<https://www.gyanblog.com/javascript/how-integrate-next-js-draft-js-strapi-create-article-upload-image-view-page/>

import React, { Component } from 'react'

import {EditorState} from "draft-js";

import dynamic from 'next/dynamic';

import apiClient from '../api/api\_client'

import { convertFromRaw, convertToRaw } from 'draft-js';

const Editor = dynamic(

() => import('react-draft-wysiwyg').then(mod => mod.Editor),

{ ssr: false }

)

import "react-draft-wysiwyg/dist/react-draft-wysiwyg.css";

export default class ArticleEditor extends Component {

constructor(props) {

super(props);

this.state = {

editorState: EditorState.createEmpty()

};

}

onEditorStateChange = (editorState) => {

this.setState({

editorState,

});

this.props.handleContent(

convertToRaw(editorState.getCurrentContent()

));

};

uploadImageCallBack = async (file) => {

const imgData = await apiClient.uploadInlineImageForArticle(file);

return Promise.resolve({ data: {

link: `${process.env.NEXT\_PUBLIC\_API\_URL}${imgData[0].formats.small.url}`

}});

}

render() {

const { editorState } = this.state;

return (

<Editor

editorState={editorState}

toolbarClassName="toolbar-class"

wrapperClassName="wrapper-class"

editorClassName="editor-class"

onEditorStateChange={this.onEditorStateChange}

// toolbarOnFocus

toolbar={{

options: ['inline', 'blockType', 'fontSize', 'fontFamily', 'list', 'textAlign', 'colorPicker', 'link', 'embedded', 'emoji', 'image', 'history'],

inline: { inDropdown: true },

list: { inDropdown: true },

textAlign: { inDropdown: true },

link: { inDropdown: true },

history: { inDropdown: true },

image: {

urlEnabled: true,

uploadEnabled: true,

uploadCallback: this.uploadImageCallBack,

previewImage: true,

alt: { present: false, mandatory: false }

},

}}

/>

)

}

}

**Event Handlers for Input**

We have set two event handlers for title and body.

For the body where we have used Draft.js WYSWYG. We are using a Draft.js function convertToRaw to save data in state.

**Image Upload**

We have configured an option in options for Draft.js

image: {

urlEnabled: true,

uploadEnabled: true,

uploadCallback: this.uploadImageCallBack,

previewImage: true,

alt: { present: false, mandatory: false }

},

And, the code from apiClient for uploading image:

async uploadInlineImageForArticle(file) {

const headers = await this.getAuthHeader();

const formData = new FormData();

formData.append('files', file);

try {

let { data } = await axios.post(

`${process.env.NEXT\_PUBLIC\_API\_URL}/upload`,

formData,

{

headers: headers,

}

)

return data;

} catch (e) {

console.log('caught error');

console.error(e);

return null;

}

}

Note, you would need to configure permission to upload image in your strapi backend.

**Create Write Page and Redirect after save**

Page: /pages/write.jsx

import SimpleLayout from '../components/layout/simple'

import React, { Component } from 'react'

import { Form, Button, Card } from 'react-bootstrap';

import { withRouter } from 'next/router'

import ArticleEditor from '../components/editor/editor'

import apiClient from '../components/api/api\_client'

class Write extends Component {

constructor(props) {

super(props);

this.state = {

title: "",

body: ""

};

this.handleInputs = this.handleInputs.bind(this);

this.submitForm = this.submitForm.bind(this);

}

handleInputs = (event) => {

let {name, value} = event.target

this.setState({

[name]: value

});

}

handleEditorContent = (content) => {

this.setState({

body: content,

articleUpdated: true

});

}

submitForm = async (event) => {

event.preventDefault()

let article = await apiClient.saveArticle({

title: this.state.title,

body: JSON.stringify(this.state.body)

})

this.props.router.push(`/articles/${article.slug}`);

}

render() {

return (

<SimpleLayout>

<div className="row">

<div className="col-8">

<Form onSubmit={this.submitForm}>

<Form.Group controlId="formBasicEmail">

<Form.Label>Headline</Form.Label>

<Form.Control type="text"

name="title"

value={this.state.title}

onChange={this.handleInputs} />

<Form.Text className="text-muted">

Give a nice title to your article

</Form.Text>

</Form.Group>

<Form.Group controlId="exampleForm.ControlTextarea2">

<Form.Label>Body</Form.Label>

<Card className="p-2">

<ArticleEditor

handleContent={this.handleEditorContent}

/>

</Card>

</Form.Group>

<Button variant="primary" type="submit">

Submit

</Button>

</Form>

</div>

<div className="col-4">

Another col

</div>

</div>

</SimpleLayout>

)

}

}

export default withRouter(Write);