

# Web Application for Individual Research Paper Management\*

Unknown

## Abstract

In this paper, we consider a problem to assist researchers for managing research papers read by them. Our broad approach is to develop a web application, where researchers could upload their interested papers with topic, year, title, and technique. Upon uploading a paper, they can read the paper, write and save comments about the paper. To develop the application, we have used React, which is an open source JavaScript library for creating interactive website and standalone application. For storing users' information and research articles, we have used Firebase database.

## Introduction

Different content management systems have been studies, such as, learning management system [1], content management system [8, 7], course management system [3], Web-based content management system [4]. In addition, Researchers uses reference management tools [9, 5, 6, 2] for maintaining the history of research papers read by them. Most of available reference website/-tools provides researchers to create bibliographies and citations. Some of them also allow researchers to collaborate with others. Using these tools, users can store, organize and search all available references. However, it is hard for the researchers to remember what they have read in the past months and years. Hence, we focus on developing an application that could assist researchers to read papers online and allow them to write their notes about the paper. Although the application is beneficial to all the researchers, it will be more useful to multidisciplinary investigators.

---

\*Copyright ©2022 by the Consortium for Computing Sciences in Colleges. Permission to copy without fee all or part of this material is granted provided that the copies are not made or distributed for direct commercial advantage, the CCSC copyright notice and the title of the publication and its date appear, and notice is given that copying is by permission of the Consortium for Computing Sciences in Colleges. To copy otherwise, or to republish, requires a fee and/or specific permission.

In the application, we allows users to upload their interested papers with topic, year, title, and techniques. This helps the researchers to find all their interested papers based on research topic, year, title, and techniques. For each uploaded paper, the application provides an interactive window for the users to read the paper, write comments on the paper. The comments are stored along with time.

The application consists of two parts, namely, front and back end. For the front-end part, we have developed interactive web pages for the users to register, login, upload and search papers based on one or more criteria, and view a selected paper and write notes/comments about the paper. All the pages have been developed using React. For the back-end part, we have used Firebase to store the uploaded papers and comments.

# 1 System Design

In this section, we describe the flow diagram of the application given in Figure 1. For the application, we have developed the following React components.

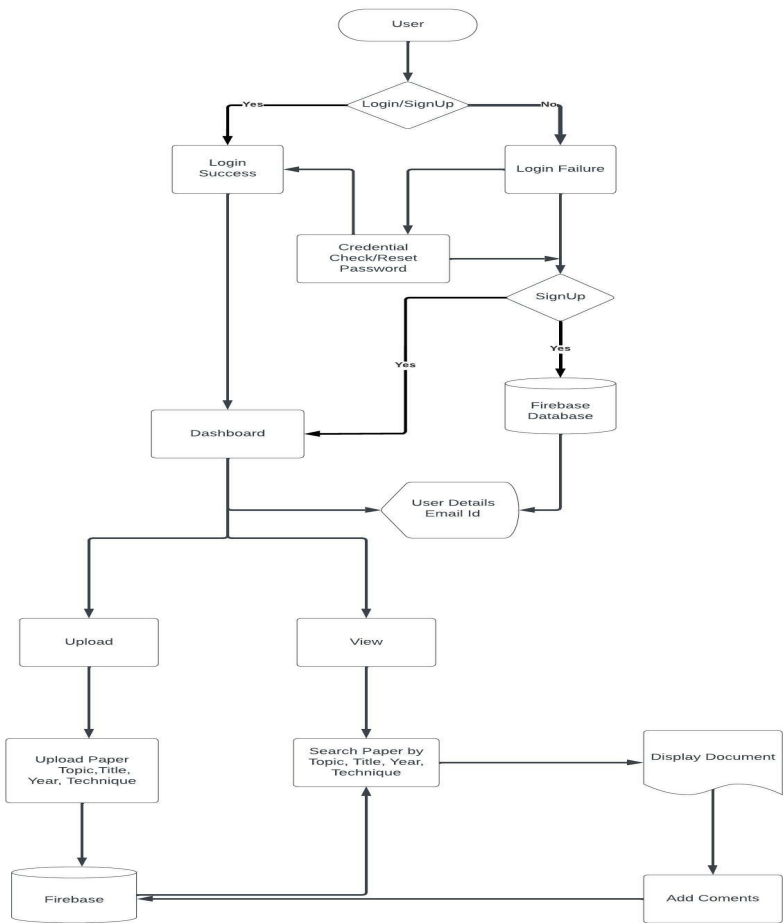


Figure 1: Research Content Management

**Register:** For the new users, we have developed a register component for creating an account on the application shown in Figure 2. To register, the users have to fill up their information, namely, name, email, and password.

