ENSF 381
Full Stack Web Development

Lecture 29: Bridging Frontend and Backend Systems

Slides: Ahmad Abdellatif, PhD

Instructor: Novarun Deb, PhD

UNIVERSITY OF CALGARY

Outline

- Recap.
- Cross-Origin Resource Sharing.
- Retrieving data sent from the frontend to the backend.
- Retrieving data sent from the backend to the frontend.

Recap: Fetch in React

- The fetch function is not specific to React.
- It is a part of the JavaScript language and is used for making HTTP requests.
- Fetch in React often involves integrating it with React's state management, component lifecycle, and JSX rendering to create dynamic and responsive user interfaces.

Recap: Fetch in React - Example

```
import React, { useState } from 'react';

function App() {
const [email, setEmail] = useState(null);
const [cellphone, setCellphone] = useState(null);
const [isLoading, setIsLoading] = useState(false);
```

Recap: Fetch in React - Example

```
async function fetchData() {
  try {
    // Set loading to true while data is being fetched
    setIsLoading(true);

    // Fetch data from an API
    const response = await fetch('https://api.randomuser.me/?nat=US&results=1');

    if (response.ok) {
      // Check if the request was successful
      let { results } = await response.json();

      // Parse the JSON data from the response
      let { email, cell } = results[0];

      // Set the fetched data to the state
      setEmail(email);
      setCellphone(cell);
    } else {
      // Handle error if the request was not successful
      console.error('Failed to fetch data:', response.statusText);
    }
  } catch (error) {
       // Handle network errors or other exceptions
       console.error('Error during data fetching:', error);
  } finally {
    setIsLoading(false); // Set loading to false once data fetching is complete
  }}
```

Recap: Fetch in React - Example

```
return (
  <div>
    <h1>Fetch Data Without useEffect</h1>
    {isLoading ? (
      <p>Loading...</p>
    ) : (
      <div>
        <button onClick={fetchData}>Fetch Data</button>
        {email && (
          <div>
            <h2>Fetched Data:</h2>
            <pre>{email}</pre>
            <pre>{cellphone}</pre>
          </div>
        )}
      </div>
    )}
  </div>
);
}
export default App;
```

Fetch in React - Example

Cross-Origin Resource Sharing (CORS)

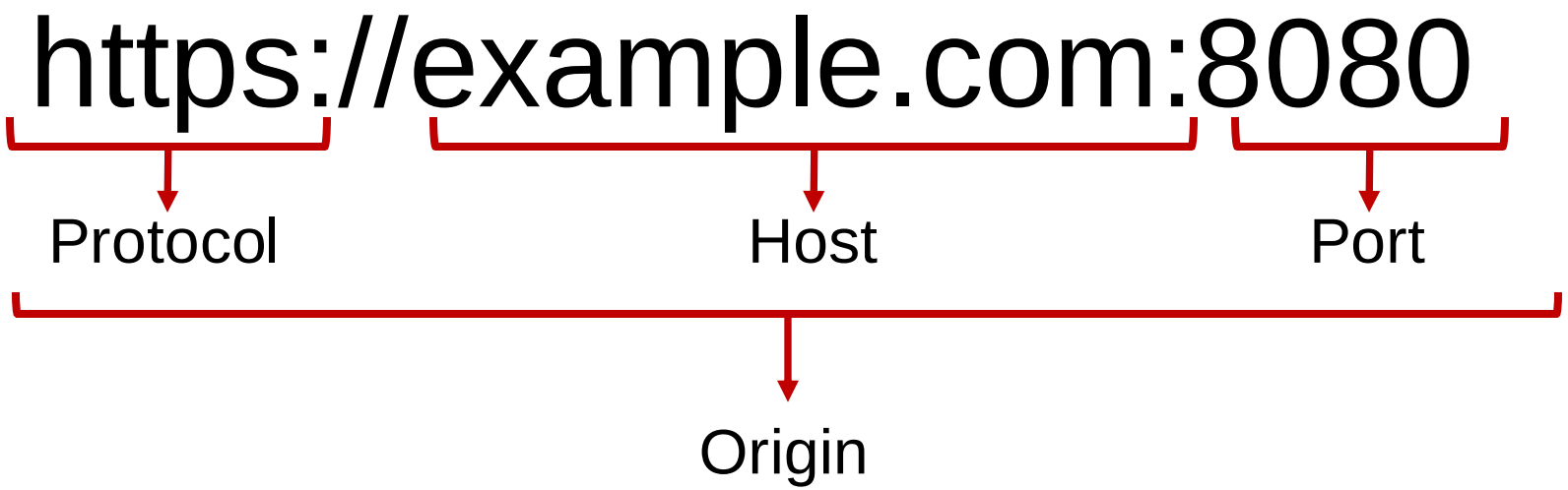
- A security feature implemented by web browsers to control how web pages in one domain can request and interact with resources hosted on another domain.

