

INCREMENT 4

RESUME BUILDER - project title



Team Name : Web Dev

Team Lead : Kota Venkata Sowmyasri

Team Members:

Kota Venkata SowmyaSri

Kota Venkata RamyaSri

Kolla Yamini

Gopu Venkata Srinivasa Baba

Story-Problem Statement

Sam is a college student who can't write her perfect resume. She keeps struggling with designing her resume. Every time she has to apply for a different type of job role she has to update the resume and change the format accordingly which takes a lot of time. Although many websites provide resume templates they charge money for downloading the resume after filling out the data and investing time in building it.

References <https://www.myperfectresume.com>

The screenshot displays the 'myPerfect resume' website interface. At the top, a progress bar indicates the user is at the 'FEATURES' stage of a 4-step process (RESUME COMPLETED, FEATURES, PAYMENT, DONE). The main heading reads 'Upgrade for Instant Access to All Features'. Below this, three subscription options are presented: '14 Day Access' for \$2.95, 'Monthly Access' for \$5.95, and 'All Subscription Features'. The 'All Subscription Features' box lists benefits such as multiple formats, resume and cover letter builder, job search package, and a money back guarantee. A 'CONTINUE' button is visible at the bottom of this box. A Trustpilot review badge is shown in the bottom right corner, indicating 4,813 reviews.

myPerfect resume

RESUME COMPLETED FEATURES PAYMENT DONE

Upgrade for Instant Access to All Features

2,342 people have used the 14-day Access in the last 14 hours!

14 Day Access

\$2.95

- Download, email & print in multiple formats (PDF, Word, TXT)
- Create unlimited resumes and cover letters
- First download, print or email included. Additional \$0.45 each
- After 14 days, unlimited downloads and auto-renews at \$20.95, billed every 4 weeks
- Cancel anytime

All Subscription Features

- Multiple Formats:** Download & save in multiple formats (PDF, Word, RTF, TXT)
- Resume and Cover Letter Builder:** Create a resume or cover letter in minutes
- Job Search Package:** **Included** Get more interviews with our enhanced resume and job search features [LEARN MORE](#)
- Money Back Guarantee:** If you are unhappy for any reason during the first 14 days, just let us know - **we'll refund your money.**

