**React Document**

**How to create a node module**

**Use following commands**

1. **npm init - for create package.json**
2. **npm login - for login into npm account**
3. **npm publish - for publish the code to npm**

**NOTE: Before Starting please ensure you’ve already install NODE**

**Check Version node-v , npm -v**

1. **Install React using CLI**

**npm install -g create-react-app**

**npx create-react-app my-react-app**

**cd my-react-app**

**npm start**

Main File Where need to work

**In SRC Folder there is index.js this is our main file (so don’t touch this file)**

**There is another app.js file so work on to that file.**

**In react everything is Component to do any work first need to create component there are two type of components  
A. Class Component**

**B. Functional Component**

**For Beginners I would suggest use functional component they are easy to use and easy to learn (in short very easy)**

**Following I Attach Basic Example of Functional Component**

**import React from 'react';**

**function HelloWorld() {**

**return (**

**<div>**

**<h1>Hello World</h1>**

**</div>**

**);**

**}**

**export default HelloWorld;**

**Firstly in every component don’t forget to import react.**

**Then create a function and use any name for it then there is a return value of function. Your most of the code goes inside the return.**

1. Install Bootstrap

**npm install bootstrap**

**import 'bootstrap/dist/css/bootstrap.min.css';**

Optional (if you want to use bootstrap js components)

**npm install react-bootstrap**

Want to use any component like modal first import

**import 'bootstrap/js/dist/modal'**

**(Then go to bootstrap official website copy modal code and use)**

1. Install Swiper Slider

**npm install swiper**

**import { Swiper, SwiperSlide } from 'swiper/react';**

**Use following code in functional component**

**<Swiper**

**spaceBetween={50}**

**slidesPerView={3}**

**onSlideChange={() => console.log('slide change')}**

**onSwiper={(swiper) => console.log(swiper)}**

**>**

**<SwiperSlide>Slide 1</SwiperSlide>**

**<SwiperSlide>Slide 2</SwiperSlide>**

**<SwiperSlide>Slide 3</SwiperSlide>**

**<SwiperSlide>Slide 4</SwiperSlide>**

**</Swiper>**

**For More Details Check Official Website**

[**https://swiperjs.com/react**](https://swiperjs.com/react)

1. Install AOS Animation in React

**npm install aos**

**import React, { useEffect } from 'react';**

**import AOS from 'aos';**

**import 'aos/dist/aos.css';**

**const AOSAnimate = () => {**

**useEffect(() => {**

**AOS.init({**

**duration: 1000,**

**easing: 'ease-in-out',**

**});**

**}, []);**

**return (**

**<div>**

**<div data-aos="fade-up" data-aos-duration="1000">**

**<h1>Content to animate</h1>**

**</div>**

**</div>**

**);**

**};**

**export default AOSAnimate;**

1. How to add Image

**import logo from './assets/logo.svg';**

**function App() {**

**return (**

**<div className="App">**

**<img src={logo} className="logoReact"/>**

**</div>**

**);**

**}**

**export default App;**

**If we want to use images from public folder then usie the following paths**

1. **For Components  
   /images/[image\_name]**
2. **For CSS  
   ../../../public/images/[image\_name]  
   Set “../” according to your folder structure.**
3. How to Use Routers

To create routes and navigation for multiple pages in a React

application, you typically use a library like React Router. React Router

allows you to define routes and handle navigation between different

components/pages in your application.

**npm install react-router-dom**

**When we build multi page website so don’t do anything in app.js & index.js**

**In app.js we create routes for all pages. See example below for better understanding**

**import React from 'react';**

**// Import Browser Route**

**import { BrowserRouter as Router, Route, Routes } from 'react-router-dom';**

**// Import All Pages**

**import Home from './inc/pages/HomePage';**

**import About from './inc/pages/AboutPage';**

**import Contact from './inc/pages/ContactPage';**

**import Services from './inc/pages/ServicesPage';**

**import NotFound from './inc/pages/NotFound';**

**function App() {**

**return (**

**<div className="App">**

**<Router>**

**<div>**

**{/\* Create Slugs using Routes \*/}**

**<Routes>**

**<Route exact path="/" element={<Home />} />**

**<Route path="/about" element={<About />} />**

**<Route path="/contact" element={<Contact />} />**

**<Route path="/services" element={<Services />} />**

**<Route path="\*" element={<NotFound />} />**

**</Routes>**

**</div>**

**</Router>**

**</div>**

**);**

**}**

**export default App;**

**Note: You need to create components for all pages then import in app.jsx and create routes for all (see above example)**

