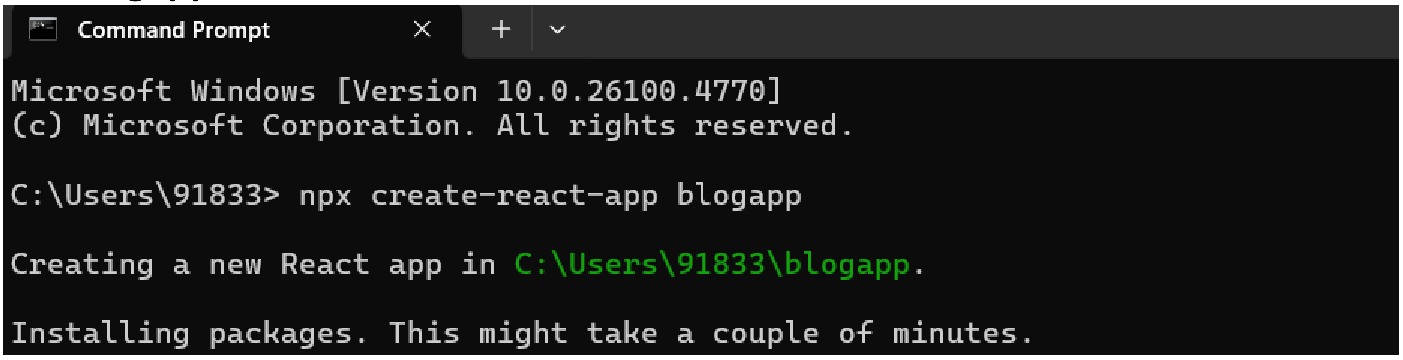
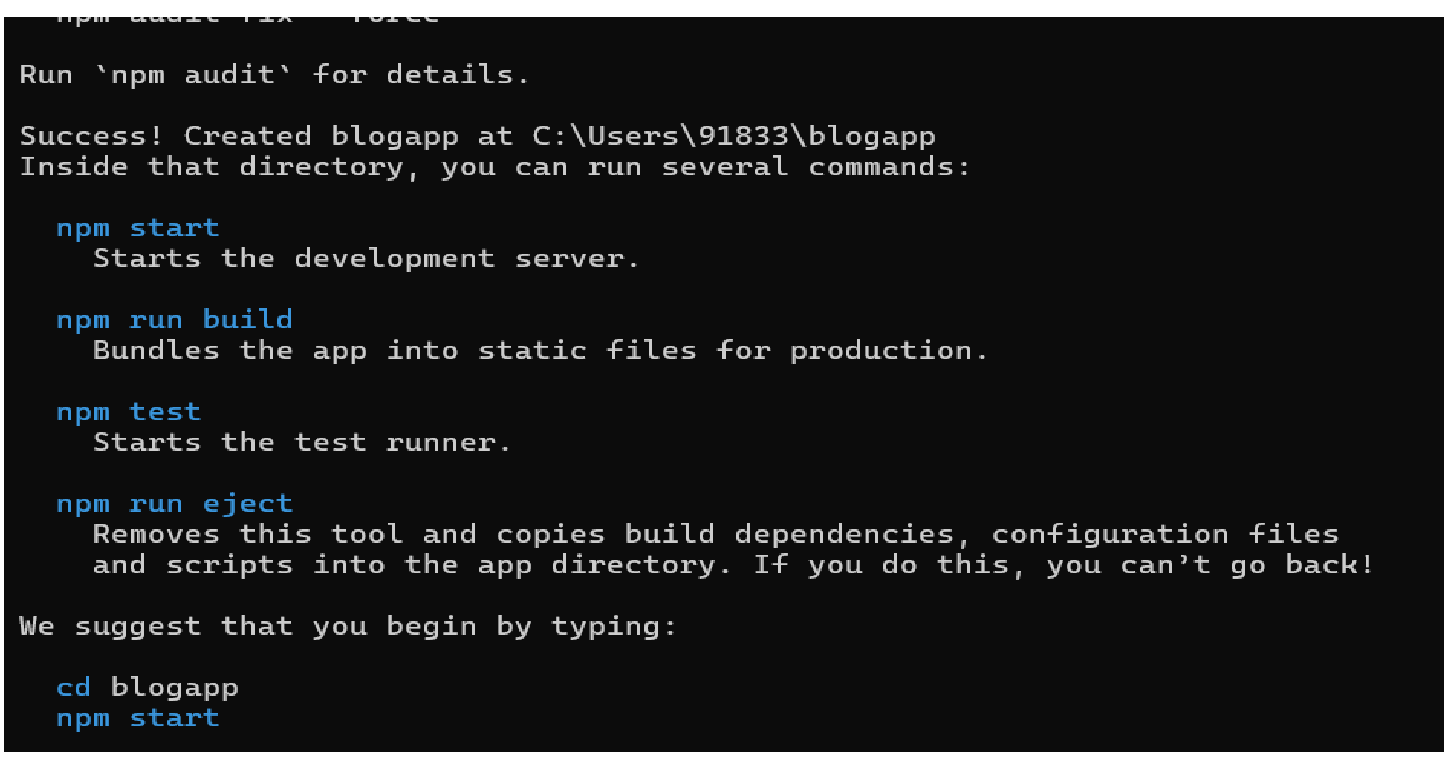
4. ReactJS-HOL From React Folder

