

JOB FINDER APPLICATION

A major project report submitted in partial fulfillment of the requirement for
the award of degree of

Bachelor of Technology

in

Computer Science & Engineering / Information Technology

Submitted by

Amritanshu Suyal (201275)

Mohnish Sharma (201217)

Under the guidance & supervision of

Mr. Faisal Firdous & Mr. Ramesh Narwal



**Department of Computer Science & Engineering and
Information Technology**

Jaypee University of Information Technology, Wagnaghat,

Solan - 173234 (India)

CERTIFICATION

We hereby declare that the work presented in this report entitled **Job Finder Application** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering / Information Technology** submitted in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat is an authentic record of my own work carried out over a period from August 2023 to May 2024 under the supervision of **Mr. Faisal Firdous (Assistant Professor) and Mr. Ramesh Narwal (Assistant Professor) Department of Computer Science & Engineering and Information Technology**).

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

(Student Signature with Date)

Student Name: Amritanshu Suyal

Roll No.: 201275

(Student Signature with Date)

Student Name: Mohnish Sharma

Roll No.: 201217

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature with Date)

Supervisor Name: Mr. Faisal Firdous

Designation: Assistant Professor

Department: CSE

Dated:

(Supervisor Signature with Date)

Supervisor Name: Mr. Ramesh Narwal

Designation: Assistant Professor

Department: CSE

Dated:

ACKNOWLEDGEMENT

Firstly, we express our heartiest thanks and gratefulness to almighty God for his divine blessing making it possible to complete the project work successfully.

We are really grateful to our supervisor Mr. Faisal Firdous (Assistant Professor, Department of Computer Science & Engineering and Information Technology) & Mr. Ramesh Narwal (Assistant Professor, Department of Computer Science & Engineering and Information Technology) for carrying out this project. His endless patience, scholarly guidance, continual encouragement, constant and energetic supervision, constructive criticism, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

I would also generously welcome each one of those individuals who have helped me straightforwardly or in a roundabout way in making this project a win. In this unique situation, I might want to thank the various staff individuals, both educating and non-instructing, which have developed their convenient help and facilitated my undertaking.

Finally, I must acknowledge with due respect the constant support and patients of my parents.

Student Name: Amritanshu Suyal

Roll No.: 201275

Student Name: Mohnish Sharma

Roll No.: 201217

JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY, WAKNAGHAT

PLAGIARISM VERIFICATION REPORT

Date:

Type of Document (Tick): PhD Thesis M.Tech Dissertation/ Report B.Tech Project Report Paper

Name: _____ Department: _____ Enrolment No _____

Contact No. _____ E-mail. _____

Name of the Supervisor: _____

Title of the Thesis/Dissertation/Project Report/Paper (In Capital letters): _____

UNDERTAKING

I undertake that I am aware of the plagiarism related norms/ regulations, if I found guilty of any plagiarism and copyright violations in the above thesis/report even after award of degree, the University reserves the rights to withdraw/revoke my degree/report. Kindly allow me to avail Plagiarism verification report for the document mentioned above.

Complete Thesis/Report Pages Detail:

- Total No. of Pages =
- Total No. of Preliminary pages =
- Total No. of pages accommodate bibliography/references =

(Signature of Student)

FOR DEPARTMENT USE

We have checked the thesis/report as per norms and found **Similarity Index** at(%). Therefore, we are forwarding the complete thesis/report for final plagiarism check. The plagiarism verification report may be handed over to the candidate.

(Signature of Guide/Supervisor)

Signature of HOD

FOR LRC USE

The above document was scanned for plagiarism check. The outcome of the same is reported below:

Copy Received on	Excluded	Similarity Index (%)	Generated Plagiarism Report Details (Title, Abstract & Chapters)	
	<ul style="list-style-type: none">• All Preliminary Pages• Bibliography/Images/Quotes• 14 Words String		Word Counts	
Report Generated on			Character Counts	
		Submission ID	Total Pages Scanned	
			File Size	

Checked by
Name & Signature

Librarian

Please send your complete thesis/report in (PDF) with Title Page, Abstract and Chapters in (Word File) through the supervisor at plagcheck.juit@gmail.com

TABLE OF CONTENT

CERTIFICATE	I
ACKNOWLEDGEMENT	II
LIST OF FIGURES	III
ABSTRACT	IV
1. CHAPTER-1 INTRODUCTION	1
1.1 INTRODUCTION	1
1.2 OBJECTIVES	3
1.3 SIGNIFICANCE & MOTIVATION OF PROJECT	4
1.4 ORGANIZATION OF PROJECT REPORT	5
2. CHAPTER-2 LITERATURE SURVEY	6
2.1 OVERVIEW OF RELEVANT LITERATURE	6
2.2 KEY GAPS IN LITERATURE	8
3. CHAPTER-3 SYSTEM DEVELOPMENT	11
3.1 REQUIREMENTS AND ANALYSIS	11
3.2 PROJECT DESIGN AND ARCHITECTURE	18
3.3 DATA PREPRATION	23
3.4 IMPLEMENTATION	24
3.5 KEY CHALLENGES	31
4. CHAPTER-4 TESTING	33
4.1 TESTING STRATEGY	33
4.2 TEST CASES AND OUTCOMES	35
5. CHAPTER-5 RESULTS AND EVALUATION	37
5.1 RESULTS	37
6. CHAPTER-6 CONCLUSION & FUTURE SCOPE	43
6.1 CONCLUSION	43
6.2 FUTURE SCOPE	44
7. REFERENCES	46

LIST OF FIGURES

Figure 3.1	Backend Flowchart
Figure 3.2	Frontend Flowchart
Figure 3.3	Home Screen Output Window
Figure 3.4	Category/Search Screen
Figure 3.5	Job Description Screen
Figure 3.6	Entry File of Backend
Figure 3.7	Fetching Data from Indian Internship
Figure 3.8	Fetching Data from foundit
Figure 3.9	Root Frontend Component
Figure 3.10	Metro Bundler
Figure 3.11	useFetch.js
Figure 3.12	UI/Server communication
Figure 5.1	Home Screen
Figure 5.2	Full time category screen
Figure 5.3	Part time category screen
Figure 5.4	Search Functionality
Figure 5.5	Search Results
Figure 5.6	Job Detail Screen
Figure 5.7	Website User Interface
Figure 5.8	Saved Jobs
Figure 5.9	Indian Internship Database
Figure 5.10	Foundit Database
Figure 5.11	Internshala Database

ABSTRACT

In response to the obstacles inherent in today's job search process, the Job Finder Application was created with the goal of delivering a universal and user-friendly solution. For job seekers, the challenge of navigating several websites and systems, each with its own application process and design, was a major obstacle. The Job Finder Application was created to solve this problem by compiling job postings from multiple sources into a single, user-friendly interface that is compatible with web, iOS, and Android platforms. Utilizing cutting-edge technologies like Node.js, Express.js, and MongoDB for backend infrastructure and React, React Native for frontend development, the application sought to improve communication between companies and job seekers while streamlining the job search process.

The Job Finder Application, which provides a thorough answer to the problems encountered by current job seekers, is an important step in the field of job search platforms. Utilizing modern technologies, the application streamlines the job search process and improves accessibility for users on various devices by combining job listings into a single interface. The application, with its careful planning and creative design, raises the bar for efficacy and efficiency in job search platforms, enabling people to easily locate fulfilling career prospects.

In conclusion, an effort to meeting the changing demands of job seekers in the fast-paced labor market of today is demonstrated by the creation of the Job Finder Application. The application streamlines and improves user experience by providing a smooth and user-friendly platform that collects job postings from several sources. The Job Finder Application, which prioritizes innovation and user-centric design, has the potential to completely transform the way people look for jobs. In the end, it will be an invaluable tool for both employers and job seekers.