1. Add pages links

<>

<Link to='../' className="">Home</Link>

<Link to='/about/' className="">About</Link>

<Link to='/contact/' className="">Contact</Link>

</>

1. Install Axios and usage

**npm install axios**

**import React, { useEffect } from 'react';**

**import axios from 'axios';**

**useEffect(() => {**

**const fetchData = async () => {**

**try {**

**const response = await axios.get('https://jsonplaceholder.typicode.com/posts');**

**console.log(response.data);**

**} catch (error) {**

**console.error('Error fetching data:', error);**

**}**

**};**

**fetchData();**

**}, []);**

**Here we use useEffect hook. The useEffect hook is a React hook**

**that allows you to perform side effects in function components.**

**Side effects include things like data fetching, subscriptions, or**

**manually changing the DOM.**

**If want to append data in DOM from API Response See Folowing**

**Example**

**import HeaderComp from '../components/Header';**

**import React, { useEffect , useState } from 'react';**

**import axios from 'axios';**

**const ServicesPage = () => {**

**const [dataMy, setData] = useState(null);**

**useEffect(() => {**

**const fetchData = async () => {**

**try {**

**const response = await axios.get('https://jsonplaceholder.typicode.com/posts');**

**console.log(response.data);**

**setData(response.data)**

**} catch (error) {**

**console.error('Error fetching data:', error);**

**}**

**};**

**fetchData();**

**}, []);**

**return (**

**<div>**

**<HeaderComp />**

**{dataMy ? ( // Check if data is not null before rendering**

**<ul>**

**{dataMy.map(item => (**

**<li key={item.id}>{item.title}</li>**

**))}**

**</ul>**

**) : (**

**<p>Loading...</p>**

**)}**

**<h1>Services</h1>**

**</div>**

**);**

**};**

**export default ServicesPage;**

1. Install Hemlet for SEO

**npm install react-helmet-async**

**Put the following code in index.js or the main entry file of your app:**

**import React from 'react';  
import ReactDOM from 'react-dom';  
import { HelmetProvider } from 'react-helmet-async';  
import App from './App';**

**ReactDOM.render(  
 <HelmetProvider>  
 <App />  
 </HelmetProvider>,  
 document.getElementById('root')  
);**

**Put following code in all of your pages {page-name}.js**

**import React from 'react';  
import { Helmet } from 'react-helmet-async';**

**function HomePage() {  
 return (  
 <div>  
 <Helmet>  
 <title>Home Page - Your Site</title>  
 <meta name="description" content="This is the homepage of Your Site." />  
 <link rel="canonical" href="https://yoursite.com/home" />  
 </Helmet>  
 <h1>Welcome to the Home Page</h1>  
 </div>  
 );  
}  
export default HomePage;**

1. Fetch data using Wordpress API’s

This code is according to axios so make sue Axios is install in your react app

<http://bookmyevents.wp/wp-json/wp/v2/pages/>

[http://bookmyevents.wp/wp-json/wp/v2/pages/21](http://bookmyevents.wp/wp-json/wp/v2/pages/) (specific page)

[http://bookmyevents.wp/wp-json/wp/v2/posts/](http://bookmyevents.wp/wp-json/wp/v2/pages/)

[http://bookmyevents.wp/wp-json/wp/v2/posts/21](http://bookmyevents.wp/wp-json/wp/v2/pages/) (specific post)

[http://bookmyevents.wp/wp-json/wp/v2/{custom\_post\_type}/](http://bookmyevents.wp/wp-json/wp/v2/pages/)

<http://bookmyevents.wp/wp-json/wp/v2/media>

<http://bookmyevents.wp/wp-json/wp/v2/media?parent=11> (specific media)

<http://originalboommyevents.wp/wp-json/wp/v2/event/?slug=friday-night> (for single blog post just pass the slug name as a query string)

**In following code we are trying to fetch home page acf content.**

import React, { useEffect , useState } from 'react';

import axios from 'axios';

