**To create React App (We shall do it using vite)**

* PS D:\MERN\MERNwithSigninSignup> npm create vite@latest client
  + - Select React
    - Select JavaScript+SWC
* PS D:\MERN\MERNwithSigninSignup> cd .\client\
* PS D:\MERN\MERNwithSigninSignup\client> npm I //For installation
* Search **tailwind css vite >> goto first link (**[**https://tailwindcss.com/docs/guides/vite**](https://tailwindcss.com/docs/guides/vite)**)**
  + - Follow the steps in the link
      * PS D:\MERN\MERNwithSigninSignup\client> npm install -D tailwindcss postcss autoprefixer
      * PS D:\MERN\MERNwithSigninSignup\client> npx tailwindcss init –p
      * Configure your template paths: Add the paths to all of your template files in your tailwind.config.js file.
      * Add the Tailwind directives to your CSS: Add the @tailwind directives for each of Tailwind’s layers to your ./src/index.css file.
      * Clean the folder structure delete unwanted files from the folder : \*.svg files, App.css

**Install some important VS code extensions**

* ES7+ React/Redux/React-Native/JS snippets
* Tailwind CSS IntelliSense
* Prettier - Code formatter

**App.jsx:**

**import React from 'react'**

**export default function App() {**

**return (**

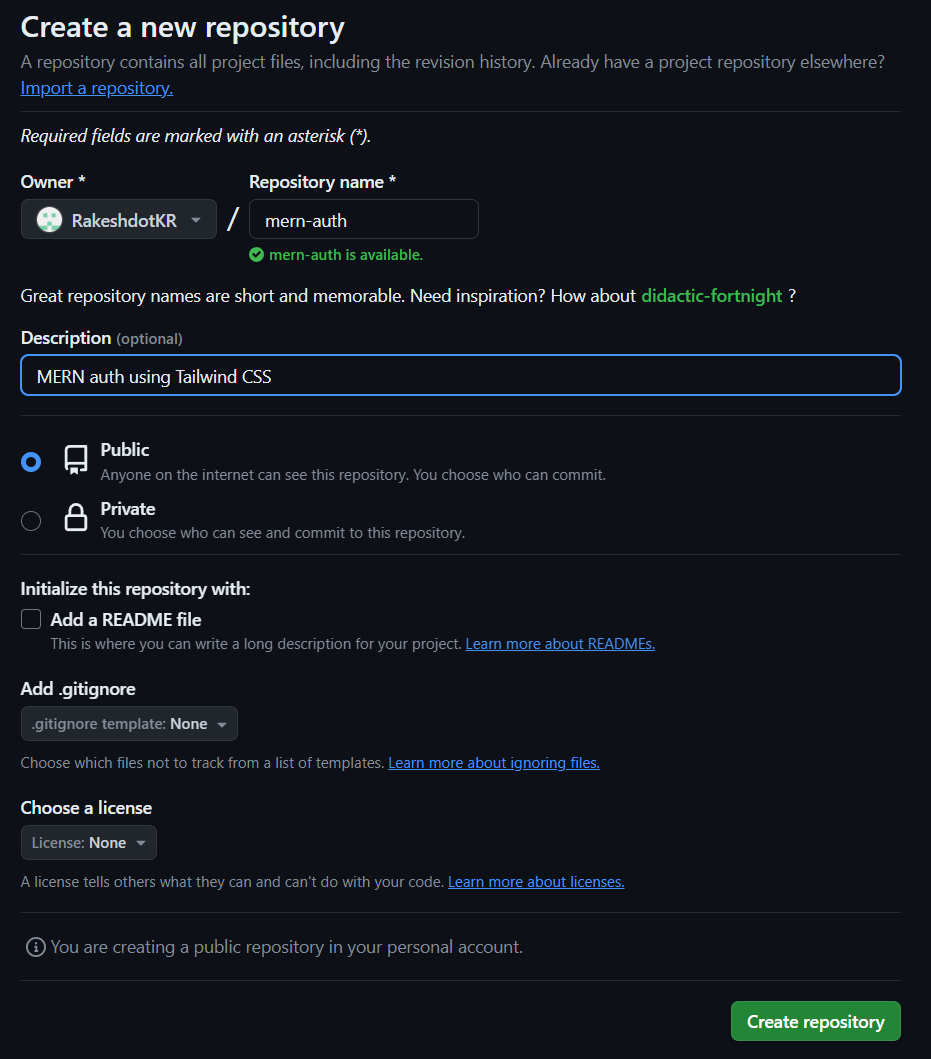
**<div>**

**<h1>Hello buddy</h1></div>**

**)}**

**GIT Integration:**

* PS D:\MERN\MERNwithSigninSignup> git init
* PS D:\MERN\MERNwithSigninSignup> git add .
* PS D:\MERN\MERNwithSigninSignup> git commit -m 'install reactjs and tailwind css'
* Go to github.com and create a repo as follows:



Run following commands in terminal:

PS D:\MERN\MERNwithSigninSignup> git remote add origin https://github.com/RakeshdotKR/mern-auth.git

PS D:\MERN\MERNwithSigninSignup> git branch -M main

PS D:\MERN\MERNwithSigninSignup> git push -u origin main

Create **pages** folder and create following files: client\src\pages

* SignIn.jsx
* SignUp.jsx
* About.jsx
* Home.jsx
* Profile.jsx

**Adding Navigation**

PS D:\MERN\MERNwithSigninSignup\client> npm i react-router-dom //for adding navigation

**App.jsx:**

**import { BrowserRouter, Routes, Route } from "react-router-dom";**

**import Home from "./pages/Home";**

**import About from "./pages/About";**

**import SignIn from "./pages/SignIn";**

**import SignUp from "./pages/SignUp";**

**import Profile from "./pages/Profile";**

**export default function App() {**

**return (**

**<BrowserRouter>**

**<Routes>**

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

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

**<Route path="/sign-in" element={<SignIn/>}/>**

**<Route path="/sign-up" element={<SignUp/>}/>**

**<Route path="/profile" element={<Profile/>}/>**

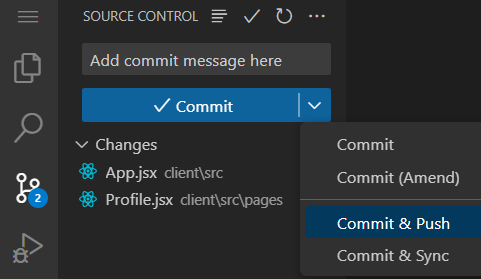
**</Routes>**

**</BrowserRouter>**

**);**

**}**

**Now whenever we need to commit the changes to GitHub do the following:**

****

**Adding Header Component:**

client\src\components\Header.jsx:

**import { Link } from "react-router-dom";**

**export default function Header() {**

**return (**

**<div className="bg-slate-300">**

**<div className="flex justify-between items-center max-w-7xl mx-auto p-3">**

**<Link to="/">**

**<h3 className="font-semibold">Auth App</h3>**

**</Link>**

**<ul className="flex gap-3">**

**<Link to="/">**

**<li>Home</li>**

**</Link>**

**<Link to="/about">**

**<li>About</li>**

**</Link>**

**<Link to="/sign-in">**

**<li>Sign In</li>**

**</Link>**

**</ul>**

**</div>**

**</div>**

**);**

**}**

**The Backend**

**Create & run the server**

Create a folder called as ‘api’ in the root folder:

In terminal:

* PS D:\MERN\MERNwithSigninSignup\api> npm init –y
* PS D:\MERN\MERNwithSigninSignup\api> npm install express nodemon

**//to be added in package.json**

**"scripts": {**

**"dev": "nodemon index.js",**

**"start": "node index.js"**

  },

PS D:\MERN\MERNwithSigninSignup\api> npm run dev //to run server

**api\index.js:**

**import express from "express"; //Add this line in package.json "description": "","type":"module",**

**const app = express();**

**app.listen(3000,()=>{**

**console.log("Server listening on port 3000");**

**});**

Connect to Database

PS D:\MERN\MERNwithSigninSignup\api> npm i mongoose

PS D:\MERN\MERNwithSigninSignup\api> npm i dotenv

api\index.js

**dotenv.config();**

**import express from "express"; //Add this line in package.json "description": "","type":"module",**

**import mongoose from "mongoose";**

**import dotenv from 'dotenv';**

**// console.log("Mongo Uri: ",process.env.MONGOURI);**

**mongoose**

**.connect(process.env.MONGOURI, { useNewUrlParser: true, useUnifiedTopology: true })**

**.then(()=>{**

**console.log("Successfully connected to MongoDB");**

**})**

**.catch((err)=>{**

**console.log(err);**

**});**

**const app = express();**

**app.listen(process.env.PORT,()=>{**

**console.log("Server listening on port 3000");**

**});**

**Create & add user model**

**api\models\user.model.js**

**import mongoose from "mongoose";**

**const userSchema = new mongoose.Schema({**

**username:{**

**type:String,**

**required:true,**

**unique:true,**

**},**

**email:{**

**type:String,**

**required:true,**

**unique:true,**

**},**

**password:{**

**type:String,**

**required:true,**

**}**

**},{timestamps:true});**

**const User = mongoose.model('User',userSchema); //Here 'User' start with Uppercase and singular**

**export default User;**

Build APIs

api\index.js (add below at end of file)