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

The job finder app is a universal and user-friendly cross platform mobile app applicable on android iOS and web, which will be able to pull the jobs across different sources in just one place. Finding a job today can entail maneuvering through different systems and websites with their individual application methods and layout. The Job Finder Application being a cross platform mobile app provides an easy way of looking for employments. The Job Finder application that supports android and IOS users will narrow down this gap as more people turn to smart phones. The primary purpose of this application is to bring together job listings from different sources in an easy-to-use platform so that individuals can easily search, view, and apply for the desired positions.

The backend infrastructure of the program is driven by a combination of Node.js, Express.js, and MongoDB. A popular NoSQL database, MongoDB, offers a scalable and effective storage option that can handle the changing nature of work data. The foundation of the application's server-side logic is provided by Node.js and Express.js, respectively. Node.js's asynchronous event-driven architecture makes developing reliable web and mobile APIs easier. When combined, these technologies create a strong backend ecosystem that can support the needs of a major job search platform.

In order to protect user data and privacy, the project also explores the usage of strict security mechanisms like OAuth and JSON Web Tokens (JWT). These procedures support the application's dedication to data integrity and protection while also enhancing user confidence in it. The project also explores the possibility of extending the application's functionality to add recruitment features. The application can provide a full solution that closes the gap between businesses and job searchers by enabling recruiters to post job postings directly within the platform and view candidate resumes. This tactical improvement strengthens the application's value proposition and expands its feature set, presenting it as a flexible resource for all parties involved in the employment market.

Including recruiter capability in the Job Finder Application offers a great chance to improve its features and fill in any holes. We can expedite the hiring process and improve user experience

by giving recruiters the option to post jobs directly within the platform and access the resumes of job seekers.

The above project report will provide an analysis on how the Job Finder application was designed towards functionality and what technology it utilized hence becoming an efficient integrated job search platform. Through meticulous planning and innovative approach the Job Finder Application seeks to offer all-encompassing solution to present day job seekers' issues making it easy, effective, useful.

The Job Finder Application is a platform for job searches that is highly efficient due to its creative design and careful planning. Its dedication to meeting the changing demands of contemporary job seekers is demonstrated by its smooth integration of cutting-edge technologies and user-centric approach. The Job Finder Application aims to transform the job search process by providing a comprehensive solution that blends strong functionality, intuitive design, and strict security measures. This will make the job search process simple, efficient, and ultimately profitable for all users.

Utilizing the Cheerio Node module can offer a practical and fast way to extract data from job listing websites. Data may be published and subscribed to on a Cheerio topic, which makes it easier to gather and manage job listing data from many sources. Scalability and fault tolerance are two further features that make Cheerio a dependable option for handling massive volumes of data. But it's crucial to make sure that Cheerio's use for data scraping complies with all applicable ethical and regulatory requirements for permission and data privacy.

PROBLEM STATEMENT

Job searchers today have a difficult time navigating through the disorganized and frequently puzzling variety of online job sites. This disorganized method not only makes the job search less effective, but it also poses a significant obstacle for consumers who find it difficult to keep track of numerous accounts and interfaces.

- The present job search process is fragmented for job searchers because they have to visit various websites and platforms in order to look through job postings.
- It is difficult for users to effectively track and manage their job applications in the absence of a centralized system.

- The job search process is made more difficult by the fact that various job platforms frequently have unique user interfaces and application procedures.
- Nowadays, looking for a job takes a lot of time because you have to visit a lot of websites, create multiple accounts, and get used to different application processes.
- The lengthy nature of this procedure reduces the job search's overall effectiveness.
- The difficulty of constantly keeping an eye on several sources may cause users to miss important job opportunities.
- The difficulties job hunters have been made worse by the lack of a single, integrated solution for both iOS and Android.
- A divided user experience is caused by the frequent requirement for users to switch between several programs depending on their mobile device.
- This negative experience could make people less motivated and involved in the job search.
- A simpler and more approachable method of job hunting is required due to the changing nature of the employment market.
- A unified platform that reduces the process is becoming more and more necessary so that consumers may concentrate on finding good job prospects instead of figuring out complex interfaces.

1.2 OBJECTIVES

- Establish a central platform that collects job postings from many sources and provides users with a single point of entry to a variety of career opportunities.
- Create the Job Finder application on both iOS and Android operating systems and also web compatible to make it accessible to a large number of users.
- To improve user efficiency and make navigating various job posts easier, provide a uniform application process within the Job Finder application.
- As a case in point, search functions would assist users find the most appropriate job opportunities like keywords.

1.3 SIGNIFICANCE & MOTIVATION OF THE PROJECT WORK

SIGNIFICANCE OF THE PROJECT

The Job Finder Application gives users an easier centralized way to explore opportunities on this platform solving the present problems concerning the split job search process. The enhancement of user experience is, therefore, crucial in an increasingly competitive employment environment that relies on accessibility and speed.

The application helps job hunters eliminate the tedious process of searching for openings.

consolidate data from various sources and present them in one interface. This is an important time saving feature, particularly for those individuals with dual responsibilities. The project will offer a detailed list obtained from various sources to enable one to get an appropriate job, possibilities more visible to users.

The application's cross-platform functionality guarantees that users can take advantage of a centralized job search platform on both Android and iOS devices. In order to maximize the application's impact and reach a wide user base.

WEB SUPPORT

We also provide web support for our project, as we also made a job finder website as well where there is job data from three websites as of right now that is from Indian internship, Foundit and Internshala.

MOTIVATION OF THE PROJECT

The project's motivation comes from a deep desire to ease the difficulties faced by job seekers. The initiative aims to offer a more user-centric solution because the present job search process is complicated and inefficient, which leads to frustration and demotivation.

The project is driven by the chance to use technological innovation to simplify and improve standard processes. Utilizing cross-platform programming and mobile application technologies is an exciting way to combine creativity with real-world problem-solving.

Make an app that is compatible with both iOS and Android so that users of all phone types can utilize it. Also an application that is easy to use and add functionalities like job sorting according to user and search functionality to search jobs for any specific keyword to make the process of job hunting as soothing as possible.

1.4 ORGANIZATION OF PROJECT REPORT

- In Chapter 2, the current state of cross-platform job search applications is explored, with particular attention paid to the use of technologies like Expo, Axios, React Native, and RapidAPI. Examining the results produced by various applications, this section prepares the reader for the comparative analysis that is a key component of this report.
- Chapter 3 presents a fresh approach that was developed to accurately summarize LinkedIn profiles, with a specific emphasis on retrieving extensive user data. This chapter provides an overview of the project's requirements, design, and implementation while addressing issues that emerged during development and offering insights into the formulation process.
- Chapter 4 provides information on the system's performance and dependability by outlining the test method, listing test cases, and presenting test results.
- A thorough analysis of the results is presented in Chapter 5, including a thorough assessment of the effectiveness of the summary technique. Not only are comparisons made within the study, but also with previous research on profile summary and job search software. The report is enhanced by this chapter's insightful context and benchmarks.
- The conclusion of the investigation is summarized in the last chapter, Chapter 6. It summarizes the project's impact on the field, making inferences from the findings and emphasizing potential directions for further research. This chapter provides a thorough summary of the main conclusions and lays the groundwork for future developments in the application's scope.

