Adding a Context Menu to the Pilots List

The majority of ***Project Mini-Mek***'s functionality thus far has involved the "Pilots" tab, so we'll work with that. We've already added the ability to select a current pilot from the list, and delete a pilot, so let's add a context menu that offers that capability as well.

> **Commit a6eb7ed: Implement a context menu for pilots list items**

features/pilots/PilotsList/PilotsListItemMenu.jsx

```
import React, { Component } from 'react'
import {connect} from "react-redux";
import { Menu } from 'semantic-ui-react'

import {selectPilot} from "../pilotsActions";
import {deleteEntity} from "features/entities/entityActions";
import {hideContextMenu} from "features/contextMenus/contextMenuActions";

const actions = {
    selectPilot,
    deleteEntity,
    hideContextMenu,
}

export class PilotsListItemMenu extends Component {
    onSelectClicked = () => {
        this.props.selectPilot(this.props.pilotId);
        this.props.hideContextMenu();
    }

    onDeleteClicked = () => {
        this.props.deleteEntity("Pilot", this.props.pilotId);
        this.props.hideContextMenu();
    }
```

```
    render() {
        return (
            <Menu vertical>
                <Menu.Item>
                    <Menu.Header>Pilot: {this.props.text} </Menu.Header>
                    <Menu.Menu>
                        <Menu.Item onClick={this.onSelectClicked}>Select P
ilot</Menu.Item>
                        <Menu.Item onClick={this.onDeleteClicked}>Delete P
ilot</Menu.Item>
                    </Menu.Menu>
                </Menu.Item>
            </Menu>
        )
    }
}

export default connect(null, actions)(PilotsListItemMenu);
```

Our pilots menu will take two props: the ID of a pilot, and some text that will be the name of the pilot that was clicked on. We show the pilot name in the header, and use `props.pilotId` to tell the action creators what pilot to select or delete.

features/pilots/PilotsList/PilotsListRow.jsx

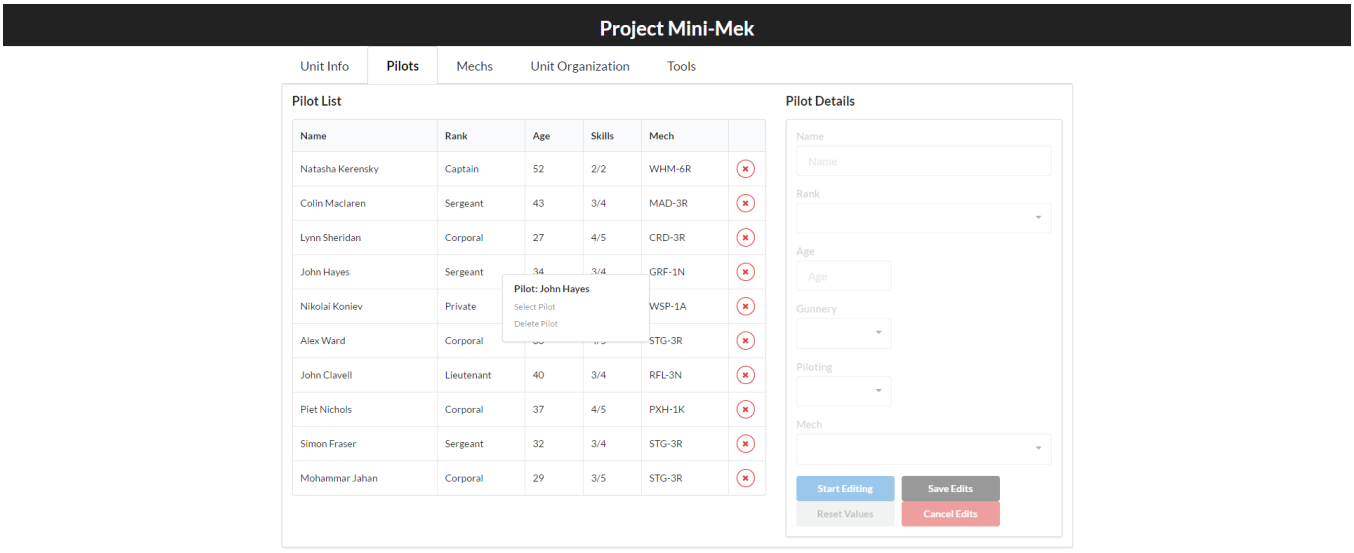
```
import {getEntitiesSession} from "features/entities/entitySelectors";
+import {deleteEntity} from "features/entities/entityActions";
import {showContextMenu} from "features/contextMenus/contextMenuActions";

const actions = {
    deleteEntity,
+   showContextMenu,
};


-const PilotsListRow = ({pilot={}, onPilotClicked=_.noop, selected, delete
Entity}) => {
+const PilotsListRow = ({pilot={}, onPilotClicked=_.noop, selected, delete
Entity, showContextMenu}) => {
    const {
        id = null,
        name = "",
```

```
// skip ahead

+   const onRowRightClicked = (e) => {
+       e.preventDefault();
+       e.stopPropagation();
+
+       const {pageX, pageY} = e;
+       showContextMenu(pageX, pageY, "PilotsListItemMenu", {text: pilot.n
ame, pilotId : id});
+   }

    return (
-       <Table.Row onClick={onRowClicked} active={selected}>
+       <Table.Row onClick={onRowClicked} onContextMenu={onRowRightClicke
d}  active={selected}>
            <Table.Cell>
                {name}
            </Table.Cell>
```

Over in the `PilotsListRow` component, we'll add a right-click handler on the entire row, and use the click coordinates in the page as the x/y location values for showing the menu. We also pass along the pilot name and ID values to the menu.

It's worth noting that the `PilotsListRow` component is a functional component that now has a lot going on inside of it. Frankly, it probably ought to be converted to a class component at this point, but I haven't yet gotten around to doing that.

With that menu implemented, let's try it out:



And since we're calling the same "Select" and "Delete" action creators as the buttons, it all Just Works :)

Let's take one final look at the application in action:

```
.App-header {
  background-color: #222;
  height: 70px;
  padding: 20px;
  color: white;
}
```