**app.get("/", (req, res) => {**

**res.json({**

**message: "GET method is working fine",**

**});**

**});**

* Start backend server and use url **http://localhost:3000/** in browser

Separating Routes into folders

api\routes\user.route.js

**import express from 'express';**

**const router = express.Router();**

**router.get("/", (req, res) => {**

**res.json({**

**message: "GET method is working fine",**

**});**

**});**

**export default router;**

api\index.js

**import userRoutes from './routes/user.route.js'; //.js at end is necessary**

**app.use('/api/user',userRoutes); //http://localhost:3000/api/user/**

Creating separate Controllers

api\controllers\user.controller.js

**export const test = (req,res) =>{**

**res.json({**

**message:'Api is working',**

**});**

**}**

api\routes\user.route.js

**import express from 'express';**

**import { test } from '../controllers/user.controller.js'; //.js at the end is necessary. This is new here**

**const router = express.Router();**

**// router.get("/", (req, res) => {**

**//     res.json({**

**//       message: "GET method is working fine",**

**//     });**

**//   });**

**router.get('/',test); //This the new line here**

**export default router;**

Create Sign up API route

api\routes\auth.route.js

**import express from 'express';**

**import {signup} from '../controllers/auth.controller.js';**

**const router = express.Router();**

**router.post('/signup',signup);**

**export default router;**

api\controllers\auth.controller.js

**import User from "../models/user.model.js";**

**export const signup = async(req,res)=>{**

**console.log(req.body); //app.use(express.json()); to be added in index.js**

**const {username,email,password} = req.body;**

**const newUser = new User({username,email,password});**

**try{**

**await newUser.save();**

**res.status(201).json({message:'User created successfully'});**

**}**

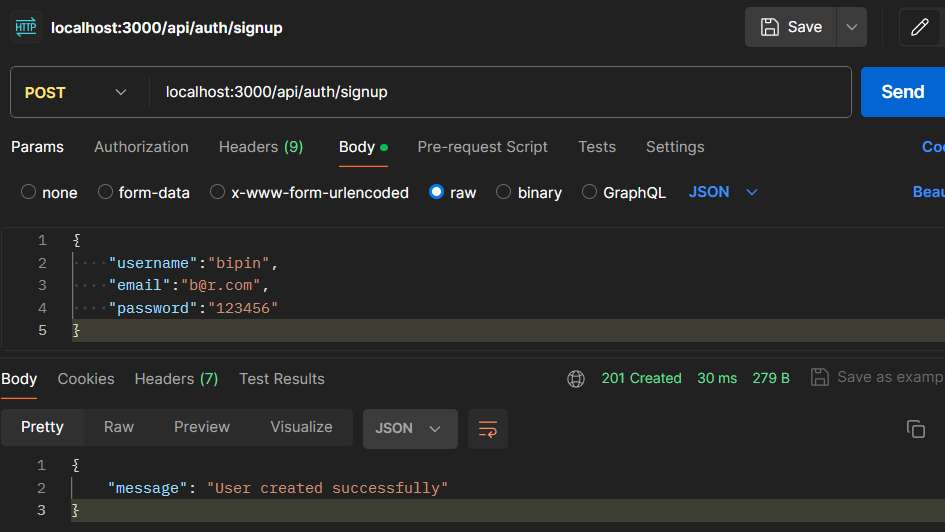
**catch(error){**

**res.status(500).json(error.message);**

**}**

**};**

Signup API testing in postman



Encrypting the password

PS D:\MERN\MERNwithSigninSignup\api**> npm i bcryptjs**

api\controllers\auth.controller.js

**import User from "../models/user.model.js";**

**import bcryptjs from 'bcryptjs';**

**export const signup = async(req,res)=>{**

**console.log(req.body); //app.use(express.json()); to be added in index.js**

**const {username,email,password} = req.body;**

**const hashPassword = bcryptjs.hashSync(password,10);**

**const newUser = new User({username,email,password:hashPassword});**

**try{**

**await newUser.save();**

**res.status(201).json({message:'User created successfully'});**

**}**

**catch(error){**

**res.status(500).json(error.message);**

**}**

**};**

Create a middleware and a function to handle the errors

api\index.js (add it at the end)

