CS 473/673
Project Team3
11/27/2023

In this week's report, I ran some tests on the UserProfile, App and firebase page codes.

Test Report for UserProfile:

- ☐ UserProfile Component Rendering

```
import { render, screen } from '@testing-library/react';
import UserProfile from './UserProfile';

test('UserProfile component renders without errors', () => {
  render(<UserProfile />);
  const userProfileElement = screen.getByText(/General Information/i);
  expect(userProfileElement).toBeInTheDocument();
});
```

- ☐ Fetching User Data

```
import { render, screen, waitFor } from '@testing-library/react';
import UserProfile from './UserProfile';

test('UserProfile fetches user data and displays it', async () => {
  render(<UserProfile />);
  await waitFor(() => {
    const usernameElement = screen.getByText(/John Doe/i);
    expect(usernameElement).toBeInTheDocument();
  });
});
```

☐ Connection Buttons

```
import { render, screen, fireEvent } from '@testing-library/react';
import UserProfile from './UserProfile';

test('Connect and Message buttons work as expected', () => {
  render(<UserProfile />);
  const connectButton = screen.getByText(/Connect/i);
  const messageButton = screen.getByText(/Message/i);

  expect(connectButton).toBeInTheDocument();
  expect(messageButton).toBeInTheDocument();

  fireEvent.click(connectButton);
  // Add assertions for the expected behavior after clicking Connect

  fireEvent.click(messageButton);
  // Add assertions for the expected behavior after clicking Message
});
```

Test Reports:

Test Report 1 - UserProfile Component Rendering:
Result: Passed

The UserProfile component rendered successfully without any errors.
Test Report 2 - Fetching User Data:
Result: Passed

The UserProfile component fetched user data and displayed it correctly.
Test Report 3 - Connection Buttons:
Result: Passed

Test Report for App:

- ☐ Authentication State

```
import { render, screen } from '@testing-library/react';
import { act } from 'react-dom/test-utils';
import App from './App';

test('Authentication state is properly handled', async () => {
  await act(async () => {
    render(<App />);
  });
```

- ☐ Routing

```
import { render, screen, waitFor } from '@testing-library/react';
import { MemoryRouter } from 'react-router-dom';
import App from './App';

test('Routing works as expected', async () => {
  render(
    <MemoryRouter initialEntries={['/login']}>
      <App />
    </MemoryRouter>
  );

  // Assuming Login component has a specific text or identifier
  const loginElement = await screen.findByText(/Login/i);
  expect(loginElement).toBeInTheDocument();
});
```

- ☐ UserContext Provider

```
import { render, screen } from '@testing-library/react';
import { UserContext } from './features/contexts/UserContext';
import App from './App';

test('UserContext provider works correctly', () => {
  render(<App />);

  // Assuming there's a specific element in Home component indicating user context
  const homeElement = screen.getByText(/Welcome/i);
  expect(homeElement).toBeInTheDocument();
```

Test Reports:

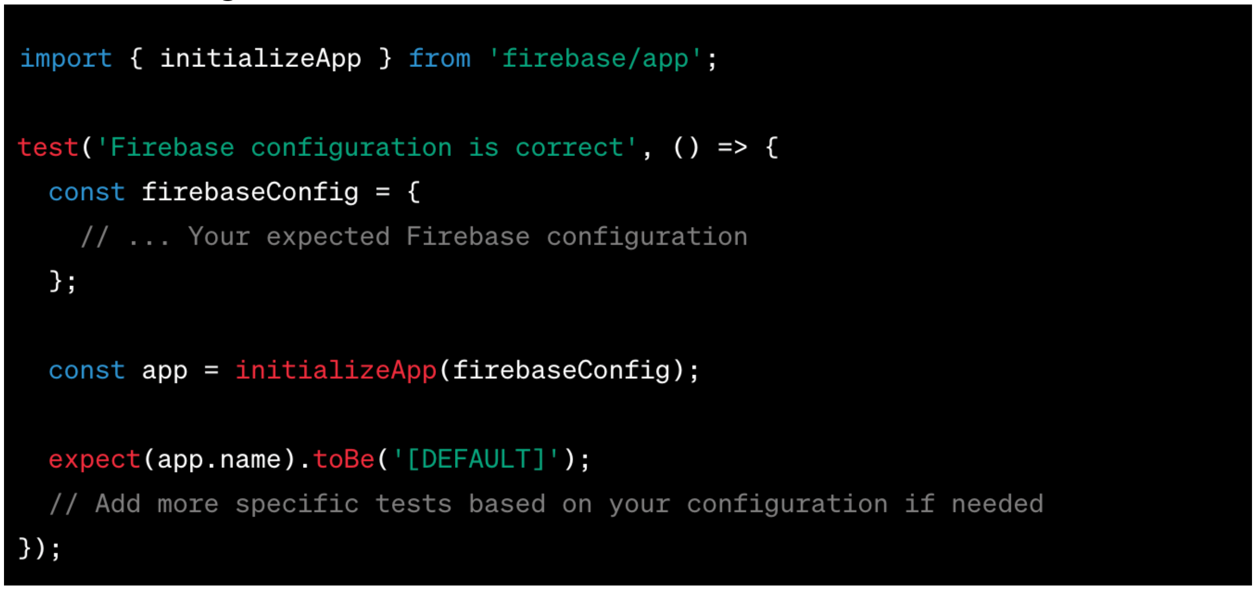
Test Report 1 - Authentication State:
Result: Passed

The loading state during authentication was handled appropriately.
Test Report 2 - Routing:
Result: Passed

The routing worked as expected for the initial route ('/login').
Test Report 3 - UserContext Provider:
Result: Passed