// Import Components

import Banner from '../components/HomeBanner';

import BannerSlider from '../components/HomeBannerSlider';

import About from '../components/AboutSection';

function Home() {

const [acfData, setAcfData] = useState(null);

// fetch data from wordpress

useEffect(() => {

const fetchData = async () => {

try {

const response = await axios.get('http://bookmyevents.wp/wp-json/wp/v2/pages/');

// console.log(response.data);

// Find the object with title "Home Page"

const homePage = response.data.find(item => item.title && item.title.rendered === "Home Page");

if (homePage && homePage.acf) {

// Store the acf data in state

setAcfData(homePage.acf);

} else {

console.log("No ACF data found for home page");

}

} catch (error) {

console.error('Error fetching data:', error);

}

};

fetchData();

}, []);

return (

<>

<div>

{acfData && acfData.sections && acfData.sections.map((section, index) => {

// You can add logic to render different components based on the section type

switch (section.acf\_fc\_layout) {

case 'main\_banner':

// return <Banner key={index} props={section} pageId={pageId} />;

return <BannerSlider key={index} props={section} />;

case 'about':

return <About key={index} props={section} />;

// Add more case blocks for other section types as needed

default:

return null;

}

})}

</div>

</>

);

}

export default Home;

**Banner Component**

import React, { useEffect, useState } from 'react';

import axios from 'axios';

function HomeBanner({ props }) {

const [backgroundImage, setBackgroundImage] = useState(null);

const bannerImageId = props.banner\_bg;

console.log(bannerImageId);

useEffect(() => {

// following code is to fetch the banner image using page id and image id

const fetchData = async () => {

try {

const response = await axios.get(`http://bookmyevents.wp/wp-json/wp/v2/media/${bannerImageId}`);

console.log(response);

// Find the media item with the specified ID

if (response.data.guid.rendered) {

setBackgroundImage(response.data.guid.rendered); // Set the image URL

} else {

console.warn("Media not found for banner\_bg:", props.banner\_bg);

}

} catch (error) {

console.error('Error fetching data:', error);

}

};

fetchData();

}, [props]);

return (

<section className="bannerSection" style={{ backgroundImage: `url(${backgroundImage})` }}>

<div className="container">

<div className="row">

<div className="col-12">

<h1>{props.banner\_heading}</h1>

<p>{props.banner\_description}</p>

</div>

</div>

</div>

</section>

);

}

export default HomeBanner;

