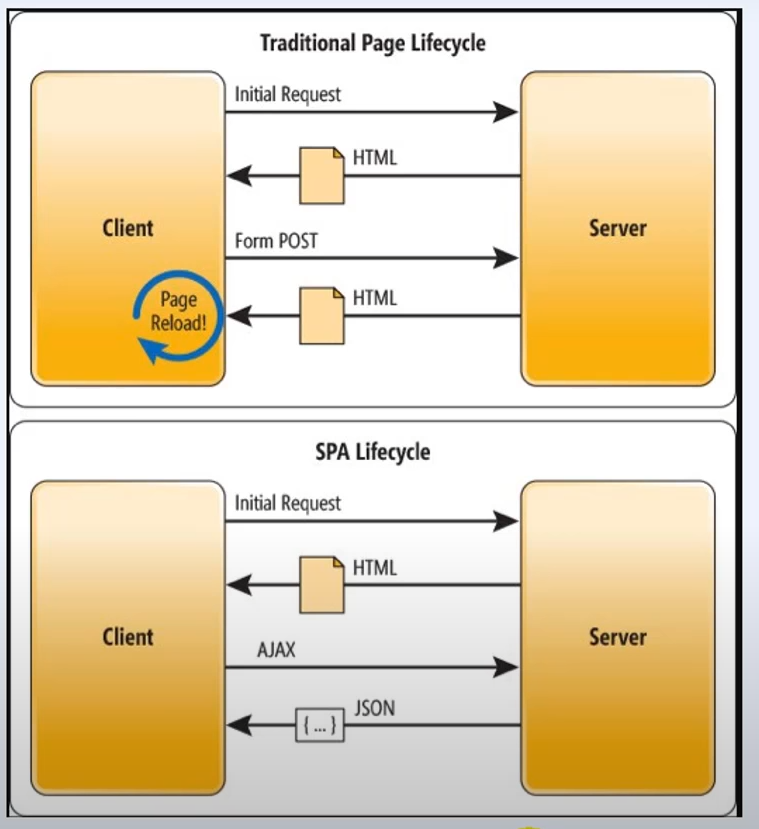
Lesson 05

SPA: single page application  


Lesson 07

Component

Component is function/class, that return web element (markup JSX).

Functional examples:

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

**let** App = **function**() {  
 **return** (  
 <**div** className="App">  
 Start  
 </**div**>  
 );  
}

**let** App = () => {  
 **return** (  
 <**div** className="App">  
 Start  
 </**div**>  
 );  
}

Classes example:

**class** App **extends** React.Component {  
 render() {  
 **return** (  
 <**div** className="App">  
 Start  
 </**div**>  
 );  
 }  
}

