

Resume Builder Web Application

Final Report

Team Leader: Ravi Krishna Teja Dachepalli (16300464)

Team members:

- 1.Jayanth Golla
- 2.Ravi Krishna Teja Dachepalli
- 3.Anudeep Thippireddy
- 4.Manoj Desetty

Project Description (Background Story):

For any working individual in any business, creating a resume is a time - consuming effort. It must be kept short, straightforward, and current with work experience, and it must be updated on a regular basis.

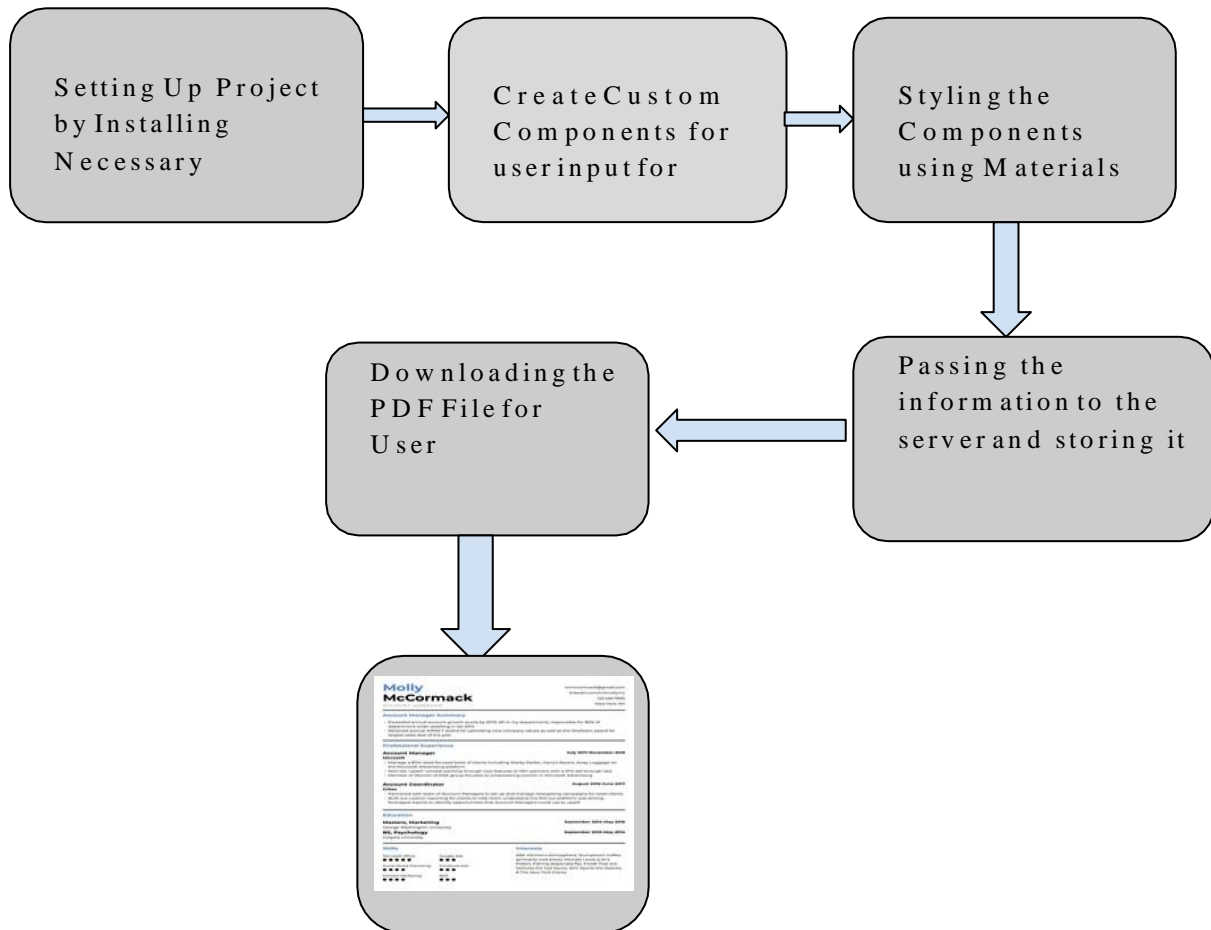
This project will guide you through the steps necessary to create a resume builder using ReactJS and NodeJS. Implementing the project will provide you the satisfaction of being able to generate it on your own while also assisting working pros.

