Project report on

ReStat Library

By

Siddhi Gavande [2020510023]

Pranali Ghag [2020510024]

Under the guidance of

Prof. Nikhita Mangaonkar

Department of Master of Computer Applications Sardar Patel Institute of Technology

Autonomous Institute Affiliated to Mumbai University 2021-22

CERTIFICATE OF APPROVAL

This is to certify that the following students.

Siddhi Gavande [2020510023]

Pranali Ghag [2020510024]

Have satisfactorily carried out work on the project entitled.

“**ReStat Library**”.

Towards the fulfillment of the project,

as laid down by Sardar Patel Institute of Technology

during the year 2021-2022.

Project Guide 1	Project Guide 2
(Dr. Pooja Raundale)	**(Prof. Nikhita Mangaonkar)**

PROJECT APPROVAL CERTIFICATE

This is to certify that the following students

Siddhi Gavande [2020510023]

Pranali Ghag [2020510024]

Have successfully completed the Project report on "**ReStat Library**", which is found to be satisfactory and is approved

At

SARDAR PATEL INSTITUTE OF TECHNOLOGY, ANDHERI (W), MUMBAI.

`

INTERNAL EXAMINER	EXTERNAL EXAMINER

`

Head of Department	Principal
(Dr. Pooja Raundale)	**(Dr. B. N. Chaudhari)**

INDEX

Abstract:

A wide number of research papers are published every year. These research papers are written by various researchers, faculty members of different institutes, students, etc. There are a lot of websites available where all such papers can be published. In case of one particular institute, the records of research papers published by their faculties & students must be maintained. All these records include data like published date, publication, author, citations, etc. To maintain all this details at one place is a tedious task. So in order to solve this problem we have come up with an idea of creating a website which can help create profiles of faculties and maintain their research paper records.

The list of figure and list of tables should be as given below:

1. INTRODUCTION

1.1 Problem Definition:

There are many publications websites where various research papers and journals are published. But there is no particular library wherein every published research papers, journals, books can be seen in which we can get the publication details of a particular research paper, its conference details, article details, etc. If an institute (for eg. S.P.I.T) wants to maintain such records there is no one stop destination for all research papers and journals where we can also maintain a researcher's profile, a platform where other researchers can search other researcher's profiles and papers.

1.2 Objective and Scope

1.2.1 Objective

- To fetch faculties (users) profile data and store it for reference.
- To fetch all published papers and journals of one user from google scholar API.
- Search function to search about different authors & papers on particular topics.
- Calculate citations, total papers of users and store for admin and user reference.

1.2.2 Scope

In this stage by using this website the faculty users can register and all their research paper , citation related data can be fetched and displayed in their profile. The total number of papers and citations per year data is also fetched.

In the later stage there can also be the feature where published papers can be segregated according to their domains, type of papers, papers per year etc,

1.3 System Requirements

1.3.1 Hardware Requirements:

Processor	Dual-Core i3 or above
Ram	8 GB or above
Storage	20 GB Hard-disk space and above

Table 1.3.1: Hardware Requirements

1.3.2 Software Requirements

Operating System	OS Independent
Frontend	ReactJs, NodeJs
Database	Firebase
Tools Used	Git for collaborative working, Visual Studio Code
API Used	SerpAPI https://serpapi.com/google-scholar-api

Table 1.3.2: Software Requirements

2. LITERATURE SURVEY

Sr.No	Name	Unique Feature
1.	Google Scholar https://scholar.google.com/	Search all scholarly literature from one convenient place. Explore related works, citations, authors, and publications.
2.	ResearchGate https://www.researchgate.net/	ResearchGate's primary feature is the individual researcher profile, which is used to promote scholarly production. The site creates profiles with information harvested from literature databases and other sources, while permitting researchers to create profiles by registering on the site. You can receive updates about recent publications by colleagues.
3.	Academia https://www. academiapu blishing.org/	Academia Publishing is an open access, peer-reviewed journals that facilitate a cost effective access to the quality research findings in the domains of engineering, medical science, scientific research, biological sciences, agricultural sciences, etc.

4.	Microsoft Academic https://microsoft.academia.edu/	Microsoft Academic Operated by the company that brings you Word, PowerPoint and Excel, Microsoft Academic is a reliable, comprehensive research tool. The search engine pulls content from over 120 million publications, including scientific papers, conferences and journals. You can search directly by topic, or you can search by an extensive list of fields of study.
5.	Science.gov http://Science.gov	Science.gov is operated and maintained by the Office of Science and Technical Information, the same department that collaborates on WorldWideScience.org. This search engine pulls from over 60 databases, over 2,200 websites and 200 million pages of journals, documents and scientific data. Search results can be filtered by author, date, topic and format (text or multimedia).

Table 2: Literature Survey

3. SOFTWARE REQUIREMENT SPECIFICATION [SRS] AND DESIGN

3.1 Introduction

The Restat website is useful for SPIT faculties to keep track of there published papers on different sites. It provides users authentication using google and gathering all there published paper data available on the internet in one place.The data such as total papers published, citations, etc is gathered.

The admin can view all the users data and delete a particular user's account if required.The users and admin can search for others profiles.

3.2 Overall Description

3.2.1 Product Perspective

The website is an open source website where faculty users can register, it is checked whether they belong to spit.ac.in domain, and if yes then their profile details are fetched from Google Scholar API and stored in firebase database. All publication related data is stored and the admin can view all data of all users registered to the system. Admin can delete any account and users can delete their particular restart account if needed. Both admin and users

3.2.2 Product Functions

The basic function of the website is that it allows the faculty user to register to the website. All the relevant data related to that user is fetched from the API only after checking whether the user’s email ID is verified at spit.ac.in. The published papers and journals of individual users are displayed in the library section. There is a search engine option available on dashboard for users to search on different topics research papers or research papers related to different authors. This feature is available to admin also. The admin has the visibility to all user accounts and can view, delete and use all users data for reporting purpose.

3.2.3 User Characteristics

User:

The people who want to search papers and add their article i.e. Research papers & Journal and update their profiles are the users of this website.

The website can be used by many of the users at a time. When the users visit the website, they can view the details of an article and can also search for research papers of other authors or different topics. The data of the articles, publications details and its links will be updated automatically.

Admin:

The person who is viewing and maintaining all the users that are the faculties and their data are the Admin of the website. They can login and check for the data that various users/faculties have made or have been automatically updated and maintain it accordingly. The Admin can then also get some analytical data in terms of tables where in the number of papers and citations can be seen of a particular faculty.

3.2.4 Dependencies And Assumptions

The system is dependent on the internet. Since the backend is done using Firebase so there is a need for the internet here as well. There is one dependency that CORS policy blocks the API data to get fetched while searching for research papers or Authors for which we have to open disabled web security There are no other known dependencies or assumptions of this system. Rest dependencies are subject to minimum hardware requirements to run this application.

3.3 System Features

3.3.1 Functional Requirements

1] Login and Registration: The application requires a User i.e. a Faculty to login either using Login with Google option else they can create a new account and register oneself. This is applicable for all Faculties.

2] Searching Research papers: Faculties can search for all the research papers which are already published and can view all the related data and links to the research papers or journals.

3] View All Users: Admin can view all Users who have registered and logged in.

4] Profile: Faculties can maintain their profile wherein they can update the data in their profiles which will then get reflected to the Admin.

5] Library: The logged in user can view all the research papers/ journals published by him/her here.

3.3.2 Non- Functional Requirements

1] Usability

- Usability is the degree to which a software can be used by specified users to achieve quantified objectives with effectiveness, efficiency, and satisfaction in a quantified context of use.
- The application can be easily used by the user and the admin.

2] Flexibility

- As soon as the customer requests the feasible research paper/journal article or updates any data of themselves, the request is quickly reflected on the Admin's account.
- When the admin logs in into the website all the users and their data can be seen.

3] Maintainability

- This website is easy to maintain.

4] Timeliness

- The website performs all the operations in less amount of time

5] Reliability

- Validations are done properly for admin registration and login.

  Avoid incorrect storage of record.

4. PROJECT ANALYSIS AND DESIGN

4.1 Methodologies Accepted

Methodology involves dividing software development work into distinct stages and coming up with tasks or activities aimed at achieving better planning and time management. It is considered a trivial part of the systems development life cycle.

Iterative Model:

The iterative model is a particular implementation of a software development life cycle (SDLC) that focuses on an initial, simplified implementation, which then progressively gains more complexity and a broader feature set until the final system is complete.

In this Model, you can start with some of the software specifications and develop the first version of the software. After the first version if there is a need to change the software, then a new version of the software is created with a new iteration. Every release of the Iterative Model finishes in an exact and fixed period that is called iteration.

The Iterative Model allows the accessing earlier phases, in which the variations made respectively. The final output of the project renewed at the end of the Software Development Life Cycle (SDLC) process.

While creating this website in the first iteration we focused on creating the simple modules like static pages and other later iterations, register , login. We focused on creating the complex modules such as databased connections, data fetching, displaying data from API and firebase in second iteration. And then as modules were completed they were evaluated and again if any glitches were there planning was done to solve those and the process went on.