CONTINUE

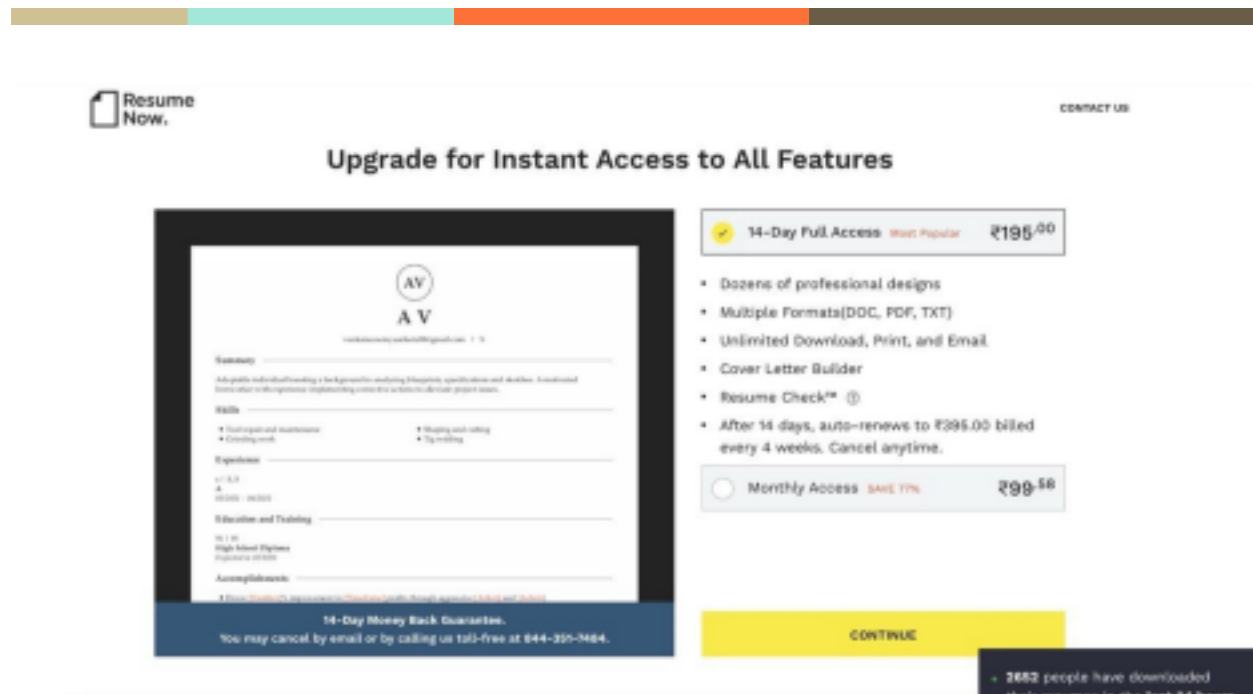
Monthly Access

\$5.95

- Pay \$11.40 up-front and save 77%
- Full access to all features including cover letters
- Automatically renews each year, cancel anytime
- Pay once, use all year long

You may cancel by email or calling us toll free at 888.213.6048

Trustpilot
★★★★★ 4,813 reviews



<https://www.resume-now.com/>

Targeted Audience

The majority of the students can get help from our application by building their resumes based on the job role and type they are applying for e.g. part-time, full-time, data engineer, front-end engineer, etc...

When sam decided to build a resume she googled for online resume building websites, then she gave out all the data required to download the application spending a lot of time on it, but then at the end when she tried to download the resume the websites asked her to upgrade her account and pay some amount to unlock other features which she didn't know earlier.

This problem is faced by many job seekers like students, professionals. Many job seekers find it hard to build resumes which takes a lot of mind-boggling and time, sometimes it can be overwhelming too.

Because of applying for many job roles Sam was tired and wanted to find the best platform to build her resumes and at least some of the templates for free and choose them depending on the job role and job type she was applying for.



Approach To The Solution:

- We are registering the users using sign-up and sign-in, the user's details and their resume details are saved to the database.
- So the users can access their resumes whenever they want by signing in to our website.
- Users can edit their details in the resume which was saved previously or can create a new resume.
- Our website is providing the skill suggestion feature where users can get the suggestion related to the skills by providing the technology which they want to add to the resume.
- The related skills to the technology are auto-populated in the resume skill field. Users can download the resume in pdf format from the website.

Data

- In our application, we mainly have two data points. One is for the user authentication, which stores the credentials of the user when signed up for the application and the second data point is for storing resume data of the users.
- These data points are hosted on the firebase platform, which is the cloud-based NoSQL data source.
- We will also scrape the job portal websites to get the keywords for skill suggestions based on various criteria like experience level, job role and store them in a database.

Source of User Data

- User credentials data is collected from the application at the time of signup.
- The user has to provide the data in the fields of the resume. The above information is stored in the form of JSON format in the real-time database of firebase.
- After the user is signed in to the application, we already have the resume details of the user in the database. We can auto-populate the data to the existing resume fields.