The main objective is to develop a web application that will auto -generate a nice and properly formatted Resume from the information filled up in a form.

Here the resume contains,

- Contact Information.
- Career Objective.
- Skills and Abilities
- Work experience
- Educational qualifications

Project Workflow:



There are different stages involved in creating this web application which gives us some good resume pdf documents as output through the process that can be followed to build your resume-builder using ReactJS and NodeJS.

Stage 1:

The first step is installing react on your system using and creating new react applications using various tools. Basically, this step is installing all the required environments for the project.

The first step is setting up an environment which is required for building an application. We start it by installing a platform i.e Webstorm or Visual Studio.

Stage 2:

In this stage, we need to build components which take the User input and store them in a form which was used for building a good resume.

Resume Builder contains the following columns as different input fields requested from the user. All these columns together form a data set that is required to build a resume of the user.

Table showing basic user inputs and their data type

Input	Data type of the input	User Reply
Image	.jp or .jpeg	
First Name	text	
Middle Name	text	
Last Name	text	
About (< 50 words)	text	
Date of Birth	date	
Address	text	
Email Id	text	
Contact	number	
Highest Degree	text	
Date of Highest Degree	date	
Score of Highest Degree	number	
Experience	number	
Technical Skills	Text	
Certifications (if any)	Text or URL	
Previous Company	text	
Publications (if any)	Text or URL	

Personal Blog	Text or URL	
Social Media	Text or URL	
Awards	Text	
Leadership Skills	Text	

Stage 3:

Here, the output of the stage 2 is styled by using Styling components like Bootstrap and CSS components.

Stage 4:

The styled components are passed to the server and stored in an user selected template.

Stage 5:

In this stage, the resume template is available in PDF format which can be easily downloaded by the user by clicking on a button.

Application working behind the scenes:

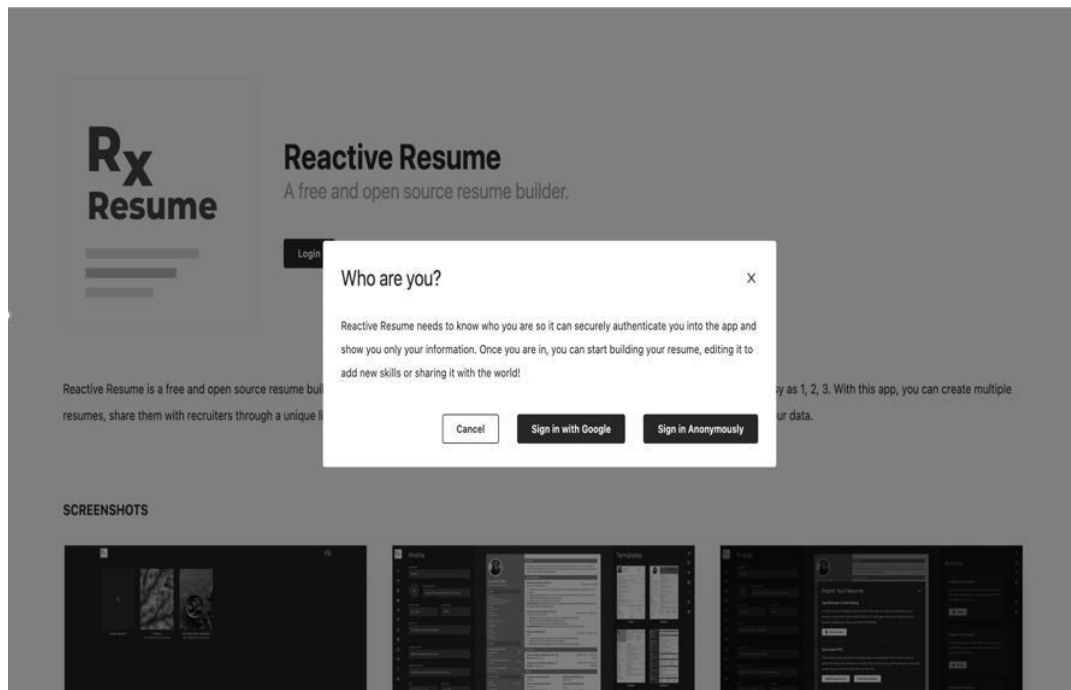
1. Users will visit the front-end website built by react.
2. When user enters the data, the frontend will connect with the API endpoints generated by NodeJS and stores the data in Firebase.
3. NodeJS will query the Firebase database and return the results to the frontend.
4. Firebase Database will load the data from the user input data into the database on a regular basis.
5. The result of the frontend is a well-designed resume which will be available in PDF format and it can be downloaded on a single click.

