Evernote clone (freeCodeCamp)

step 1 → set up firebase.

step 2 → install dependencies → npm install @material-ui/core
→ npm install @material-ui/icons
→ npm install react-quill
→ npm install firebase

step 3 → In index.js → require('firebase') & firestore
→ also paste the config. copied from firebase

step 4 → create App comp. as class component.

step 5 → create a constructor in App.js, inside it declare 3 states → selectedNoteIndex : null,
→ selectedNote : null,
→ notes : null

step 6 → Create a componentDidMount inside class, and consde.log whatever there is inside firestore collection

step 7 → Create 3 folders editor, sidebar & sidebaritem for all three create a styles.js & folderName.js files

step 8 → write boiler plate code for all the 3 comp.

eg → This is editor.

Step 9 → call all the three comps. in App.js

30 mins

Step 10 → link reactquill in index.html.

Step 11 → In editor.js, destructure classes variable.

```
const { classes } = this.props;
```

Include <ReactQuill> component in editor.js

Step 12 → Create three states in editor.js →

```
this.state = {
  text : '',
  title : '',
  id : ''
};
```

Step 13 → Add attributes to <ReactQuill> value & onChange.

```
value = { this.state.text }
onChange = { this.updateBody } //define func" updateBody.

updateBody = async (val) => {
await this.setState ({ text : val });
this.update ();  //defing update func".
};

update = debounce (() => {
.
}, 1500);
```

once you stop typing for 1.5 sec it will do its action.

step 1 → In sidebar component,
create 2 states addingNote : false,
title : null

Restructure props →
const { notes, classes, selectedNoteIndex } = this.props;

step 2 →
Add a Button, add attribute onclick & className.
onclick = {this.newNoteBtnClick} //define the func'.

step 3 → Add a block below button
{
this.state.addingNote ?
add an input with attr. type, className = {classes.newNoteInput},
placeholder, onKeyUp
:
}

step 4 → change button Name with this code →
{this.state.addingNote ? 'Cancel' : 'New Note'}

step 5 → Below input, add a Button with attribute.
ClassName = {classes.newNoteSubmission}
onClick = {this.newNote}
Name of the button is Submit Note.

step 6 → we have to show all the notes in a List in sidebar.
After submit Note Button, come out of the block,
and add a <List> component.
and inside List comp. add a block. { }

Step 20 → Inside the block, add

```
<List>
{ div key={_index} >
<sidebarItemComponent
  _note={_note}
  _index={_index}
  selectedNoteIndex={selectedNoteIndex}
  selectNote={this.selectNote}
  deleteNote={this.deleteNote}
</sidebarItemComponent>
```

Step 21 → Create a material ui <Divider> below <sidebarItemComponent>

Step 22 → Go to sidebar item component →

Destructure props :-

```
const {_index, _note, classes, selectedNoteIndex}
= this.props;
```

Step 23 → Add key={_index} to div.

Step 24 → Add <ListItem> component from material-ui

```
<ListItem
  className={classes.listItem}
  selected={selectedNoteIndex === _index}
  alignItems='flex-start'>

</ListItem>
```

Step 25 → Inside ListItem, add a div

```
<div
  className = {classes.textSection}
  onClick = {() => this.selectNote(_note, _index)}>
```

Step 26 → Inside , add <ListItemText>

```
<ListItemText
  primary = {_note.title}
  secondary = {removeHTMLTags(_note.body.
    substring(0,30)) + '...'}
</ListItemText>
```

Step 27 → Add a delete icon, in <ListItem>, below div.

```
<DeleteIcon onClick = {() => this.deleteNote(_note)}
            className = {classes.deleteIcon}
</DeleteIcon>
```

Step 28 → Define selectNote and deleteNote functions

```
selectNote = (n, i) => this.props.selectNote(n, i);
deleteNote = (note) => {
  if (window.confirm('Are you sure you want to
  delete: ${note.title}')) {
    this.props.deleteNote(note);
  }
}
```

Step 29 → In App.js →

Inside <SideBarComponent>, pass attributes.

```
deleteNote = {this.deleteNote}
selectNote = {this.selectNote}
newNote = {this.newNote}
```

step 30 → Define selectNote in App.js

```
selectNote = (note, index) =>
this.setState({
    selectNoteIndex : index ,
    selectedNote : note });
```

step 31 → Go to Sidebar.js →
(make some changes in selectNote func")

```
selectNote = (n,i) => this.props.selectNote(n, i)
```

step 32 → In App.js, add attr. to EditorComponent.

```
selectedNote = {this.state.selectedNote}
selectedNoteIndex = {this.state.selectedNoteIndex}
notes = {this.state.notes}
noteUpdate = {this.noteUpdate}

{ this.state.selectedNote ?
  <EditorComponent> ---
  </EditorComponent> :
    null
}
```

step 33 → Go to editor.js, add some code.

```
componentDidMount = () => {
this.setState({
  text : this.props.selectedNote.body,
  title : this.props.seletedNote.title,
  id : this.props.selectedNote.id,
});
}
```

componentDidUpdate = () => {
if (this.props.selectedNote.id !== this.state.id) {
this.setState({
text: this.props.selectedNote.body,
title: this.props.selectedNote.title,
id: this.state.id,
});
}

Step 54 → Define func" noteUpdate in App.js

Step 55 → Define func" newNote in App.js

App.js ✓
editor.js ✓