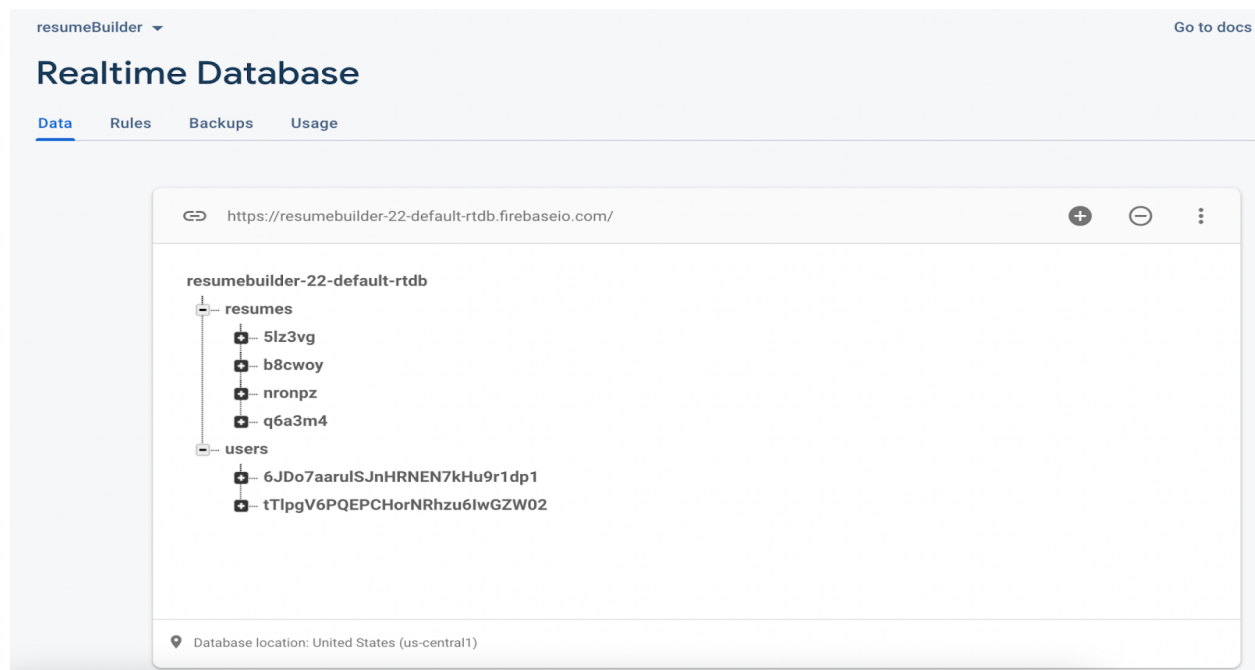
Source of Keyword Data


EMSI API

- We are using EMSI API for accessing the skill keywords, based on the technology the user enters
- This API provides an endpoint to call the GET method and access the skills.
- We have used the OAuth bearer token for securely accessing the API.
- Emsi API Endpoint:
<https://api.emsidata.com/apis/skills#versions-version-changes/meta>

Location of Data

Real-Time Database: For Storing User Authentication Details and resume details of users. We are going to a real-time database of firebase.





We are using a real-time database because whenever the user tries to log in to another device. They should be able to see the saved resumes. So, we are storing the data in a persistent location.

JSON

Some of the information like font sizes, colors, themes, and templates that are used for customizing the resume is stored in the files having .json as an extension.

Relevance of Data

Since We are getting keywords from EMSI API, data will be relevant since the API designers will update the data with the trending technologies at that point in time . We will be able to suggest the appropriate keywords that will be relevant.

Working Features

- Sign up and Sign In to the website
- Download Resume
- Saving the Resume
- Get User Data
- Skill Suggestion
- Chatbot for help

Sign up and Sign In to the website

- When the new user visits our application, the user needs to sign up for the website. We are storing the resume details in the firebase database. So, whenever the user signs in to the website the resumes that the user has created will be accessed.
- We are using firebase authentication with email and password enabled for Sign Up and Sign In.
- The working screens are as follows:

Sign Up Page

Resume Builder

SignupSignin


SignUp here

Email *


Password *

Re-enter Password *

Sign Up



Hello! How can I help you?



Sign In Page

Resume Builder


SignupSignin

Signin here


Email *

Password *

Sign In



Greetings! How can I assist?



The code snippet to handle the user signin and signup is as follows. Here we are using a context in order to handle the data related to the user throughout the application.

```
import React from "react";  
  
export const UserContext = React.createContext();  
|
```