**In following code we are trying to fetch gallerie in pages.**

**import React, { useEffect, useState } from 'react';**

**import axios from 'axios';**

**function HomeBannerSlider({ props }) {**

**const [MediaItem, setMediaItems] = useState([]);**

**// console.log(props.banner\_gallery);**

**const bannerIds = props.banner\_gallery;**

**useEffect(() => {**

**const fetchAllMedia = async () => {**

**try {**

**// Use Promise.all to fetch all media items**

**const responses = await Promise.all(**

**bannerIds.map((bannerId) =>**

**axios.get(`http://bookmyevents.wp/wp-json/wp/v2/media/${bannerId}`)**

**)**

**);**

**// Extract the data from each response and update state**

**const items = responses.map((response) => response.data);**

**setMediaItems(items);**

**console.log(items);**

**} catch (error) {**

**console.error("Error fetching data:", error);**

**}**

**};**

**fetchAllMedia();**

**}, [bannerIds]);**

**return (**

**<section className="bannerSection">**

**<div className="container">**

**<div className="row">**

**<div className="col-12">**

**<h1>{props.banner\_heading}</h1>**

**<p>{props.banner\_description}</p>**

**{MediaItem.map((item) => (**

**<img**

**key={item.id}**

**src={item.guid.rendered}**

**alt={`Media ID ${item.id}`}**

**/>**

**))}**

**</div>**

**</div>**

**</div>**

**</section>**

**);**

**}**

**export default HomeBannerSlider;**

1. Basic Form using Spread Operator and console the values

**import React, { useState } from 'react';**

**function Contact() {**

**const [getInputValues, holdInputValues] = useState({**

**fName : '',**

**lName : '',**

**eMail : '',**

**pHone : ''**

**});**

**const GetInputEvent = (event) => {**

**const {name, value} = event.target;**

**holdInputValues((prevVal) => {**

**//console.log(prevVal);**

**return{**

**...prevVal,**

**[name]: value,**

**}**

**})**

**}**

**const onSubmitForm = (event) =>{**

**event.preventDefault();**

**console.log(getInputValues);**

**}**

**return (**

**<>**

**<section className="formContact">**

**<div className="container">**

**<div className="row">**

**<div className="col-md-7">**

**<form className="formWrapper" onSubmit={onSubmitForm}>**

**<div className="ap\_form">**

**<h1>Hello</h1>**

**<input type="text" placeholder="Username" value={getInputValues.fName} name="fName" onChange={GetInputEvent}/>**

**<input type="text" placeholder="Password" value={getInputValues.lName} name="lName" onChange={GetInputEvent}/>**

**<input type="email" placeholder="Email" value={getInputValues.eMail} name="eMail" onChange={GetInputEvent}/>**

**<input type="tel" placeholder="Phone" value={getInputValues.pHone} name="pHone" onChange={GetInputEvent}/>**

**<button>Submit</button>**

**</div>**

**</form>**

**</div>**

**</div>**

**</div>**

**</section>**

**</>**

**);**

**}**

**export default Contact;**

1. Basic Form submission in cf7 wordpress using API’s and mails are sent

**import React, { useState } from 'react';**

**import axios from 'axios';**

**import GlobalComp from '../global/globalComponent';**

**function Contact() {**

**const [getInputValues, holdInputValues] = useState({**

**fName : '',**

**eMail : '',**

**pHone : ''**

**});**

**const [responseMessage, setResponseMessage] = useState('');**

**const GetInputEvent = (event) => {**

**const {name, value} = event.target;**

**holdInputValues((prevVal) => {**

**//console.log(prevVal);**

**return{**

**...prevVal,**

**[name]: value,**

**}**

**})**

**}**

**const onSubmitForm = async (event) => {**

**event.preventDefault();**

**console.log(getInputValues);**

**// Prepare the payload**

**const payload = new FormData();**

**payload.append('your-name', getInputValues.fName); // Replace 'your-name' with the CF7 field name**

**payload.append('your-email', getInputValues.eMail); // Replace 'your-email' with the CF7 field name**

**payload.append('your-phone', getInputValues.pHone); // Replace 'your-phone' with the CF7 field name**

**payload.append('\_wpcf7\_unit\_tag', 123);**

**for (let pair of payload.entries()) {**

**console.log(pair[0] + ": " + pair[1]);**

**}**

**try {**

**const response = await axios.post(**

**'http://bookmyevents.wp/wp-json/contact-form-7/v1/contact-forms/60/feedback',payload,{**

**headers: {**

**'Content-Type': 'multipart/form-data',**

**},**

**}**

**);**

**if (response.data.status === 'mail\_sent') {**

**setResponseMessage('Your message has been sent successfully!');**

**holdInputValues({ fName: '', eMail: '', pHone: '' });**

**} else {**

**setResponseMessage('There was an issue sending your message. Please try again.');**

**}**

**} catch (error) {**

**setResponseMessage('An error occurred. Please check your connection and try again.');**

**console.error('Error sending the form:', error);**

**}**

**}**

**return (**

**<>**

**<section className="formContact">**

**<div className="container">**

**<div className="row">**

**<div className="col-md-7">**

**<form className="formWrapper" onSubmit={onSubmitForm}>**

**<div className="ap\_form">**

**<h1>Hello</h1>**

**<input type="text" placeholder="Username" value={getInputValues.fName} name="fName" onChange={GetInputEvent}/>**

**<input type="email" placeholder="Email" value={getInputValues.eMail} name="eMail" onChange={GetInputEvent}/>**

**<input type="tel" placeholder="Phone" value={getInputValues.pHone} name="pHone" onChange={GetInputEvent}/>**

**<button>Submit</button>**

**{responseMessage && <p>{responseMessage}</p>}**

**</div>**

**</form>**

**</div>**

**</div>**

**</div>**

**</section>**

**</>**

**);**

**}**

**export default Contact;**

1. How to create single blog posts.

Create a component for single blog posts

import React from 'react';

import { useParams } from 'react-router-dom';

const BlogPost = () => {

// Get the slug from the URL

const { slug } = useParams();

// Use the slug to fetch the blog post content (optional for API calls)