Data and its Flow:

- Here the data is Users Input. User's input is taken as raw data and this data will be formatted to give a resume that can be used to apply to various jobs.
- The data is completely about the user and the user enters the data in the provided HTML input areas.
- This project doesn't need any sampling as the user gives only the requested data.
- This data will be fresh for every user and this application will not be working with any global or previous data.
- This data doesn't contain geographical information or any other factors like regional, national, or global as the data is user specific.

Working Screens and Features of Application:

- User Authentication: User can create an account or sign in through Gmail or sign in anonymously



User Auth Screen

- Following image is the Main screen, where user can fill out the details for our resume.

- This data will be used to build resume in different templates according to the user selection.

The Main Screen is divided into three main sections. On the left is a 'Profile' form with fields for personal information: First Name (Jayanth), Last Name (Golla), Subtitle (Student), Date of Birth (10/31/2021), Address Line 1 (525 E Armour Blvd, Apt# 605), Address Line 2 (Apt# 605), City (Kansas City), and ZIP Code (64109). In the center is a preview of the resume for 'Jayanth Golla', showing sections for Profile, Objective, Work Experience, Education, and Contact Information. On the right is a 'Templates' sidebar displaying six different resume templates: Onyx, Pikachu, Genagar, Castform, and two others partially visible. A vertical sidebar on the far left contains icons for various app functions like profile, settings, and export.

