class **TimersDashboard** extends **React**.**Component** {

state = {

timers: [],

};

**componentDidMount**() {

this.**loadTimersFromServer**();

**setInterval**(this.loadTimersFromServer, 5000);

}

**loadTimersFromServer** = () => {

client.**getTimers**((serverTimers) => (

this.**setState**({ timers: serverTimers })

)

);

};

*// ...*

**handleCreateFormSubmit** = (timer) => {

this.**createTimer**(timer);

};

**handleEditFormSubmit** = (attrs) => {

this.**updateTimer**(attrs);

};

**handleTrashClick** = (timerId) => {

this.**deleteTimer**(timerId);

};

**handleStartClick** = (timerId) => {

this.**startTimer**(timerId);

};

**handleStopClick** = (timerId) => {

this.**stopTimer**(timerId);

};

**createTimer** = (timer) => {

const t = helpers.**newTimer**(timer);

this.**setState**({

timers: this.state.timers.**concat**(t),

});

};

**updateTimer** = (attrs) => {

this.**setState**({

timers: this.state.timers.**map**((timer) => {

if (timer.id === attrs.id) {

return **Object**.**assign**({}, timer, {

title: attrs.title,

project: attrs.project,

});

} else {

return timer;

}

}),

});

};

**deleteTimer** = (timerId) => {

this.**setState**({

timers: this.state.timers.**filter**(t => t.id !== timerId),

});

};

**startTimer** = (timerId) => {

const now = **Date**.**now**();

this.**setState**({

timers: this.state.timers.**map**((timer) => {

if (timer.id === timerId) {

return **Object**.**assign**({}, timer, {

runningSince: now,

});

} else {

return timer;

}

}),

});

};

**stopTimer** = (timerId) => {

const now = **Date**.**now**();

this.**setState**({

timers: this.state.timers.**map**((timer) => {

if (timer.id === timerId) {

const lastElapsed = now - timer.runningSince;

return **Object**.**assign**({}, timer, {

elapsed: timer.elapsed + lastElapsed,

runningSince: null,

});

} else {

return timer;

}

}),

});

};

**render**() {

return (

<div *className*='ui three column centered grid'>

<div *className*='column'>

<**EditableTimerList**

*timers*={this.state.timers}

*onFormSubmit*={this.handleEditFormSubmit}

*onTrashClick*={this.handleTrashClick}

*onStartClick*={this.handleStartClick}

*onStopClick*={this.handleStopClick}

/>

<**ToggleableTimerForm**

*onFormSubmit*={this.handleCreateFormSubmit}

/>

</div>

</div>

);

}

}

class **ToggleableTimerForm** extends **React**.**Component** {

state = {

isOpen: false,

};

**handleFormOpen** = () => {

this.**setState**({ isOpen: true });

};

**handleFormClose** = () => {

this.**setState**({ isOpen: false });

};

**handleFormSubmit** = (timer) => {

this.props.**onFormSubmit**(timer);

this.**setState**({ isOpen: false });

};

**render**() {

if (this.state.isOpen) {

return (

<**TimerForm**

*onFormSubmit*={this.handleFormSubmit}

*onFormClose*={this.handleFormClose}

/>

);

} else {

return (

<div *className*='ui basic content center aligned segment'>

<button

*className*='ui basic button icon'

*onClick*={this.handleFormOpen}

>

<i *className*='plus icon' />

</button>

</div>

);

}

}

}

class **EditableTimerList** extends **React**.**Component** {

**render**() {

const timers = this.props.timers.**map**((timer) => (

<**EditableTimer**

*key*={timer.id}

*id*={timer.id}

*title*={timer.title}

*project*={timer.project}

*elapsed*={timer.elapsed}

*runningSince*={timer.runningSince}

*onFormSubmit*={this.props.onFormSubmit}

*onTrashClick*={this.props.onTrashClick}

*onStartClick*={this.props.onStartClick}

*onStopClick*={this.props.onStopClick}

/>

));

return (

<div *id*='timers'>

{timers}

</div>

);

}

}

class **EditableTimer** extends **React**.**Component** {

state = {

editFormOpen: false,

};

**handleEditClick** = () => {

this.**openForm**();

};