Lesson 14

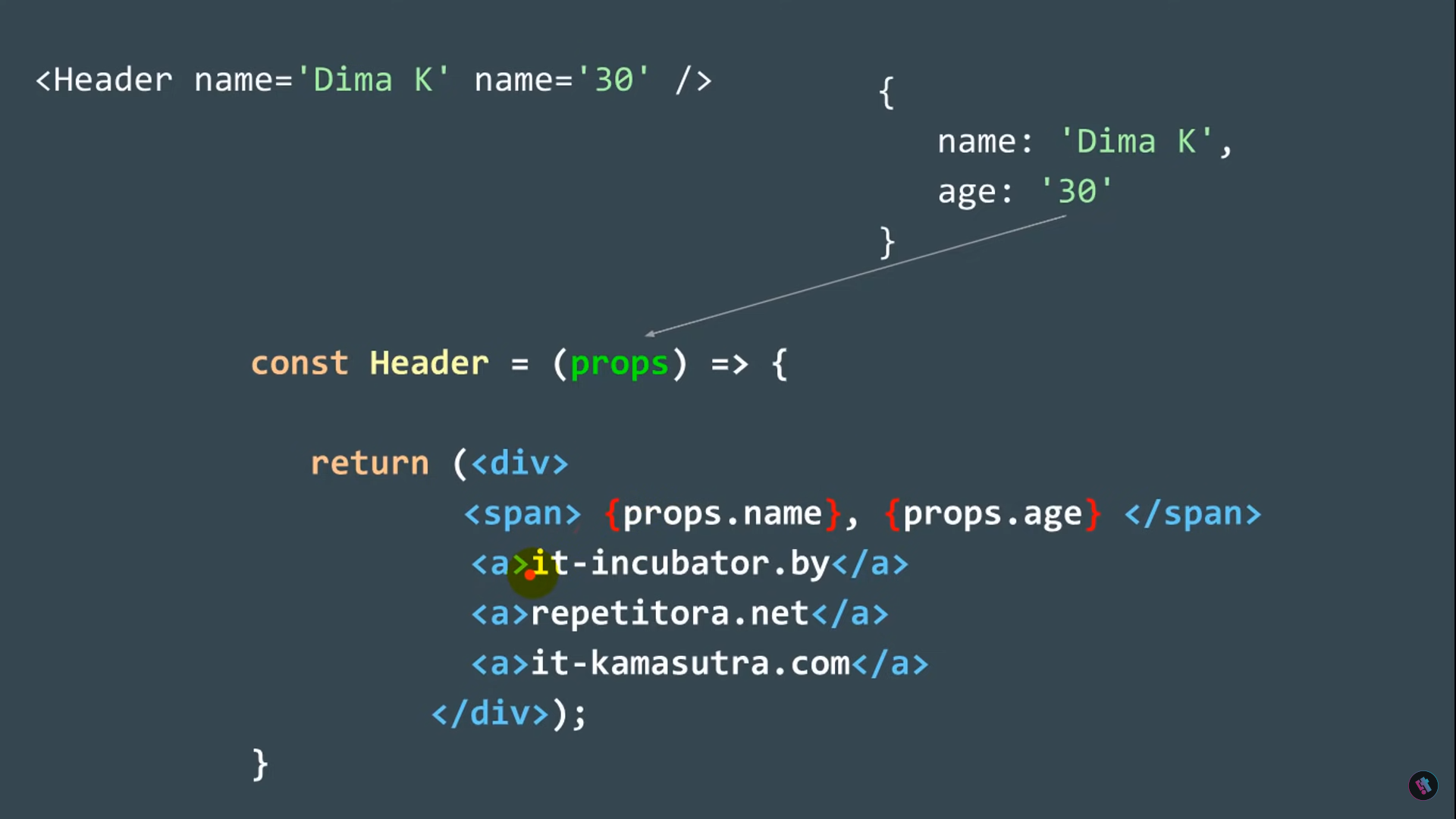
\*.module.css 🡪 import s from “./\*.module.css”

Inside s there will be an object with classes from \*.module.css

**import** s **from** "./Profile.module.css";  
  
**export let** Profile = () => (  
 <**div** className={s.content}>  
 <**img** src={header} alt="Header" className={s.header} />  
 <**PageBlock** />  
 <**Posts** />  
 </**div**>  
);

Lesson 14

Props



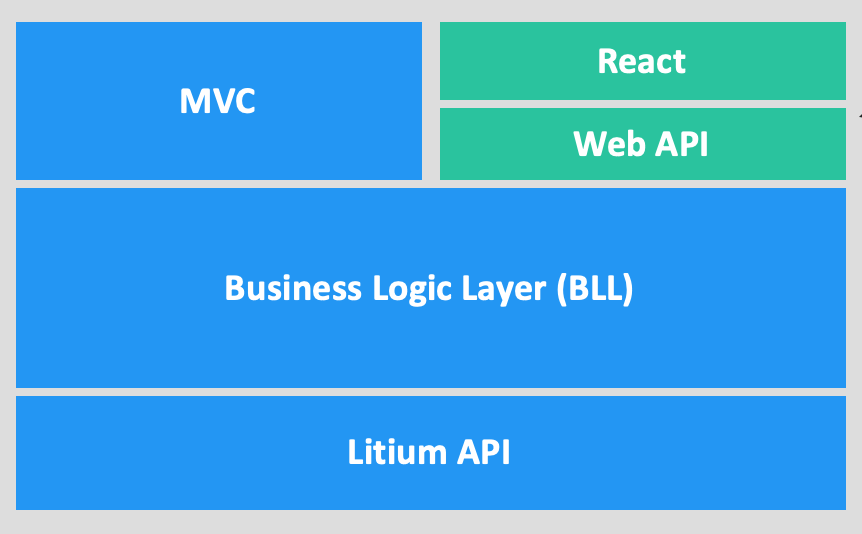
props.children 🡪 text or others components is embedded here.