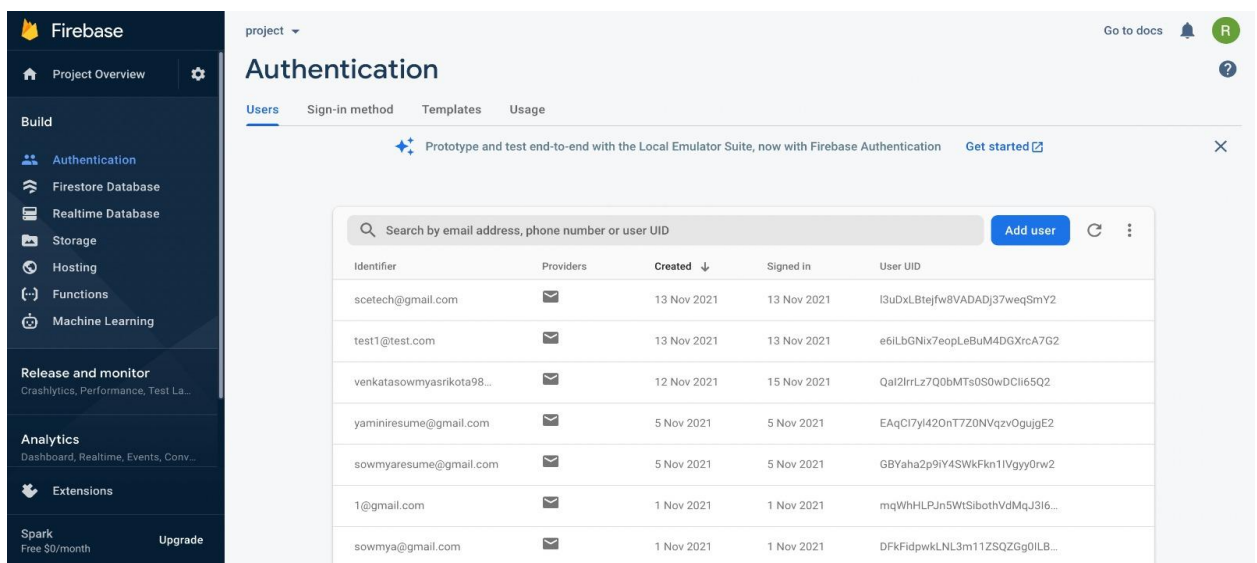
The context that has been created will be called in the application which requires the information related to the user and to update the user details using userContext functionality from React. Then we will create states for the email and password and any extra fields necessary and handle these fields with the state management.

```
const context = useContext(UserContext);  
  
const [email, setEmail] = useState("");  
const [password, setPassword] = useState("");  
const [reenteredPassword, setReenteredPassword] = useState("");
```

Firestore Authentication Integration

The image below shows all the signed-up users in the firestore. Each user will be assigned a unique ID namely UID.

Using this UID, we can distinguish between the users and save their resumes in our database

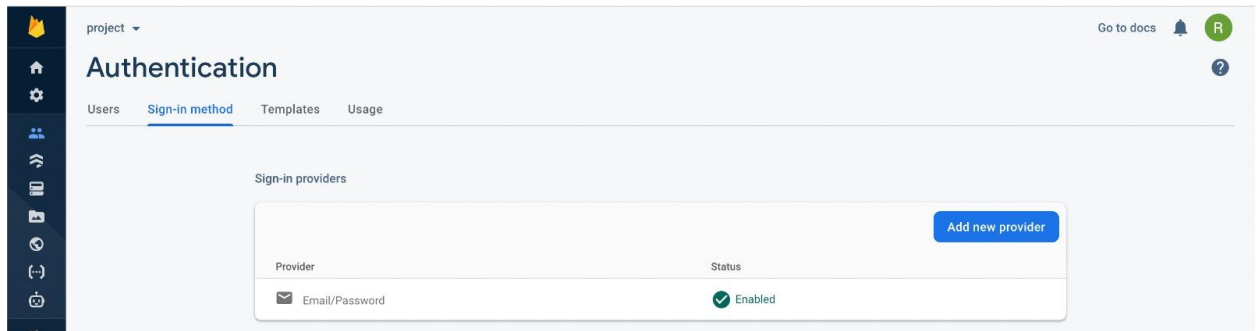


The screenshot displays the Firebase Authentication console. On the left is a sidebar with navigation options: Project Overview, Build (Authentication, Firestore Database, Realtime Database, Storage, Hosting, Functions, Machine Learning), Release and monitor, Analytics, Extensions, and Spark. The main area is titled 'Authentication' and includes tabs for Users, Sign-in method, Templates, and Usage. Below the tabs is a search bar and an 'Add user' button. A table lists the following users:

Identifier	Providers	Created ↓	Signed in	User UID
scetech@gmail.com	📧	13 Nov 2021	13 Nov 2021	I3uDxLBtejfw8VADADj37weqSmY2
test1@test.com	📧	13 Nov 2021	13 Nov 2021	e6iLbGNix7eopLeBuM4DGXrcA7G2
venkatasowmyasrikota98...	📧	12 Nov 2021	15 Nov 2021	QaI2IrrLz7Q0bMTs0S0wDCli65Q2
yaminiresume@gmail.com	📧	5 Nov 2021	5 Nov 2021	EAqCi7yl420nT7Z0NVqzvOgujgE2
sowmyaresume@gmail.com	📧	5 Nov 2021	5 Nov 2021	GBYaha2p9iY4SWkFkn1IVggy0rw2
1@gmail.com	📧	1 Nov 2021	1 Nov 2021	mQWhHLpJn5WtSibothVdMqJ3i6...
sowmya@gmail.com	📧	1 Nov 2021	1 Nov 2021	DFKFdpwKLNl3m11ZSQZGg0ILB...

Enabling Email and password for authentication

The below image shows that the authentication in firebase is email and password enabled.



Download Resume

- After signing into the application, the user needs to fill in all required details in each section as follows :
 - Personal Details, Education Details, Work Experience, Projects, Skills, Interests
- Separate screens are designed for each section.
- When the user clicks on the download resume button the resume is downloaded in PDF format.

The image displays the 'Resume Builder' application interface. At the top, a teal header bar contains the text 'Resume Builder' followed by the email 'venkatasowmyasrikota98@gmail.com' and a 'Logout' link. The main content area is titled 'Extra Details' and is divided into two sections: 'Skills/Languages' and 'Interest'. The 'Skills/Languages' section has a checked checkbox and six input fields labeled 'Skill 1' through 'Skill 6', containing the text 'Angular (Web Framework)', 'Angular Ui', 'Angular Material', 'Angular CLI', 'Angular Components', and 'Angular Reactive Forms' respectively. The 'Interest' section also has a checked checkbox and six empty input fields labeled 'Interest 1' through 'Interest 6'. Below these sections, there is a 'skill' dropdown menu currently showing 'angular', and a row of buttons: 'GET SKILL', '< BACK', and 'NEXT >'. At the bottom of the form, there are two buttons: 'DOWNLOAD RESUME' and 'SAVE DETAILS', both with download icons.

We are creating an endpoint **/create-pdf** using express, this will be called by the client web app to create the resume pdf. We are using have created a htmlTemplate

and the req.body will be the props.values which we stored after the user filled in details. The resume pdf will be stored in Resume.pdf file.

```
// POST route for PDF generation....
app.post("/create-pdf", (req, res) => {
  pdf.create(pdfTemplate(req.body), options).toFile("Resume.pdf", (err) => {
    if (err) {
      console.log(err);
      res.send(Promise.reject());
    } else res.send(Promise.resolve());
  });
});
```

The below code snippet shows a part of htmlTemplate that we have designed to generate a resume template. The values are dynamically assigned when a request is made with this template. We can just change the CSS properties and HTML alignment and generate different templates.

```
htmlTemplate += `
  </div>
  <div class="rela-block content-container">
    <div class="rela-block caps greyed">Education</div>
    <h3 class="mb-0">${college}</h3>
    <p class="text-muted light mt-0 mb-1">${fromyear1}<span class="mx-2">to</span>
    <p class="justified mt-0 mb-1" style="font-size: 17px;">${qualification1}</p>
    <p class="justified mt-0 mb-3" style="font-size: 17px;">${description1}</p>
```