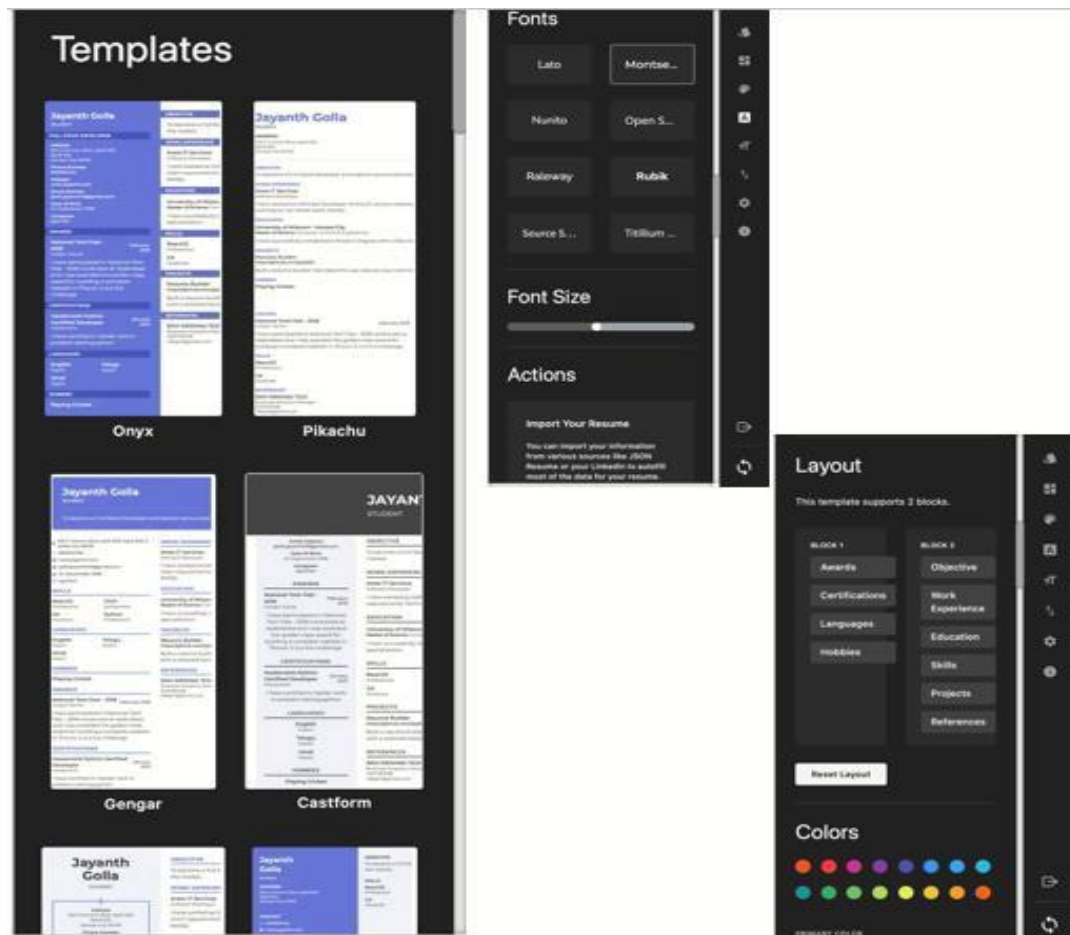
Main Screen

- Another important feature is that user can import, export, print and download resume. This enables user friendly experience.

The Print Screen shows the 'Export Your Resume' dialog box overlaid on the profile form. The dialog has three main sections: 'Use Browser's Print Dialog' with a 'Print Resume' button; 'Download PDF' with 'Single Page Resume' and 'Multi Page Resume' buttons; and 'Export to JSON Format' with an 'Export JSON' button. On the right side of the screen, a sidebar contains settings for 'Font Size' (with a slider) and 'Actions' (with buttons for 'Import' and 'Export').

Print Screen

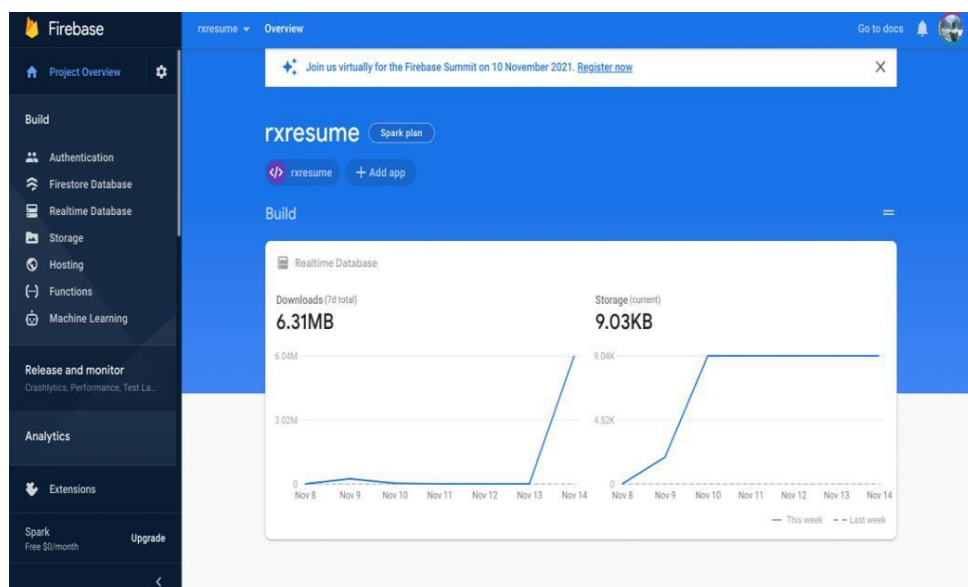
- User can select any template out of the 6 provided templates.
- User can change the color of the template.
- User can access font, font size, layout colors etc. in the side bar provided.