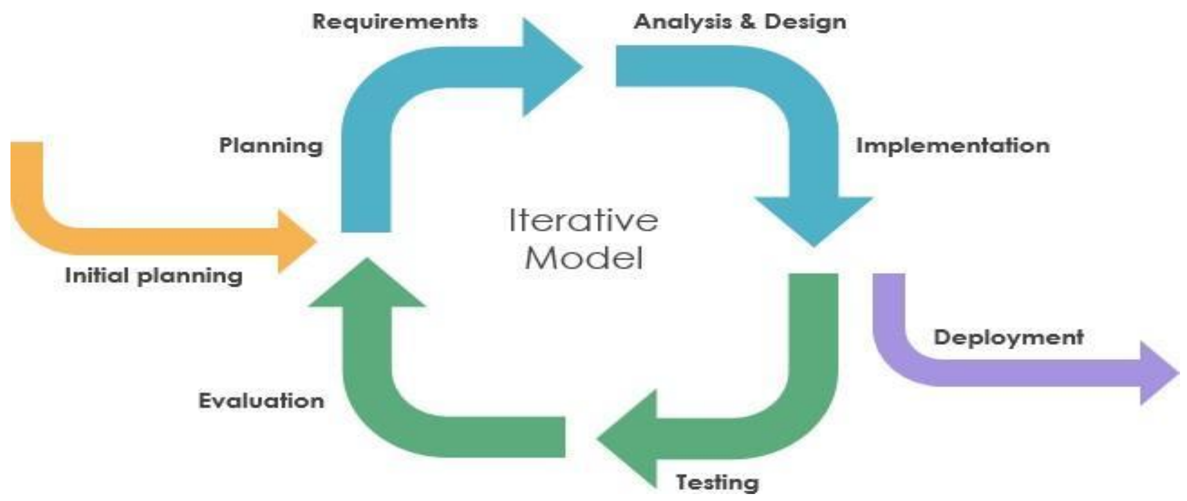


FIG 4.1 Iterative Model The Happy Plate

4.2 Architectural Design

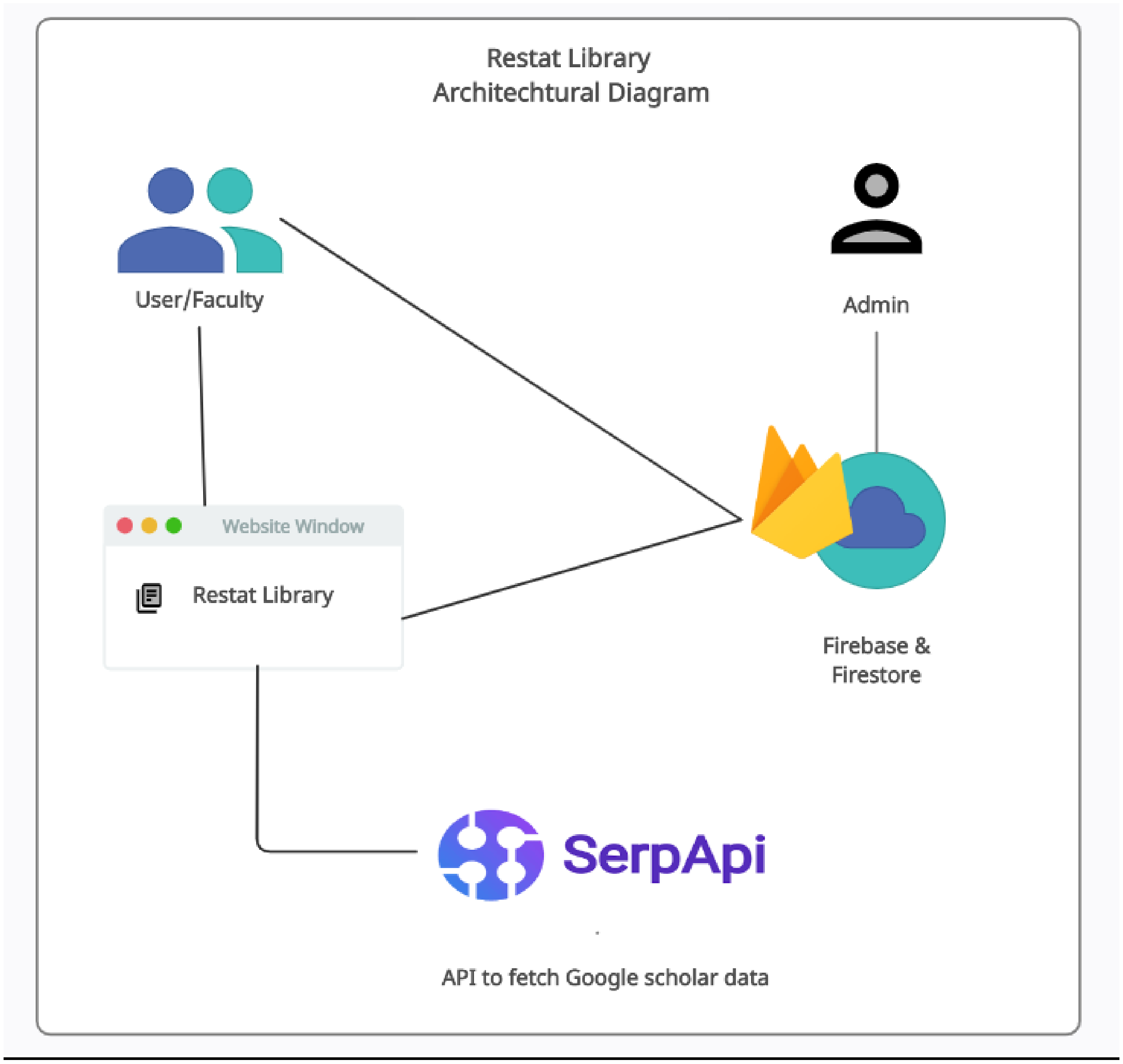


FIG 4.2 Architectural Design: Restat Librar

4.3 Diagrams

4.3.1 Use Case Diagram

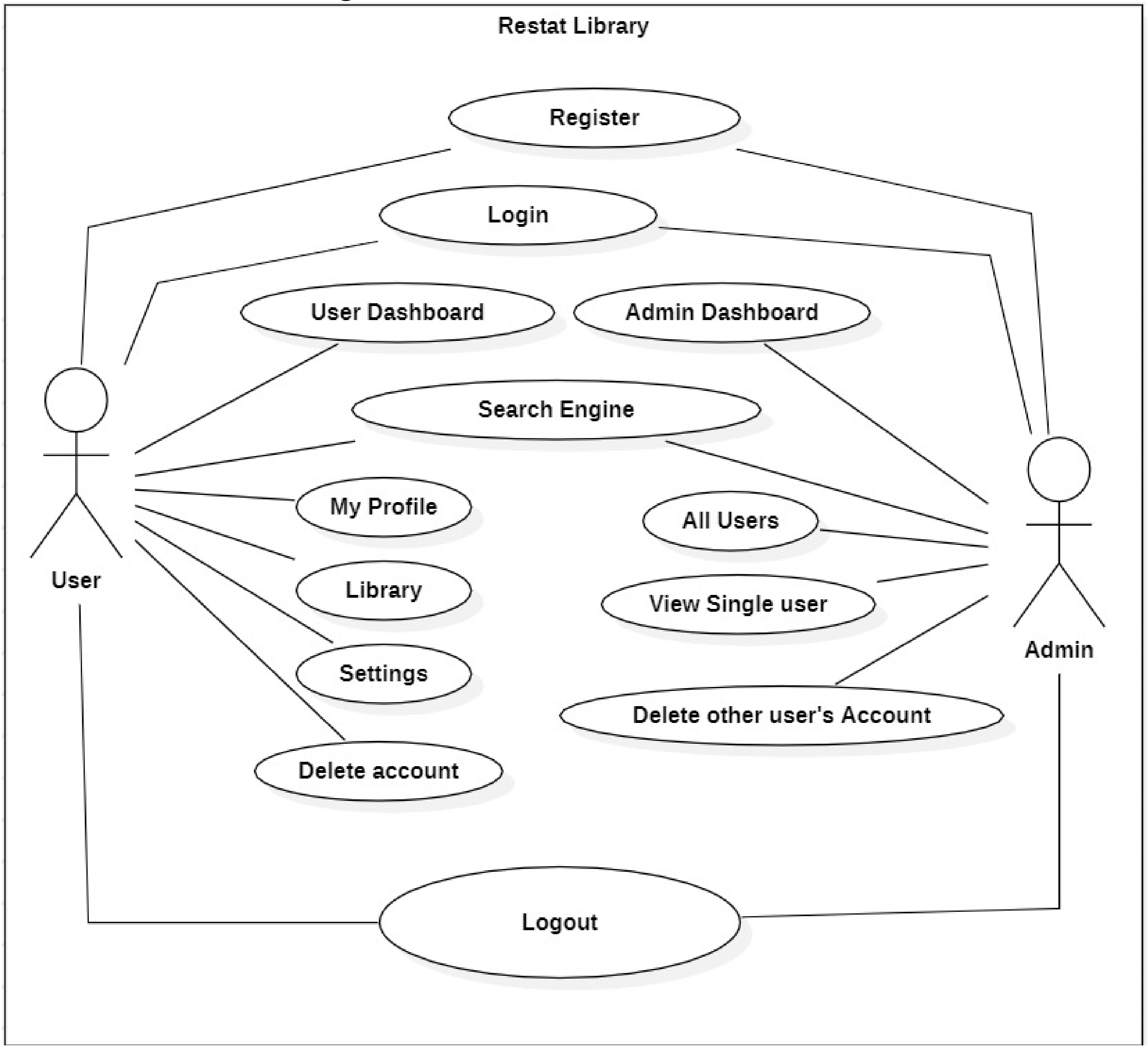


FIG 4.1 Use Case Diagram: Restart Library the use cases of the system

4.3.2. Activity Diagram

- User/Faculty

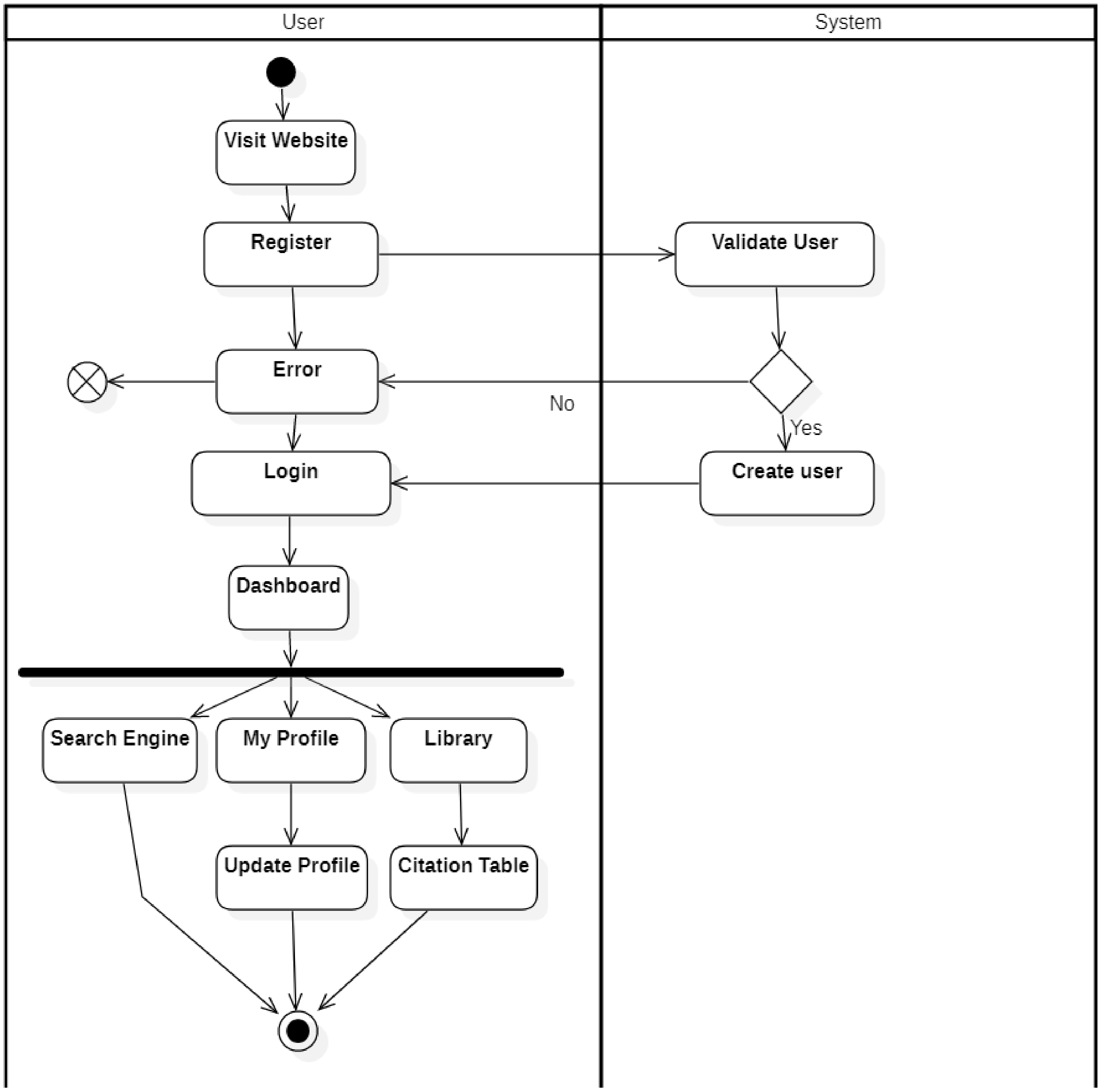
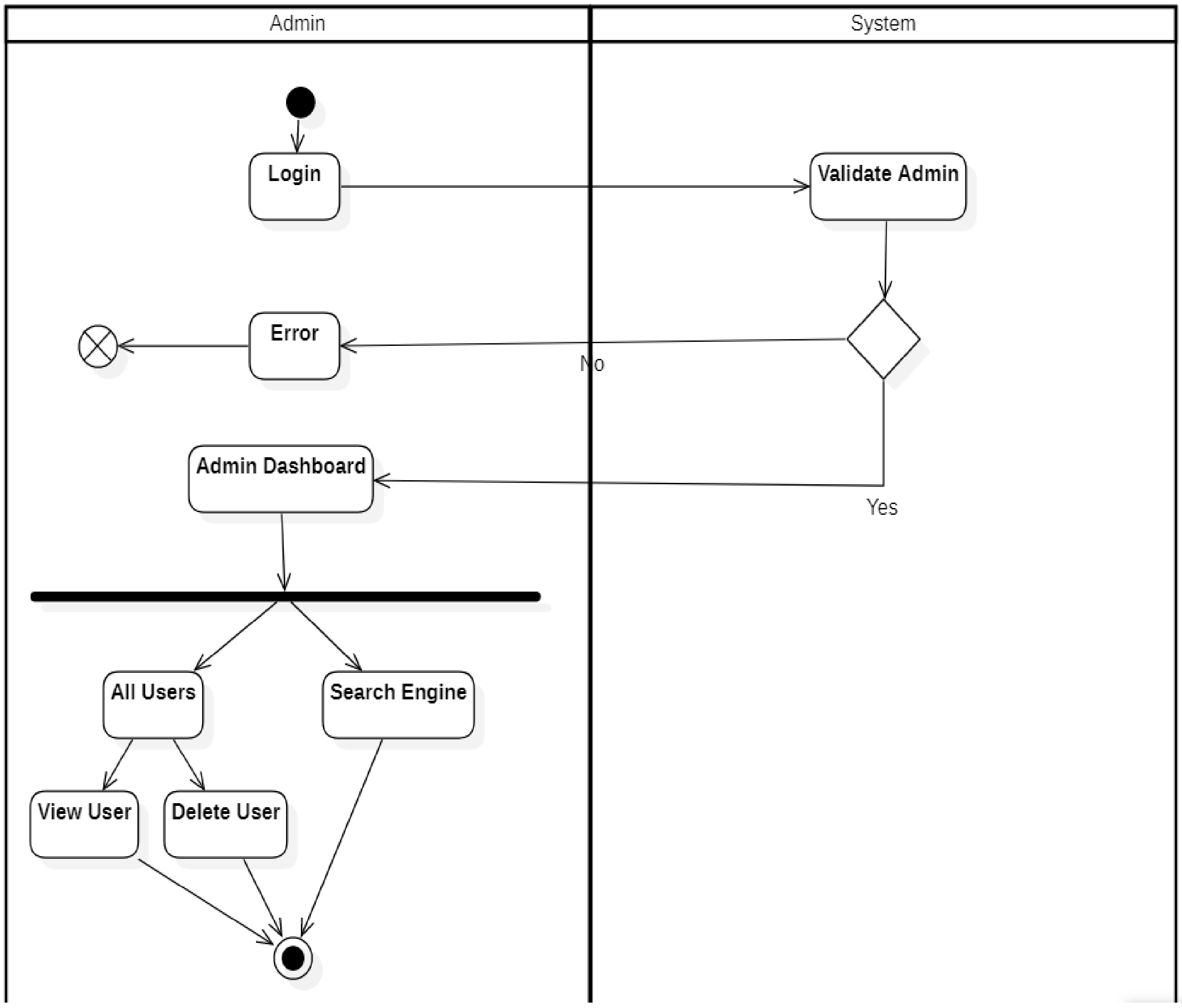


