

## 연습문제 12

The screenshot shows a web browser at localhost:3000 displaying a Redux News Viewer application. The application shows a list of news items with titles, images, and brief descriptions. The Redux DevTools extension is open, showing the state of the application. The state is an array of news items, each with an id, title, author, image, and description. The Redux DevTools interface shows the state being updated with new news items.

src/index.js

```
import React from 'react';
import ReactDOM from 'react-dom/client';
import App from './App';
import { BrowserRouter } from 'react-router-dom';
import { Provider } from 'react-redux';
import store from './store';
const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
  <React.StrictMode>
    <Provider store={store}>
      <BrowserRouter>
        <App />
      </BrowserRouter>
    </Provider>
  </React.StrictMode>
);
```

src/App.js

```
import React, { memo } from 'react';
import News from './pages/News';
const App = memo(() => {
```

```
    return (  
      <div>  
        <h1>Redux News Viewer</h1>  
        <hr />  
        <News />  
      </div>  
    );  
  });  
export default App;
```

## src/store.js

```
import { configureStore } from '@reduxjs/toolkit';  
import NewsSlice from './slices/NewsSlice';  
const store = configureStore({  
  reducer: {  
    news: NewsSlice,  
  },  
  middleware: (getDefaultMiddleware) =>  
    getDefaultMiddleware({ serializableCheck: false }),  
  devTools: true,  
});  
export default store;
```

## src/slices/NewsSlice.js

```
import { createSlice, createAsyncThunk } from '@reduxjs/toolkit';  
import axios from 'axios';  
export const getNewsList = createAsyncThunk(  
  'NewsSlice/getNewsList',  
  async (payload, { rejectWithValue }) => {  
    let result = null;  
    try {  
      result = await axios.get('http://localhost:3001/news');  
    } catch (err) {  
      result = rejectWithValue(err.response);  
    }  
    return result;  
  }  
);  
const NewsSlice = createSlice({  
  name: 'news',  
  initialState: {  
    data: null,  
    loading: false,  
  },  
  reducers: {},  
  extraReducers: (builder) => {  
    builder.addCase(getNewsList.pending, (state, action) => {  
      state.loading = true;  
    });  
    builder.addCase(getNewsList.fulfilled, (state, action) => {  
      state.data = action.payload;  
      state.loading = false;  
    });  
    builder.addCase(getNewsList.rejected, (state, action) => {  
      state.loading = false;  
    });  
  },  
});  
export const { data, loading } = NewsSlice.reducer;
```

```
    error: null,
  },
  reducers: {},
  extraReducers: {
    [getNewsList.pending]: (state, { payload }) => {
      return { ...state, loading: true };
    },
    [getNewsList.fulfilled]: (state, { payload }) => {
      return {
        data: payload?.data,
        loading: false,
        error: null,
      };
    },
    [getNewsList.rejected]: (state, { payload }) => {
      return {
        data: payload?.data,
        loading: false,
        error: {
          code: payload?.status ? payload.status : 500,
          message: payload?.statusText ? payload.statusText : 'Server
Error',
        },
      };
    },
  },
});
export default NewsSlice.reducer;
```

---