CHAPTER – 2: LITERATURE SURVEY

2.1 OVERVIEW OF RELEVANT LITERATURE

This process becomes challenging for the job seekers looking for positions that suit their interests and skills. These challenges arise because people do not know what the firm wishes to achieve, work culture, and advertised job vacancies. Another task that may be critical to the recruiters in any organization is identifying an appropriate candidate who can fit well and possess the required qualification to take up the available slots. It is true that online job search portals have made searching for jobs easier for both employers and job seekers. Job portal is a place where both employers can interact on the basis of getting what they really need at the end. This is the fastest, cheapest method of communication that reaches many people with a single click irrespective of their location on earth[1].

For job seekers and workers, finding a job that matches their interests and skills is very hard. The problem is caused by failure to understand the organizational objectives, working culture, and job positions. Recruiters' jobs are also an important task that must be performed by agencies, firms, contractors or other places in search of right-skilled people to fit into open vacancies. Yes, web-based job search applications have made job search more convenient for both the employers and the job seekers. Job Search Application facilitates collaboration between employers and jobseekers in addressing their requirements [2].

Science and modern technology are the source of what we call the internet today. There are many people who take advantage of the internet for varied purposes depending on their needs. Nowadays in this digital era, people must be connected to the internet in order to keep life going on. The internet has modified people's lives and office environments as well. The spread of the Corona virus has led to a significant rise in people's use of the internet and its services. Most job seekers turn to e-job portals when conducting an Internet job search. To this end, this paper aims at building a portal that considers stakeholder's views, fixing problems and adding more features into the existing systems [3].

This paper studies job instability in a crisis to investigate the consequences of job insecurity. Writers explore how workers perceive and respond to job uncertainty. To this end, the study

contributes valuable insights into the organizational and personal dimensions related to job insecurity and how employees experience their job during such tough economic times. In his paper, Ajzen introduces the Theory of Planned Behavior, which is a psychological theory aimed at predicting as well as understand human behavior. The theory postulates that attitudes towards the behavior, subjective norms, and perceived behavioral control significantly influence behavioral intentions. Combined, these components aim to explain as well as predict certain kinds of behavior such as in organizations. It has had a very big influence on the disciplines of decision-making, organizational behavior, and psychology [4].

Aims to give job seekers and current employees access to online job portals with information on various companies' openings. In addition to managing job openings, job details, bio data, interviews, call letters, and other information, this system also integrates SMS. Because users must upload their resumes through their accounts and because job opportunities are posted by visitors and other users, this system is very helpful to users. Many features are offered by this paper to help manage all the data efficiently. This application has many sophisticated modules, which give the back end system a great deal of power [5].

As Internet technology advances quickly, an increasing number of job searchers are disclosing their personal information online, while employers are posting job openings. The development of Web 2.0 technology has resulted in a significant rise in the amount of personal data that employers and job seekers share. Consequently, there is an overload of information, which lowers the utilization rate. It has been suggested that the job recommender system, an online recruiting platform with tailored recommendations, can help both employers and job seekers with the aforementioned problem. As a recommender system, the job recommender system uses recommendation technology—such as content-based recommender and collaborative filtering recommender, which have demonstrated success in various recommender systems—to retrieve a list of job positions that suit a job seeker's desire or a list of talented candidates that meet a recruiter's requirement. Some issues that have received a lot of attention in the job recommender system have been identified based on the papers we have studied during our preliminary research [6].

An expert system for evaluating the unemployed at specific offered posts is presented in this paper. The expert application analyzes corporate databases of jobless and company profile data using Neuro-Fuzzy techniques. A Sugeno-type Neuro-Fuzzy inference system is used to match jobless people with available positions. The Greek General Secretariat of Social Training

provided large training sets of historical records of unemployed people (of the same social class) who were either rejected or approved at multiple posts. These records were used to define the weights of the system parameters. The training set is expanded with newly received cases, whether they are accepted or denied. Retraining takes place following a predetermined number of newly available cases. Fuzzy logic and neural networks combined produce strong expert decision systems. The field of hybrid and neural processing research has experienced remarkable growth in the last few years. Additionally, the successful application of hybrid intelligent systems has grown significantly in a wide range of fields, including robotics, education (Vrettaros, 1996), e-commerce (Kouremenos, Vrettos & Stafylopatis, 2003), speech/natural language understanding, medical diagnosis, and information retrieval (Vrettos & Stafylopatis, 2001). But there hasn't been any prior research on expert job matching in the literature. Too frequently, the straightforward Boolean matching method that is widely used online needs to be replaced with a more organized and comprehensive process for matching an unemployed person with a specific job [7].

2.2 KEY GAPS IN THE LITERATURE

[1] Job Search Portal

User Experience (UX) - Deficits in knowledge or solutions for particular UX problems in the current online job search engines.

Technological Gaps - Possible shortcomings in the features or technology provided by the job search apps available today, which may point to areas in need of innovation or improvement.

Gaps in accessibility and inclusivity that need to be filled in order to guarantee that a diverse user base, including people with disabilities, can utilize the platform.

Data Security and Privacy - Problems or weaknesses pertaining to these two essential components of any internet application, particularly those that handle sensitive and personal data.

[2] Online Job Search Application

Assisting job seekers in locating positions that are unique to companies as well as functional areas.

Functional Domains - Determine your ideal functional areas by evaluating your abilities, background, and strengths.

Examine employment portals - To locate relevant job listings, use job portals such as Indeed, LinkedIn, Glassdoor, and specialized industry platforms.

Sector-Specific Websites - Examine job boards or platforms that are specific to your industry and functional area. Take GitHub for technology, Behance for artists, or Medscape for medical information.

[3] A Study of Issues in Job Portals: Research Analysis

Assuring the validation of a greater number of users is a substantial task for administrators, requiring a thorough and organized strategy. Administrators must put strong systems and procedures in place because the difficulties with authentication and validation increase with the number of users.

Administrators should think about utilizing sophisticated verification techniques that balance user convenience and security in order to carry out this project successfully. For example, multi-factor authentication, which combines passwords, biometrics, and one-time passcodes, can significantly improve the security posture of user verification procedures overall. Multiple forms of identification make it much less likely that someone will gain unauthorized access.

[4] Job Search and Employment Success: A Quantitative Review and Future Research Agenda

In the world of technology and business, scalability and performance concerns are crucial factors that greatly influence the effectiveness and prosperity of different kinds of systems, apps, and enterprises. As we explore the complex facets of performance and scalability, it becomes clear that resolving these issues is essential to preserving a competitive advantage in the quickly changing digital landscape of today.

[5] Job Seeking: The Process and Experience of looking for a job

Data Accessibility - The accuracy and dependability of any analysis or study are significantly impacted by the caliber and volume of available data.

Difficulties: Insufficient data availability may make it difficult to conduct thorough research or create machine learning models that work well. There are situations when data may exist but be unavailable because of technological obstacles, privacy issues, or legal limitations.

Generalizability - Beyond the specific conditions of the study, generalizability is essential for deriving significant conclusions and making wise decisions.

The generalizability of results is influenced by several factors, including the diversity of the dataset, the representativeness of the sample, and the similarity of study conditions to those of the target population.

[6] Job Recommender Systems: A Survey

Data Security - It is critical to guarantee user data availability, confidentiality, and integrity. Use encryption techniques to protect private data while it's being transferred and stored.

Verification and Permission:

Establish reliable authentication procedures to safely confirm user identities. Use multi-factor authentication to add an additional degree of protection. Establish and implement stringent access controls to suitably restrict user permissions.

[7] An Expert System For Job Matching of the Unemployed

Retraining the dataset occurs when a predetermined number of new test cases are added.

Gathering of Data - Your model may come across fresh situations and data points during operation that weren't included in the initial training set.

Gather fresh data on a regular basis to account for the dynamic nature of the issue your model is attempting to solve.

