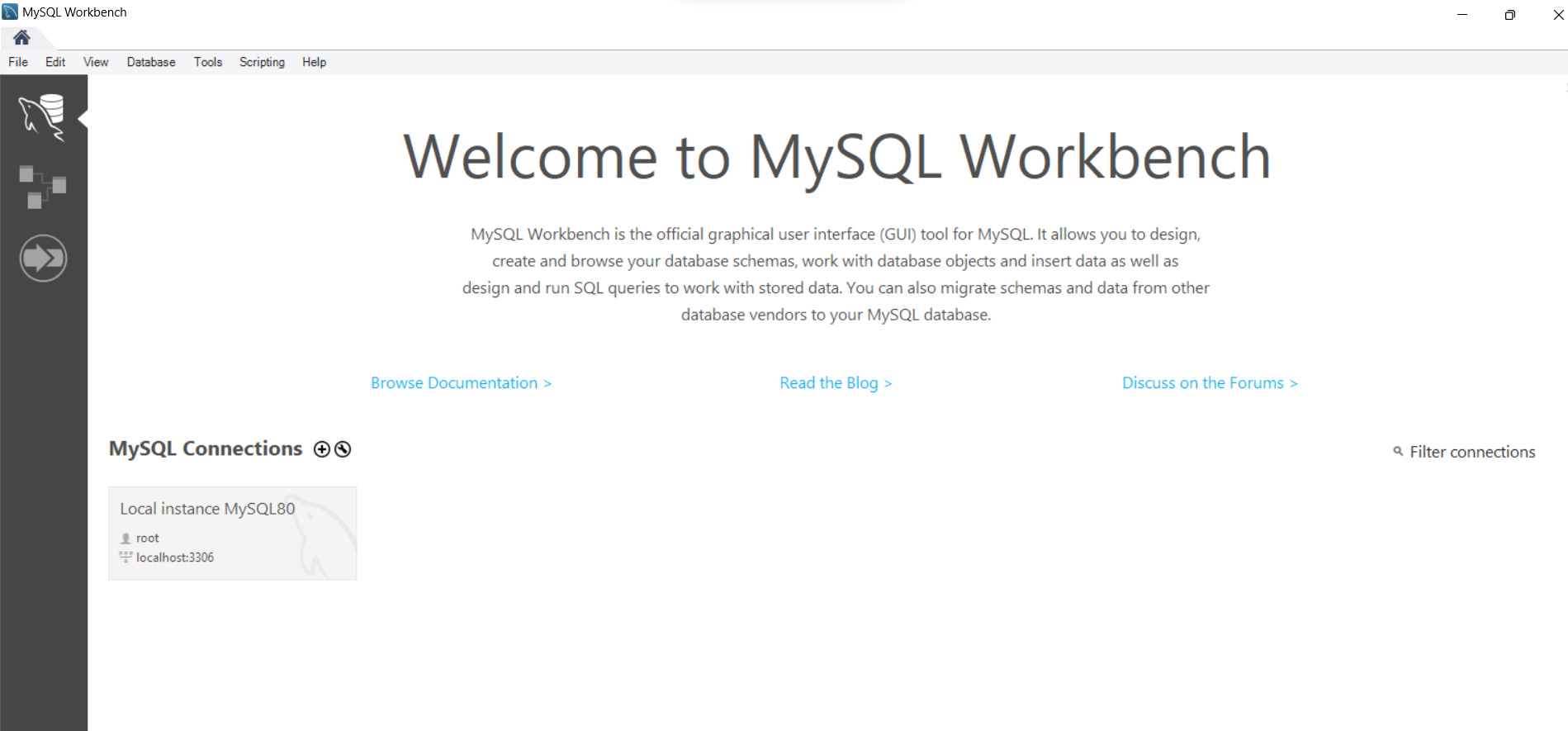
**Ecommerce App**

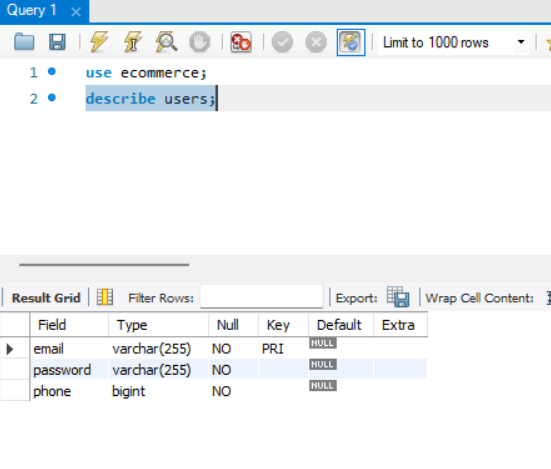
**This is an ecommerce application for shopping electronic products. The product was built using React , Express and Mysql. The project was build by making maximum use of Gen AI techniques. The Gen AI tool used here was Sourcegraph Cody**