**app.use((err,req,res,next)=>{**

**const statusCode = err.statusCode || 500;**

**const message = err.message || 'Internal server error';**

**return res.status(statusCode).json({**

**success:false,**

**message,**

**statusCode,**

**});**

**});**

**Now in the controllers add the following**

export const signup = async(req,res,**next**)=>{

    try{

…

    }

    catch(error){

**next(error);** //This line to be added

    }

};

**Signup UI page**

client\src\pages\SignUp.jsx

**// import React from "react";**

**import { useState } from "react";**

**import { Link } from "react-router-dom";**

**export default function SignUp() {**

**const [formData, setFormData] = useState({});**

**/\* Adding state individually of the form:**

**const [username,setusername] = useState('');**

**const [email,setemail] = useState('');**

**const [password,setpassword] = useState(''); \*/**

**const handleChange = (e) => {**

**setFormData({ ...formData, [e.target.id]: e.target.value });**

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

**};**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**const res = await fetch('/api/auth/signup', {**

**method: 'POST',**

**headers: {**

**'Content-Type': 'application/json',**

**},**

**body: JSON.stringify(formData),**

**});**

**const data = await res.json();**

**console.log(data);**

**// {message:'User created successfully'}**

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

**};**

**return (**

**<div className="p-3 max-w-lg mx-auto">**

**<h1 className="text-3xl text-center font-semibold my-7">Sign up</h1>**

**<form onSubmit={handleSubmit} className="flex flex-col gap-4">**

**<input**

**type='text'**

**placeholder='Username'**

**id='username'**

**className='bg-slate-100 p-3 rounded-lg'**

**onChange={handleChange}**

**/>**

**<input**

**type='email'**

**placeholder='Email'**

**id='email'**

**className='bg-slate-100 p-3 rounded-lg'**

**onChange={handleChange}**

**/>**

**<input**

**type='password'**

**placeholder='Password'**

**id='password'**

**className='bg-slate-100 p-3 rounded-lg'**

**onChange={handleChange}**

**/>**

**<button**

**className="bg-slate-900 text-white p-3 rounded-lg uppercase hover:opacity-95**

**disabled:opacity-80"**

**>**

**Sign Up**

**</button>**

**</form>**

**<div className="flex gap-2 mt-5">**

**<p>Have an account?</p>**

**<Link to="/sign-in">**

**<span className="text-blue-600">Sign In</span>**

**</Link>**

**</div>**

**</div>**

**);**

**}**

**Adding:**

* **Error handling in signup page,**
* **Page Loading icon/message concept**
* **navigate to signin after signup**

**client\src\pages\SignUp.jsx**

// import React from "react";

import { useState } from "react";

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

export default function SignUp() {

  const [formData, setFormData] = useState({});