Assessment of the Model's Performance - Evaluate your model's performance on a regular basis with the updated data. Determine any areas where the model might be performing poorly or having trouble with new cases.

Extension of the Test Set - Incorporate the recently added test cases into your assessment collection to replicate real-world circumstances and obstacles. To capture the variety of possible inputs, make sure the new test cases include a wide range of scenarios.

CHAPTER – 3: SYSTEM DEVELOPMENT

3.1 REQUIREMENT & ANALYSIS

The following are the requirements explained that are required to build a react native project:

NODE JS AND NODE PACKAGE MANAGER (NPM)

The runtime environment for javascript code can be accessed outside of a browser thanks to Node.js. For server-side programming, this is utilized. Node.js is used to create input- and output-heavy web applications. Node.js is an event-driven, asynchronous framework. Because Node.js is asynchronous, the server never waits for the API to return the data; instead, it operates without stopping. Node.js does not use data buffering; instead, it outputs data in chunks.

Single threaded models and event looping are used in Node.js.

Npm stands for node package manager in its full form. The largest software library in the world is NPM. It is a JavaScript package manager.

NPX

Npx stands for node package execute in its full form. A javascript package can be executed by developers using npx if it is not installed in the npm library. A command-line utility called npx was added to the Node.js package management, npm, with version 5.2. Without needing to install them globally or to their project, it enables developers to quickly execute packages that are installed in their local or remote npm registry.

Running a command-line interface (CLI) tool that is available as a npm package without the requirement to install it locally or globally in your project is the main use case for npx. This makes it especially helpful for one-time use of packages, testing out new packages, or testing packages before adding them to your project dependencies.

The requested package is first temporarily installed by npx before being executed. Upon completion of the command, the package undergoes automatic uninstallation. This implies that

you won't have to worry about corrupting your project or global environment while using any package that is listed on the npm repository.

Running a package version other than the one installed in your project is another use case for npx. You can run a different version of the package without needing to install it individually by providing the package version or location.

Additionally, npx has a number of options that let you modify how it behaves. These include giving command-line arguments, deciding how to handle global packages, and indicating the package version or location. Because of its degree of customization, npx is a versatile tool that works well in a variety of development settings.

Besides package execution, npx offers several more helpful functions. It can be used, for instance, to run scripts that are part of packages or to quickly write and run shell scripts. It's also possible to use it to run commands from a file URL or a GitHub repository, which simplifies testing and using packages that aren't listed in the npm registry.

Taking into account everything, npx is an effectual and flexible instrument for running packages directly from the npm registry without having to install them either locally or globally in your project. It will prove as a great tool for developers to test latest packages or newer version of existing packages. Moreover, having extensive features set will ease any node JS development activity.

CREATE-REACT-APP:

This command helps to build the react-native applications and installs all the basic packages at one go. The developer need not to install the packages one by one. It saves the developers time to setup and configure. To use this command the developer needs to install node and npm/yarn. It creates a file named package.json that writes all the dependencies and libraries that were installed in that project.

API DATA FROM SEARCH API:

Search jobs posted on LinkedIn, indeed, ZipRecruiter, and others on Google for Jobs in real-time, all in a single API. JSearch is the best Global Job Search API solution. As the most comprehensive and highly-maintained option available, JSearch empowers us to seamlessly access most-up-to-date job postings and salary information in real-time from Google for Jobs - the largest job board in the world.

TO RUN THE PROJECT:

To start the expo application, we use a command `expo start`. This command needs a virtual android emulator running as well and after giving the command `expo start` it automatically opens in the emulator and whenever any changes are made in the code the changes are reflected in the application without reloading the application. We can also run our application for testing purposes in our mobile phone also by scanning the QR code that appears in the CLI.

NODEMON:

It ensures that the `node.js` applications can be restarted when changes are detected in the directory. To start using nodemon, one has to install this tool first before opening the backend file to do it. Nodemon is responsible for wrapping the application which makes it possible to pass the normal arguments onto the app. The command used to run the backend file should be `nodemon filename.js`.

One of the widely used utilities in the Node.js application's development workflow is Nodemon. Auto restart for Node.js keeps track of file changes within the project folder and reboots the app for a fluid programming flow.

This is especially beneficial for developers involved in bigger and more complicated `node.js` based projects that entail constant coding changes. Nodemon automates the repetitive task of restarting the server anytime anything is changed. This leads to higher production levels and shorter development cycles.

DOTENV:

Dotenv manages configuration variables and environmental parameters through JavaScript, making it an invaluable tool for React Native development. Dotenv greatly enhances the handling of sensitive data and environment-specific configurations in React Native applications with its simple and lightweight methodology.

Dotenv makes it easier to save sensitive data in a secure `.env` file, such as server URLs, API keys, and other configuration variables. This improves the application's overall security posture by guaranteeing that sensitive data stays secret and is not revealed in the source code. Dotenv's adaptability is especially useful when switching between environments, including development, testing, and production. React Native developers can increase flexibility and

durability by using Dotenv to quickly adjust configurations for every environment instead of hardcoding data straight into the codebase.

Dotenv offers a simple implementation approach and integrates with React Native projects with ease. Utilizing Dotenv, developers can quickly load environment variables, guaranteeing that customizations are readily available across the application without compromising security.

Dotenv helps create a cleaner and more structured project structure by centralizing configurations in a `special.env` file. Because developers can easily find and handle all environment-specific variables in one convenient area, this promotes code readability and maintenance.

CHEERIO:

Within the React Native development framework, Cheerio presents itself as an efficient instrument that offers effective JavaScript-based HTML parsing capabilities. Cheerio's adaptability allows for the easy extraction and manipulation of data from HTML text, which improves the usefulness of React Native applications.

Because Cheerio uses a syntax similar to jQuery, developers with prior expertise with jQuery will find it easy to use. For individuals who are already familiar with jQuery's syntax, this makes the switch easier and shortens the learning curve for integrating HTML parsing functionality into React Native apps.

Cheerio's main advantage is that it can easily extract and manipulate data from HTML documents. Cheerio helps React Native developers to quickly integrate external data into their applications by navigating the DOM, picking individual items, and extracting pertinent information.

When it comes to data scraping in React Native applications, Cheerio is quite helpful. Cheerio makes the process of obtaining product information from e-commerce websites or gathering content from internet sources easier by offering a reliable and effective way to parse HTML content.

Cheerio's lightweight and speedy design maximizes HTML text processing performance. This efficiency is especially helpful when developing React Native applications, as resource optimization is essential to providing a seamless user experience.

Cheerio's smooth integration with the larger JavaScript ecosystem is consistent with the language's extension and flexibility. This enables developers using React Native to integrate Cheerio with additional JavaScript frameworks and tools to produce feature-rich and reliable applications.

JSON WEB TOKEN:

By using JavaScript to create an efficient and reliable method of token-based authentication, JSON Web Token (jsonwebtoken) is essential to guaranteeing safe authentication in React Native applications. React Native developers can improve the security of user authentication procedures and protect critical data inside their apps by easily adding jsonwebtoken.

Token-based authentication is a popular and safe way to verify user identity, and jsonwebtoken makes it easier. Tokens can be generated and verified by React Native developers using JSON Web Tokens, which improves the overall security of user authentication procedures in their apps.

Encoding a payload into the token is one of the main functions of jsonwebtoken. User-specific data or other information useful to the application's authentication procedure may be included in this payload. Sensitive data is protected during transmission and storage thanks to its encrypted payload.

With the help of jsonwebtoken, developers can easily build tokens with programmable attributes like issuer, audience, and expiration time. Because of this flexibility, React Native developers can customize token characteristics to meet the unique needs of their applications, which provides an additional degree of security and control.