The screenshot shows a web browser window with the address bar displaying 'localhost:3000/register'. The page has a light blue header with the text 'Research Content Management'. Below the header, there is a registration form with the following fields and buttons:

- Name:** A text input field with the placeholder 'Full Name'.
- Email:** A text input field with the placeholder 'E-mail Address'.
- Password:** A text input field with the placeholder 'Password'.
- Buttons:** A blue 'Register' button and a blue 'Register with Google' button.
- Footer:** A link that says 'Already have an account? [Login now.](#)'

Figure 2: Register Page

We also provide another option to users for registering through their google account, that is, if one has a valid Google account saved in his local machine, he/she can register at a glance using the above mentioned feature. The authentication requirements provided by the Firebase are needed to meet like password should be minimum of 6 characters in length and email address should end up with .com.

**Login:** For existing users, we have develop a login component shown in Figure 3. Once a user has successfully created/registered an account he/she can login

The screenshot shows a web browser window with the address bar displaying 'localhost:3000'. The page has a light blue header with the text 'Research Content Management'. Below the header, there is a login form with the following fields and buttons:

- Email:** A text input field with the placeholder 'E-mail Address'.
- Password:** A text input field with the placeholder 'Password'.
- Buttons:** A blue 'Login' button and a blue 'Login with Google' button.
- Footer:** A link that says 'Forgot Password' and a link that says 'Don't have an account? [Register now.](#)'

Figure 3: Login Page

to website using his/her email address and password. Alternatively, the user

can also login with Google account as we have the option of registering using Google account. If the user forget his/her password, the user can reset the password via forgot password link, which we provide on the login page. Upon clicking the link, the user has to enter his registered email address in order to reset the password.

**Dashboard:** We have developed a dashboard component for the users to upload, search and view papers upon login. The dashboard consists of website name to the left and the user can see his name, email address and logout button to the right. This dashboard gives the user a quick glance of user details.

**Upload:** We have developed an upload component for the users to upload their interested papers shown in Figure 4. This page allows a user to upload a research paper on which he/she is currently working or previous research papers which he/she worked on. For making the uploaded paper unique, there are several factors taken into consideration. The paper which user wants to upload should be of pdf format, should choose a topic for the paper, year of publishing for the paper, title of the paper, and finally technique of the paper. By choosing the above metrics, each paper represents unique. When the user clicks on submit button given in Figure 4, the paper is directly uploaded to the Firebase server which later can be accessed.

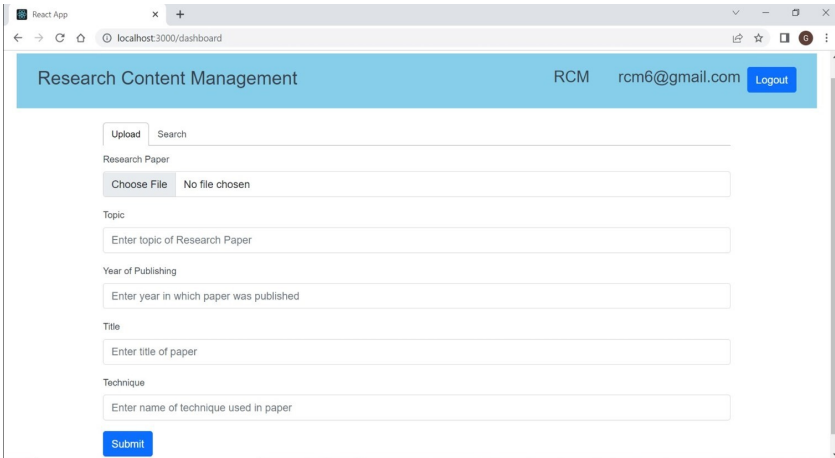


Figure 4: Upload Page

**Search:** Upon login, a user can search interested papers that have been already uploaded by him/her. For this, we have developed a search component.