- It is a set of rules and policies that enable or restrict cross-origin (cross-site) HTTP requests initiated by client-side JavaScript code running in a web browser.

- The Same-Origin Policy (SOP) is a default security measure in web browsers that restricts web pages from making requests to a different domain than the one that served the web page.

Cross-Origin Resource Sharing (CORS)

- An origin is composed of a combination of protocol (HTTP or HTTPS), domain, and port (if specified) :



- When a web page from one origin makes a request to a different origin using client-side JavaScript (e.g., an AJAX request), it is considered a cross-origin request.

Why do we need to use CORS?

- In modern web development, it is common for web applications to fetch resources from APIs or servers hosted on different domains.

- **Without CORS**, these cross-origin requests would be blocked by the browser.

- We can enable CORS from the backend.

Flask-CORS

- A Flask extension to handle CORS.

- Install the extension with using pip: `pip install -U flask-cors`

- Syntax:

```
from flask import Flask
from flask_cors import CORS

app = Flask(__name__)
CORS(app)

@app.route("/")
def helloWorld():
  return "Hello, cross-origin-world!"
```

Connect frontend and backend – example (backend)

```
from flask import Flask
from flask_cors import CORS

app = Flask(__name__)
CORS(app)

# Assigning multiple routes to the same function
@app.route('/',methods=['POST'])
@app.route('/main',methods=['POST'])
def home():
  print('This is the home API')
  return 'Welcome to the home page!'

@app.route('/about')
def about():
  return 'Welcome to the About Us page!'

if __name__ == '__main__':
  app.run()
```

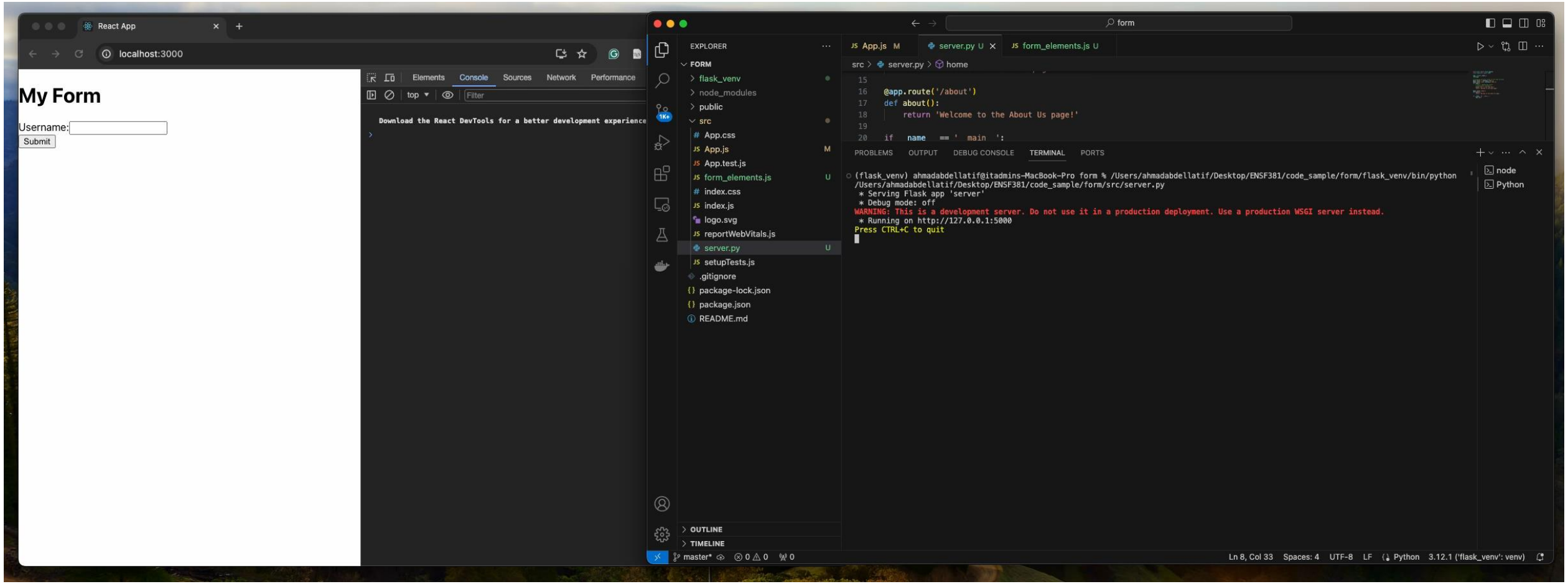
Connect frontend with backend – example (frontend)

```
import React, { useState } from 'react';
function MyFormComponent() {
// State to hold the username
const [username, setUsername] = useState('');

// Function to handle form submission
async function handleSubmit(event) {
  event.preventDefault();

  const backendEndpoint = 'http://127.0.0.1:5000';
  try {
    const response = await fetch(backendEndpoint, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify({'username':username }), //Converts a JavaScript object or value into a JSON string.
    });

    if (response.ok) {
      console.log('Form submitted successfully!');
    } else {
      console.error('Form submission failed.');
    }
  } catch (error) {
    console.error('Error during form submission:', error);  }};
```