Stateless authentication is made possible via JSON Web Tokens, which do away with the necessity for server-side user session storage. This is consistent with JavaScript's asynchronous and stateless design, which enables React Native apps to effectively handle user authentication without depending on server-side sessions.

React Native apps and backend services can communicate more easily thanks to jsonwebtoken. Developers can facilitate interoperability and consistency by following the JSON Web Token standard, which guarantees that authentication tokens issued on the client side can be readily validated by server-side components.

MONGODB:

Using JavaScript to offer a scalable and adaptable NoSQL database solution, MongoDB shows to be a flexible and effective option for data management in React Native applications. Because of its smooth interaction with React Native, developers can easily manage duties related to data storage and retrieval, which improves the overall performance and responsiveness of mobile applications.

Because MongoDB is a NoSQL database, data modeling is more flexible, enabling React Native developers to dynamically modify the database schema. This adaptability is especially useful in situations when data structures change or differ amongst various React Native application components.

MongoDB adheres to the JavaScript object notation and stores data in BSON (Binary JSON) documents that resemble JSON. Because of this inherent connectivity, working with MongoDB is simple for React Native developers, allowing for simple data manipulation and application integration.

Because of MongoDB's scalable architecture, React Native apps can easily handle expanding datasets and rising user loads. Because of its document-oriented format and support for horizontal scalability, it operates at peak efficiency and offers a snappy user interface.

React Native apps can be updated in real-time with database updates thanks to MongoDB's real-time data synchronization features. This capability is especially helpful for programs that need real-time updates, such chat apps or teamwork platforms.

The query language used by MongoDB is recognizable to developers of JavaScript. React Native programmers don't need to have a deep learning curve to retrieve and handle data from databases because they can create queries using the JavaScript syntax they already know.

MONGOOSE:

Mongoose is a valuable tool for developing React Native applications since it serves as a strong Object Data Modeling (ODM) library for MongoDB. By utilizing JavaScript, Mongoose simplifies communication with MongoDB databases and offers React Native developers an organized and user-friendly approach to handling data models, queries, and validations.

Schema validation is integrated into Mongoose, allowing React Native developers to guarantee data consistency and integrity. Through the definition of needed fields, certain data types, and validation rules inside the schema, Mongoose makes sure that data stored in MongoDB follows set guidelines, minimizing the possibility of errors.

With the help of Mongoose's fluid API, developers of React Native may create intricate queries with a familiar JavaScript syntax for MongoDB. The process of dealing with the database is made easier and more efficient by this abstraction.

Population is a feature of Mongoose that makes data linking and referencing in MongoDB easier. Population is a tool that React Native developers can use to automatically replace paths in a document with real data, which can speed up the process of getting relevant data from various collections.

EXPRESS:

Express is a flexible and lightweight online application framework for Node.js that is essential to the creation of React Native applications because it offers a strong base upon which to construct scalable backend services. Express, which uses JavaScript, makes it easier to create middleware, routing, and APIs, allowing React Native developers to easily integrate server-side features into their mobile applications.

Express gives React Native developers the flexibility to organize their backend code anyway they see fit because of its simple and neutral design. Because of its adaptability to different project requirements, Express facilitates a customized and effective development experience.

With its robust routing mechanism, Express makes it easier to define API endpoints. React Native programmers find it easy to establish a well-structured and transparent API, which facilitates handling various routes and HTTP methods and improves the maintainability of backend code.

For React Native applications to handle requests from several origins, Cross-Origin Resource Sharing management is a must. Express has built-in support for this. The establishment of controlled and secure cross-origin communication between the client and the server is made easier by this functionality.

3.2 PROJECT DESIGN AND ARCHITECTURE

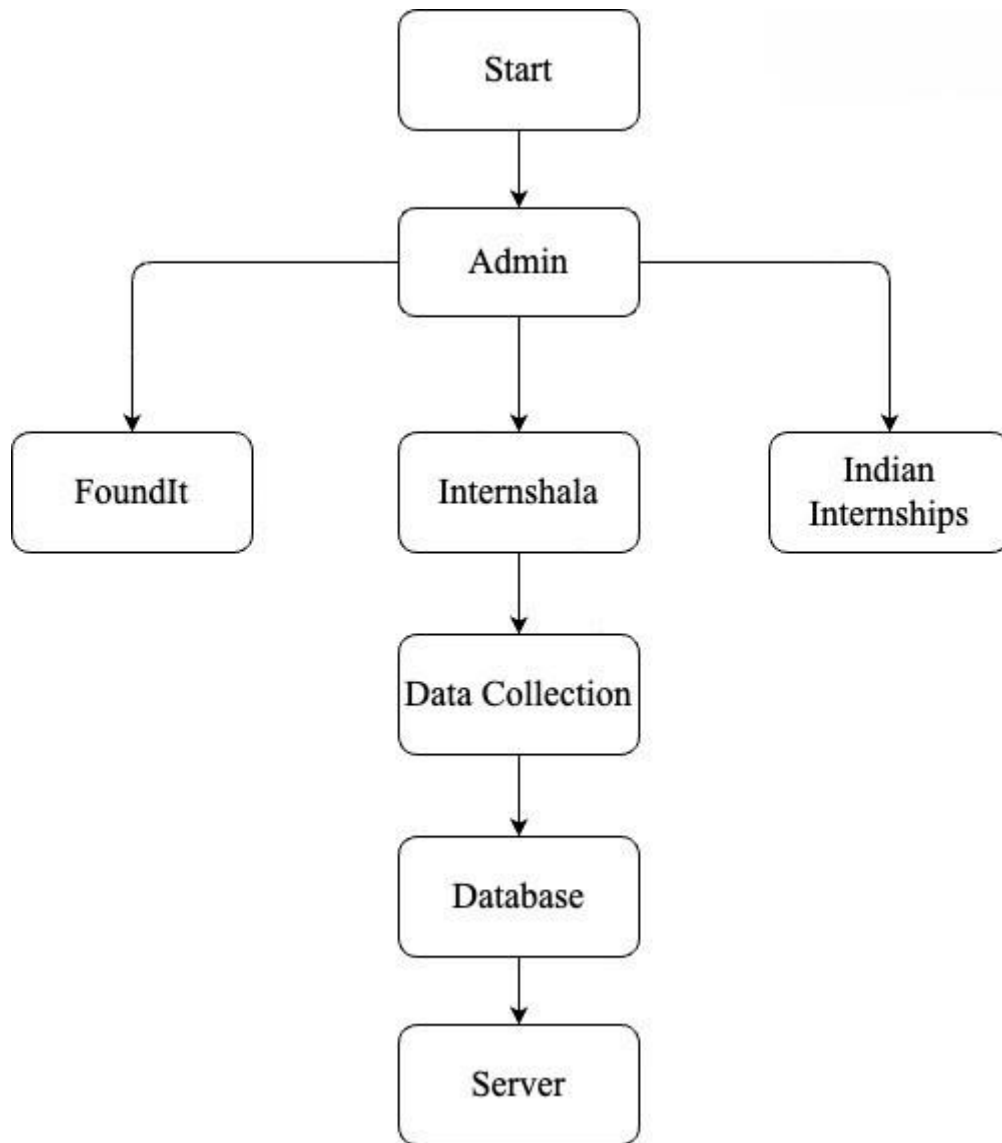


Figure 3.1: Backend Flowchart

This flowchart of the application creation process is displayed in the chart. The administrator is the first to gather information, doing so from FoundIt, Internshala, and Indian Internships. Next, a database is used to store this data. The webpage is then created by the Server using the information from the Database. Although the chart lacks specificity, it provides a broad picture of the procedure. An application's development, testing, and design phases are probably only a

few of the numerous processes that go into making one. Nonetheless, the process's primary steps are depicted in the chart:

1. Collect Data
2. Store data in a database
3. Use data to create the application

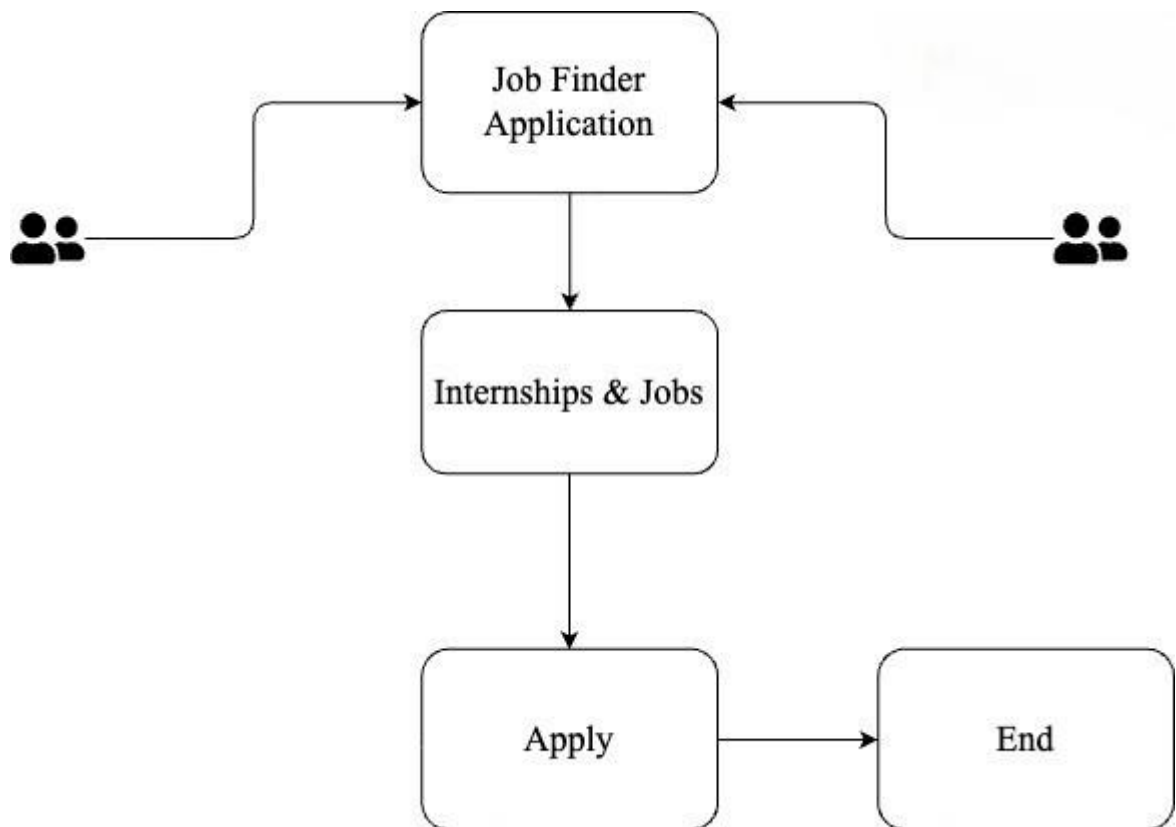


Figure 3.2: Frontend Flowchart

The Main Output Window

The main output window that consists of parent component i.e index.js has other components that are also called as the child component of the index.js. The parent component consists of Welcome component, Popular Jobs component, Nearby Jobs component etc.

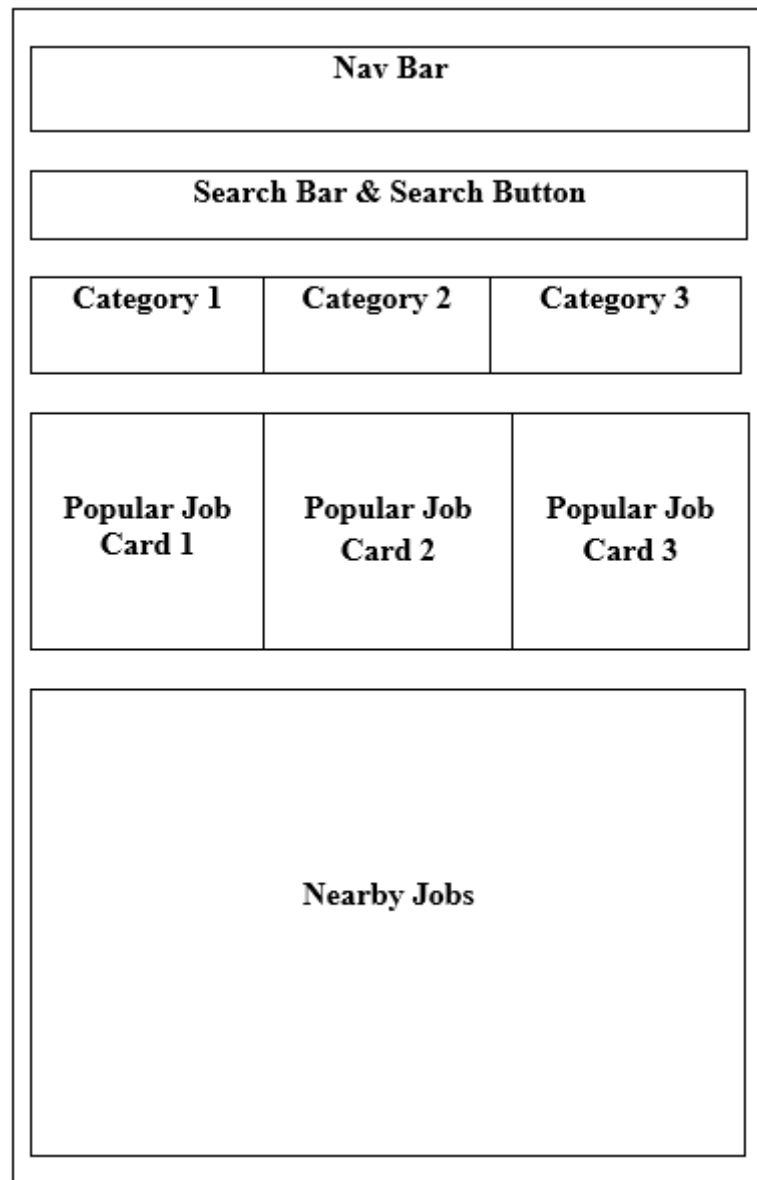


Figure 3.3: Home Screen Output Window

The Navbar consists of various icons that will be further connected to perform their activities like the menu icon will be responsible to open the sidebar then the image on top right will be responsible for the credentials of the user. Search bar consists of a search bar, in there we can search for any keyword and we will find job suggestions. According to that particular keyword it also has a little search icon on the right of the search bar, the following operation will be performed after clicking that search button. Then after this there is the category of the jobs like full time jobs, part time jobs etc. By clicking on these buttons made for categorizing the jobs we will find jobs according to that. After this there are popular job cards which we can slide towards to explore more

jobs. Then in the end of the home screen there is a Nearby jobs container which will show you the jobs nearby you.

Search & Category Screen

This component consists of a Navbar component with a back button, then after that there is a Job Opportunities component with searched keyword or category type, then below that is all the jobs falling under that category, and then finally there are buttons for navigating the screen with screen numbers.