// For now, just display the slug

return (

<div>

<h1>Blog Post: {slug}</h1>

<p>Content of the blog post goes here...</p>

</div>

);

};

export default BlogPost;

Use the following api ke if you are using wordpess as a backend   
<http://originalboommyevents.wp/wp-json/wp/v2/event/?slug=friday-night>   
  
Pass the **post slug** inslug parameter

And don't forget to import this in **App.js** or wherever you have written your routes.

import BlogPost from './BlogPost';

**Put the following code in routes**

<Route path="/blog/:slug" element={<BlogPost />} />

1. Login using Wordpress API’s | Local Storage & Cookie

#### **a. Enable WordPress REST API**

WordPress already includes a REST API, so ensure it's enabled by default.   
For user authentication, you'll need the JWT Authentication plugin.  
<https://wordpress.org/plugins/jwt-authentication-for-wp-rest-api/>

Add the following lines to your wp-config.php file:

define('JWT\_AUTH\_SECRET\_KEY', 'jagga-is-back');

define('JWT\_AUTH\_CORS\_ENABLE', true);

Ensure your WordPress .htaccess file supports the API by adding:

RewriteCond %{HTTP:Authorization} ^(.\*)

RewriteRule ^(.\*) - [E=HTTP\_AUTHORIZATION:%1]

