Submitted by: Sazzad Alam Tomal

Email: sazzadalamtomal00786@gmail.com

So, there are multiple types of real-time data-fetching solutions. The most common are HTTP short and long polling, Server-Side Events, and WebSocket's. Here we will discuss in general terms how to troubleshoot the problem.

Steps of troubleshooting

General-Troubleshooting

- Running test cases.
- Check Internet connection.
- Client-server is operational.
- API server is online.
- Check the console for errors.
- Cross-browser Compatibility.

API Testing

First of all, we need to test the REST API endpoint to determine whether the API is responding to expected data or not.

We can use API testing tools like Postman to determine it.

If it is not working properly then-

Some questions we, as developers, can focus on to troubleshoot

- Is the REST endpoint still supported by the API?
- Is there any additional configuration needed?
- Are we using the latest API version?
- Can we find any changes in API Documentation that can break our app?
- Has our quota for the API somehow fulfilled?
- Has our API key been somehow rejected or compromised?

Potential Challenges

- Don't have access to proper API documentation.
- The quota for the API cannot be extended.
- The protocol of the API is fully Changed. Ex- Http to WebSocket's.
- Cannot get a new API key in time.

If the API is working properly then we can move on to Our React App testing

General Troubleshooting in React App

First, we have to check how the data is being fetched in the app-

Some questions we, as developers, can focus on to troubleshoot

Data Fetching

- Is the endpoint for the API correct or not?
- Are we fetching the data in a particular interval that is short enough to assume it is a real-time data fetching?
- Are we using other ways of fetching data, such as Server-Side events? If yes, then are the events configured properly or not?
- Which method are we using for the data fetching and whether it is properly configured or not?
- If we are using third-party data fetching libraries like Axios, React-query, or Rtk-query, is there any implantation problem in that? Like using interceptors in Axios which can create problems in the normal data flow if it is not configured properly or in Rtk-query and React-query the interval is properly configured or not?
- In react, we use the useEffect hook which is triggered by components' life cycle events where we fetch data, is properly configured or not?
- If we are using any custom hook for fetching data, is it properly configured or not?

State Management

After we have validated that the app is properly fetching data from the API-

Some questions we, as developers, can focus on to troubleshoot: -

- Is the state of the app properly updating or not?
- Configuration of the useState hook properly configured or not?
- Is the data fetched in the useEffect hook properly setting the state or not?
- The structure of the state is right or not?
- If the data has to be sent in a child as a prop, the data flow is correct or not?
- If we are using third-party state management tools like Redux, is it properly updating and passing the state or not?
- Is there any middleware causing some problems or not?

Third-party Library for Charts

- Is the data received by useState or props properly used by the third-party chart library or not?
- Is there any caching mechanism that is stopping it from updating the data?

Resolving The Issue

After we figured out the potential cause of the problem, we had to configure the React App accordingly.

- **Have to make sure to change the state properly, every time new data is being received.**
- **Making sure that react is re-rendering the component every time the state is being changed.**
- **Sending props to the child components properly so that React can re-render them also.**
- **Giving each child element a unique key so that they can re-render properly.**
- **Properly configure the useEffect hook, especially the dependency array.**
- **Fetch Data with low enough intervals.**
- **Reconfigure custom hooks if needed.**
- **Make new test cases to troubleshoot this kind of problem.**