Create a new react application using *create-react-app* tool with the name as **“blogapp”**

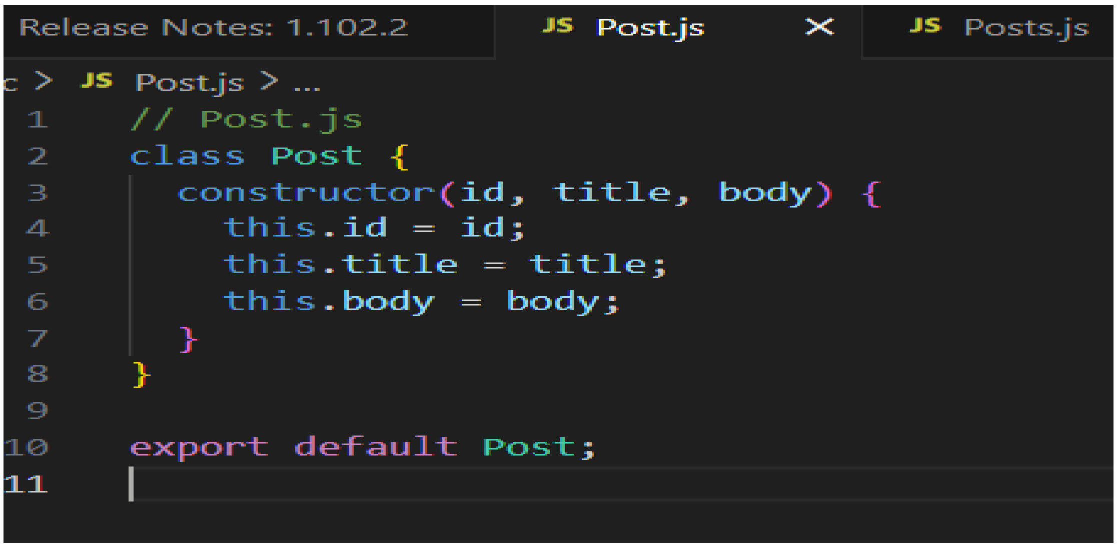


```
Microsoft Windows [Version 10.0.26100.4770]
(c) Microsoft Corporation. All rights reserved.

C:\Users\91833> npx create-react-app blogapp

Creating a new React app in C:\Users\91833\blogapp.

Installing packages. This might take a couple of minutes.
```



```
Run `npm audit` for details.

Success! Created blogapp at C:\Users\91833\blogapp
Inside that directory, you can run several commands:

  npm start
    Starts the development server.

  npm run build
    Bundles the app into static files for production.

  npm test
    Starts the test runner.

  npm run eject
    Removes this tool and copies build dependencies, configuration files
    and scripts into the app directory. If you do this, you can’t go back!

We suggest that you begin by typing:

  cd blogapp
  npm start
```