#### b. Login Form

**You'll use axios to communicate with the WordPress REST API.**

**import React, { useState } from 'react';**

**import axios from 'axios';**

**import { useNavigate } from 'react-router-dom';**

**// use navigate is used for redirect the user after successful login**

**function LoginForm() {**

**const [getInputValues, holdInputValues] = useState({**

**fName : '',**

**eMail : '',**

**pHone : '',**

**pWord : ''**

**});**

**const [responseMessage, setResponseMessage] = useState('');**

**const navigate = useNavigate(); // Initialize useNavigate for redirects**

**const GetInputEvent = (event) => {**

**const {name, value} = event.target;**

**holdInputValues((prevVal) => {**

**//console.log(prevVal);**

**return{**

**...prevVal,**

**[name]: value,**

**}**

**})**

**}**

**const onSubmitForm = async (event) => {**

**event.preventDefault();**

**console.log(getInputValues.eMail);**

**console.log(getInputValues.pWord);**

**try {**

**const response = await axios.post('http://bookmyevents.wp/wp-json/jwt-auth/v1/token', {**

**username: getInputValues.eMail,**

**password: getInputValues.pWord,**

**});**

**localStorage.setItem('token', response.data.token); // Save the JWT token**

**setResponseMessage('Login Successfull');**

**// Redirect to dashboard after login**

**navigate('/dashboard'); // redirects user**

**} catch (error) {**

**// console.error('Login failed', error.response.data);**

**setResponseMessage('Invalid username or password.');**

**}**

**}**

**return (**

**<>**

**<section className="formContact">**

**<div className="container">**

**<div className="row">**

**<div className="col-md-7">**

**<form className="formWrapper" onSubmit={onSubmitForm}>**

**<div className="ap\_form">**

**<h1>Hello</h1>**

**<input type="email" placeholder="Email" value={getInputValues.eMail} name="eMail" onChange={GetInputEvent}/>**

**<input type="password" placeholder="Password" value={getInputValues.pWord} name="pWord" onChange={GetInputEvent}/>**

**<button>Submit</button>**

**{responseMessage && <p>{responseMessage}</p>}**

**</div>**

**</form>**

**</div>**

**</div>**

**</div>**

**</section>**

**</>**

**);**

**}**

**export default LoginForm;**

#### c.Use following for protected Components

**Add the Code Open ProtectedRoute.js and paste the code:**

**import React from 'react';**

**import { Navigate } from 'react-router-dom';**

**const ProtectedRoute = ({ children }) => {**

**const token = localStorage.getItem('token');**

**return token ? children : <Navigate to="/login" />;**

**};**

**export default ProtectedRoute;**

**Import those component in App.js which are opens only after login**

**import ProtectedRoute from './inc/global/ProtectedRoute';**

**import Dashboard from './inc/pages/Dasboard';**

**Then add all those components in routes like following in App.js**

**<Route**

**path="/dashboard"**

**element={**

**<ProtectedRoute>**

**<Dashboard />**

**</ProtectedRoute>**

**}**

**/>**

#### d. Check if user is already login

**Check if user is already login and open login page then automatically redirect to dashboard page**

**Import “useEffect” in login component**

**import React, { useState, useEffect } from 'react';**

**Add the following component in inside login component.**

**Redirect if already logged in**

**useEffect(() => {**

**const token = localStorage.getItem('token');**

**if (token) {**

**navigate('/dashboard'); // Redirect to dashboard if token exists**

**}**

**}, [navigate]);**

#### e. Login Functionality using Cookie

**npm install js-cookie**

**Then import js-cookie in login component.