Connect frontend with backend – example (frontend)

```
return (
  <div>
    <h1>My Form</h1>
    <form onSubmit={handleSubmit}>
      <label>
        Username:
      </label>
        <input
          name='username'
          type="text"
          onChange={(e) => setUsername(e.target.value)}
        />
      <br />
      <button type="submit">Submit</button>
    </form>
  </div>
);
};

export default MyFormComponent;
```

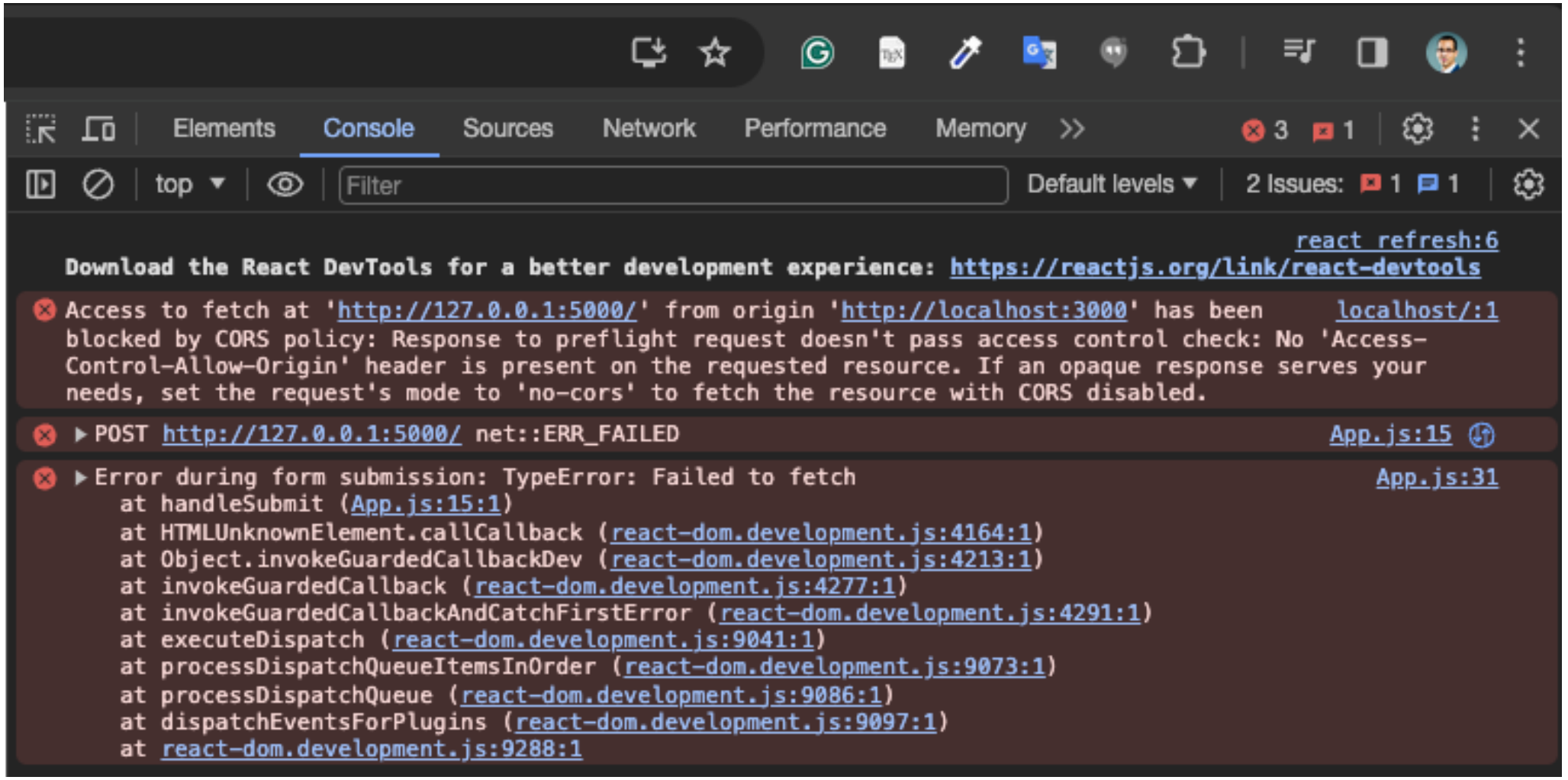
Connect frontend with backend – example (frontend)