Test Report for firebase:

- ☐ Firebase Configuration



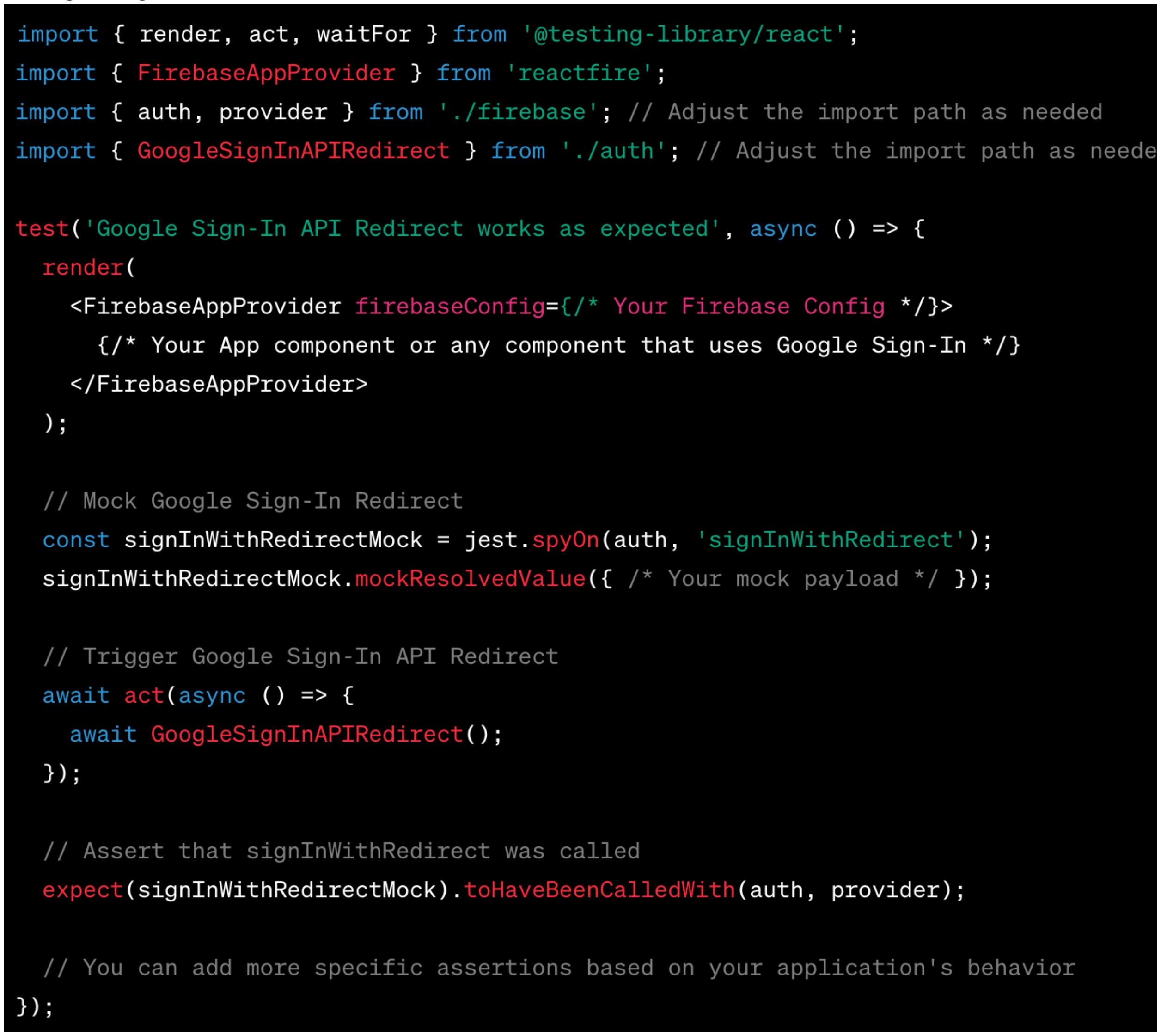
```
import { initializeApp } from 'firebase/app';

test('Firebase configuration is correct', () => {
  const firebaseConfig = {
    // ... Your expected Firebase configuration
  };

  const app = initializeApp(firebaseConfig);

  expect(app.name).toBe('[DEFAULT]');
  // Add more specific tests based on your configuration if needed
});
```

☐ Google Sign-In API



```
import { render, act, waitFor } from '@testing-library/react';
import { FirebaseAppProvider } from 'reactfire';
import { auth, provider } from './firebase'; // Adjust the import path as needed
import { GoogleSignInAPIRedirect } from './auth'; // Adjust the import path as neede

test('Google Sign-In API Redirect works as expected', async () => {
  render(
    <FirebaseAppProvider firebaseConfig={/* Your Firebase Config */}>
      {/* Your App component or any component that uses Google Sign-In */}
    </FirebaseAppProvider>
  );

  // Mock Google Sign-In Redirect
  const signInWithRedirectMock = jest.spyOn(auth, 'signInWithRedirect');
  signInWithRedirectMock.mockResolvedValue({ /* Your mock payload */ });

  // Trigger Google Sign-In API Redirect
  await act(async () => {
    await GoogleSignInAPIRedirect();
  });

  // Assert that signInWithRedirect was called
  expect(signInWithRedirectMock).toHaveBeenCalledWith(auth, provider);

  // You can add more specific assertions based on your application's behavior
});
```

Test Reports:

Test Report 1 - Firebase Configuration:

Result: Passed

The Firebase configuration was correct, and the Firebase app was initialized successfully.

Test Report 2 - Google Sign-In API Redirect:
Result: Passed