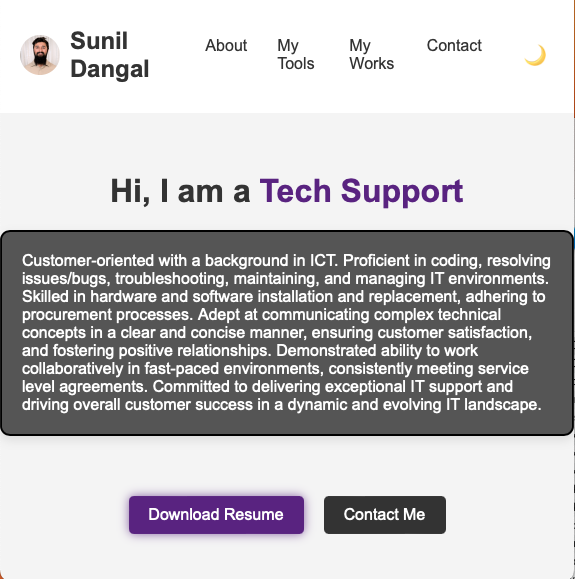
Portfolio

Website



By Sunil Dangal

TAble of  
Contents

Introduction 1

Setting Up React Project 1

AWS DEPLOY 2

Conclusion 4

## Introduction

So, I have created three files with simple Designs, Layouts and structure. There are index.html, styles.css and script.js files. This is a portfolio website, and it is responsive. Now, I am planning to use Node.js, react.js for the frontend and backend tasks, PostgreSQL for database and deploying it to the AWS.

## Setting Up React Project

Now, let’s begins by installing Node.js. Before we proceed, we got to make sure Node.js is installed in our machine. To download, Node.js go to this link <https://nodejs.org/en> and download and install.

After the installation of Node.js, we are going to create a new react app. In terminal, in our directory, we type:

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

**cd my-portfolio**

After the successful installation, we are going to navigate to our project directory. Now, we are going to organize our files. So, I have moved the HTML elements, JSX components, JavaScript logics inside App.js. Likewise, I have also moved all the styles.css to App.css file.

Now I have made a well-structured React component with features like dark mode, dynamic charts using Chart.js and smooth scrolling to Top functionality. This is how my structure looks like after, refactoring original code and after creating a build folder for production.



## AWS Deploy

Now, I have login to AWS console and created a S3 bucket and uploaded all the build items inside S3 bucket. After the upload, I tested, and everything looks awesome. Now I want to make the form work in the Contact menu. For example: If a user presses the submit button after putting the details, then I want to receive those data in my personal email. So, let’s do that.

So, I also added Amazon Simple Email Service and registered my personal email to use it later to receive form contents.

After registering in AWS SES, Now, it is time to add API, I created **ContactSubmissionAPI using** AWSAPI Gateway and create ContactSubmissionAPI. I have also added Route inside as /contact and created a POST request to connect to the Lambda Function. After, creation of Route which is /contact, we need to integrate with Lambda Function so, we deploy.

Now for the lambda function, inside the code, we need to run the JSX logic to make the form work. So, In, lambda function, I create a function named as Create function, and selected Author from scratch, and configured latest Node.js for the runtime and created a new role with basic lambda permissions for debugging later if issues arise. So, the function is created. Now it is time to implement a function logic inside the function configuration page. And we must code the logic to make the form send form information to our email which is registered in AWS SES.

I renamed the code file to index.js from index.msj, I also added another package.json file and added code = {"type": "module"}. After that, I deployed, saved and tested and I was able to receive test email in my personal email.

Now, that we know the Lambda Function is working. We need to configure Cross-Origin Resource Sharing to allow our frontend application to interact with our API. So, we go to API and inside CORS and configure following:

* **Access-Control-Allow-Origin**: \* (or our specific domain)
* **Access-Control-Allow-Headers**: Content-Type
* **Access-Control-Allow-Methods**: OPTIONS, POST
* **Access-Control-Expose-Headers**: (leave blank)
* **Access-Control-Max-Age**: 3600
* **Access-Control-Allow-Credentials**: NO (or YES if needed)

Now after writing CORS, we were able to route form details to my email address.

Lastly, we need to npm run build and update the build folder to S3 bucket and we must put Access-Control-Allow Origin: \* (or our specific domain). Before, I did **Access-Control\_Allow-Origin**: <http://localhost:3000> because I was testing from a local machine. But when we upload the new Build to S3, we got to put \*. And that’s it all done. Now we can see the website running and people can easily send information through contact page, and I will be able to receive in my email.

The intermediate part is finished, now I was going to use **AWS Route 53** and set up a domain [**www.sunildangal.com.au**](http://www.sunildangal.com.au/) to load the portfolio website hosted on AWS S3. But I find out AWS Route 53 to be bit expensive so I have chosen some other cheaper domain provider in order to host my AWS website.

So, I chose, crazydomains.com. Now I bought 1 year domain subscription which is the basic one. Now, we need to go to the DNS setting tab. Inside there, we are going to delete the default DNS settings if there are any. Since, S3 buckets don't use IP addresses for static website hosting, we should delete both existing **A Records.** Now, we need to add a new record. In the selection, we select CNAME record, and press ADD.

## Conclusion

So, this project is 💻 hosted in an AWS S3 bucket, my portfolio includes a fully functional contact form that allows users to submit their information, which goes straight to my email via AWS services.

Key AWS services used:

* S3 for hosting 🌐
* API Gateway and Lambda for dynamic contact form functionality 🛠️
* IAM for secure access controls 🔐
* Amazon SES for email integration ✉️
* CloudWatch for monitoring ⏱️

this project showcases not only my technical skills in leveraging AWS services but also my commitment to creating seamless and functional user experiences. I'm excited to keep building and learning as I continue my journey in tech!