Template and Side Bar Screen

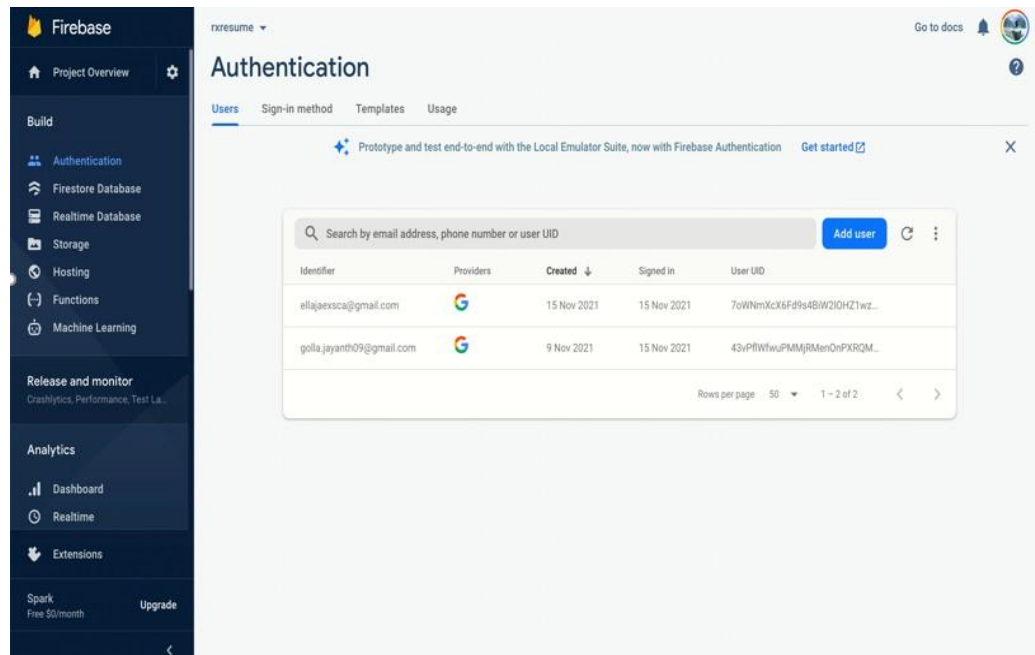
Database:

- Firebase Realtime Database is used to store the data and database schema is created according to the requirement.



Firebase Main Screen

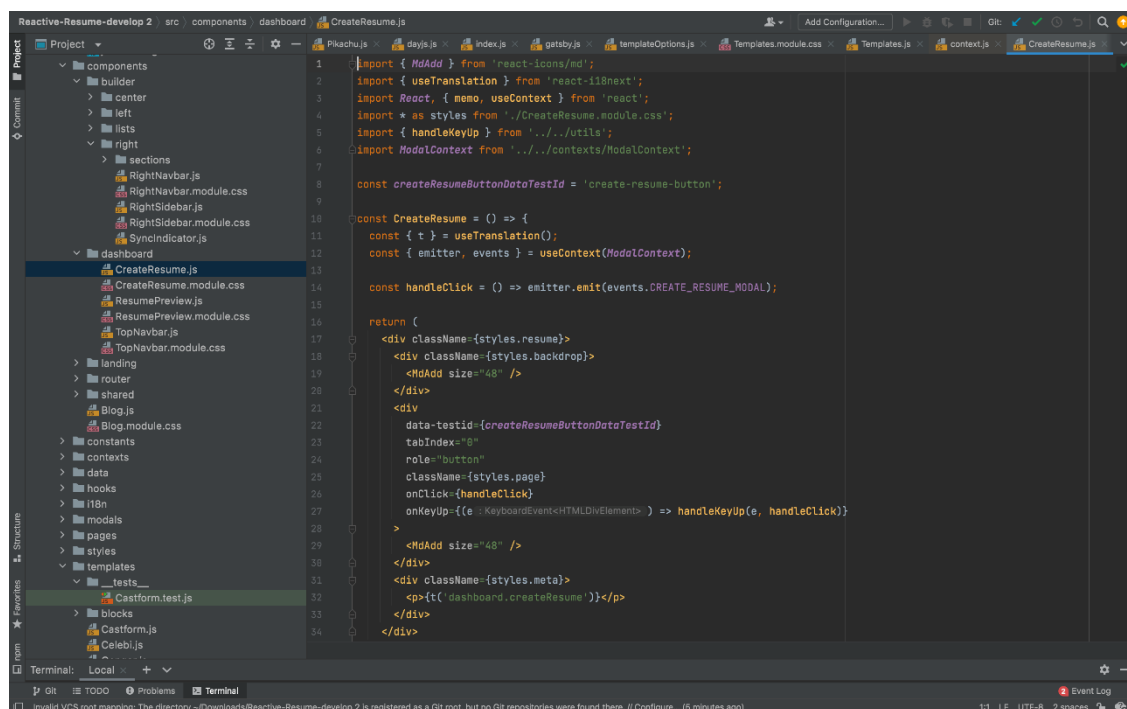
- For Authentication, you can login through Gmail or Anonymously. The following picture displays the list of users auth data stored in the database.



