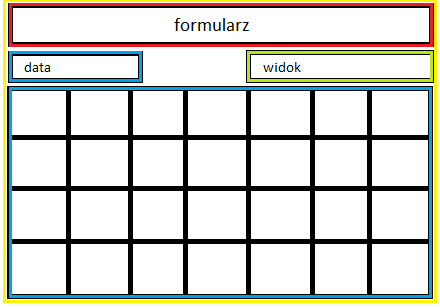
**Krok 1: Podziel interfejs użytkownika na zhierarchizowany układ komponentów**



Kolor czerwony - wyświetla formularz z dodawaniem wydarzeń.

Kolor zielony - wyświetla zmianę widoku.

Kolor niebieski - wyświetla kalendarz.

Kolor żółty - wyświetla wszystkie komponenty.

**Krok 2: Zbuduj wersję statyczną w Reakcie**

let eventsT = [

{

start: '2022-03-09', end: '2022-03-25', title: 'dsdsdsd'

}

];

let additionalData = [

{

}

];

class Calendar extends React.Component

{

render() {

return (

<div>

<label>

Poczatek:

<input type="date"/>

</label>

<label>

Koniec:

<input type="date"/>

</label>

<label>

Miejsce:

<input type="text"/>

</label>

<label>

Godzina:

<input type="time"/>

</label>

<label>

Tytuł:

<input type="text"/>

</label>

<label>

Opis:

<input type="text"/>

</label>

<button>Wyslij</button>

<FullCalendar

editable={true}

selectable={true}

selectMirror={true}

dayMaxEvents={true}

plugins={[ dayGridPlugin, timeGridPlugin ]}

initialView="dayGridMonth"

events={this.state.event}

headerToolbar=

{{

left: "prev,next",

center: "title",

right: "today,dayGridMonth,timeGridWeek,timeGridDay",

}}

/>

</div>

);

}

}

ReactDOM.render(

<Calendar />,

document.getElementById('root')

);

reportWebVitals();

**Krok 3: Określ minimalne (ale kompletne) odwzorowanie stanu interfejsu użytkownika**

Stan przedstawia się następująco:

- Każdy input w formularzu

- Wydarzenia w kalendarzu

**Krok 4: Określ, gdzie powinien mieścić się stan**

let eventsT = [

{

}

];

let additionalData = [

{

}

];

class Calendar extends React.Component

{

constructor(props)

{

super(props);

this.state = {startInput: '', endInput: '', placeInput: '', timeInput: '', titleInput: '', descriptionInput: '', update: '', event: ''};

}

render() {

return (

<div>

<label>

Poczatek:

<input type="date" value={this.state.value} onChange={this.handleChangeStart} />

</label>

<label>

Koniec:

<input type="date" value={this.state.value} onChange={this.handleChangeEnd} />

</label>

<label>

Miejsce:

<input type="text" value={this.state.value} onChange={this.handleChangePlace} />

</label>

<label>

Godzina:

<input type="time" value={this.state.value} onChange={this.handleChangeTime} />

</label>

<label>

Tytuł:

<input type="text" value={this.state.value} onChange={this.handleChangeTitle} />

</label>

<label>

Opis:

<input type="text" value={this.state.value} onChange={this.handleChangeDescription} />

</label>

<button onClick={this.handleSubmit}>Wyslij</button>

<FullCalendar

editable={true}

selectable={true}

selectMirror={true}

dayMaxEvents={true}

plugins={[ dayGridPlugin, timeGridPlugin ]}

initialView="dayGridMonth"

events={this.state.event}

headerToolbar=

{{

left: "prev,next",

center: "title",

right: "today,dayGridMonth,timeGridWeek,timeGridDay",

}}

/>

</div>

);

}

}

ReactDOM.render(

<Calendar />,

document.getElementById('root')

);

**Krok 5: Dodaj przepływ danych w drugą stronę**

let eventsT = [

{

}

];

let additionalData = [

{

}

];

class Calendar extends React.Component

{

constructor(props)

{

super(props);

this.state = {startInput: '', endInput: '', placeInput: '', timeInput: '', titleInput: '', descriptionInput: '', update: '', event: ''};

this.handleChangeStart = this.handleChangeStart.bind(this);

this.handleChangeEnd = this.handleChangeEnd.bind(this);

this.handleChangePlace = this.handleChangePlace.bind(this);

this.handleChangeTime = this.handleChangeTime.bind(this);

this.handleChangeTitle = this.handleChangeTitle.bind(this);

this.handleChangeDescription = this.handleChangeDescription.bind(this);

this.handleSubmit = this.handleSubmit.bind(this);

}

handleChangeStart(event) {

this.setState({startInput: event.target.value});

}

handleChangeEnd(event) {

this.setState({endInput: event.target.value});

}

handleChangePlace(event) {

this.setState({placeInput: event.target.value});

}

handleChangeTime(event) {

this.setState({timeInput: event.target.value});

}

handleChangeTitle(event) {

this.setState({titleInput: event.target.value});

}

handleChangeDescription(event) {

this.setState({textInput: event.target.value});

}

handleSubmit(event) {

eventsT.push({title: this.state.titleInput, start: this.state.startInput, end: this.state.endInput});

additionalData.push({place: this.state.placeInput, time: this.state.timeInput, description: this.state.descriptionInput});

this.setState({event: eventsT});

}

render() {

return (

<div>

<label>

Poczatek:

<input type="date" value={this.state.value} onChange={this.handleChangeStart} />

</label>

<label>

Koniec:

<input type="date" value={this.state.value} onChange={this.handleChangeEnd} />

</label>

<label>

Miejsce:

<input type="text" value={this.state.value} onChange={this.handleChangePlace} />

</label>

<label>

Godzina:

<input type="time" value={this.state.value} onChange={this.handleChangeTime} />

</label>

<label>

Tytuł:

<input type="text" value={this.state.value} onChange={this.handleChangeTitle} />

</label>

<label>

Opis:

<input type="text" value={this.state.value} onChange={this.handleChangeDescription} />

</label>

<button onClick={this.handleSubmit}>Wyslij</button>

<FullCalendar

editable={true}

selectable={true}

selectMirror={true}

dayMaxEvents={true}

plugins={[ dayGridPlugin, timeGridPlugin ]}

initialView="dayGridMonth"

events={this.state.event}

headerToolbar=

{{

left: "prev,next",

center: "title",

right: "today,dayGridMonth,timeGridWeek,timeGridDay",

}}

/>

</div>

);

}

}

ReactDOM.render(

<Calendar />,

document.getElementById('root')

);