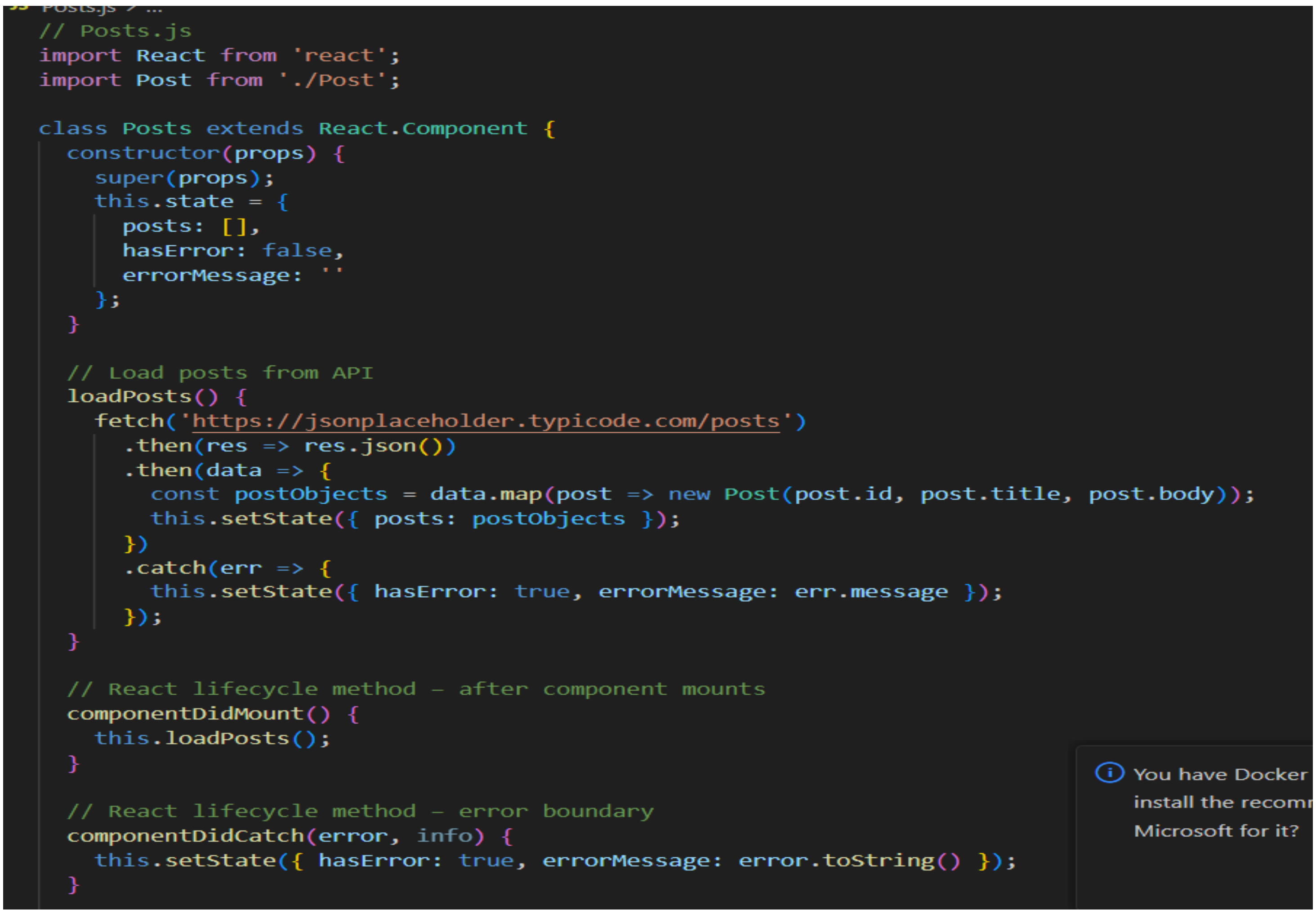
Create the Post.js in the src folder



```javascript
// Post.js
class Post {
  constructor(id, title, body) {
    this.id = id;
    this.title = title;
    this.body = body;
  }
}

export default Post;
```