**handleFormClose** = () => {

this.**closeForm**();

};

**handleSubmit** = (timer) => {

this.props.**onFormSubmit**(timer);

this.**closeForm**();

};

**closeForm** = () => {

this.**setState**({ editFormOpen: false });

};

**openForm** = () => {

this.**setState**({ editFormOpen: true });

};

**render**() {

if (this.state.editFormOpen) {

return (

<**TimerForm**

*id*={this.props.id}

*title*={this.props.title}

*project*={this.props.project}

*onFormSubmit*={this.handleSubmit}

*onFormClose*={this.handleFormClose}

/>

);

} else {

return (

<**Timer**

*id*={this.props.id}

*title*={this.props.title}

*project*={this.props.project}

*elapsed*={this.props.elapsed}

*runningSince*={this.props.runningSince}

*onEditClick*={this.handleEditClick}

*onTrashClick*={this.props.onTrashClick}

*onStartClick*={this.props.onStartClick}

*onStopClick*={this.props.onStopClick}

/>

);

}

}

}

class **Timer** extends **React**.**Component** {

**componentDidMount**() {

this.forceUpdateInterval = **setInterval**(() => this.**forceUpdate**(), 50);

}

**componentWillUnmount**() {

**clearInterval**(this.forceUpdateInterval);

}

**handleStartClick** = () => {

this.props.**onStartClick**(this.props.id);

};

**handleStopClick** = () => {

this.props.**onStopClick**(this.props.id);

};

**handleTrashClick** = () => {

this.props.**onTrashClick**(this.props.id);

};

**render**() {

const elapsedString = helpers.**renderElapsedString**(

this.props.elapsed, this.props.runningSince

);

return (

<div *className*='ui centered card'>

<div *className*='content'>

<div *className*='header'>

{this.props.title}

</div>

<div *className*='meta'>

{this.props.project}

</div>

<div *className*='center aligned description'>

<h2>

{elapsedString}

</h2>

</div>

<div *className*='extra content'>

<span

*className*='right floated edit icon'

*onClick*={this.props.onEditClick}

>

<i *className*='edit icon' />

</span>

<span

*className*='right floated trash icon'

*onClick*={this.handleTrashClick}

>

<i *className*='trash icon' />

</span>

</div>

</div>

<**TimerActionButton**

*timerIsRunning*={!!this.props.runningSince}

*onStartClick*={this.handleStartClick}

*onStopClick*={this.handleStopClick}

/>

</div>

);

}

}

class **TimerActionButton** extends **React**.**Component** {

**render**() {

if (this.props.timerIsRunning) {

return (

<div

*className*='ui bottom attached red basic button'

*onClick*={this.props.onStopClick}

>

Stop

</div>

);

} else {

return (

<div

*className*='ui bottom attached green basic button'

*onClick*={this.props.onStartClick}

>

Start

</div>

);

}

}

}

class **TimerForm** extends **React**.**Component** {

state = {

title: this.props.title || '',

project: this.props.project || '',

};

**handleTitleChange** = (e) => {

this.**setState**({ title: e.target.value });

};

**handleProjectChange** = (e) => {

this.**setState**({ project: e.target.value });

};

**handleSubmit** = () => {

this.props.**onFormSubmit**({

id: this.props.id,

title: this.state.title,

project: this.state.project,

});

};

**render**() {

const submitText = this.props.id ? 'Update' : 'Create';

return (

<div *className*='ui centered card'>

<div *className*='content'>

<div *className*='ui form'>

<div *className*='field'>

<label>Title</label>

<input

*type*='text'

*value*={this.state.title}

*onChange*={this.handleTitleChange}

/>

</div>

<div *className*='field'>

<label>Project</label>

<input

*type*='text'

*value*={this.state.project}

*onChange*={this.handleProjectChange}

/>

</div>

<div *className*='ui two bottom attached buttons'>

<button

*className*='ui basic blue button'

*onClick*={this.handleSubmit}

>

{submitText}

</button>

<button

*className*='ui basic red button'

*onClick*={this.props.onFormClose}

>

Cancel

</button>

</div>

</div>

</div>

</div>

);

}

}

ReactDOM.**render**(

<**TimersDashboard** />,

document.**getElementById**('content')

);