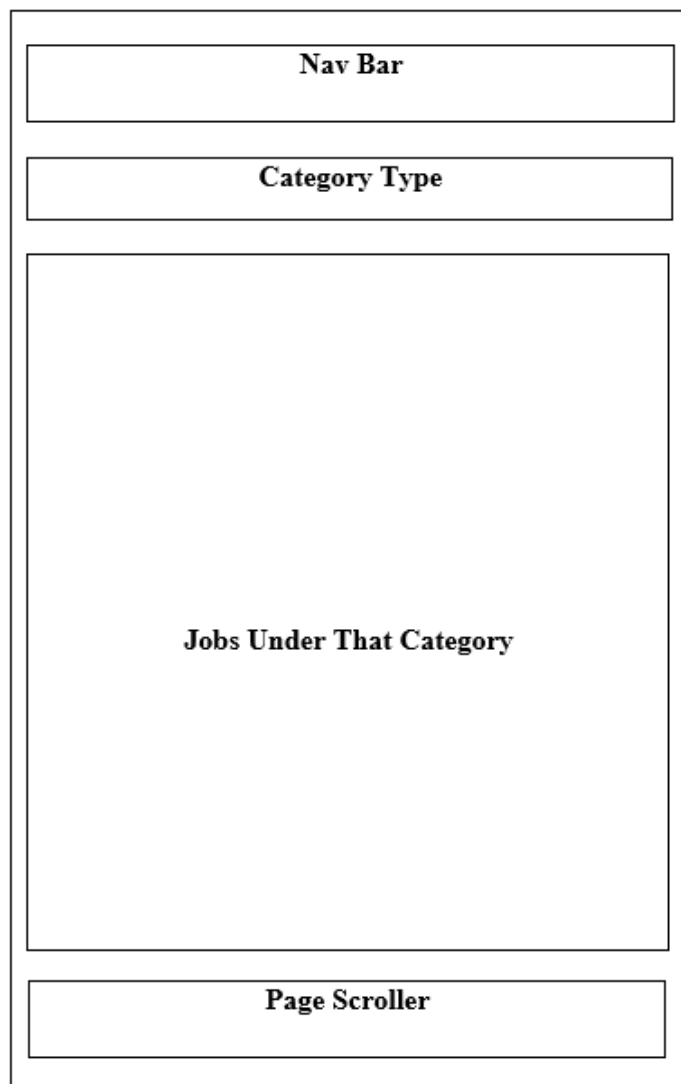


Figure 3.4: Category/Search Screen

Job Description Screen

In this section first there is a navbar with back button and a share button then below the navbar component there is the section containing about the logo of company and the job role then below this there is three buttons (About, Qualification, Responsibilities), then below that is description according to the button clicked in the above section, and then finally there is Apply button for the job.

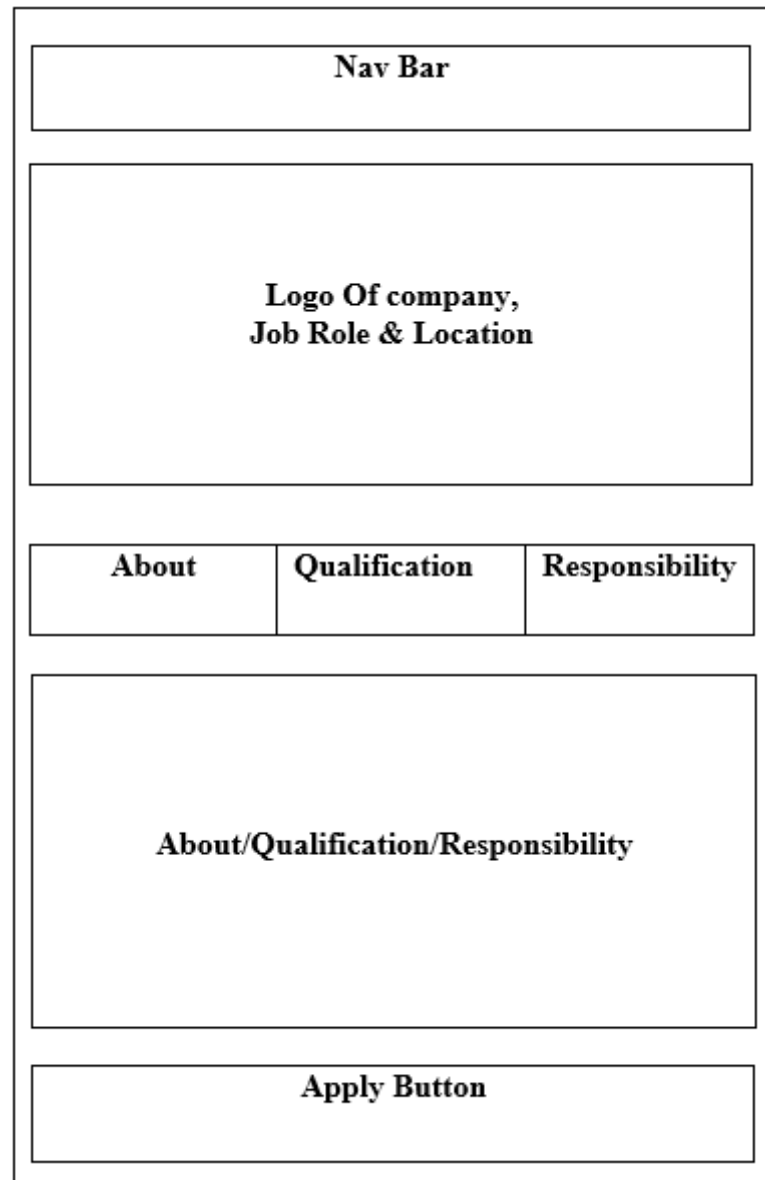


Figure 3.5: Job Description Screen

ARCHITECTURE

CLIENT-SIDE ARCHITECTURE

React Native is the framework used in the application's cross-platform mobile development. The architecture is composed of reusable and independent components that handle the logic and rendering of particular user interface elements.

STATE MANAGEMENT

To handle the state of the application in a centralized and predictable way, a state management solution like Redux or React Context API is used. This guarantees effective data flow and makes it easier for various components to synchronize their states.

NETWORK REQUESTS (AXIOS)

This is achieved by using a popular JavaScript HTTP client known as Axios for handling network queries. Moreover, this solution allows communicating with external APIs efficiently to retrieve user details, job ads as well as relevant data.

SERVER-SIDE ARCHITECTURE (EXPRESS WITH MONGODB)

For management of APIs (backend architecture), there is a stable base on the Express framework for Node.js. It is decided to use the flexible NoSQL database MongoDB that has an easy interoperability with Java Script.

DATA MODELLING (MONGOOSE)

In regards to structured data modeling, Mongoose is an ODM package used with MONGO Db system. This improves schema validation, better performance of query, and integration with MongoDB.

3.3 DATA PREPARATION

A key component of the development process for the "Job Finder" project was the careful compilation of data to guarantee a rich and varied pool of job listings for users. This round of data preparation involved gathering employment data from multiple sources, such as Indian Internship, Foundit, and Internshala, and organizing it into a MongoDB database. The goal was

to build an extensive and current database of job openings that could be accessed via the application.