What happens if we run the code without CORS?

```
from flask import Flask
from flask_cors import CORS

app = Flask(__name__)
CORS(app)
# Assigning multiple routes to the same function
@app.route('/',methods=['POST'])
@app.route('/main',methods=['POST'])
def home():
  print('This is the home API')
  return 'Welcome to the home page!'

@app.route('/about')
def about():
  return 'Welcome to the About Us page!'

if __name__ == '__main__':
  app.run()
```

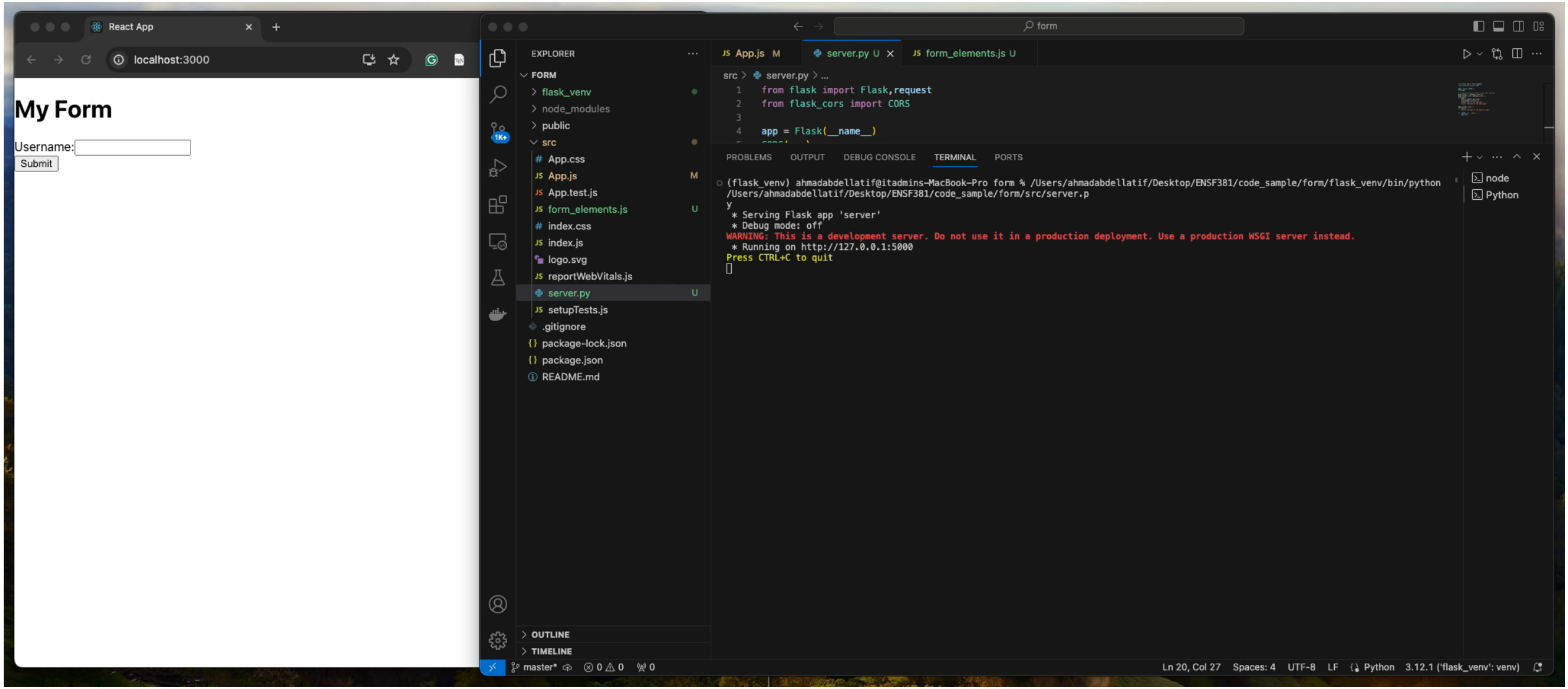
What happens if we run the code without CORS?