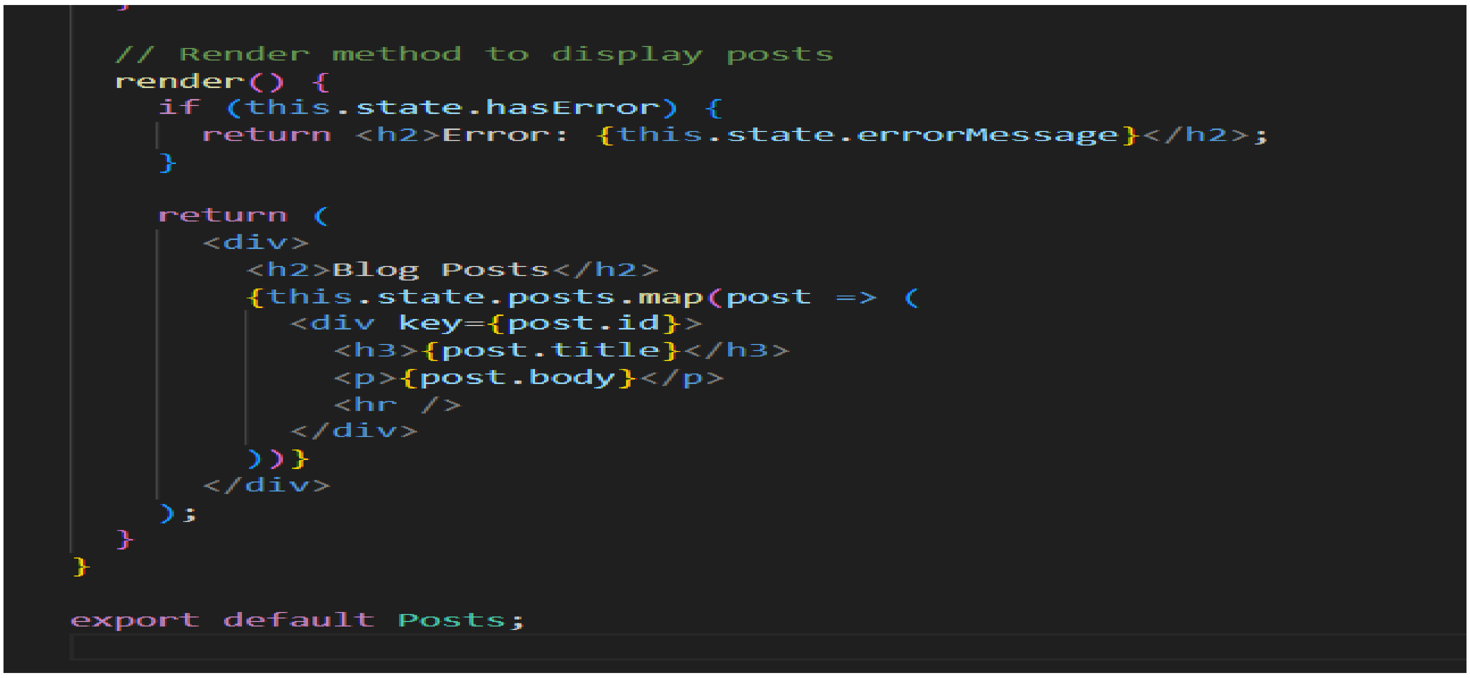
➔ Create a new class based component named as Posts inside Posts.js file create loadmethods in that , Implement the componentDidMount() hook to make calls to loadPosts(),Implement the render(),Define acomponentDidCatch() method. In this file.