User Auth Data in Firebase

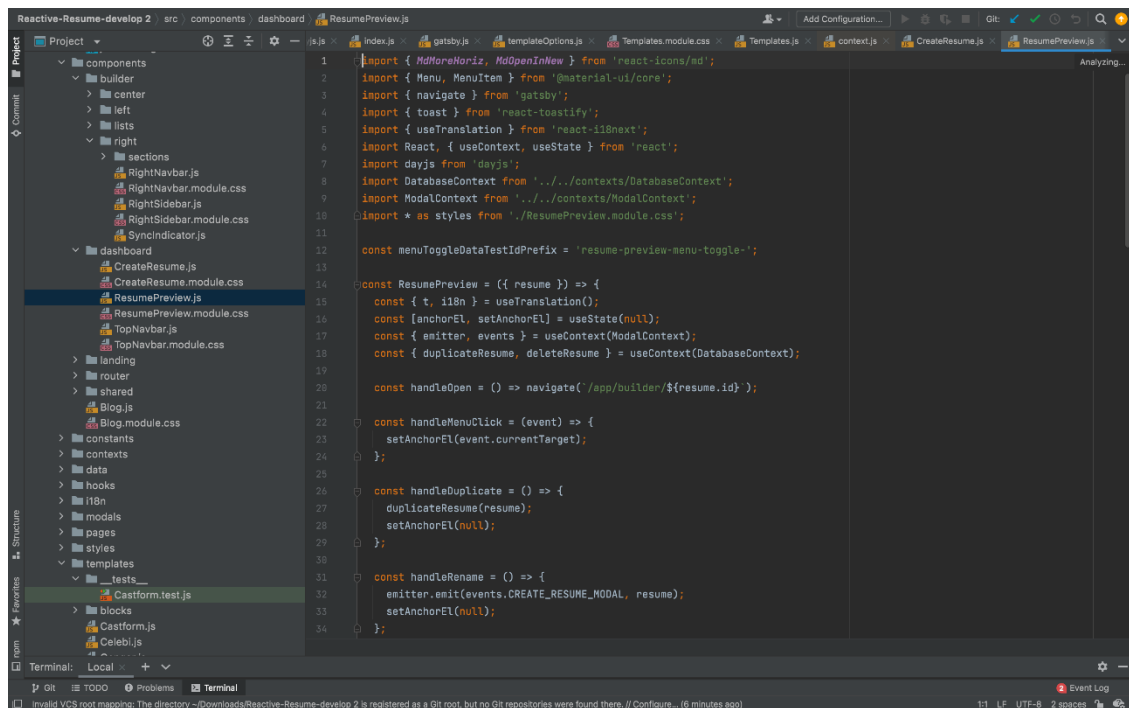
Code Snippets

Following code snippet is related to the information of user profile. It seeks all the required information from the user and this data will be stored to obtain resume in required template/format.