Retrieving data sent from the frontend to the backend - example (backend)

```python
from flask import Flask,request
from flask_cors import CORS

app = Flask(__name__)
CORS(app)

# Assigning multiple routes to the same function
@app.route('/',methods=['POST'])
@app.route('/main',methods=['POST'])
def home():
  data = request.get_json() # Retrieves the JSON data from the incoming request and store it in a dictionary
  print(data['username']) # Accesses the 'username' field from the data dictionary
  print('This is the home API')
  return 'Welcome to the home page!'

@app.route('/about')
def about():
  return 'Welcome to the About Us page!'

if __name__ == '__main__':
  app.run()
```

Retrieving data sent from the frontend to the backend - example (backend)



Retrieving text data sent from the backend to the frontend – example (backend)

```
from flask import Flask,request
from flask_cors import CORS

app = Flask(__name__)
CORS(app)

# Assigning multiple routes to the same function
@app.route('/',methods=['POST'])
@app.route('/main',methods=['POST'])
def home():
  data = request.get_json()
  print(data['username'])
  return 'Welcome {} to the home page!'.format(data['username'])

@app.route('/about')
def about():
  return 'Welcome to the About Us page!'

if __name__ == '__main__':
  app.run()
```

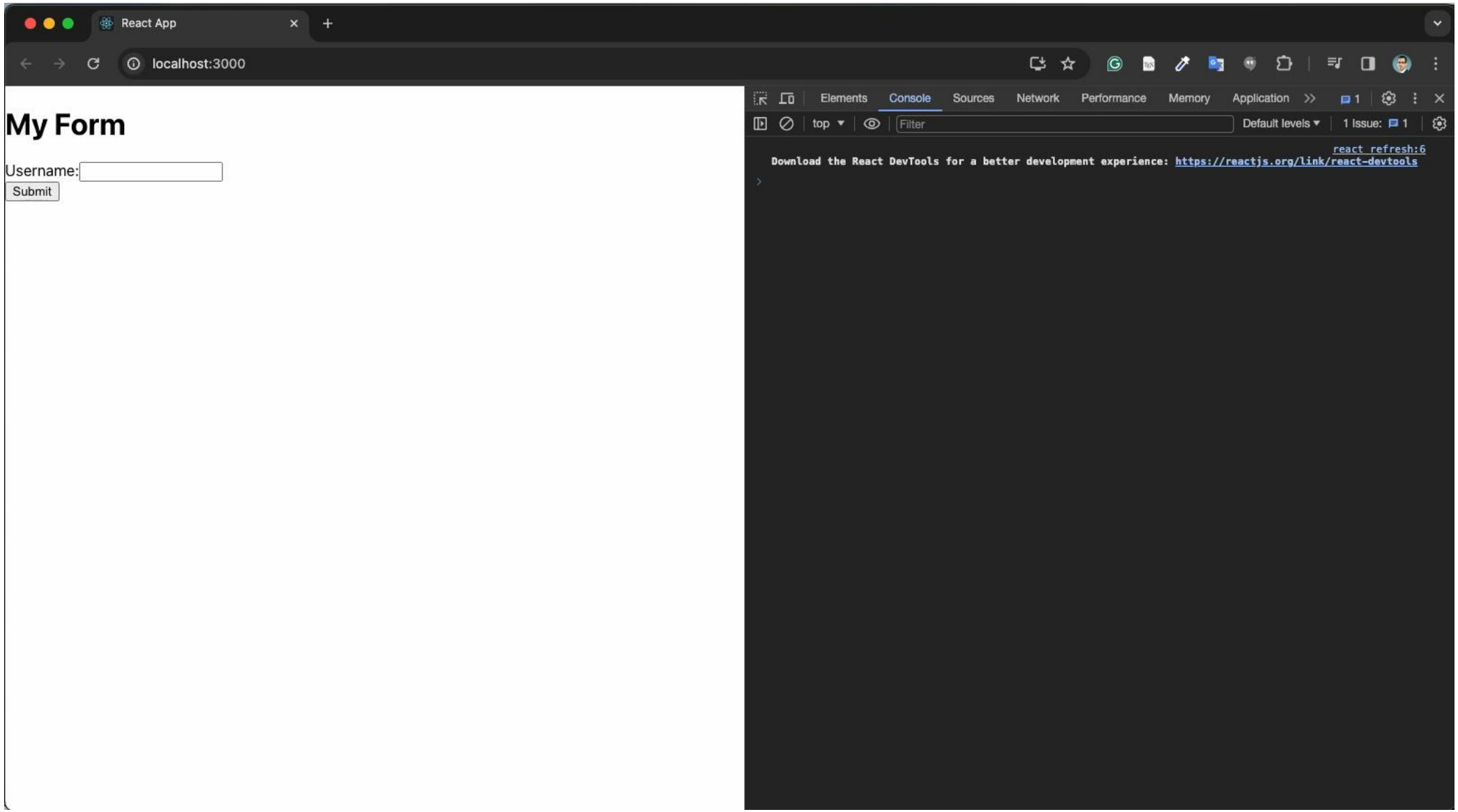
Retrieving text data sent from the backend to the frontend – example (frontend)

```
import React, { useState } from 'react';
function MyFormComponent() {
// State to hold the username
const [username, setUsername] = useState('');

// Function to handle form submission
async function handleSubmit(event) {
