React-Redux Guide for XSPACE and Best Practices

There are three methods that the front end use to promise the data to the front end.

- `fetch`
- `axios`
- `dispatch()` through react-redux and RSAA

When to use fetch

- Use `fetch()` for **GET** data that does not need to be persisted and can be disposed of after viewing
- Be aware of back-nav

When to Axios

- Use when a **GET** request to another API that doesn't require a token
- Use when a **POST** request to another API that doesn't require a token
- Safer to wrap with React Component state
- Better at handling other request types in addition to Content-Type: Application/JSON

```
axios.get('/user?ID=12345')
```

```
  .then(function (response) {
    console.log(response);
  })
  .catch(function (error) {
    console.log(error);
  });
```

When to use RSAA (Redux Standard API-Calling Actions)

- Must use if data requires access token
- When data needs to be persisted across the app
- Need to block acccess to data if user is unauthorized
- Long upload jobs

Below is an RSAA action

```
//actions/model.js

import { RSAA } from `redux-api-middleware`;
{
  [RSAA]: {
    endpoint: 'http://www.example.com/api/users',
    method: 'GET',
    types: ['X_REQUEST', 'X_SUCCESS', 'X_FAILURE']
  }
}
```

Here is a matching reducer.

```
//reducers/model.js
export default (state=initialState, action) => {
  switch(action.type) {

    case products.X_REQUEST:
      return state
    case products.X_SUCCESS:
      return {
        ...state,
        product: action.payload
      }
    case products.X_FAILURE:
      return {
        ...state,
        product: action.payload
      }
      ...
```

For the retamining of the redux steps follow this guide.