Resume Creation

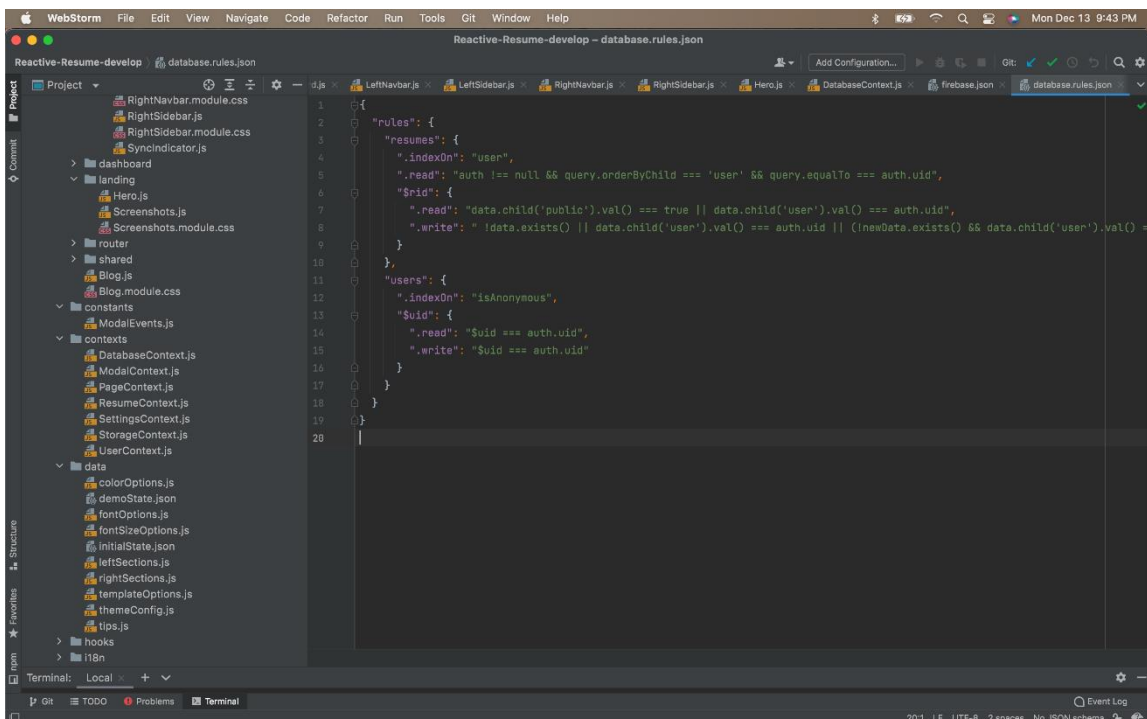
Following code snippet is related to the templates that helps to build resume from user given data. This consists of all the templates, and it is responsible for inserting the data into template opted by the user. User can preview the resume before going to download / print it.



```
1 import { MdMoreHoriz, MdOpenInNew } from 'react-icons/md';
2 import { Menu, MenuItem } from '@material-ui/core';
3 import { navigate } from 'gatsby';
4 import { toast } from 'react-toastify';
5 import { useTranslation } from 'react-i18next';
6 import React, { useContext, useState } from 'react';
7 import dayjs from 'dayjs';
8 import DatabaseContext from '../../contexts/DatabaseContext';
9 import ModalContext from '../../contexts/ModalContext';
10 import * as styles from './ResumePreview.module.css';
11
12 const menuToggleDataTestIdPrefix = 'resume-preview-menu-toggle-';
13
14 const ResumePreview = ({ resume }) => {
15   const { t, i18n } = useTranslation();
16   const [anchorEl, setAnchorEl] = useState(null);
17   const { emitter, events } = useContext(ModalContext);
18   const { duplicateResume, deleteResume } = useContext(DatabaseContext);
19
20   const handleOpen = () => navigate('/app/builder/${resume.id}');
21
22   const handleMenuClick = (event) => {
23     setAnchorEl(event.currentTarget);
24   };
25
26   const handleDuplicate = () => {
27     duplicateResume(resume);
28     setAnchorEl(null);
29   };
30
31   const handleRename = () => {
32     emitter.emit(events.CREATE_RESUME_MODAL, resume);
33     setAnchorEl(null);
34   };
35 }
```

Resume Preview

Following code snippet is related to the firebase where data and user information is stored in the schema created.



```
1 {
2   "rules": {
3     "resumes": {
4       ".indexOn": "user",
5       ".read": "auth !== null && query.orderByChild === 'user' && query.equalTo === auth.uid",
6       ".write": "data.child('public').val() === true || data.child('user').val() === auth.uid || (newData.exists() && data.child('user').val() === auth.uid)",
7       ".delete": "data.child('user').val() === auth.uid || (newData.exists() && data.child('user').val() === auth.uid)",
8     },
9     "users": {
10      ".indexOn": "isAnonymous",
11      ".read": "$uid === auth.uid",
12      ".write": "$uid === auth.uid"
13    }
14  }
15 }
```

Firebase

Issues:

- Establishing proper connection with Firebase Realtime Database
- Creating appropriate schema in the database
- Issues with cascading style sheets while designing the website.

Work Distribution:

- Front end: Anudeep & Jayanth
- Back end: Jayanth, Manoj, Anudeep & Ravi
- Documentation: Ravi

GitHub URL: <https://github.com/ravi4080/ResumeBuilder>

Presentation URL:

<https://github.com/ravi4080/ResumeBuilder/blob/main/Documentation/Final%20Presentation/Team-01-Final-Presentation-Resume%20Builder.pdf>

Video URL:

<https://umsystem.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=78bbb867-8c2b-4235-ad49-ade101309e7c>

References:

- Resume Genius: <https://resumegenius.com/>
- Article: <http://www.prosperoverseas.com/sample-resume>
- Firebase Article: <https://css-tricks.com/intro-firebase-react/>
- Resume Templates: <https://novoresume.com/resume-templates>