  event.preventDefault();

  // Assuming you have a backend endpoint for form submissions
  const backendEndpoint = 'http://127.0.0.1:5000';

  try {
    const response = await fetch(backendEndpoint, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify({ 'username':username }),
    });
    const data = await response.text();

    if (response.ok) {
      console.log(data);
      console.log('Form submitted successfully!');
    } else {
      console.error('Form submission failed.');
    }
  } catch (error) {
    console.error('Error during form submission:', error); }};
```

Retrieving text data sent from the backend to the frontend – example (frontend)

```
return (
  <div>
    <h1>My Form</h1>
    <form onSubmit={handleSubmit}>
      <label>
        Username:
        <input
          type="text"
          value={username}
          onChange={(e) => setUsername(e.target.value)}
        />
      </label>
      <br />
      <button type="submit">Submit</button>
    </form>
  </div>
);
};

export default MyFormComponent;
```

Retrieving text data sent from the backend to the frontend – example



Retrieving JSON data sent from the backend to the frontend – example (backend)

```
from flask import Flask,request
from flask_cors import CORS

app = Flask(__name__)
CORS(app)

# Assigning multiple routes to the same function
@app.route('/',methods=['POST'])
@app.route('/main',methods=['POST'])
def home():
  data = request.get_json()
  print(data['username'])
  msg = 'Welcome {} to the home page!'.format(data['username'])
  return {'msg':msg}

@app.route('/about')
def about():
  return 'Welcome to the About Us page!'

if __name__ == '__main__':
  app.run()
```