Code:



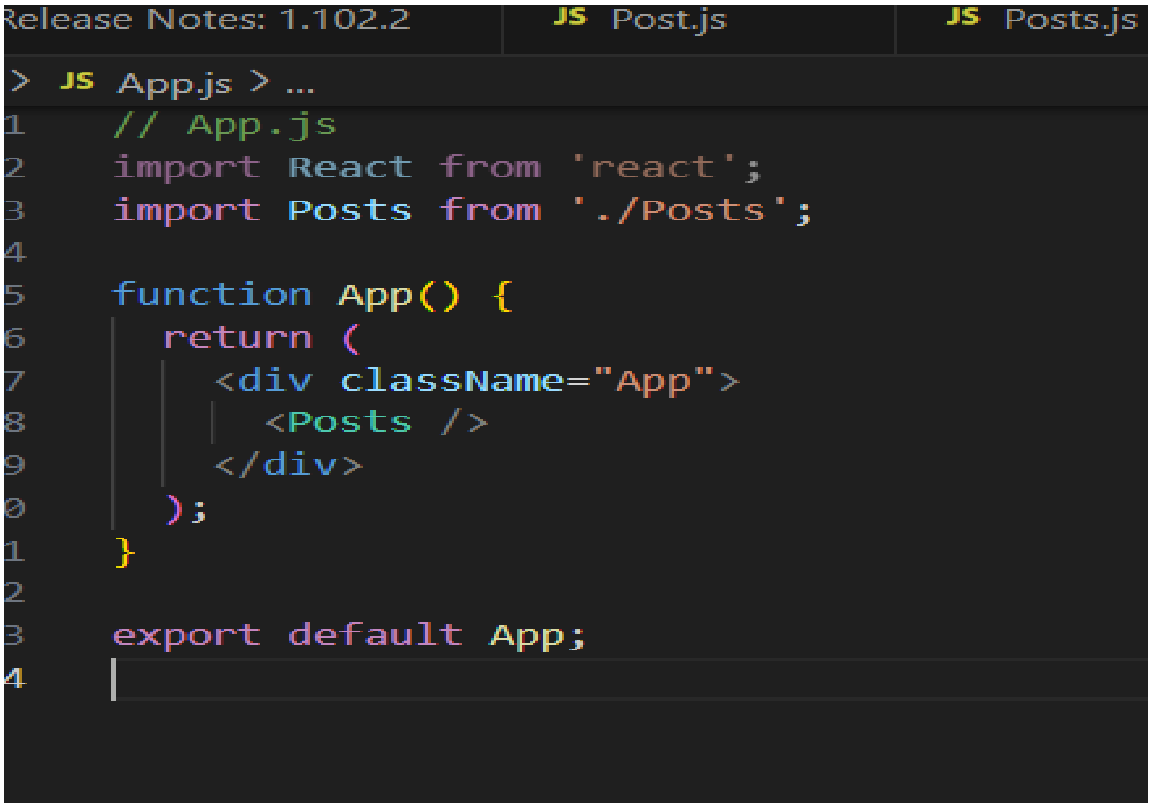
```
// Posts.js
import React from 'react';
import Post from './Post';

class Posts extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      posts: [],
      hasError: false,
      errorMessage: ''
    };
  }

  // Load posts from API
  loadPosts() {
    fetch('https://jsonplaceholder.typicode.com/posts')
      .then(res => res.json())
      .then(data => {
        const postObjects = data.map(post => new Post(post.id, post.title, post.body));
        this.setState({ posts: postObjects });
      })
      .catch(err => {
        this.setState({ hasError: true, errorMessage: err.message });
      });
  }

  // React lifecycle method - after component mounts
  componentDidMount() {
    this.loadPosts();
  }

  // React lifecycle method - error boundary
  componentDidCatch(error, info) {
    this.setState({ hasError: true, errorMessage: error.toString() });
  }
```

You have Docker
install the recomm
Microsoft for it?



```
  // Render method to display posts
  render() {
    if (this.state.hasError) {
      return <h2>Error: {this.state.errorMessage}</h2>;
    }

    return (
      <div>
        <h2>Blog Posts</h2>
        {this.state.posts.map(post => (
          <div key={post.id}>
            <h3>{post.title}</h3>
            <p>{post.body}</p>
            <hr />
          </div>
        ))}
      </div>
    );
  }
}

export default Posts;
```