**

**import Cookies from 'js-cookie';**

**const onSubmitForm = async (event) => {**

**event.preventDefault();**

**console.log(getInputValues.eMail);**

**console.log(getInputValues.pWord);**

**try {**

**const response = await axios.post('http://bookmyevents.wp/wp-json/jwt-auth/v1/token', {**

**username: getInputValues.eMail,**

**password: getInputValues.pWord,**

**});**

**Cookies.set('authToken', response.data.token, { expires: 4, secure: true });**

**setResponseMessage('Login Successfull');**

**navigate('/dashboard');**

**} catch (error) {**

**// console.error('Login failed', error.response.data);**

**setResponseMessage('Invalid username or password.');**

**}**

**}**

#### f.Use following for protected Components using js-cookie

**Add the Code Open ProtectedRoute.js and paste the code:**

**import React from 'react';**

**import { Navigate } from 'react-router-dom';**

**const ProtectedRoute = ({ children }) => {**

**const token = Cookies.get('authToken'); // Read token from cookies**

**return token ? children : <Navigate to="/login" />;**

**};**

**export default ProtectedRoute;**

#### g. Check is user is already login using js-cookie

**Check if user is already login and open login page then automatically redirect to dashboard page**

**Import “useEffect” in login component**

**import React, { useState, useEffect } from 'react';**

**Add the following component in inside login component.**

**Redirect if already logged in**

**useEffect(() => {**

**// const token = localStorage.getItem('token');**

**const token = Cookies.get('authToken'); // Read token from cookies**

**if (token) {**

**navigate('/dashboard'); // Redirect to dashboard if token exists**

**}**

**}, [navigate]);**

#### h. Logout using cookies

**import Cookies from 'js-cookie';**

**import { useNavigate } from 'react-router-dom';**

**const Logout = () => {**

**const navigate = useNavigate();**

**const handleLogout = () => {**

**Cookies.remove('authToken'); // Remove the cookie**

**alert('Logged out successfully!');**

**navigate('/login');**

**};**

**return <button onClick={handleLogout}>Logout</button>;**

**};**

**export default Logout;**

#### i. Logout using local storage

**import React from 'react';**

**import { useNavigate } from 'react-router-dom';**

**const Logout = () => {**

**const navigate = useNavigate();**

**const handleLogout = () => {**

**localStorage.removeItem('token'); // Remove the token from localStorage**

**alert('You have been logged out!'); // Optional: Notify the user**

**navigate('/login'); // Redirect to the login page**

**};**

**return <button onClick={handleLogout}>Logout</button>;**

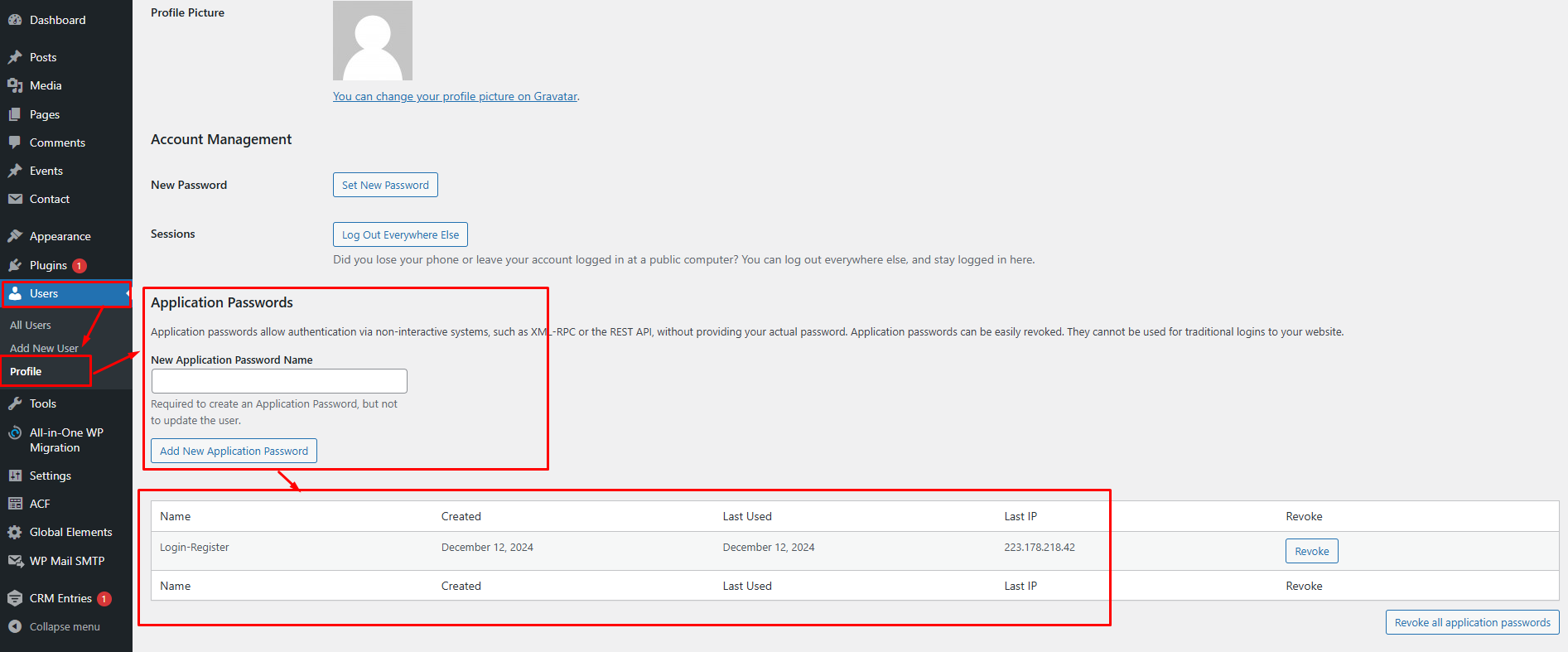
**};**

**export default Logout;**

1. Signup using Wordpress API’s

#### First we need to enable few option in wordpress

#### Create application password using main user’s profile Save application password which will be used in code in future.



#### Basic Sign up form with normal form validations

import React, { useState, useEffect } from 'react';

import axios from 'axios';

import { useNavigate , Link } from 'react-router-dom';

//import module

import { Helmet } from 'react-helmet-async';

function RegisterForm({ siteUrl }) {

const [responseMessage, setResponseMessage] = useState('');

const [responseClass, setResponseClass] = useState('');

const [loader, setLoader] = useState('');

const navigate = useNavigate(); // Initialize useNavigate for redirects

useEffect(() => {

const token = localStorage.getItem('token');

if (token) {

navigate('/dashboard'); // Redirect to dashboard if token exists

}

}, [navigate]);

const [getInputValues, holdInputValues] = useState({

uName: '',

fName: '',

lName: '',

eMail: '',

pHone: '',

pWord: '',

cpWord: ''

});

const GetInputEvent = (event) => {

const { name, value } = event.target;

holdInputValues((prevVal) => {

return {

...prevVal,

[name]: value,

}

})

}

function removeAllErrors() {

setResponseMessage(false);

setResponseClass(false);

setLoader(false);

}

function redirectAfterFewSec() {

navigate('/login');

}

const onSubmitForm = async (event) => {

event.preventDefault();

setLoader(true);

// normal validation form field on the basis of empty

for (let x in getInputValues) {

if(!getInputValues[x]){

setResponseMessage("Please fill all fields");

setResponseClass("alert-danger");

setLoader(false);

setTimeout(removeAllErrors, 4000);

return;

}

}

// check if password and confirm password is same or not

if(getInputValues.pWord !== getInputValues.cpWord){

setResponseMessage("Password and Confirm Password is not same");

setResponseClass("alert-danger");

setTimeout(removeAllErrors, 4000);

setLoader(false);

return;