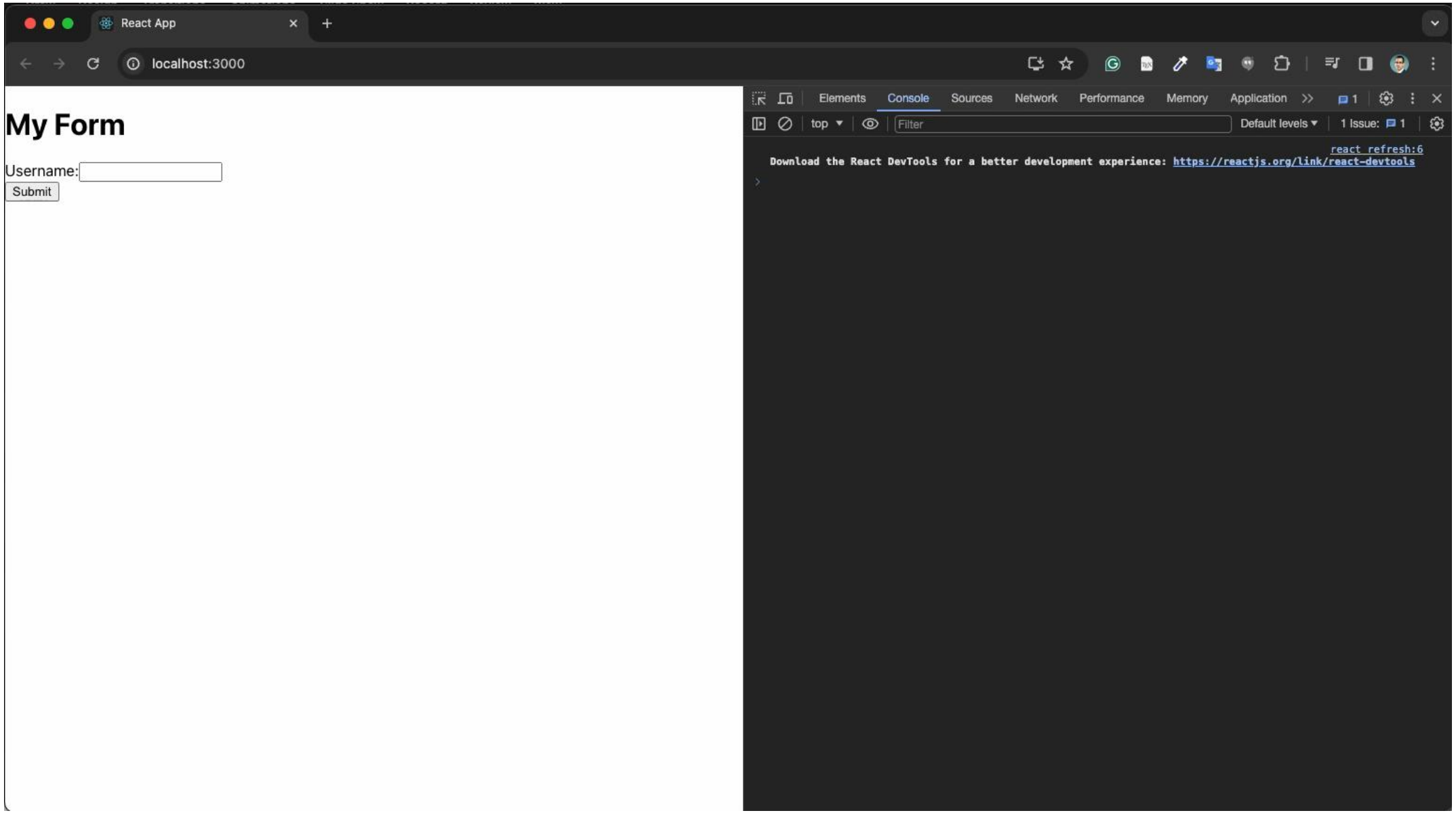
Retrieving JSON data sent from the backend to the frontend – example (frontend)

```
import React, { useState } from 'react';

const MyFormComponent = () => {
// State to hold the username
const [username, setUsername] = useState('');

// Function to handle form submission
const handleSubmit = async (event) => {
  event.preventDefault();

  // Assuming you have a backend endpoint for form submissions
  const backendEndpoint = 'http://127.0.0.1:5000';

  try {
    const response = await fetch(backendEndpoint, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json',
      },
      body: JSON.stringify({ 'username':username }),
    });
    const data = await response.json();

    if (response.ok) {
      console.log(data['msg']);
      console.log('Form submitted successfully!');
    } else {
      console.error('Form submission failed.');
    }
  } catch (error) {
    console.error('Error during form submission:', error);}};
```

Retrieving JSON data sent from the backend to the frontend – example (frontend)

```
return (
  <div>
    <h1>My Form</h1>
    <form onSubmit={handleSubmit}>
      <label>
        Username:
        <input
          type="text"
          value={username}
          onChange={(e) => setUsername(e.target.value)}
        />
      </label>
      <br />
      <button type="submit">Submit</button>
    </form>
  </div>
);
};

export default MyFormComponent;
```

Retrieving JSON data sent from the backend to the frontend – example



Questions