It uses the Firebase database finding papers based on topic, title, year or technique. Upon entering these key values, the user has to enter the search button shown in Figure 5. Upon clicking on search button, the component interacts with the Firebase database and find all matching papers stored in the Firebase database. For this, we have implemented effective search technique.

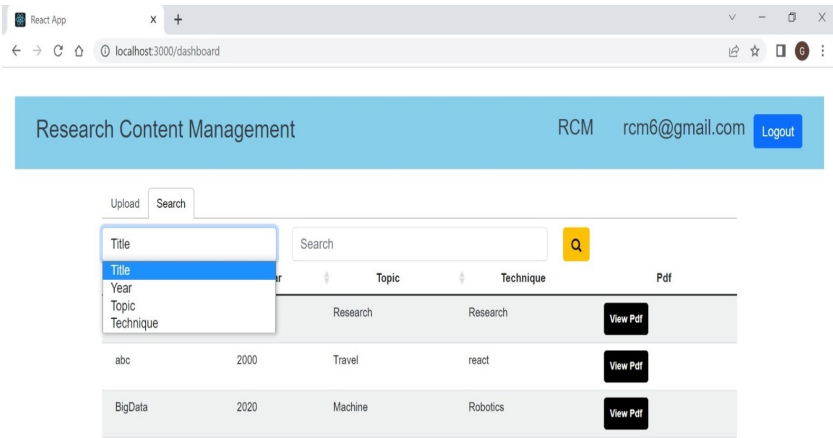


Figure 5: Search Page

**View:** Upon searching papers, the user may wish to view/read a specific paper. For this, we have developed view component shown in Figure 6. After search operation, there is a view button on each matched paper. When the user clicks on a specific paper, view component will help the user to read/view the paper along with comments facility. If the user wants to add comments for the paper, he/she needs to enter the note in comments box and to click on post comment button. Then, the comment for the paper will be stored in database with timestamp. When user clicks on show comments button, the comments which are posted for the paper is displayed as list with the timestamp.

## 2 Features

In this section, we will provide the details about certain features of the application.

**Performance** The application is built in React and is integrated with Firebase. The performance of the application is fast and user-friendly. Users can navigate between different sections of the application with ease.

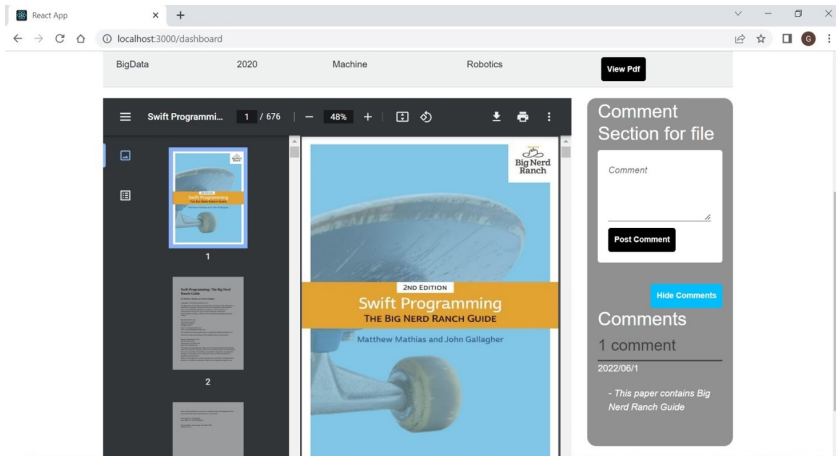


Figure 6: Search Page

**Reliability** The application delivers correct output to the user from start to end. Users can retrieve any stored paper with ease and add comments for further review and update it accordingly. The application delivers what it needs to perform and asks inputs accordingly so that it can work accordingly.