FIG 4.3.2(i) Activity Diagram Resat: Depicts the flow of User activities

- Admin



<u>FIG 4. 3.2(ii)Activity Diagram Restat Library: Depicts the flow of Admin activities</u>

4.3.3 Deployment Diagram

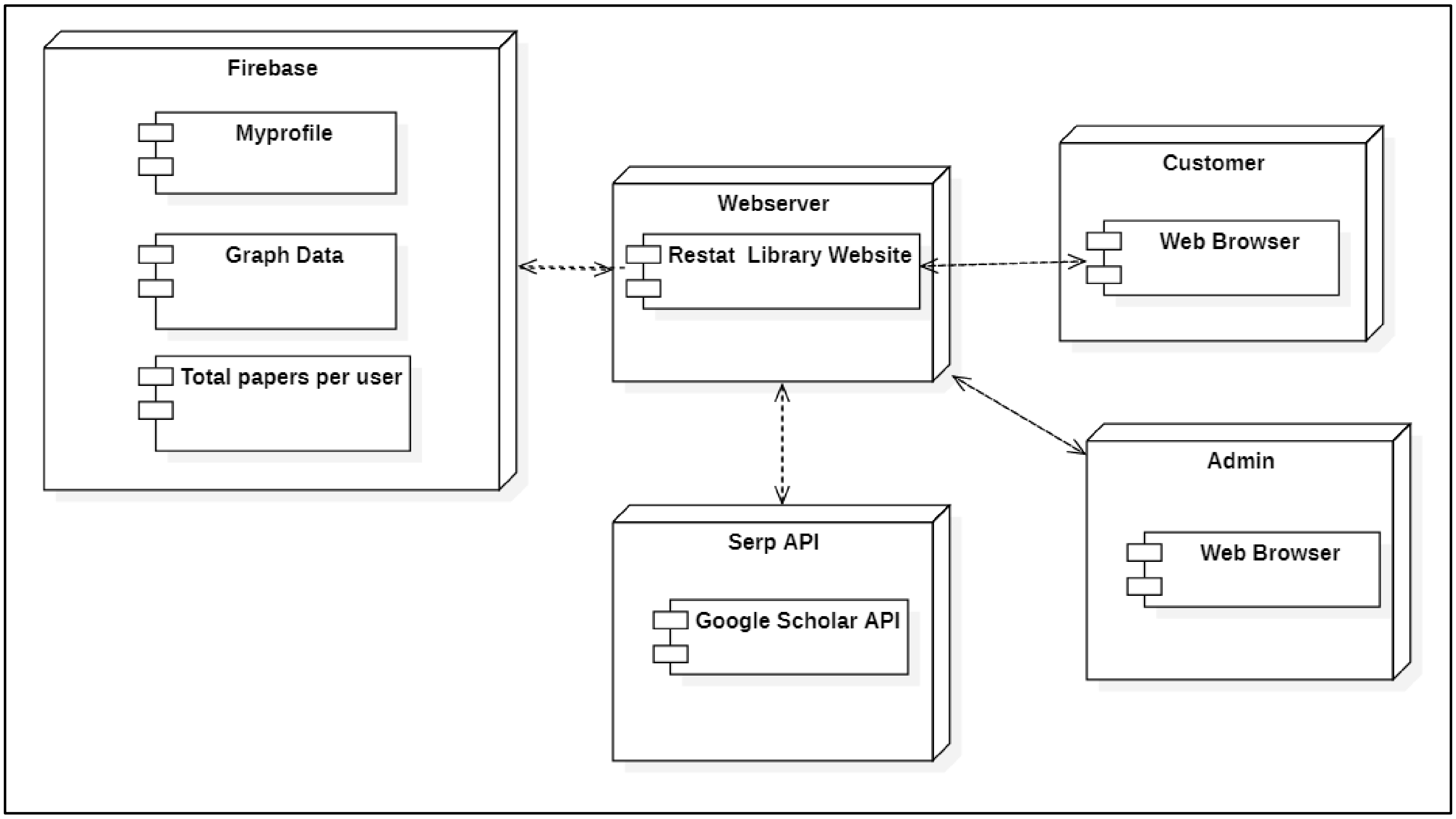


FIG 4.3.3 Deployment Diagram Restat Library

4.4 WORK BREAKDOWN STRUCTURE

4.4.1 Gantt Chart

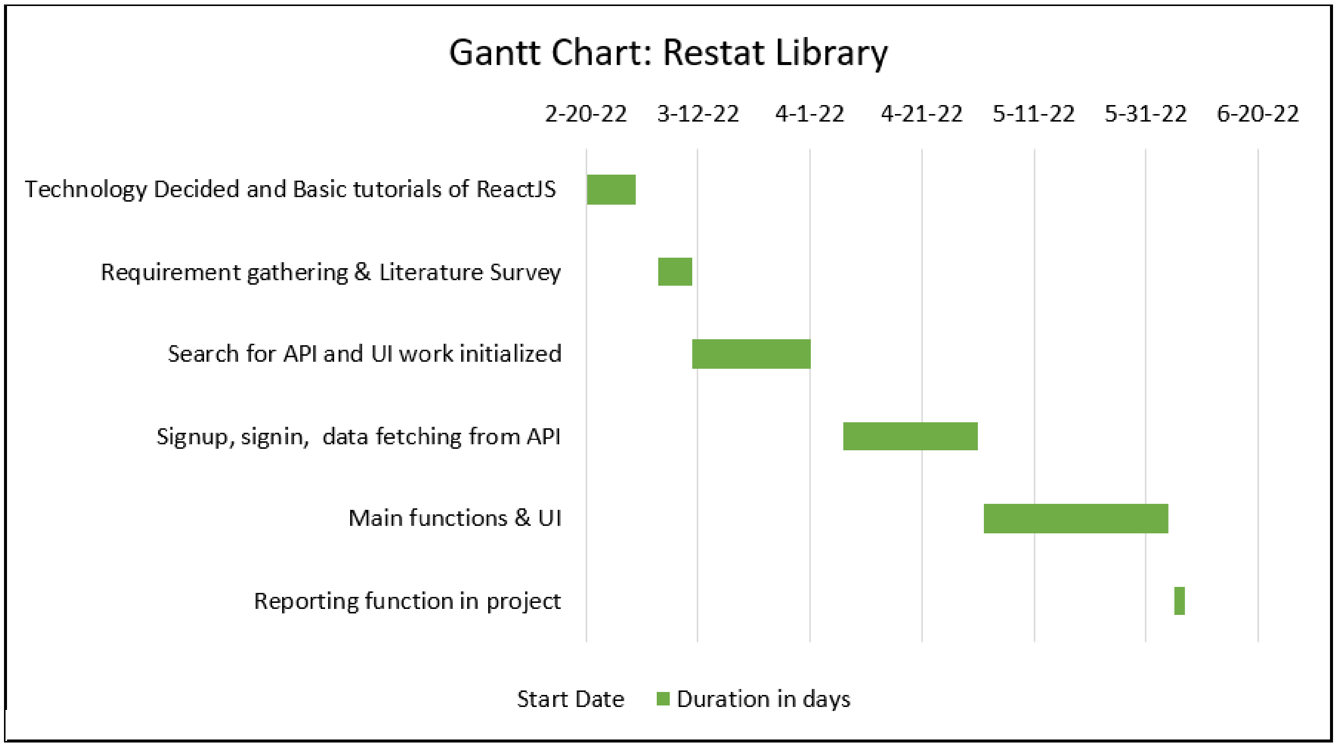


FIG 4.4.1 Gantt Chart The Happy Plate: Depicts the work breakdown Structure

5. PROJECT IMPLEMENTATION AND TESTING

5.1 Code

USERS:

- **Dashboard.js**

```
import React, { Component, useState, useEffect } from 'react';
import { AppContent, AppSidebar, AppFooter, AppHeader } from
'src/components/index'
import { cilMagnifyingGlass } from '@coreui/icons';
import { getStyle, hexToRgba } from '@coreui/utils'
import CIcon from '@coreui/icons-react'
import './dashboard.css'

const Dashboard = () => {
  const [user, setUser] = useState([])
  const [pub_sum, setPubsum] = useState([])
  const [cite, setCite] = useState([])
  const [resr, setResr] = useState([])
  const [loading, setLoading] = useState(false);
  const [count, setCount] = useState(0);
  const [s, setS] = useState("")

  const fetchData = () => {

    const url =
`https://serpapi.com/search.json?engine=google_scholar&q=${s}&hl=en&num=20
&api_key=452c8d62a3109b7126306267dc125e951050339de3b68057816fdd0893fbd2f9`
;

    fetch(url)
      .then(response => {
        return response.json();

      }).then(data => {
         setUser(data.organic_results);
         console.log(data.organic_results);
      })
      .catch(err => {
         console.log(err)
      })
  }
  const fetchData2 = () => {
    const url2 =
`https://serpapi.com/search.json?engine=google_scholar&q=${s}&hl=en&start=
20&api_key=452c8d62a3109b7126306267dc125e951050339de3b68057816fdd0893fbd2f
9`;
    fetch(url2)
      .then(response => {
        return response.json();
      }).then(data => {
         setUser(data.organic_results);
        setCite(data.organic_results.inline_links.cited_by);
        setResr(data.organic_results.resources[0]);
      })
      .catch(err => {
         console.log(err)
      })
  }
  return (
    <>
      <div>
        <AppSidebar />
        <div className="wrapper d-flex flex-column min-vh-100 bg-light">
          <AppHeader />
          {
            <div className="p-4 box">
              <h2 className="mb-3"
style={{padding:'10px'}}><b>Dashboard</b></h2>

              <div style={{padding:'10px'}}>
                <div className="form-outline" id="same-line">
                  <input type="search" id="search" className="form-
```

```
control" placeholder="Search Article or Name here..." onChange={(e) =>
{setS(e.target.value)}} aria-label="Search" />
                  <button className="search-button" type="submit"
id="submit" name="search-go" onClick={fetchData} >
                    <CIcon icon={cilMagnifyingGlass} customClassName="nav-
icon" style={{ height: '25px', width: '25px' }}/>
                  </button>
                </div>
              </div>

              <div style={{padding:'10px'}} className="mainData">
              <h3 style={{padding:'10px'}}>{s}</h3>
                {user.map((item, index) =>
                    <div key={index}>
                    <p >
                    <hr/>
                    <a href= {item.link}> <h5>{item.title}</h5></a>
                     <h6>{item.publication_info.summary}</h6>
                     <h7>{item.snippet}</h7><br/>
                     {item.resources !== undefined ? <div>
                      {item.resources.map((d, index) => <div key={index}>
                          <p>
                          &nbsp;<div
style={{color:'blue'}}>[{d.file_format}] <a href=
{d.link}>{d.title}</a></div>
                          </p>
                        </div>
                      )}
                    </div> : null}
                    {item.inline_links.cited_by != undefined ? <div>
                      &nbsp;<b>Cited By:
</b>{item.inline_links.cited_by.total}
                    </div> : null}
                     </p>
                   </div>
                )}
              </div>
                <button className="next-button" type="submit" id="submit"
```

```
name="next" onClick={fetchData2}>Next...</button>
            </div>
          }
          <AppFooter />
        </div>
      </div>
    </>
  )}
export default Dashboard
```

- **libraryapi.js**

```
import React, { Component, useState, useEffect } from 'react'
import { AppContent, AppSidebar, AppFooter, AppHeader } from
'src/components/index'
import Table from 'react-bootstrap/Table'
import { DataGrid } from '@mui/x-data-grid';
import { useUserAuth } from 'src/context/UserAuthContext'
import { firestore } from 'src/firebase';
import { addDoc, collection, getDocs, setDoc, doc, getDoc, onSnapshot,
updateDoc } from "firebase/firestore"
import './library.css'

const Libraryapi = () => {

  const { user } = useUserAuth();
  const uid = user.uid
  const n = user.displayName
  const usersCollectionRef = collection(firestore, "myprofile")
  const [users, setUsers] = useState([]);
  const [papers, setPapers] = useState([]);
  const [profile, setProfile] = useState([]);
  const [citedBy, setCitedBy] = useState([]);
  const [authid, setAuthid] = useState(" ");
  const [totalc, setTotalc] = useState(0);
  const [loading, setLoading] = useState(false);
  var totalPapers = 0;
  useEffect(() => {
    const getUsers = async () => {
      const data = await getDocs(usersCollectionRef)
      setUsers(data.docs.map((doc) => ({ ...doc.data(), id: doc.id })))
      // setU(data.docs.map((doc) => ({ ...doc.data(), id: doc.id})))
    }

    getUsers();
  }, []);
  useEffect(() => {
    users.map(item => {
      if (user.uid === item.id) {
        setAuthid(item.AuthorID);
      }
      console.log(user.uid);
      console.log(authid);
    }
    )
  }, []);


  useEffect(() => {
    setLoading(true);
    if (authid !== " ") {
      Fetchpapers()
    }
  }, [authid])

  const Fetchpapers = () => {

    console.log(authid);

    const url =
`https://serpapi.com/search.json?engine=google_scholar_author&author_id=${
authid}&hl=en&api_key=452c8d62a3109b7126306267dc125e951050339de3b68057816f
dd0893fbd2f9`;
    console.log(url);
    fetch(url)
```

```
      .then(response => {
        return response.json();
      }).then(data => {
        console.log(data);
        setPapers(data.articles);
        setProfile(data.author);
        setCitedBy(data.cited_by.graph)
        //  console.log(papers);
      })
      .catch(err => {
        console.log(err)
      })
      .finally(() => {
        setLoading(false);
      });
    }
      const countOfPapers = async () => {
        await setDoc(doc(firestore, "Total Paper per User", uid),
          {
            Name: n,
            Count: totalPapers
          })
      }
      const graphData = async () => {
        await setDoc(doc(firestore, "Graph Data", uid),
          {
            Year_18: citedBy[0].citations,
            Year_19: citedBy[1].citations,
            Year_20: citedBy[2].citations,
            Year_21: citedBy[3].citations,
            Year_22: citedBy[4].citations,
          })
      }
    return (
      <div>
     <AppSidebar />
                <div className="wrapper d-flex flex-column min-vh-100 bg-
light">
```

```
                    <AppHeader />
                     <div className="wrap" style={{padding:'20px'}}>
                      <div className="inline">
                      <img className='style' src={ user.photoURL }
referrerPolicy="no-referrer"/>
                      <div className='profile'>
                        <h4>{profile.name}</h4>
                        <h6>{profile.affiliations}</h6>
                        <h6>{profile.email}</h6><br/>
                      </div>
                    </div>
                    <div >
                  </div>
                  {
                    papers.map((item, index) =>
                      <div key={index}>
                        {console.log(totalPapers = totalPapers + 1)}
                        <hr />
                        <a href={item.link}> <h5>{item.title}</h5></a>
                        <h6>Authors: {item.authors}</h6>
                        <h6>Publication: {item.publication}</h6>
                        {item.cited_by !== undefined ? <div>
                          <h6>Cited By: {item.cited_by.value}</h6>
                        </div> : null}
                        <h6>Year: {item.year}</h6>
                      </div>
                    )
                  }
                  <br/><br/>
                  <p onClick={countOfPapers}>All Papers</p>

                  <b><p style={{marginLeft:'500px'}}
onClick={graphData}>GraphData</p></b>
                    {
                      citedBy.map((i, index) =>
                        <div className="stylemap" key={index}>
                          <table className='abc'>
                            <tr>
```

```
                              <td>Year</td>
                              <td>{i.year}</td>
                            </tr>
                            <tr>
                            <td>Citations</td>
                              <td>{i.citations}</td>
                            </tr>
                          </table>
                        </div>
                      )
                    }

                    </div>
                    <AppFooter />
                </div>
                </div>
    );
  }
export default Libraryapi;
```