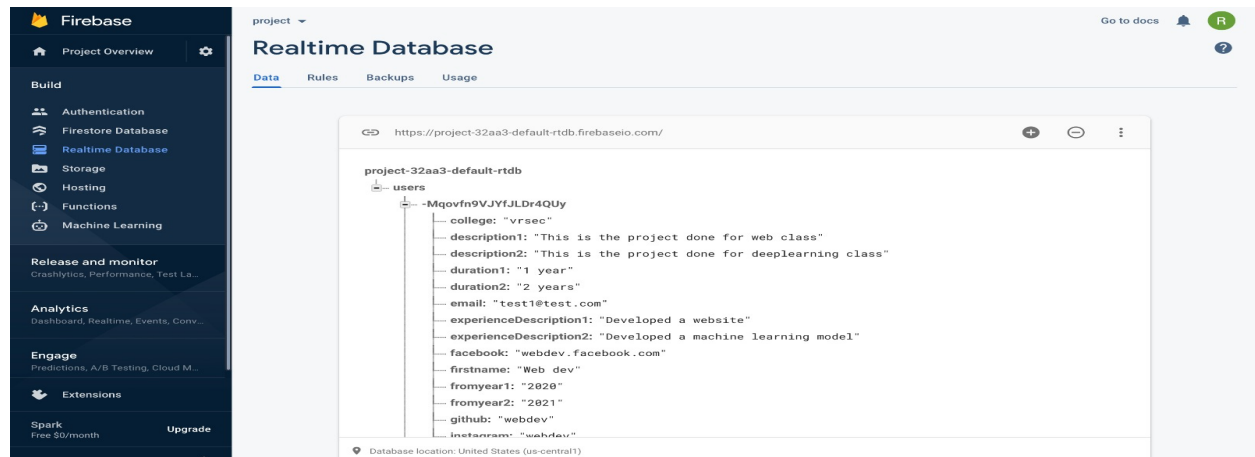
The Sample Resume after the download is as follows



Save Resume

The user can save their data by clicking on the save details button which is next to the download resume button, when the user clicks on the save details button then all the values entered by the user in all the fields will be stored in the database.

We are storing the data in the database because we can auto-populate the same data whenever the user needs it.



The code snippet below is written to save the database to the firebase. Here we are using the fetch method for saving the details to the database asynchronously. Since we are sending the data the Method will be POST as the request type. We will be storing the data in **json** format since firebase stores in that format. props.values will have all the data filled up by the user, so we will assign that to the body of the request. The endpoint is given to the real-time database.

```
66  async saveToDB() {
67      const response = await fetch(
68          "https://project-32aa3-default-rtdb.firebaseio.com/users.json",
69          {
70              method: "POST",
71              body: JSON.stringify(this.props.values),
72              headers: {
73                  "Content-Type": "application/json",
74              },
75          }
76      );
77      const data = response.json();
78      console.log(data);
79  }
```

Get User Data:

The user can click on the get details button as shown on the screen below. The user has to enter their email id before getting the data because we will identify the user data based on their email id, based on that we can retrieve the user data and fill all the fields of the resume.

This feature really helps the user to get the details and edit them instead of filling up all the details from scratch.

Resume Builder sowmyakota07@gmail.com Logout

Personal Details

First Name *
VENKATA SOWMYASRI

Last Name *
KOTA

Email *
sowmyakota07@gmail.com

Phone Number
9134619389

Your Website
abc

GitHub

Linked In

Twitter

Facebook

Instagram

< BACK

NEXT >

Page 1

GET DETAILS

The below code snippet shows the `getFromDB` method implementation. This method is called inorder to get the user data from the database. The user needs to provide the email inorder to get the data from the database. The endpoint is given as a real-time firebase database. We will check for the email, if it gets matched then we will assign that data to the fields of the resume.

```
getFromDB() {  
  fetch("https://project-32aa3-default-rtdb.firebaseio.com/users.json")  
    .then((response) => {  
      return response.json();  
    })  
    .then((data) => {  
      //data.filter(data.email === mail)  
      const keyValues = Object.keys(data);  
      for (let i = 0; i < keyValues.length; i++) {  
        const key = String(keyValues[i]);  
        console.log(key);  
        if (data[key].email === this.props.values.email) {  
          console.log(data[key]);  
        }  
      }  
    })  
  }  
}
```

Skill Suggestion

- When the user fills the required skill section in the application, the user can choose to customize their own skill or can get the skill suggestion from our application.
- When given the technology in the get skill field, the skills are suggested related to that technology.
- Emsi API Endpoint: We are using Emsi skills API for Skill suggestions.