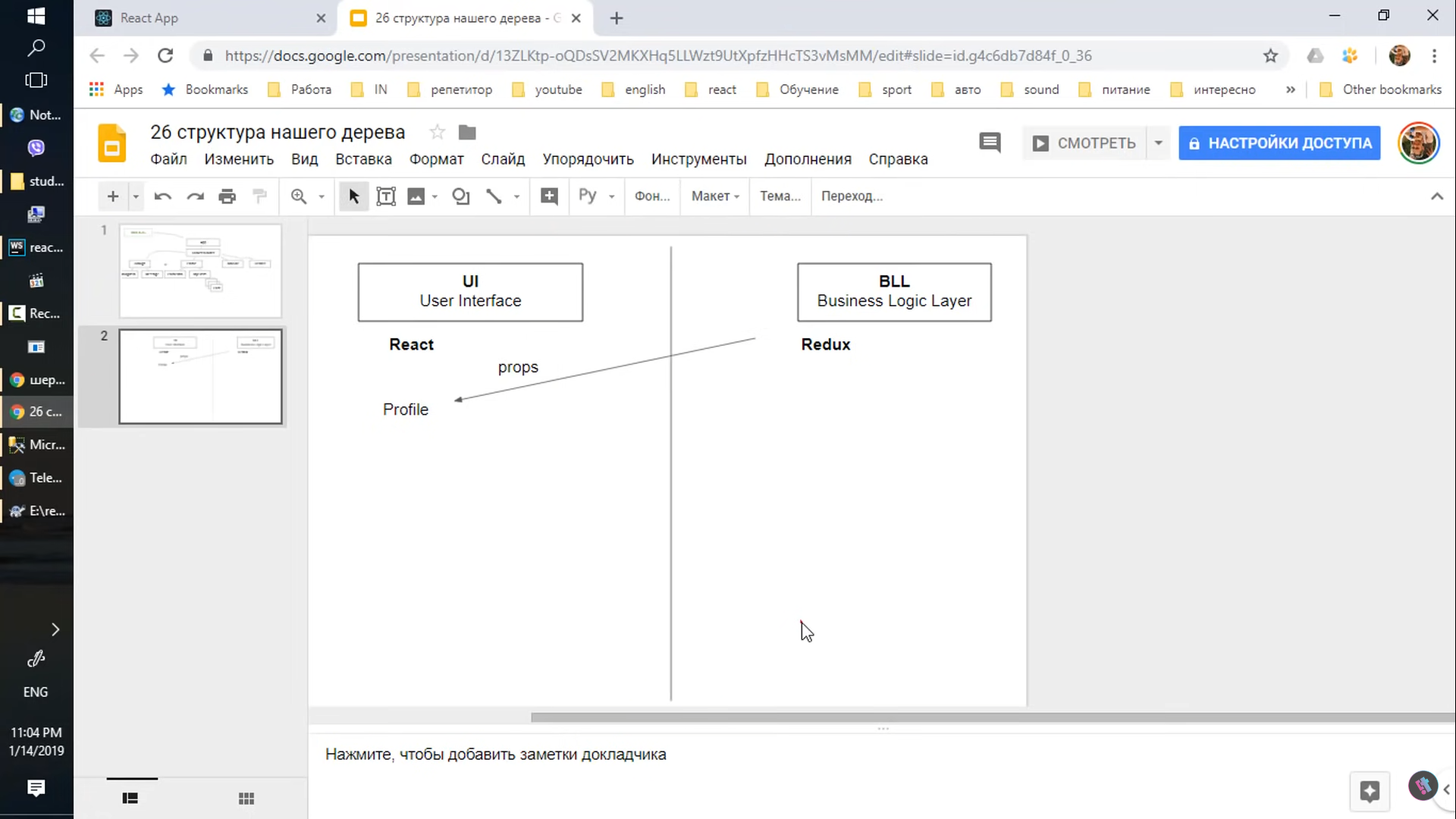
Lesson 24/26

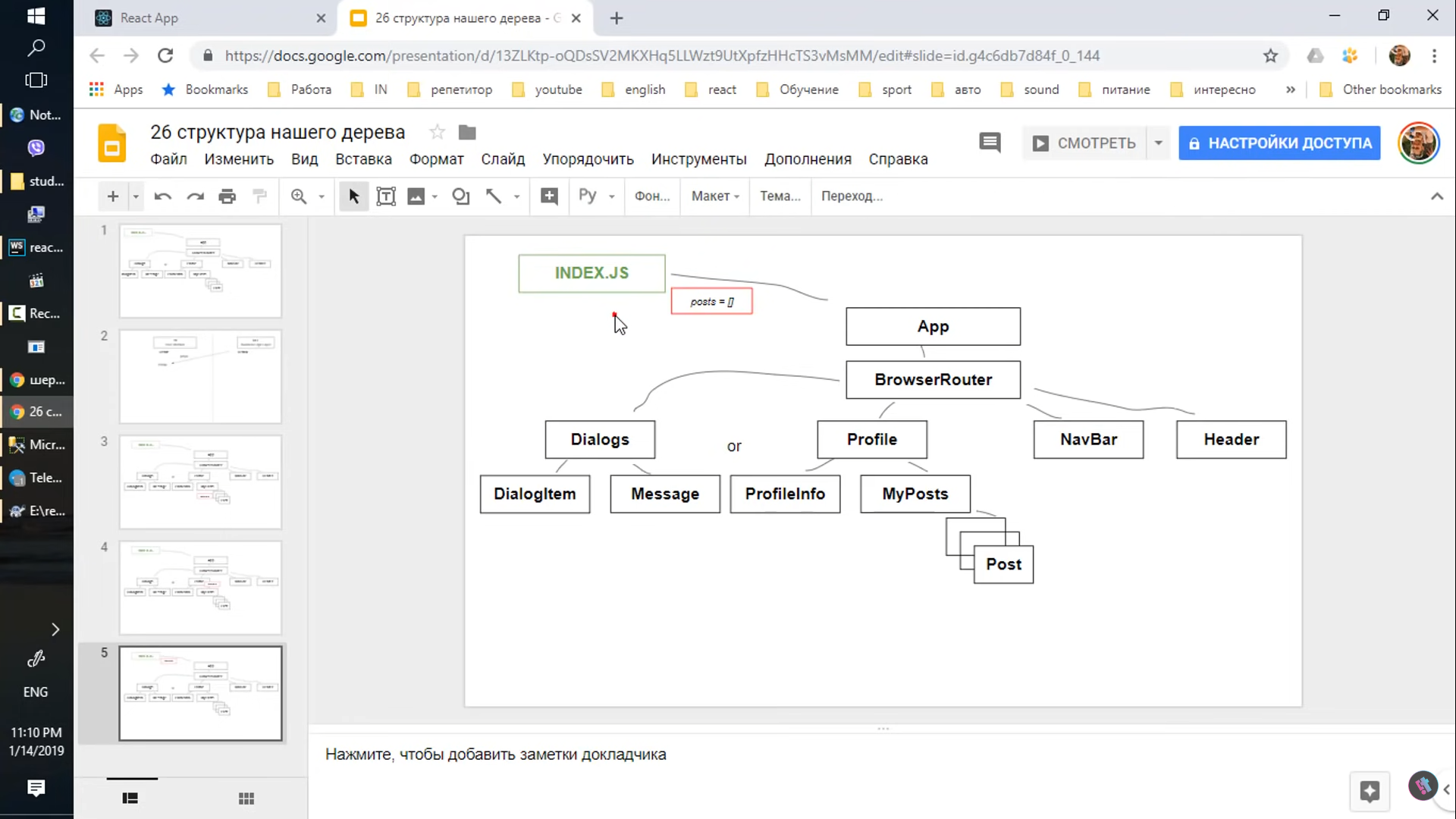
UI, BLL

UI – User Interface

BLL – Business Logic Layer

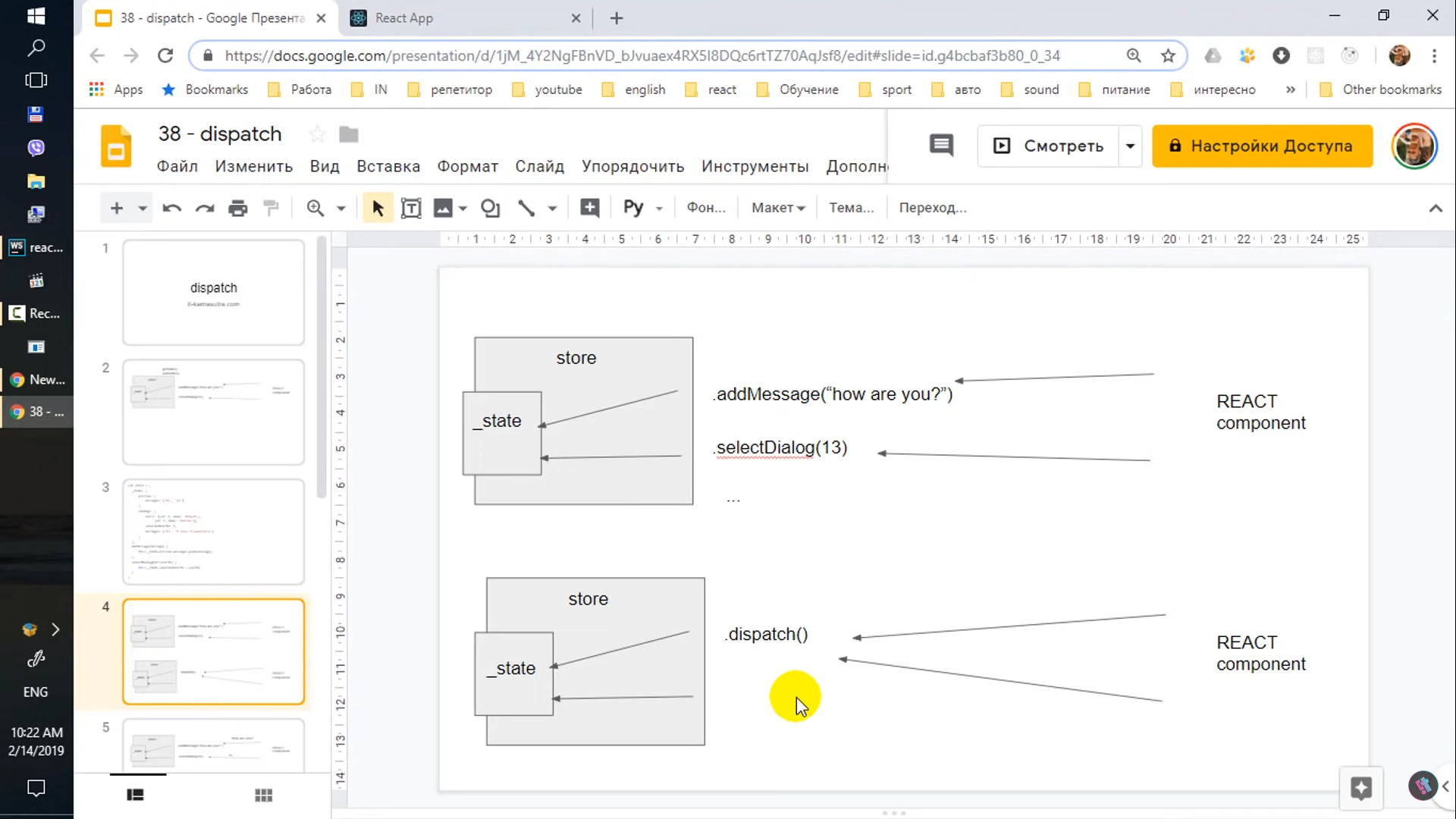






Lesson 38

Dispatch, action

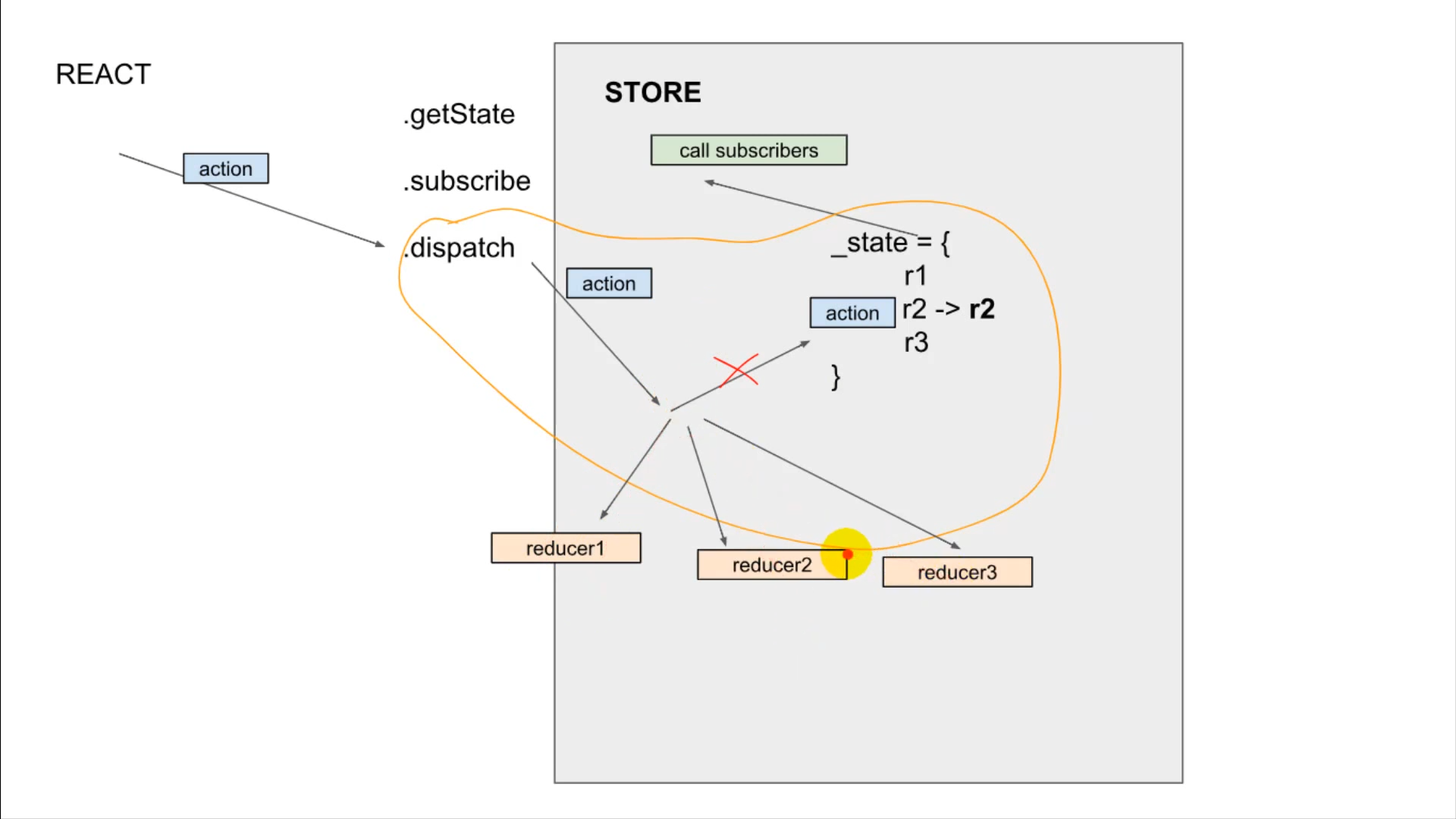


dispatch(action) {  
 **switch**( action.type ) {  
 **case** "ADD-POST":  
 **this**.\_addPost(action.post);  
 **break**;  
 **case** "SET-VALUE-INPUT":  
 **this**.\_setValueInput(action.text);  
 **break**;  
 }  
}

Lesson 41

Reducers

Reducer is just a clear function; it returns modified state or state without changes. It is necessary for partition dispatch.



Lesson 44

Context API.

Context has 2 children: Context.Provider and Context.Consumer.

To create context, you need to create Context-Object.

Example: const StoreContext = React.createContext(null);

After the creating, we can get Provider and Consumer, like that: StoreContext.Provider and StoreContext.Consumer.

Provider and Consumer are React components:

Provider is a wrapper over children, often over the whole application.

Example:

<**BrowserRouter**>  
 <**StoreContext**.**Provider** value={store}>  
 <**App**/>  
 </**StoreContext.Provider**>  
</**BrowserRouter**>

Provider has required field – value.

To get value from children, you need to use Consumer, like that:

<**StoreContext.Consumer**>  
 {store => {  
 **const** state = store.getState().profile;  
 **return** <**Profile** {...props} headerImage={state.headerImage} avatar={state.avatar} info={state.info}/>  
 }}  
</**StoreContext.Consumer**>

Consumer must contain js-insert, which return jsx.