- **myprofile.js**

```
import React, { Component, useState, useEffect } from 'react';
import PropTypes from 'prop-types';
import { Form, Button } from "react-bootstrap";
import DefaultLayout from 'src/layout/DefaultLayout';
import { useUserAuth } from '../../context/UserAuthContext'
import { AppContent, AppSidebar, AppFooter, AppHeader } from
'src/components/index'
import { toast } from 'react-toastify';
import { Link, useNavigate } from "react-router-dom";
import app from 'src/firebase'
import { firestore } from 'src/firebase';
import { addDoc, collection, getDocs, setDoc, doc, getDoc, onSnapshot }
from "firebase/firestore"
const Myprofile = () => {
    var count = 0;
    const [users, setUsers] = useState([]);
    const [emailstat, setEmailStat] = useState("")
    const [authorid, setAuthorID] = useState("")
    const [citiedby, setCitedBy] = useState("")
    const [newAffiliation, setNewAffiliation] = useState({})
    const [newAOI, setNewAOI] = useState([])

    const { user } = useUserAuth();
    const id = user.uid;
    const n = user.displayName;
    const e = user.email;
    const pic = user.photoURL
    let navigate = useNavigate();
    //const docSnap = DocumentSnapshot<>
    const usersCollectionRef = collection(firestore, "myprofile")
    // const userDoc = doc(firestore, "myprofile", id);
    // const docSnap = getDoc(userDoc);
    const verify = "Verified email at spit.ac.in"
    var num = 0
    var lenarr = 0
    const createMyProfile = async () => {
        fetchData();
        await setDoc(doc(firestore, "myprofile", id),
            {
                Name: n,
                Affiliation: newAffiliation,
                Email: e,
                EmailStatus: emailstat,
                AOI: newAOI,
                AuthorID: authorid,
                CitiedBy: citiedby,
                photo: pic
            })
    }
    const q="";
    const fetchData = () => {
        const url =
`https://serpapi.com/search.json?engine=google_scholar_profiles&mauthors=$
```

```
{n}&hl=en&api_key=452c8d62a3109b7126306267dc125e951050339de3b68057816fdd08
93fbd2f9`;
        fetch(url)
            .then(response => {

                return response.json();

            }).then((data) => {

                const xyz = data.profiles;
                lenarr = xyz.length;
                //console.log(lenarr);
                for (var i = 0; i <= lenarr; i++) {
                    //console.log(data.profiles[i].email)
                    if (data.profiles[i].email === verify) {
                        setNewAffiliation(data.profiles[i].affiliations);
                        setEmailStat(data.profiles[i].email);
                        setNewAOI(data.profiles[i].interests);
                        setAuthorID(data.profiles[i].author_id);
                        setCitedBy(data.profiles[i].cited_by);
                    }
                    else {
                        console.log("not spit account!!");
                        //num = num + 1
                    }
                }

            })
            .catch(err => {
                console.log(err)
            })

    }



    useEffect(() => {
        const getUsers = async () => {
            const data = await getDocs(usersCollectionRef)
            setUsers(data.docs.map((doc) => ({ ...doc.data(), id: doc.id
}))))
        }

        getUsers();
    }, []);

    return (
        <>
            <div>
                <AppSidebar />
                <div className="wrapper d-flex flex-column min-vh-100 bg-
light">
                    <AppHeader />
                    {
                        users.map((u) => {
                            if (user.uid === u.id) {
                                return (
                                    <div key={u.id} className="p-4 box">
                           {/* <h2 className="mb-3"> { u.Name }</h2> */}
               <h2 className="mb-3">My Profile details</h2>
               <Form>
               <Form.Group className="mb-3" controlId="formName">
           <Form.Label>Name</Form.Label>
                 <Form.Control type="text" value={u.Name} disabled />
               </Form.Group>

               <Form.Group className="mb-3" controlId="formAffiliation">
               <Form.Label>Affiliation</Form.Label>
               <Form.Control type="text" value={u.Affiliation} disabled />
          </Form.Group>

              <Form.Group className="mb-2" controlId="formEmail">
                <Form.Label>Email address</Form.Label>
                <Form.Control type="email" value={u.Email} disabled />
              </Form.Group>
              <Button variant="primary" type="submit" onClick={() =>
navigate("/updateprofile")}>Update
              </Button>
                                        </Form>
                                    </div>
                                );
                            }
                            else {
                                count = count + 1;
                            }

                        })
                    }
                    {users.map((u) => {
                        if (count === users.length) {
                            createMyProfile();
                            //fetchData();
                            return (
                                <div key={u.id} className="p-4 box">

                                    <h2 className="mb-3">My Profile
details</h2>
      <Form>
       <Form.Group className="mb-3" controlId="formName">
           <Form.Label>Name</Form.Label>
           <Form.Control type="text" value={u.Name} disabled />
       </Form.Group>

       <Form.Group className="mb-3" controlId="formAffiliation">
           <Form.Label>Affiliation</Form.Label>
           <Form.Control type="text" value={u.Affiliation} disabled />
       </Form.Group>

      <Form.Group className="mb-2" controlId="formEmail">
           <Form.Label>Email address</Form.Label>
           <Form.Control type="email" value={u.Email} disabled />
      </Form.Group>
      <Button variant="primary" type="submit" onClick={() =>
navigate("/updateprofile")}>Update
      </Button>
                                    </Form>
                                </div>

                            );

                        }

                    })
                    }
                    <AppFooter />
                </div>
            </div>
        </>
    );
}

export default Myprofile;
```

- **Settings.js**

```
import React, { Component, useState, useEffect } from 'react'
import Table from 'react-bootstrap/Table'
import { DataGrid } from '@mui/x-data-grid';
import { useUserAuth } from 'src/context/UserAuthContext';
import { AppContent, AppSidebar, AppFooter, AppHeader } from
'src/components/index'
import { firestore } from 'src/firebase';
import { Link, useNavigate } from "react-router-dom";
import { getAuth, deleteUser } from "firebase/auth";
import { addDoc, collection, getDocs, setDoc, doc, deleteDoc} from
"firebase/firestore"
import './settings.css'
import { async } from '@firebase/util';
```

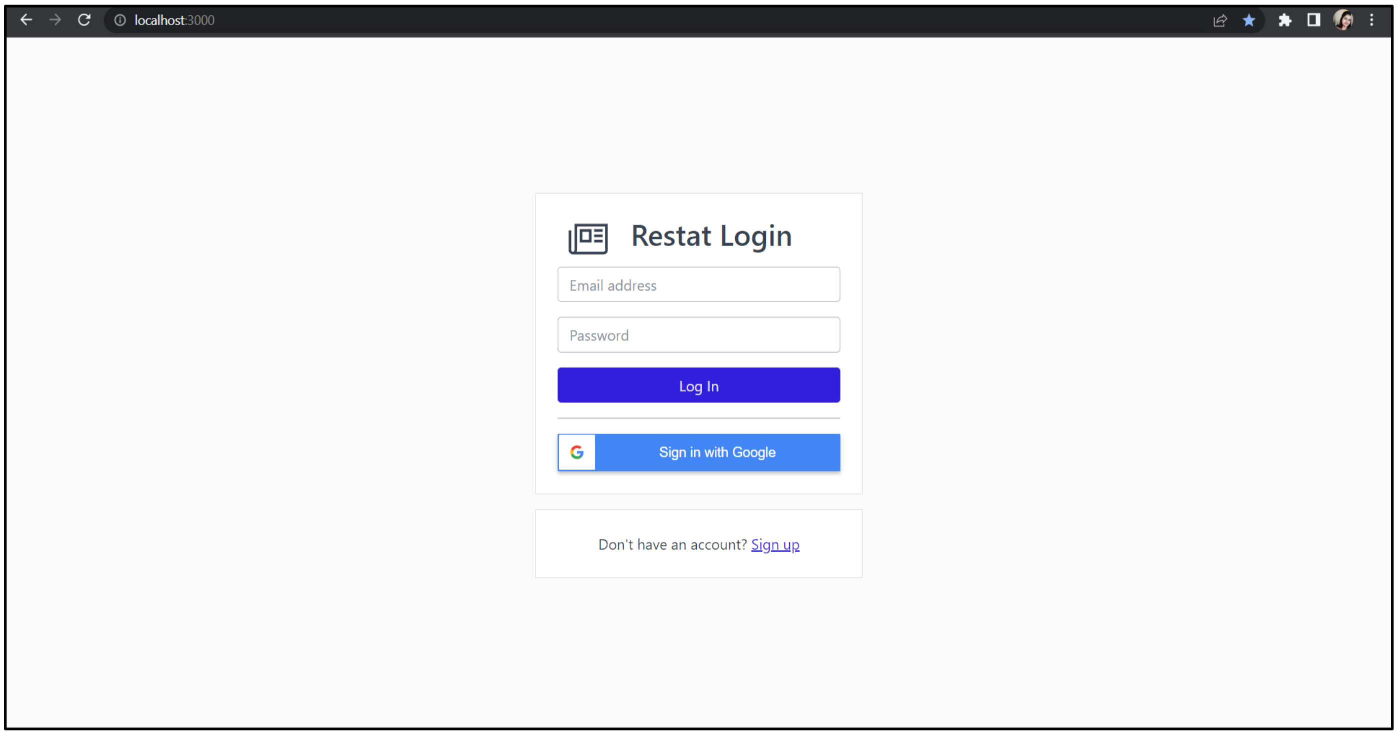
```
const Settings = () => {


  let navigate = useNavigate();

  const { user } = useUserAuth();
  const e = user.email;
  const id = user.uid;

  const delUser = async () => {
    await deleteDoc(doc(firestore, "myprofile", id));
    deleteUser(user).then(() => {
      alert('Successfully deleted user');
      navigate("/");
    }).catch((error) => {
      console.log('Error deleting user:', error);
    });

  }

  return (
      <div>
        <AppSidebar />
                <div className="wrapper d-flex flex-column min-vh-100 bg-
light">
                    <AppHeader />
                    <div style={{padding:'25px'}}>
                    <h2> Settings</h2><br/>
                    <div style={{padding:'10px'}}>
                    <h6>You are currently signed in as: <b
style={{color:'blue'}}><i>{e}</i></b></h6>
                    <br/><br/>
                    <b><i>Language </i>  </b>
                    <br/><br/>
                    Current Language: <b
style={{color:'blue'}}>English</b>
                    <br/><br/><br/>
                    <b><i>Account Settings</i></b>
                    <br/><br/>
                    <button className="setbutton" type="submit"
id="submit" name="next" onClick={() => navigate("/logout")}>Sign
Out</button><br/><br/>
                    <button className="setbutton" type="submit"
id="submit" name="next" onClick={ delUser }>Delete Restat Account</button>

                    </div>
                    </div>
                    <AppFooter />
                </div>
      </div>

  );
}




export default Settings;
```

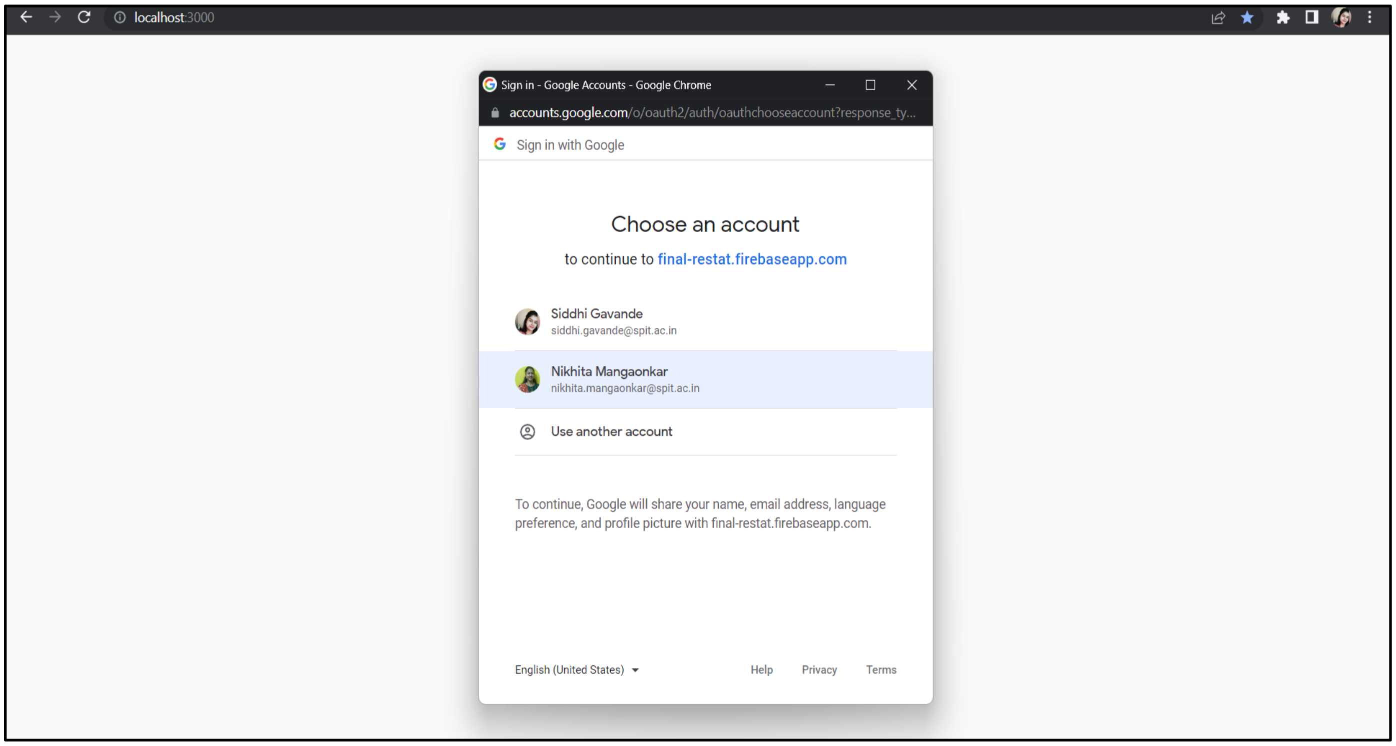
ADMIN:

● allUsers.js

```
import React, { Component, useState, useEffect } from 'react'
import {AppFooter, AppHeader, AppHeaderAdmin } from 'src/components/index'
import { AppSidebarAdmin } from 'src/components/index'
import Table from 'react-bootstrap/Table'
import { useUserAuth } from "src/context/UserAuthContext";
import { firestore } from 'src/firebase';
import { getAuth, deleteUser } from "firebase/auth";
import CIcon from '@coreui/icons-react'
import { addDoc, collection, doc, getDocs, updateDoc, runTransaction,
setDoc, deleteDoc } from "firebase/firestore"
import { cilPen, cilInfo, cilTrash } from '@coreui/icons'
import { useNavigate } from 'react-router-dom';

const AllUsers = () => {

  let navigate = useNavigate()
    const { user } = useUserAuth();
    const [users, setUsers] = useState([]);
    const usersCollectionRef = collection(firestore, "myprofile")

    useEffect(() => {
        const getUsers = async () => {
            const data = await getDocs(usersCollectionRef)
            setUsers(data.docs.map((doc) => ({ ...doc.data(), id: doc.id
})))
        }

        getUsers();
    }, []);

  return (
      <div>
        <AppSidebarAdmin />
                <div className="wrapper d-flex flex-column min-vh-100 ">
                    <AppHeaderAdmin />
                    <h2>All Users</h2>
                    <Table  >
                    <thead >
                        <tr>
                          <th><center>Sr. No</center></th>
                          <th><center>Name</center></th>
                          <th><center>Email</center></th>
                          <th><center>Affiliation</center></th>
                          <th><center>Citied By</center></th>
                          <th><center>Action</center></th>
                        </tr>
                      </thead>
                      <tbody>
                        {users.map((u,index) => (
                         <tr key={ index }>
                          <td scope='row'><center>{index +
1}</center></td>
                          <td>
                          <center>
                              <img style={{borderRadius: '50%', height:
'40px'}} src={ u.photo } referrerPolicy="no-referrer"/>
                              &nbsp;&nbsp; &nbsp;&nbsp;
                              {u.Name}
                            </center></td>
                          <td><center>{u.Email}</center></td>
                          <td><center>{u.Affiliation}</center></td>
                          <td><center>{u.CitiedBy}</center></td>
                          <td>
                            <center>
                              <button style={{background: 'none',
border:'none', height:'40px',width:'40px'}} variant="success"
type="submit" onClick={() => navigate(`/viewUser/${u.id}`)} ><CIcon
icon={cilInfo} /></button>&nbsp;&nbsp;
                              {/* <button  type="submit" onClick={
delUser(u.id) }><CIcon icon={cilTrash} /></button> */}
                            </center>
                          </td> </tr>))} </tbody> </Table> <AppFooter />
                </div></div>

 );
}

export default AllUsers
```

5.2 Screen Shots





localhost:3000

Sign in - Google Accounts - Google Chrome

accounts.google.com/o/oauth2/auth/oauthchooseaccount?response_ty...

Sign in with Google

Choose an account

to continue to final-restat.firebaseapp.com

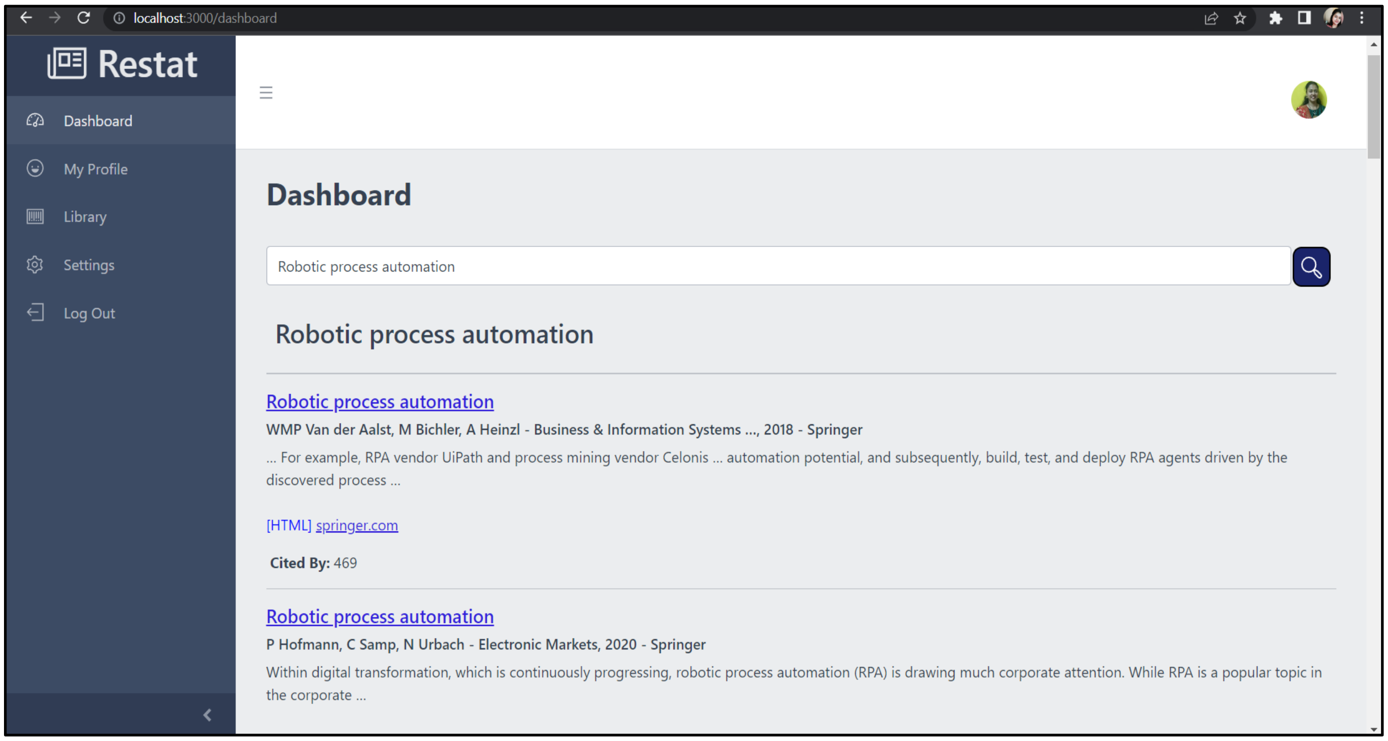
Siddhi Gavande
siddhi.gavande@spit.ac.in

Nikhita Mangaonkar
nikhita.mangaonkar@spit.ac.in

Use another account

To continue, Google will share your name, email address, language preference, and profile picture with final-restat.firebaseapp.com.

English (United States) Help Privacy Terms



localhost:3000/dashboard

Dashboard

Robotic process automation

Robotic process automation

Robotic process automation

WMP Van der Aalst, M Bichler, A Heinzl - Business & Information Systems ..., 2018 - Springer

... For example, RPA vendor UiPath and process mining vendor Celonis ... automation potential, and subsequently, build, test, and deploy RPA agents driven by the discovered process ...

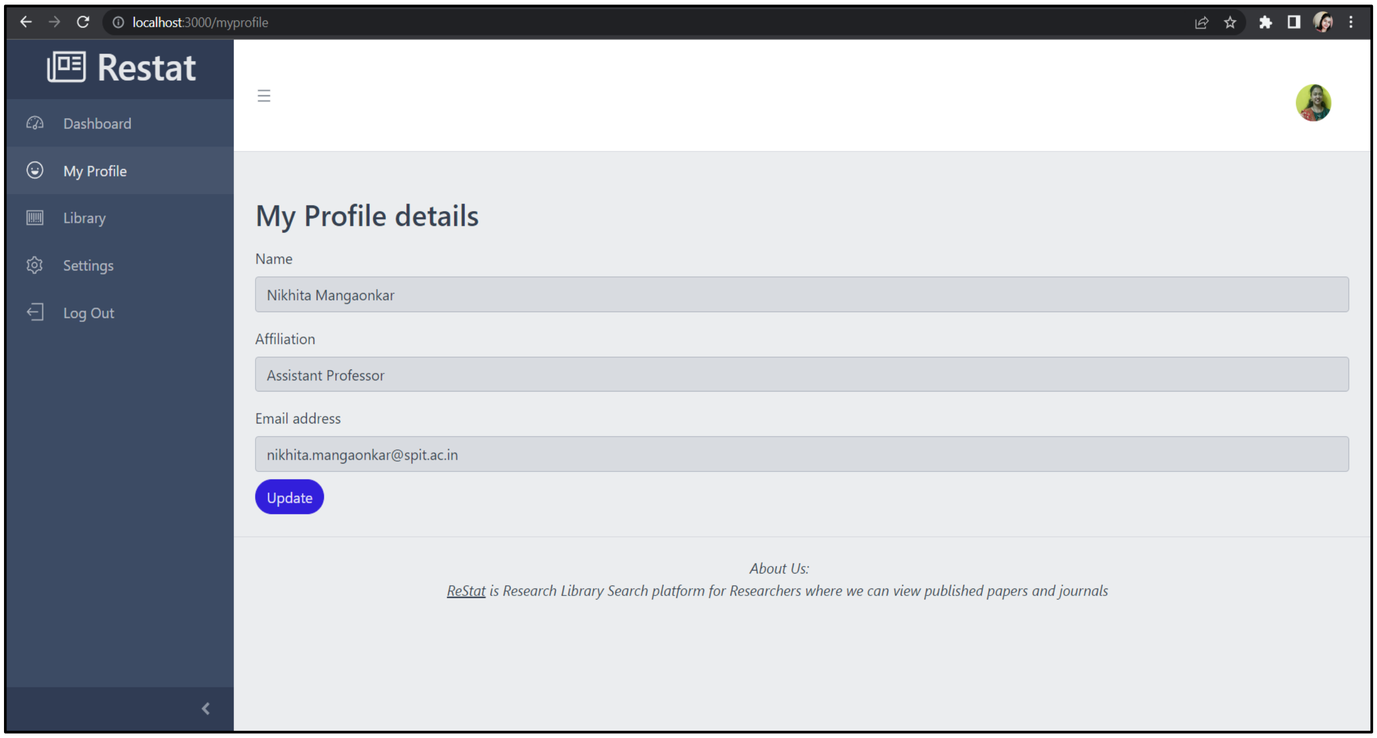
[HTML] springer.com

Cited By: 469

Robotic process automation

P Hofmann, C Samp, N Urbach - Electronic Markets, 2020 - Springer

Within digital transformation, which is continuously progressing, robotic process automation (RPA) is drawing much corporate attention. While RPA is a popular topic in the corporate ...