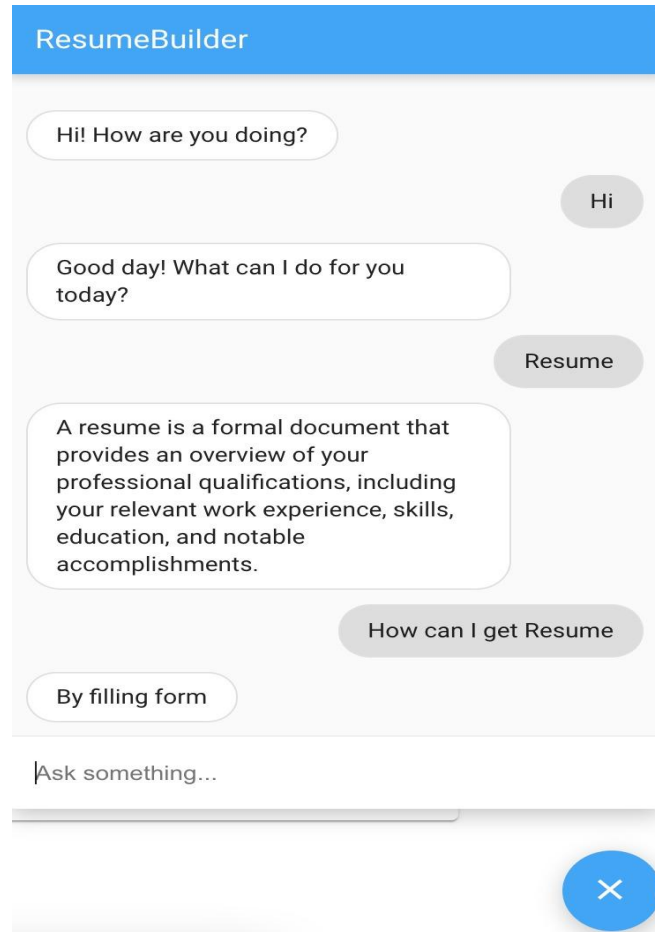
<https://api.emsidata.com/apis/skills#versions-version-changes/meta>

```
101 |  
102 | const getData = async (query) => {  
103 |   await axios  
104 |     .get(  
105 |       'https://emsiservices.com/skills/versions/latest/skills?q=${query}',  
106 |       {  
107 |         headers: {  
108 |           Authorization:  
109 |             'Bearer eyJhbGciOiJSUzI1NiIsImtpZCI6IjNDNjZCRjIzZmJhZmY4RDQ2QzJERDhCMjI0MEVGMTFENTZlZkY3MUY1LCJ0eXAiOiJKV1QiLCJ4NXQiOiJ...' ,  
110 |         },  
111 |       },  
112 |     )  
113 |     .then((data) => {  
114 |       console.log(data.data);  
115 |       const skills = data.data;  
116 |       values.skill1 = skills.data[0].name;  
117 |       values.skill2 = skills.data[1].name;  
118 |       values.skill3 = skills.data[2].name;  
119 |       values.skill4 = skills.data[3].name;  
120 |       values.skill5 = skills.data[4].name;  
121 |       values.skill6 = skills.data[5].name;  
122 |       this.forceUpdate();  
123 |     })  
124 |     .catch((err) => alert(err));  
125 | };
```

- The above code snippet will help us to get the skills from the API, as stated above we can use the get method provided by EMSI API
- We are using Axios for making the API request asynchronously
- We are passing a query as the parameter to the request, the value of the query will be the technology that the user enters.
- We are using bearer tokens for Authorization and securely accessing the API
- After getting the skills data we are assigned to the skills fields. So this data will be populated on the webpage.