**Design:-**

1. **Front End:-**The front end was built using react. The front end consists of home page   
   login page , register page and forgot password page. Upon logging in user can view home, product catalog , cart and payment page upon checking out.
2. **Back End:-** The backend was built using Express. The backendend has all the functionalities to interact with the database. All requests from react UI to the database are handled using the express API endpoints.
3. **SQL database:-** Thesql database stores users information in a database.

**Architecture:-**

1. **Front End Architecture:-** Made full use of react features to split the app into components. The components are as follows:-  
   **Home:-** Carousel consisiting of the products available on the app.   
   **Header:-** Header consisiting of title and navigation links . The navigation link changes based on whether user is logged in or not.  
   **Cart:-** Cart consisting of all the products use wants to purchase  
   **Login:-** Login component where user logins using username and password and OTP. Also provides a facility for user to change password.  
   **Product**:- Product component which consists of list of all available products their description and price. Products are static but there is scope to extend them to database  
   **Register:-** This component provides users option to create accounts on the app. **Payment:-** This component provides users option to make payment using credit card.  
   **PasswordChange:-**This component provides users options to change password.  
     
   In addition to the components use of redux state was also made to store information about whether user is logged in or not
2. **Back End Architecture:-** The backend consists of API endpoints to interact with the database.Following are the API endpoints:- **/login:-** This is for the user to login . A check is done if credentials are present in the database and if present , mails OTP and returnsOTP to frontend to validate if otp entered by user matches with the generated one.  
   **/register:-** This for users to register   
   **/passwordChange:-** This is for users to change password  
   **/forgot-password:-**Thie resets users password, updates the database and sneds reset password as a mail to the user
3. **SQL Database:-** Database service used was mysql workbench. It contain an ecommerce database. The database contains users table:- Below it the architecture of the database  
   



**Microservice:-**

This is the high level microservice diagram of how all the microservices interact with each other. React makes API calls to express app to perform database related operations. Express interacts with database to modify content in it and returns the operation status to React.