My Profile details

Name

Nikhita Mangaonkar

Affiliation

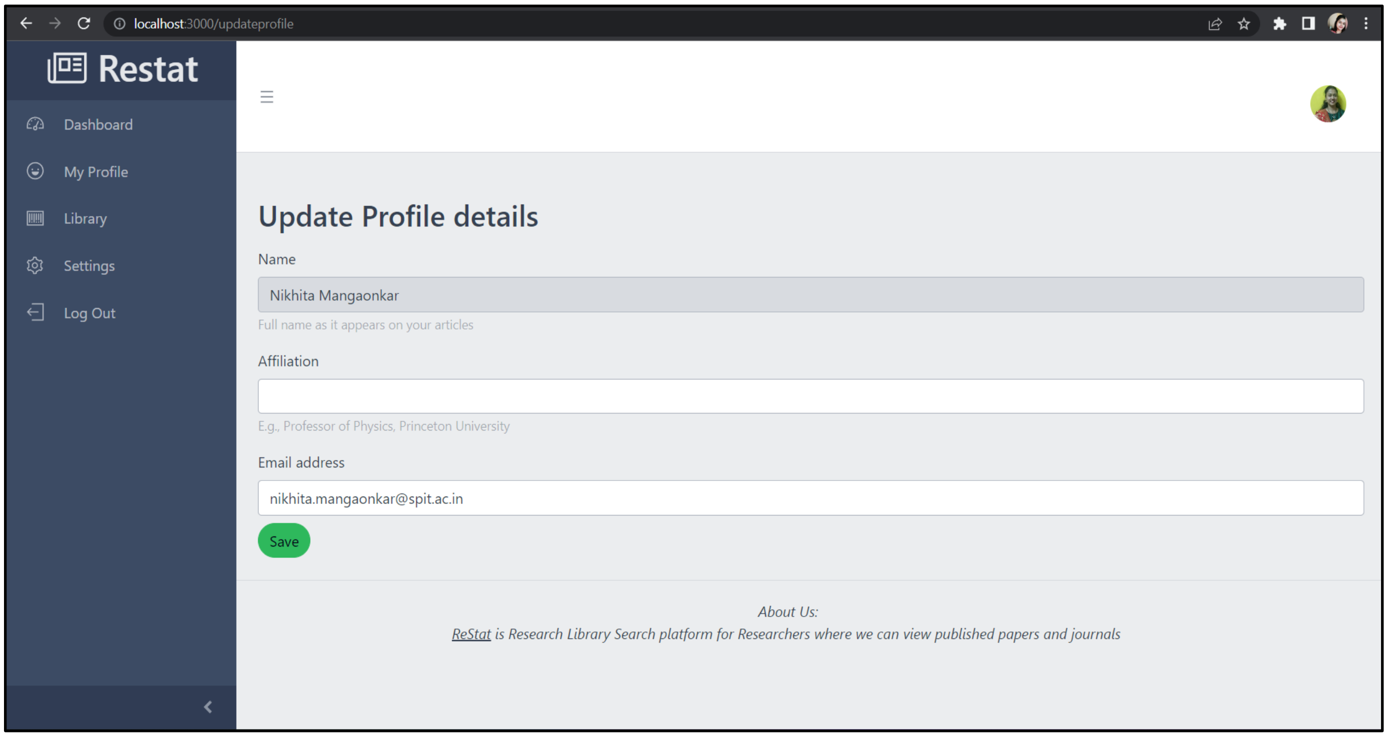
Assistant Professor

Email address

nikhita.mangaonkar@spit.ac.in

About Us:

ReStat is Research Library Search platform for Researchers where we can view published papers and journals



Update Profile details

Name

Nikhita Mangaonkar

Full name as it appears on your articles

Affiliation

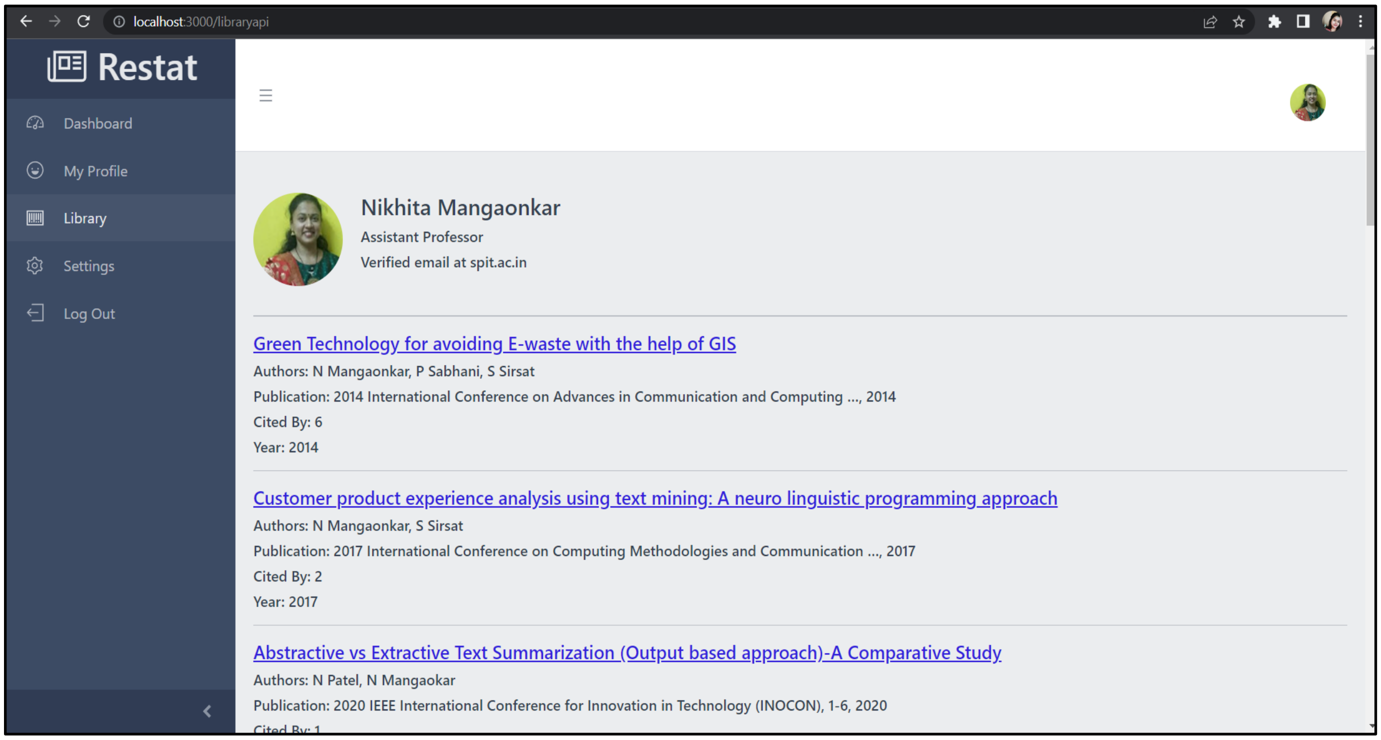
E.g., Professor of Physics, Princeton University

Email address

nikhita.mangaonkar@spit.ac.in

About Us:

ReStat is Research Library Search platform for Researchers where we can view published papers and journals

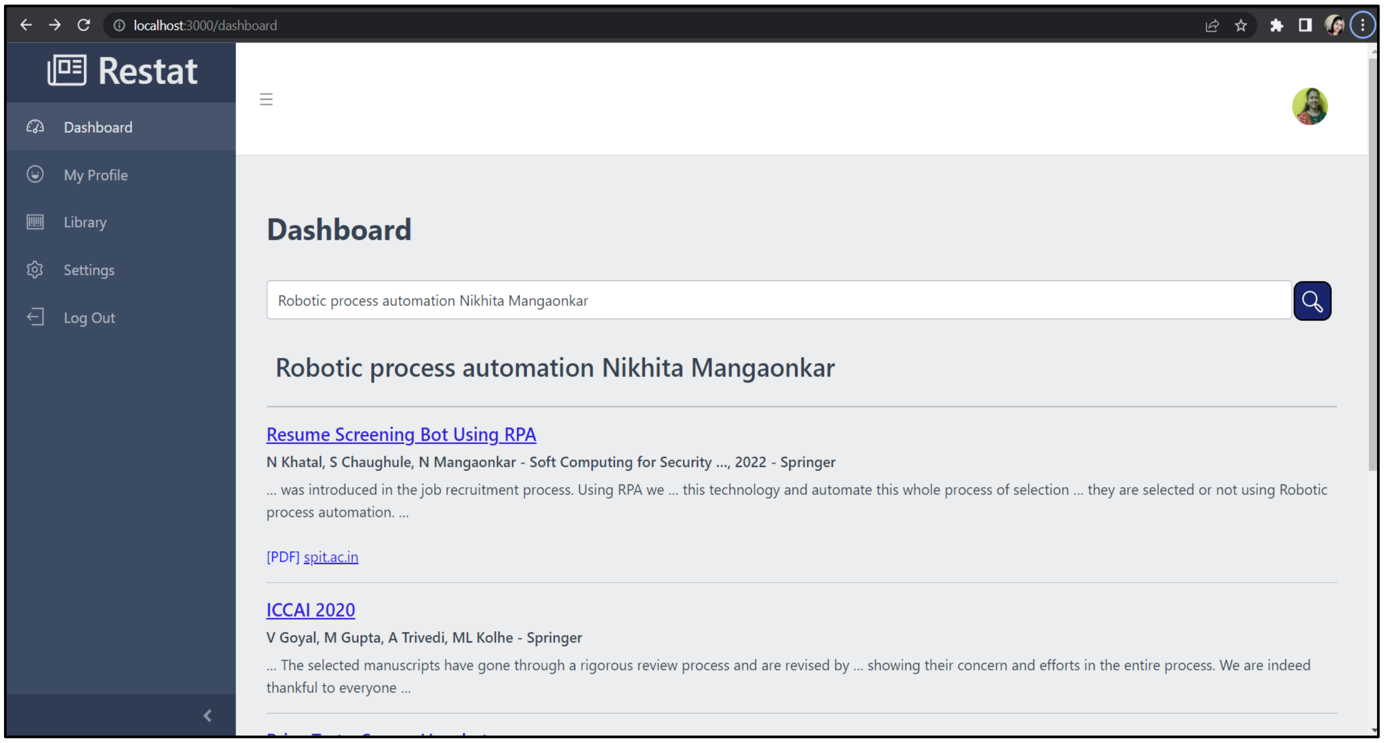


Nikhita Mangaonkar
Assistant Professor
Verified email at spit.ac.in

Green Technology for avoiding E-waste with the help of GIS
Authors: N Mangaonkar, P Sabhani, S Sirsat
Publication: 2014 International Conference on Advances in Communication and Computing ..., 2014
Cited By: 6
Year: 2014

Customer product experience analysis using text mining: A neuro linguistic programming approach
Authors: N Mangaonkar, S Sirsat
Publication: 2017 International Conference on Computing Methodologies and Communication ..., 2017
Cited By: 2
Year: 2017