**Availability** The application is free to use for any users. As the application is designed keeping in view of various users where they can store their research work and retrieve for further use. Users can create an account and login with their credentials, upload and retrieve papers for further use.

**Security** The application is protected with authentication provided by Firebase where it doesn't allow other users to use some others account until and unless the login credentials are misused. Firebase has various inbuilt authentication steps to register or login just by enabling and using the features in code.

**Portability** Currently, the application is designed as web application. The application runs on server side so there is no restriction for any systems other than mobile devices.

### 3 Implementation of the Application

In this section, we provide the details about the implementation and execution of the application.

**Creating a new React project** First, we have created a React project via the following steps:

- A new React project can be created by using command “`npx create-react-app my-app`” where `my-app` is your app name.
- Developer can use any IDE for developing a react application.
- After the successful execution of the above command, a basic react application will be created.

**Managing package.json** We use `package.json` for storing information about the application. The `package.json` file consists of metadata where the file includes all the details about the application that includes application name, version, dependencies installed and their version and debugging dependencies.

**Components** Components are building blocks for the React component. A React application can have many components where a developer writes clean and efficient code in a component and embed the component to a main file. There are two types of components in React JavaScript, namely, functional and class component.

**Adding Database to React Application** To our react application we have chosen the Firebase as primary database where our application has authentication, storage and managing the users. To make the Firebase in sync with react application, a developer needs to create a project in the Firebase and choose the application which he/she needs to develop so that the Firebase provides a snippet code which is ready to use. We use the provided snippet by the Firebase in our react application by creating a new `.js` extension file. By using above file we can import the Firebase functionalities to our react application.

**Embedding Components to App.js** All the components which were created in React application needs to embed in main file i.e., `App.js` so that whenever react application call `index.js`, it renders `App.js` in strict mode and all the components in `App.js` will be displayed whenever application is run.

**Running React Application** After adding all the components to `App.js` the application needs to be run on the server side. React application can be run on the server side using command “`npm start`” and the application can be viewed on `localhost:3000`. In order to run the application some of the dependencies need to be installed and they can be installed by running the below commands in VS code terminal.



- npm install react-scripts --save
- npm install -S react-router-dom
- npm install --save react-firebase-hooks

## 4 Conclusion

In this paper, we have developed a web application to assist researchers for the effective management of the articles read by them. We have developed the application using React and Firebase. The application is easy to use and all the interaction pages are user friendly. In the future, we plan to add more features, such as, word documents, automatic detection of topic, title, and technique.

## References

- [1] Lejla Abazi-Bexheti. "Development of a learning content management system". In: *WSEAS Transactions on Information Science and Applications* (2008).
- [2] Joeran Beel et al. "Docear: An academic literature suite for searching, organizing and creating academic literature". In: *Proceedings of the 11th annual international ACM/IEEE joint conference on Digital libraries*. 2011.
- [3] Christopher Brooks, Rupī Panesar, and Jim Greer. "Awareness and collaboration in the ihelp courses content management system". In: *European Conference on Technology Enhanced Learning*. Springer. 2006.
- [4] Maciej Dobecki and Wojciech Zabierowski. "Web-based content management system". In: *International Journal of Computing* (2010).
- [5] Bell Eapen. "EndNote 7.0". In: *Indian Journal of Dermatology, Venereology, and Leprology* (2006).
- [6] Raphaël Grolimund. *Citation and bibliography made easy with Zotero*. Tech. rep. 2012.
- [7] Dimitrios Michelinakis. "Open source content management systems: An argumentative approach". In: *The University of Warwick: Warwick Manufacturing Group, A report submitted for the award of MSc Electronic Business Management* (2004).
- [8] Sean D Mooney and Peter H Baenziger. "Extensible open source content management systems and frameworks: a solution for many needs of a bioinformatics group". In: *Briefings in bioinformatics* (2008).

- [9] Jatinder Singh. “Mendeley: A free research management tool for desktop and web”. In: *Journal of Pharmacology and Pharmacotherapeutics* (2010).