**const [error, setError] = useState(null);**

**const [loading, setLoading] = useState(false);**

**/\* Adding state individually of the form:**

**const [username,setusername] = useState('');**

**const [email,setemail] = useState('');**

**const [password,setpassword] = useState(''); \*/**

**const navigate = useNavigate();**

**const handleChange = (e) => {**

**setFormData({ ...formData, [e.target.id]: e.target.value });**

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

**};**

**const handleSubmit = async (e) => {**

**e.preventDefault();**

**try {**

**setLoading(true);**

**setError(false);**

**const res = await fetch("/api/auth/signup", {**

**method: "POST",**

**headers: {**

**"Content-Type": "application/json",**

**},**

**body: JSON.stringify(formData),**

**});**

**const data = await res.json();**

**console.log(data);**

**setLoading(false);**

**if (data.success === false) {**

**setError(true);**

**return;**

**}**

**navigate("/sign-in");**

**} catch (error) {**

**setLoading(false);**

**setError(true);**

**}**

**};**

**return (**

**<div className="p-3 max-w-lg mx-auto">**

**<h1 className="text-3xl text-center font-semibold my-7">Sign up</h1>**

**<form onSubmit={handleSubmit} className="flex flex-col gap-4">**

**<input**

**type="text"**

**placeholder="Username"**

**id="username"**

**className="bg-slate-100 p-3 rounded-lg"**

**onChange={handleChange}**

**/>**

**<input**

**type="email"**

**placeholder="Email"**

**id="email"**

**className="bg-slate-100 p-3 rounded-lg"**

**onChange={handleChange}**

**/>**

**<input**

**type="password"**

**placeholder="Password"**

**id="password"**

**className="bg-slate-100 p-3 rounded-lg"**

**onChange={handleChange}**

**/>**

**<button disabled={loading}**

**className="bg-slate-900 text-white p-3 rounded-lg uppercase hover:opacity-95**

**disabled:opacity-80"**

**>**

**{loading ? 'In Progress' : 'Sign Up'}**

**</button>**

**</form>**

**<div className="flex gap-2 mt-5">**

**<p>Have an account?</p>**

**<Link to="/sign-in">**

**<span className="text-blue-600">Sign In</span>**

**</Link>**

**</div>**

**<p className='text-red-700 mt-5'>{error && 'Something went wrong!'}</p>**

**</div>**

**);**

**}**

Create sign in API route

api\controllers\auth.controller.js

**import User from "../models/user.model.js";**

**import bcryptjs from "bcryptjs";**

**import { errorhandler } from "../utils/error.js";**

**import jwt from "jsonwebtoken";**

**export const signup = async (req, res, next) => {**

**console.log(req.body); //app.use(express.json()); to be added in index.js**

**const { username, email, password } = req.body;**

**const hashPassword = bcryptjs.hashSync(password, 10);**

**const newUser = new User({ username, email, password: hashPassword });**

**try {**

**await newUser.save();**

**res.status(201).json({ message: "User created successfully" });**

**} catch (error) {**

**// res.status(500).json(error.message);**

**next(error);**

**}**

**};**

**export const signin = async (req, res, next) => {**

**const { email, password } = req.body;**

**try {**

**const validUser = await User.findOne({ email });**

**if (!validUser) return next(errorhandler(404, "User not found"));**

**const validPassword = bcryptjs.compareSync(password, validUser.password);**

**if (!validPassword) return next(errorhandler(401, "Wrong credentials"));**

**const token = jwt.sign({ id: validUser.\_id }, process.env.JWT\_SECRET);**

**const { password: hashedPassword, ...rest } = validUser.\_doc; //\_doc: destructure**

**const expiryDate = new Date(Date.now() + 3600000); // 1 hour**

**res**

**.cookie("access\_token", token, { httpOnly: true, expires: expiryDate })**

**.status(200)**

**//.json(validUser) //gives password as part of the response so we need to avoid this**

**.json(rest);**

**} catch (error) {**

**next(error);**

**}**

**};**

api\routes\auth.route.js

**import express from "express";**

**import { signup, signin } from "../controllers/auth.controller.js";**

**const router = express.Router();**

**router.post("/signup", signup);**

**router.post("/signin", signin);**

**export default router;**

api\.env (append following)

**JWT\_SECRET=shivashiva369**

Add Redux Toolkit

Create a ‘**redux**’ folder inside src folder and in that create file called **store.js**

Install “**npm i @reduxjs/toolkit react-redux**”

client\src\user\userSlice.js

**import { CreateSlice } from '@reduxjs/toolkit';**

**const initialState = {**

**currentUser: null,**

**loading: false,**

**error: false,**

**};**

**const userSlice = CreateSlice({**

**name:'user',**

**initialState,**

**signInStart: (state) =>{**

**state.loading = true;**

**},**

**signInSuccess: (state,action) => {**

**state.currentUser = action.payload;**

**state.loading = false;**

**state.error = false;**

**},**

**signInFailure: (state,action) => {**

**state.loading = false;**

**state.error = action.payload;**

**}**

**});**

**export const {signInStart, signInSuccess, signInFailure} = userSlice.actions;**

**export default userSlice.reducer;**

client\src\redux\store.js

**import {configureStore} from '@reduxjs/toolkit';**

**import userReducer from './user/userSlice';**

**export const store = configureStore({**

**reducer: {user: userReducer},**

**middleware:(getDefaultMiddleware) => getDefaultMiddleware({**

**serializableCheck: false,**

**}),**

**});**

Install redux extention [Redux DevTools in the browser](https://chromewebstore.google.com/detail/lmhkpmbekcpmknklioeibfkpmmfibljd)

Errors:

1. MongooseError: The `uri` parameter to `openUri()` must be a string, got "undefined". Make sure the first parameter to `mongoose.connect()` or `mongoose.createConnection()` is a string.

Solution:

* Make sure the .env file is in the same location as the index.js
* And in the .env file it should be like follows (not within quotes)
  + MONGOURI = mongodb://127.0.0.1:27017/mern-auth;