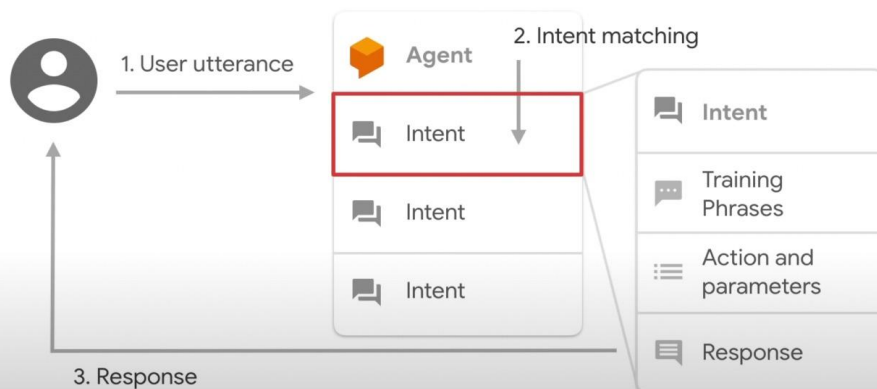
Chatbot for help

- We have integrated a chatbot that can be accessed throughout the application.
- This chatbot is small talk enabled. i.e, the bot can answer basic conversation-related questions.
- Users can ask questions related to the application, the bot can help the user by answering their questions.
- The UI related to the chatbot is as follows:



- We are using **Dialogflow** to design and develop the chatbot
- The input to the chatbot is provided by the user. Based on that the intent gets matched with the training phase, action, and parameters that we have trained the bot with, and the bot will provide the response.
- We are training the bot using **NLP Techniques** that can generalize well with the new questions.

Intent - Response



The code snippet related to integrating chatbot is as follows

Here we are integrating the chatbot using predefined ScriptTag from react library. We are providing source as the URL that is provided by the Dialogflow and the code snippet provided by the Dialogflow.

```
client > src > components > ChatBot.js > ChatBot
1  import React from 'react';
2  import ScriptTag from 'react-script-tag';
3  const ChatBot = () => {
4    return(
5      <div>
6        <ScriptTag src="https://www.gstatic.com/dialogflow-console/fast/messenger/bootstrap.js?v=1"></ScriptTag>
7        <df-messenger
8          chat-icon="https://www.gstatic.com/dialogflow-console/fast/messenger/bootstrap.js?v=1"
9          resume-examples-1024x731-landscape.png"
10         intent="WELCOME"
11         chat-title="ResumeBuilder"
12         agent-id="504905e1-9fa5-44fc-a043-b2146f1b0b1c"
13         language-code="en"
14       ></df-messenger>
15     </div>
16   )
17 }
18 export default ChatBot;
```

Tech Stack:

Node js

We have used Node js for back-end technology and ExpressJs for routing with the server and API calls.

React

We have used react for front-end website design. For Styling, We are using Material UI.

Firebase

We have used firebase for authentication and Database. In the firebase, we will be using the firestore database.

Dialog Flow

Designed and developed chatbot using Google Dialog Flow

Challenges:

- Emsi Api uses JsonWebToken as auth token. While accessing it, JWT is a new concept to us. Understanding JWT and saving it as a cookie has been a challenge for us.

- It took some time for us to design an API to download the resume in pdf format.
- Making the chatbot available throughout the application.

Work Sharing

Sign up/Sign In - Kota Venkata Ramyasri

Firestore Configuration for Authentication - Kolla Yamini

Saving resume- Kolla Yamini and Gopu Venkata Srinivasa Baba

Get User Details - Kota Venkata Ramyasri and Kota Venkata Sowmyasri

Resume Screens - All Team Members

UX/UI Design - Gopu Venkata Srinivasa Baba

Integrating Skill suggestion API - Kota Venkata Sowmyasri

Designing, Developing and Integrating Chat Bot - Kota Venkata Ramyasri

Server Side Code - Kota Venkata Sowmyasri and Gopu Venkata Srinivasa Baba

Future Work:

- We are going to provide multiple resume templates.
- We are going to provide skill recommendations based on the job role for the user in the application.
- We can set the payment gateway for some of the templates and provide some templates as free templates.

GithubLink: https://github.com/kvsowmyasri/ResumeBuilder_WebDevTeam

VideoLink:

<https://pro.panopto.com/Panopto/Pages/Viewer.aspx?tid=148cffce-2763-46a2-bf1f-adfe001d316b&start=0>

PresentationLink:

<https://docs.google.com/presentation/d/1p-7mqLKyYfLolb7DXwyMq9zLJs9WnxASR5C1hH4EOAg/edit?usp=sharing>



References

[1] <https://cloud.google.com/dialogflow/docs>

[2] <https://api.emsidata.com/apis/skills#versions-version-changes/meta>

[3] <https://reactjs.org/docs/getting-started.html>

[4] <https://firebase.google.com/docs>