The background features three glowing, translucent 3D torus shapes against a dark blue gradient. One ring is positioned in the upper left, another in the lower left, and a larger one in the lower right.

# Đồ án ReactJS

# Tables of contents

I.

KIẾN THỨC  
CƠ BẢN

II.

KIẾN THỨC  
NÂNG CAO

III.

RUN CODE

# I. Kiến thức cơ bản



# I. Kiến thức cơ bản:

## 1/ Sử dụng jsx

- Jsx là gì ? JSX là một phần của React, một thư viện JavaScript phổ biến được sử dụng để xây dựng giao diện người dùng. JSX là viết tắt của "JavaScript XML" và là một phần quan trọng của cú pháp của React.
- JSX cho phép bạn viết mã HTML trong các tệp JavaScript mà không cần sử dụng các phương thức tạo đối tượng DOM hay chuỗi JavaScript. Nó giúp kết hợp mã JavaScript và các phần tử giao diện người dùng một cách thuận tiện hơn và đồng thời tạo ra mã nguồn dễ đọc và dễ hiểu hơn.

## - Ví dụ nhỏ jsx

```
website-v1 > src > components > User > DetailUser.js > DetailUser > state

return (
  <>
  <div className="backrou">

    <div className="title">Thành viên thứ {this.props.match.params.id}</div>
    <br></br>
    {checkempty === false && (
      <div className="main-user">
        <div className="main-user_name">
          {" "}
          Name {user.first_name} - {user.last_name}
        </div>
        <div className="main-user_email"> Email {user.email}</div>
        <div className="main-user_img">
          {" "}
          <img src={user.avatar} />
        </div>
      </div>
    )}
  </div>
);

}
```



## 2. Sử dụng State

- "state" là một đối tượng đặc biệt chứa thông tin có thể thay đổi trong quá trình thực thi ứng dụng. State được sử dụng để lưu trữ dữ liệu cục bộ và quản lý sự thay đổi của dữ liệu đó theo thời gian.
- Mỗi thành phần React có thể có state riêng của nó, và thay đổi state sẽ gây ra việc render lại giao diện người dùng để phản ánh trạng thái mới. Sử dụng state giúp ứng dụng React giữ lại trạng thái của nó và tương tác với người dùng một cách động.



## Ví dụ về state

State này dùng để sau khi check xem id nào của data được chọn thì state sẽ được setstate nó sẽ mang giá trị mới chứa id data mình vừa chọn để xem thông tin

```
import React from 'react';
import './DetailUser.scss';
Complexity is 11 You must be kidding
▼ class DetailUser extends React.Component { █
  |   state = { user: {} };
  |   Complexity is 6 It's time to do something...
  ▼   async componentDidMount() { █
    ▼     if (this.props.match.params && this.props.match) {
      |       let id = this.props.match.params.id;
      |       let res = await axios.get(`https://reqres.in/api/users/${id}`);
      |       this.setState({ user: res.data });
      |     }
    ▌   }
  ▌ }
  |   Complexity is 10 It's time to do something...
  |   render() {
  |     return (
  |       

|         

# User Detail


  |         

ID: {this.state.user.id}


  |         

Name: {this.state.user.name}


  |         

Email: {this.state.user.email}


  |


  |     );
  |   }
  ▌ }
```

### 3. Sử dụng Hook

Trong React, "hook" là một chức năng mở rộng của React, giúp bạn sử dụng state và các tính năng React khác trong các thành phần được viết dưới dạng hàm (functional components) thay vì các thành phần dựa trên lớp (class components). Hook được giới thiệu trong phiên bản React 16.8 và đã trở thành một phần quan trọng trong việc quản lý state và hiệu suất trong các ứng dụng React.