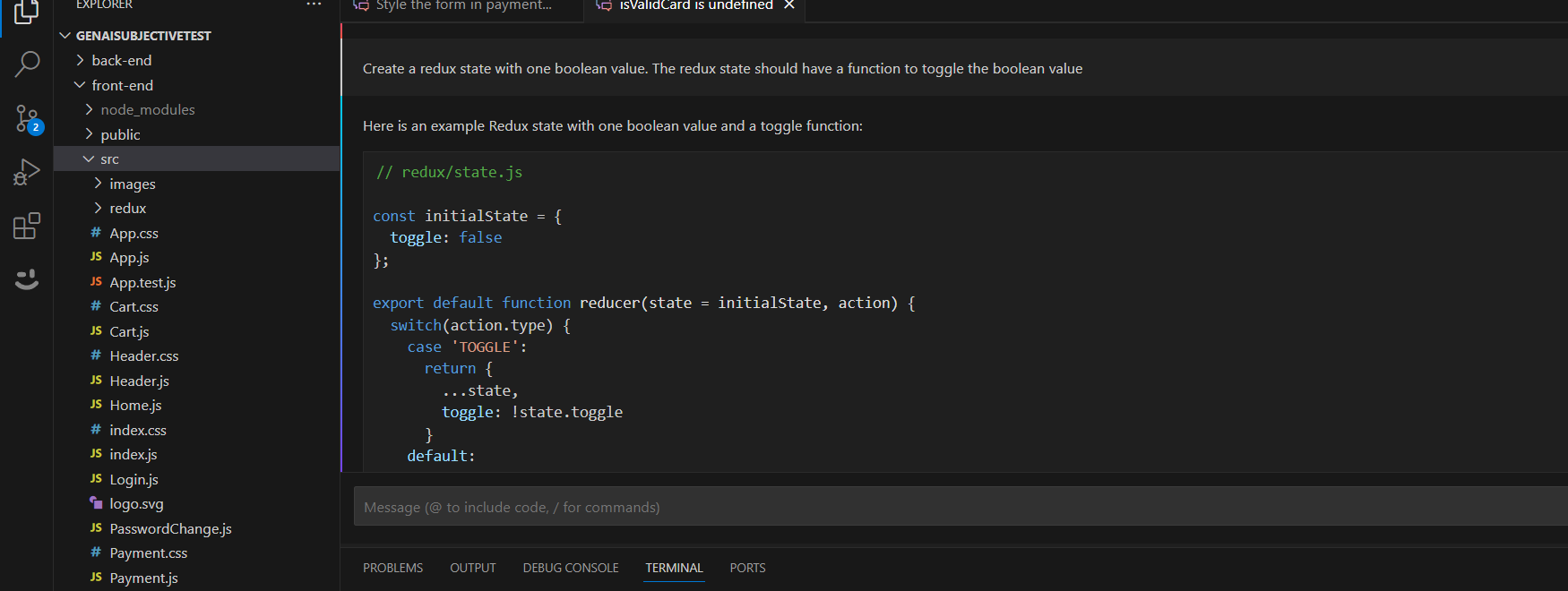
React

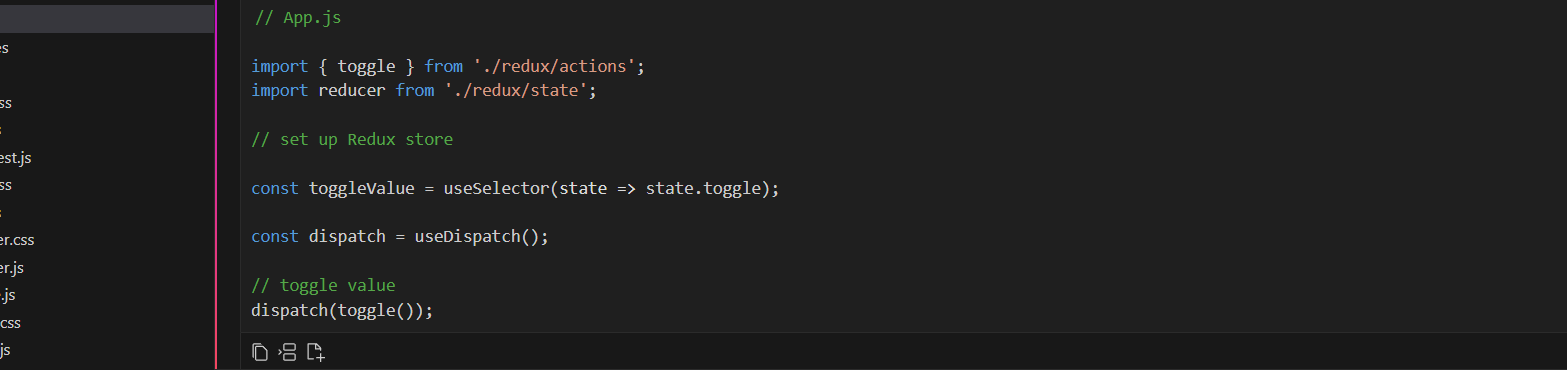
Expresss

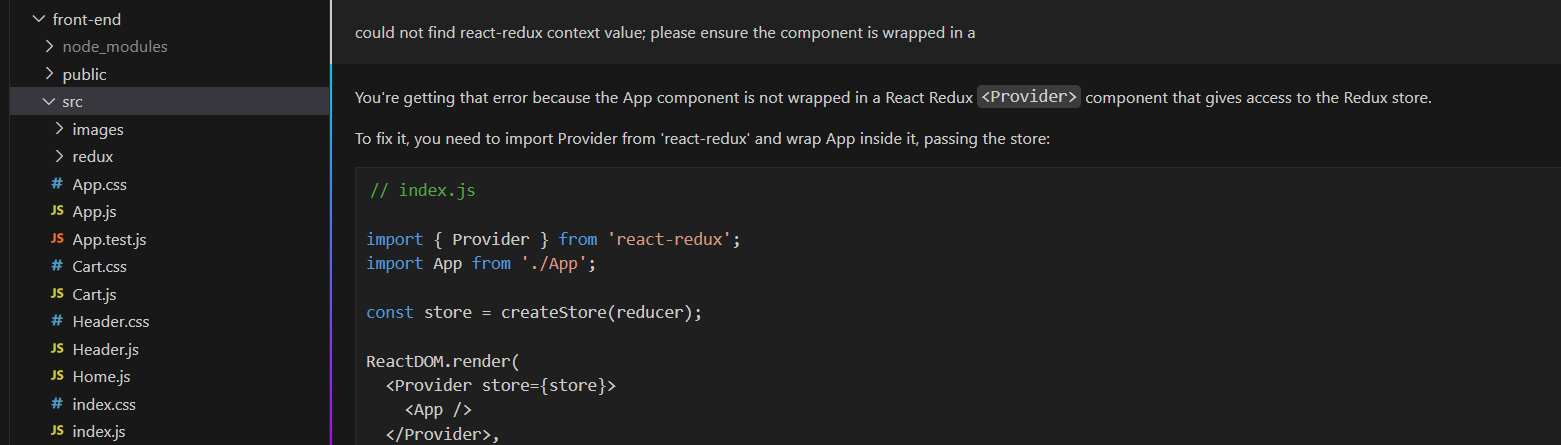
Database

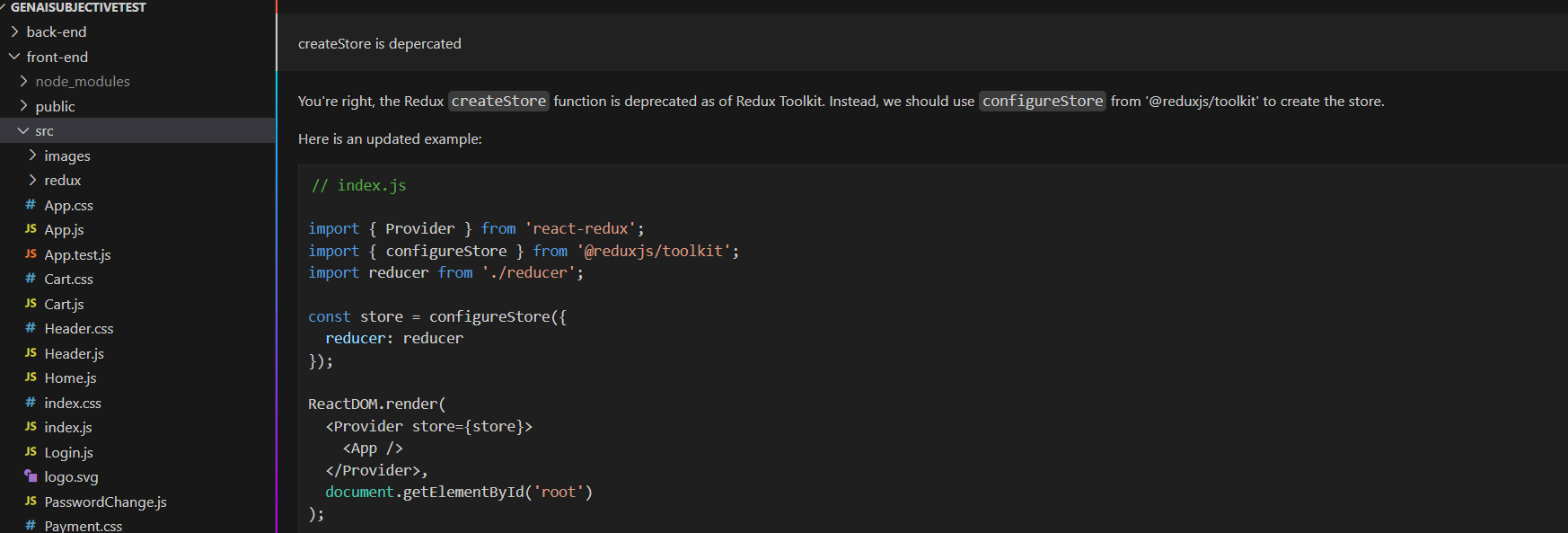
**Database scripts:-**The database script used to create users table:-  
mysql create database users(  
email varchar(255) primary key not null,  
password varchar(255) not null,  
phone bigint not null);  
The remaining queries are present in back-end/index.js in the repo  
  
**Code:-**The frontend react code is present in front-end folder  
The backend express code is present in back-end folder  
  
**Unit test cases:-**The unit test cases generated by Cody for backend are present in back-end/unit-test/unit-test.js  
The unit test cases generated by Cody for frontend are present in the form front-end/\*test.js

**Prompts:-**Below are some of the snapshots of using cody. List of all the prompts used are present in Prompts-BackEnd.txt and Prompts-FrontEnd.txt  
   
An example showing how I setup redux store using Cody . It has the prompts I have used to modify Cody’s output which was initially not working and was refined lateron to get the expected output.









Example of setting up a 2 x 2 grid on products page. Refined the output to my need.