}

// axios

try {

const auth = btoa('main\_admin:Ze5L lZ7I aOeb kmIc jD3c DZB9'); // Replace with your admin username and application password

await axios.post(`${siteUrl}wp-json/wp/v2/users`, {

username: getInputValues.uName,

email: getInputValues.eMail,

first\_name: getInputValues.fName,

last\_name: getInputValues.lName,

password: getInputValues.pWord,

// roles: ['administrator'],

acf: { // if some fields are created by using ACF then use following

field\_6763b91bfec4b: 'MALE', // Gender

field\_6763b95ffec4c: '20241213', // DOB

field\_6763b99ffec4d: 'Sector 15', // Address

field\_6763b9aefec4e: 'Panchkula', // City

field\_6763b9b7fec4f: 'Haryana', // State

field\_6763b9c2fec50: '134203', // PinCode

field\_6763b9cefec51: 'IN', // Country

field\_6763ba5ffec52: 'A+', // Blood Group

field\_6763baebfec53: 'Waoa', // Emer Name

field\_6763baf9fec54: '9898878798', // Emer Num

field\_6763bb0bfec55: 'XL' // Shirt Size

},

},

{

headers: {

Authorization: `Basic ${auth}`,

},

});

setResponseMessage('Signup successful! You can now login.');

setResponseClass("alert-success");

setLoader(false);

setTimeout(redirectAfterFewSec, 3000);

} catch (error) {

setResponseMessage('Signup Failed - Please try again');

setResponseClass('alert-danger');

setTimeout(removeAllErrors, 4000);

setLoader(false);

//console.error('Signup failed', error.response.data);

//alert('Error signing up: ' + error.response.data.message);

}

}

return (

<>

<Helmet>

<title>Register Page - Book my event</title>

<meta name="description" content="This is the contactpage of Bookmyevent." />

</Helmet>

<section className="common\_banner\_section pt\_100 pb\_100">

<div className="common\_banner\_section\_wrap">

<div className="container">

<div className="banner\_heading">

<h1 className="hdng fs\_55">Register</h1>

</div>

</div>

</div>

</section>

<section className="contact\_us\_section pt\_100 pb\_100">

<div className="contact\_us\_section\_wrap">

<div className="container">

<div className="row d-flex justify-content-center">

<div className="col-md-6">

<form className="form\_wrap" onSubmit={onSubmitForm}>

<div className="ap\_form">

{responseMessage && <div className="col-12 col-md-12 mt-4"><div className={`alert ${responseClass}`}>{responseMessage}</div></div>}

<div className="row">

<div className="col-12 col-md-6">

<input type="text" placeholder="First name" value={getInputValues.fName} name="fName" onChange={GetInputEvent} />

</div>

<div className="col-12 col-md-6">

<input type="text" placeholder="Last Name" value={getInputValues.lName} name="lName" onChange={GetInputEvent} />

</div>

<div className="col-12 col-md-12">

<input type="text" placeholder="Username" value={getInputValues.uName} name="uName" onChange={GetInputEvent} />

</div>

<div className="col-12 col-md-6">

<input type="email" placeholder="Email" value={getInputValues.eMail} name="eMail" onChange={GetInputEvent} />

</div>

<div className="col-12 col-md-6">

<input type="tel" placeholder="Phone" value={getInputValues.pHone} name="pHone" onChange={GetInputEvent} />

</div>

<div className="col-12 col-md-6">

<input type="password" placeholder="Password" value={getInputValues.pWord} name="pWord" onChange={GetInputEvent} />

</div>

<div className="col-12 col-md-6">

<input type="password" placeholder="Confirm Password" value={getInputValues.cpWord} name="cpWord" onChange={GetInputEvent} />

</div>

<div className="col-12">

<div className="submitWrapper">

<input type="submit" value="Signup" />

{loader &&

<div className="loaderWrap">

<span className="loader"></span>

</div>

}

</div>

</div>

<div className="col-12 col-md-6">

<p class="formUnderMessage">Already have an account ? <Link to="/login">Log in here</Link></p>

</div>

</div>

</div>

</form>

</div>

</div>

</div>

</div>

</section>

</>

);

}

export default RegisterForm;

Install MD5 Module

**npm i md5**

//just used following code

var md5 = require('md5');

console.log('admin = ' + md5('admin'));