# Ứng dụng HOOK để check input đã nhập đủ ký tự

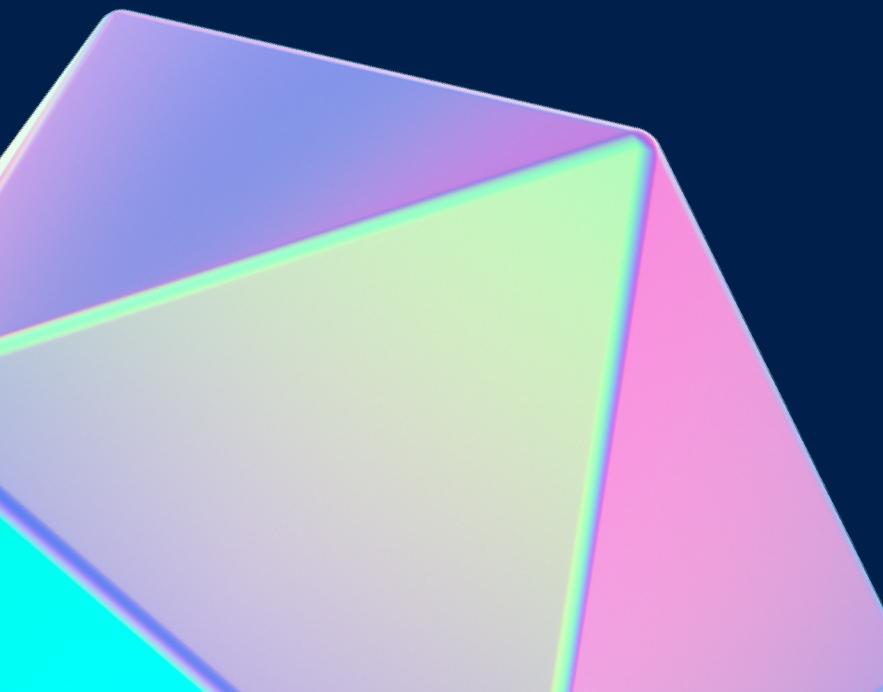
```
// đăng nhập
const [inputValue, setInputValue] = useState("");
const [isErrorVisible, setIsErrorVisible] = useState(false);
const [isInputClicked, setIsInputClicked] = useState(false);

Complexity is 7 It's time to do something...
useEffect(() => {
    // Kiểm tra nếu người dùng đã click vào input và độ dài của giá trị là 10, bạn có thể
    // thực hiện các hành động tương ứng ở đây
    if (isInputClicked && inputValue.length === 10 && isErrorVisible) {
        setIsErrorVisible(false); // Ẩn thông báo khi đã nhập đủ
    } else if (isInputClicked && inputValue.length < 10 && !isErrorVisible) {
        setIsErrorVisible(true); // Hiển thị thông báo khi chưa nhập đủ và isErrorVisible là true
    }
}, [inputValue, isErrorVisible, isInputClicked]); // useEffect sẽ chạy lại khi giá trị
// của inputValue, isErrorVisible, hoặc isInputClicked thay đổi

const handleInputChange = (event) => {
    setInputValue(event.target.value);
};

const handleInputClick = () => {
    setIsInputClicked(true); // Đánh dấu rằng người dùng đã click vào input
    setIsErrorVisible(true); // Hiển thị thông báo khi người dùng bắt đầu nhập liệu
};
```

## II. Kiến thức nâng cao



# Sử dụng API

**Q1**

Sử dụng fetch API là một cách tiêu biểu để gọi các yêu cầu API trong ứng dụng React. fetch trả về một Promise, giúp bạn xử lý dễ dàng các yêu cầu và phản hồi.

**Q2**

Axios là một thư viện JavaScript phổ biến được sử dụng để gọi các yêu cầu HTTP. Nó cung cấp các tính năng như tự động chuyển đổi JSON, hỗ trợ Promise, và xử lý lỗi dễ dàng

**Q3**

XMLHttpRequest là một cách cổ điển để thực hiện các yêu cầu HTTP. Tuy nhiên, trong các ứng dụng React hiện đại, việc sử dụng fetch hoặc thư viện như Axios thường được ưa chuộng hơn.

# API trong dự án:

```
function App() {
  const {register, handleSubmit, formState: {errors}} = useForm({
    resolver: yupResolver(schema)
  });

  const onSubmit = async (data) => {
    axios.post("https://dummyjson.com/auth/login", data)
      .then((response) => {
        alert(response.data.token);
        console.log(data)
      })
      .catch((error) => {
        console.log(error);
      });
  }
}
```