Abstractive vs Extractive Text Summarization (Output based approach)-A Comparative Study
Authors: N Patel, N Mangaokar
Publication: 2020 IEEE International Conference for Innovation in Technology (INOCON), 1-6, 2020
Cited By: 1



Dashboard

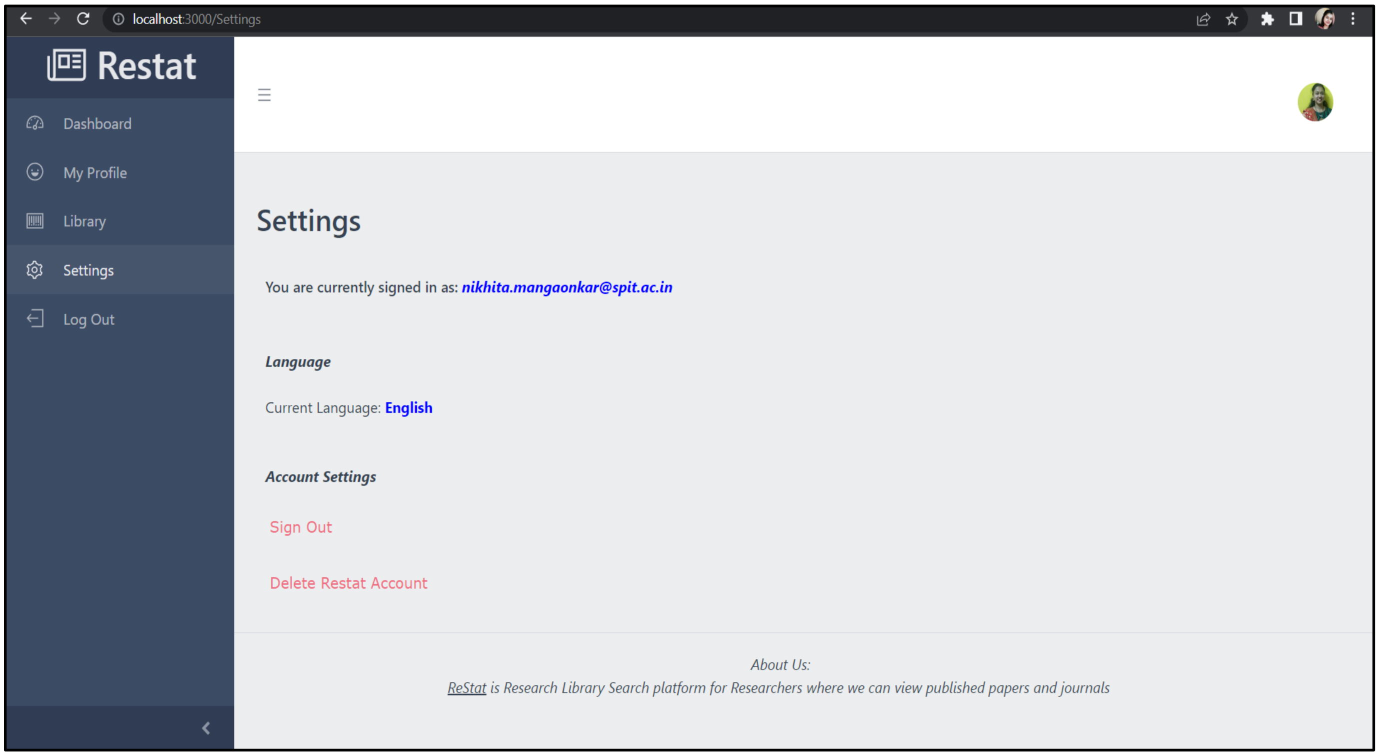
Robotic process automation Nikhita Mangaonkar

Robotic process automation Nikhita Mangaonkar

Resume Screening Bot Using RPA
N Khatal, S Chaughule, N Mangaonkar - Soft Computing for Security ..., 2022 - Springer
... was introduced in the job recruitment process. Using RPA we ... this technology and automate this whole process of selection ... they are selected or not using Robotic process automation. ...

[PDF] spit.ac.in

ICCAI 2020
V Goyal, M Gupta, A Trivedi, ML Kolhe - Springer
... The selected manuscripts have gone through a rigorous review process and are revised by ... showing their concern and efforts in the entire process. We are indeed thankful to everyone ...



Settings

You are currently signed in as: ***nikhita.mangaonkar@spit.ac.in***

Language

Current Language: **English**

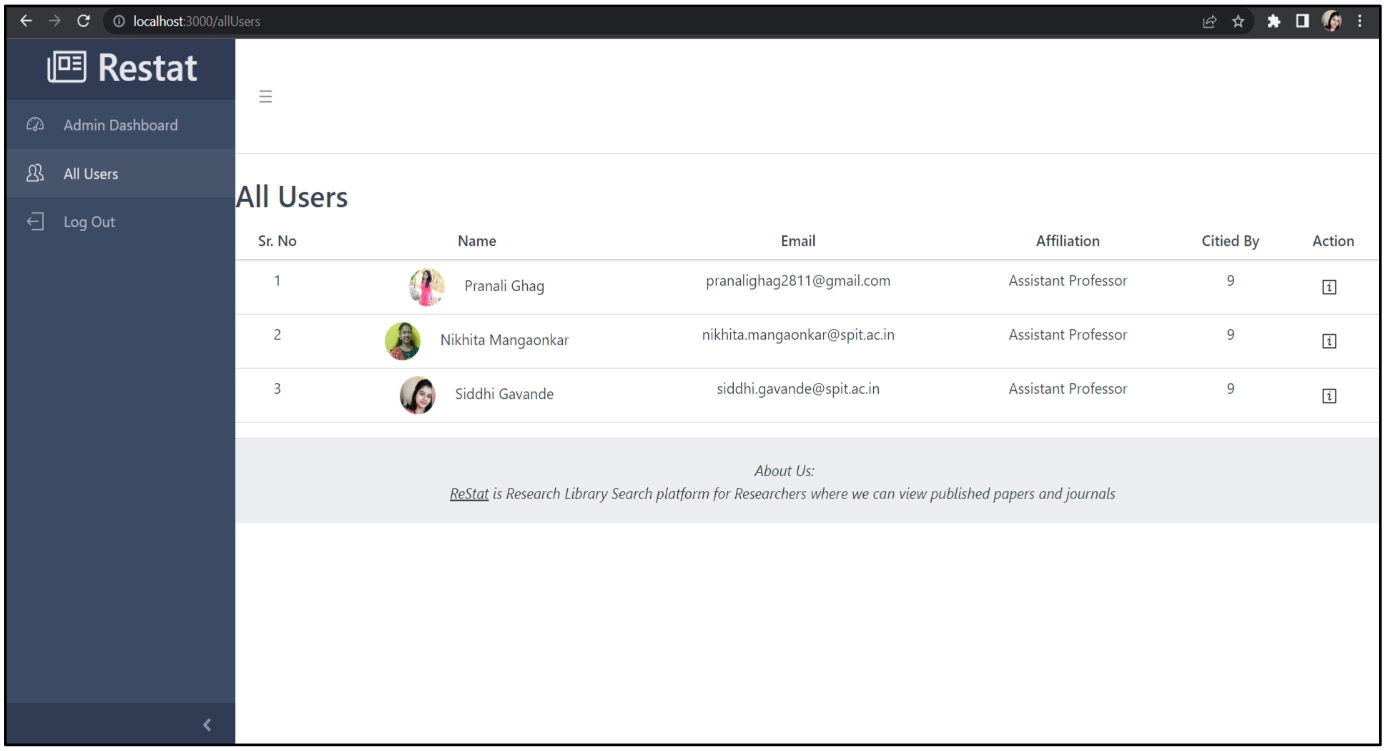
Account Settings

Sign Out

Delete Restat Account

About Us:

ReStat is Research Library Search platform for Researchers where we can view published papers and journals



All Users

Sr. No	Name	Email	Affiliation	Citied By	Action
1	Pranali Ghag	pranalighag2811@gmail.com	Assistant Professor	9	
2	Nikhita Mangaonkar	nikhita.mangaonkar@spit.ac.in	Assistant Professor	9	
3	Siddhi Gavande	siddhi.gavande@spit.ac.in	Assistant Professor	9	

About Us:

ReStat is Research Library Search platform for Researchers where we can view published papers and journals

5.3 Testing

Software testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest. It can be either done manually or using automated tools.

5.3.1 Test Cases

Testcase ID	Testcase name	Test Data	Expected Output	Actual Output	Result
1	Admin Login	Admin enters right email id and password	Log in successfully	Logged in successfully	Pass
2	Admin Login	Admin enters wrong email id and password	Check your credentials	Log in fail. Check credentials	Fail
3	User/Faculty Login	User enters right email id and password	Log in successfully	Logged in Successfully	Pass
4	User/Faculty Login	User enters Wrong email id and password	Check your credentials	Log in fail. Check credentials	Fail
5	User's Library	User's Email is checked, if spit.ac.in verified	Published papers data is fetched of that user	Published papers data is fetched of that user and displayed on the screen	Pass
6	User's Library	User's Email is checked, if not spit.ac.in verified	Published papers data is not fetched	Published papers data is not fetched and library is empty	Fail

Table 5.3.1: Test Cases

6. SYSTEM MAINTENANCE

- System Maintenance is needed when a new version of ReStat Library is released.
- System Maintenance will be carried out at least quarterly to check whether everything is working fine.
- Entire System Maintenance work is also carried out when the system fails to work properly.
- Application will require maintenance when there is a major release or change of functionality if some new functionality is added to the website.
- Maintenance is also required when this application fails to run in a certain environment, then proper debug and update will be released so as to increase compatibility.
- System Maintenance also takes place when we have to make some changes in the database either in its structure or migrations of data.
- Backup of Database should be taken frequently to prevent loss of data in case of any problems.

7. FUTURE ENHANCEMENTS

- Graphs can be made out of the data fetched from the API.
- Reports can be generated from the data by the Admin where they can create various analytics of data in terms of Domains, Number of Papers, Year-wise graphical representation of the Faculties papers, etc.
- Admin can be given an option to update the user's information if needed.
- Notifications can be sent through mails to the Users/ Faculties by the Admin to update their profiles frequently if required.

8. LIMITATION

- Requires Internet connection.
- No Support for iOS and MacOS Application.
- Requires a web browser & at least one E-mail id to work on the website.
- Fetching the API Data takes more time.

9. CONCLUSION

- The Website is developed using React JS, Redux and firebase database. The Website can be easily used by the user. The Website was built keeping in mind the ease of use and simple but fast.
- Website will be used by Admin and Users/Faculties who want to maintain their research profile.
- This Website will be beneficial to institutes if they want to maintain the records of published papers & journals by their faculty members.

10. BIBLIOGRAPHY

1. React JS- https://reactjs.org/docs/getting-started.html
2. NPM modules - https://docs.npmjs.com/cli/v6/commands
3. Database Connections- https://firebase.google.com/docs
4. UI- https://getbootstrap.com/docs/5.0/getting-started/introduction/ , https://mui.com/ , https://coreui.io/react/
5. API - https://serpapi.com/google-scholar-api
6. Charts - https://www.chartjs.org/docs/latest/