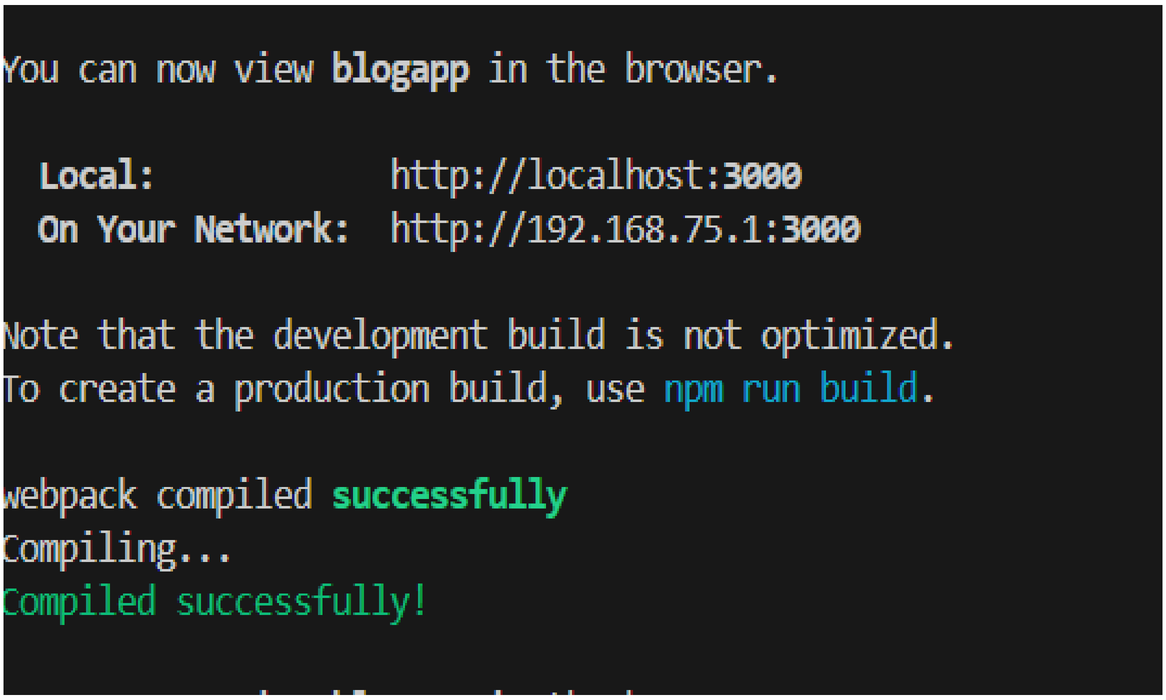
Change the App.js code to this:



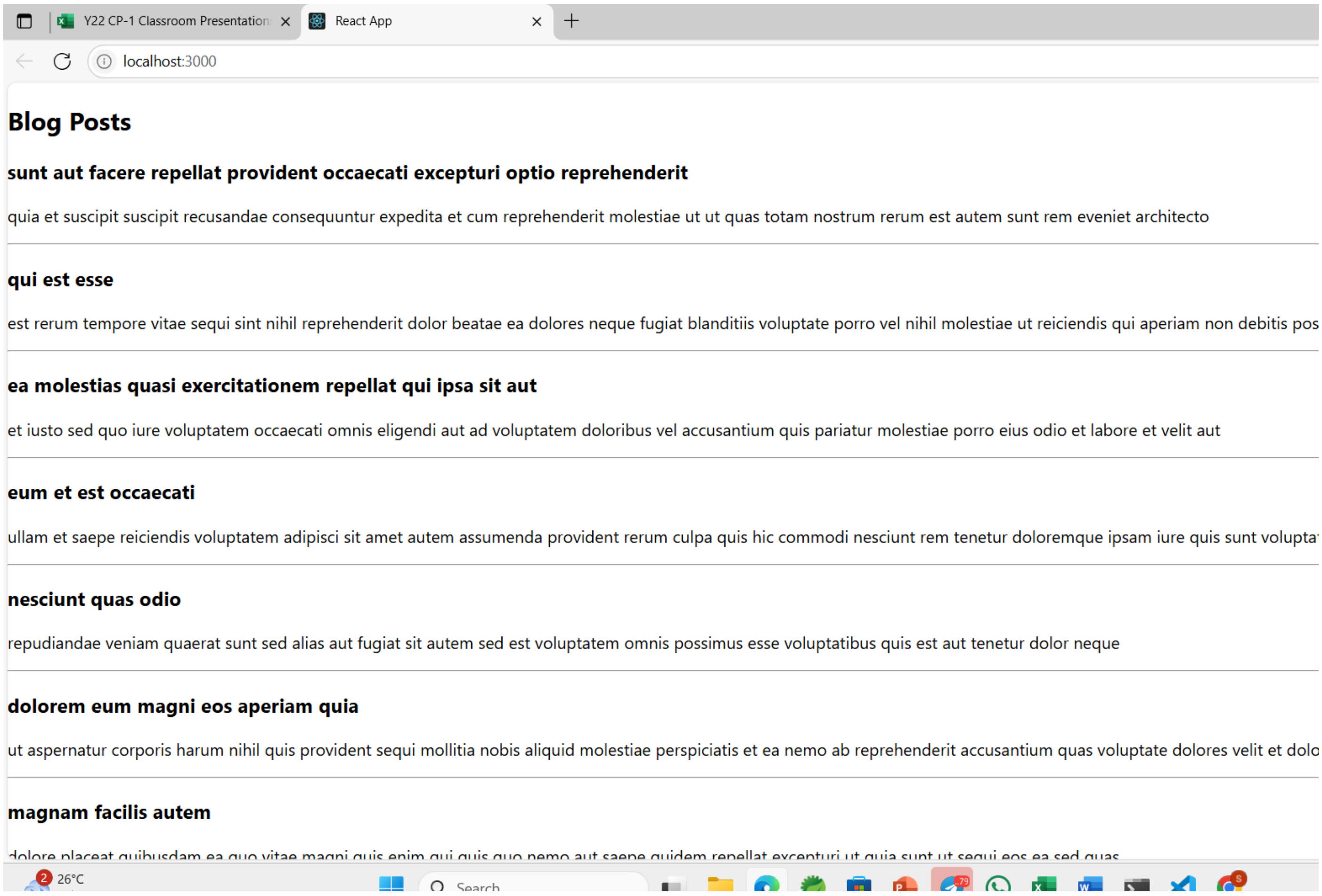
```javascript
// App.js
import React from 'react';
import Posts from './Posts';

function App() {
  return (
    <div className="App">
      <Posts />
    </div>
  );
}

export default App;
```

Output:

npm start



```
You can now view blogapp in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://192.168.75.1:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
Compiling...
Compiled successfully!
```



Blog Posts

sunt aut facere repellat provident occaecati excepturi optio reprehenderit

quia et suscipit suscipit recusandae consequuntur expedita et cum reprehenderit molestiae ut ut quas totam nostrum rerum est autem sunt rem eveniet architecto

qui est esse

est rerum tempore vitae sequi sint nihil reprehenderit dolor beatae ea dolores neque fugiat blanditiis voluptate porro vel nihil molestiae ut reiciendis qui aperiam non debitis pos

ea molestias quasi exercitationem repellat qui ipsa sit aut

et iusto sed quo iure voluptatem occaecati omnis eligendi aut ad voluptatem doloribus vel accusantium quis pariatur molestiae porro eius odio et labore et velit aut

eum et est occaecati

ullam et saepe reiciendis voluptatem adipisci sit amet autem assumenda provident rerum culpa quis hic commodi nesciunt rem tenetur doloremque ipsam iure quis sunt volupta

nesciunt quas odio

repudiandae veniam quaerat sunt sed alias aut fugiat sit autem sed est voluptatem omnis possimus esse voluptatibus quis est aut tenetur dolor neque

dolorem eum magni eos aperiam quia

ut aspernatur corporis harum nihil quis provident sequi mollitia nobis aliquid molestiae perspiciatis et ea nemo ab reprehenderit accusantium quas voluptate dolores velit et dolo

magnam facilis autem

dolore placeat quibusdam ea quo vitae magni quis enim qui quis quo nemo aut saepe quidem repellat excepturi ut quia sunt ut sequi eos ea sed quas