DATA SCRAPING FROM MULTIPLE SOURCES:

1. FOUNDTIT

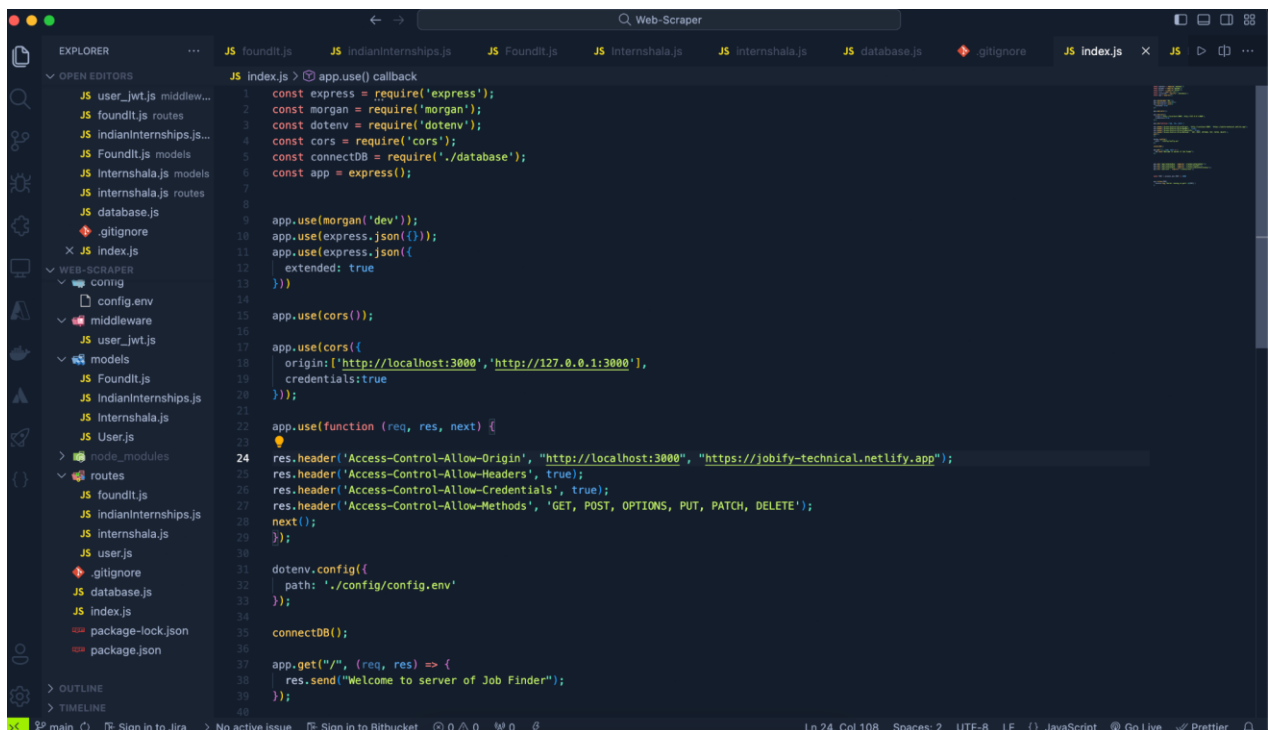
The database foundit (data scrape) provided a useful repository of job postings; information related to applications was also retrieved among others (application, name, description). Data was retrieved from a structured approach of web browsing using web scraping tools or libraries.

2. INDIAN INTERNSHIP

Data scraping techniques have been used to get job advertisements, company information, and other relevant material by utilizing the professional network Indian Internship.

3.4 IMPLEMENTATION

CODE SNIPPETS



```
JS index.js > app.use() callback
1  const express = require('express');
2  const morgan = require('morgan');
3  const dotenv = require('dotenv');
4  const cors = require('cors');
5  const connectDB = require('./database');
6  const app = express();
7
8
9  app.use(morgan('dev'));
10 app.use(express.json());
11 app.use(express.json({
12   extended: true
13 }));
14
15 app.use(cors());
16
17 app.use(cors({
18   origin: ['http://localhost:3000', 'http://127.0.0.1:3000'],
19   credentials: true
20 }));
21
22 app.use(function (req, res, next) {
23
24   res.header('Access-Control-Allow-Origin', 'http://localhost:3000', 'https://jobify-technical.netlify.app');
25   res.header('Access-Control-Allow-Headers', true);
26   res.header('Access-Control-Allow-Credentials', true);
27   res.header('Access-Control-Allow-Methods', 'GET, POST, OPTIONS, PUT, PATCH, DELETE');
28   next();
29 });
30
31 dotenv.config({
32   path: './config/config.env'
33 });
34
35 connectDB();
36
37 app.get("/", (req, res) => {
38   res.send("Welcome to server of Job Finder");
39 });
40
```

Figure 3.6: Entry file of backend

```
routes > JS indianInternships.js > Intern
1 const cheerio = require('cheerio');
2 const express = require('express');
3 const axios = require('axios');
4 const Intern = require('../models/IndianInternships');
5 const router = express.Router();

router.post('/addIndianInternship', async (req, res, next) => {

  var IndianInternships = [];

  try {

    await axios.get("https://www.indianinternship.com/")
      .then(response => {
        const htmlContent = response.data;
        const $ = cheerio.load(htmlContent);

        $('article').each((index, element) => {
          const title = $(element).find('h2.entry-title a').text();
          const ctc = "Performance Based";
          const url = $(element).find('h2.entry-title a').attr('href');
          const startDate = "Starts Immediately";
          const image = "data:image/jpeg;base64,/9j/4AAQSkZJRgABAQAAQABAAIDvCEAAKGBwGHBUrBwgWfHUWGBkbFhgYFSEg1BoYH1EdIBoHJRoICgk1
          IndianInternships.push({ title, ctc, url, startDate, image });
        });

      });

    IndianInternships = IndianInternships.filter((month, idx) => idx < 25);

    Intern.insertMany(IndianInternships).then(function (err, result) {
      console.log("Inserted documents into the collection");
    });
  } catch (err) {
    console.log(err);
  }
});
```

Figure 3.7: fetching data from Indian Internships

```
routes > JS foundit.js > ...
1 const cheerio = require('cheerio');
2 const express = require('express');
3 const axios = require('axios');
4 const Intern = require('../models/FoundIt');
5 const router = express.Router();

router.post('/addFoundIt', async (req, res, next) => {

  var FoundIt = [];

  try {

    await axios.get('https://www.foundit.in/search/it-jobs')
      .then(response => {
        const htmlContent = response.data;
        const $ = cheerio.load(htmlContent);

        $('cardContainer').each((i, el) => {
          const result = $(el).find('div.infoSection h3 a').text() + "-" + $(el).find('div.infoSection companyName span a').text();
          let title = result.substring(0, result.indexOf("-") + 0);
          const ctc = "Performance Based";
          const url = $(el).find('div.infoSection h3 a').attr('href');
          const startDate = "Starts Immediately";
          const image = "data:image/png;base64,iVBORw0KGgoAAANSUHUgAAAEAAADhCAMAAAABJbSjIAAAAbLBHVEX///8KCwAAADX7fsAk93///78/

          FoundIt.push({ title, ctc, url, startDate, image });
        });

      });

    } catch (err) {
      console.log(err);
    }

    Intern.insertMany(FoundIt).then(function (err, result) {
      console.log("Inserted documents into the collection");
      res.status(200).json({ FoundIt });
    });
  }
});
```

Figure 3.8: fetching data from foundit


```

1 import { useState } from "react";
2 import { View, ScrollView, SafeAreaView, Text } from "react-native";
3 import { Stack, useRouter } from "expo-router";
4 import { COLORS, icons, images, SIZES } from "../constants";
5 import {
6   NearbyJobs,
7   PopularJobs,
8   ScreenHeaderBtn,
9   Welcome,
10 } from "../components";
11
12 const Home = () => {
13   const router = useRouter();
14   const [searchTerm, setSearchTerm] = useState("");
15
16   return (
17     <SafeAreaView style={{ flex: 1, backgroundColor: COLORS.lightwhite }}>
18       <Stack.Screen
19         options={{
20           headerStyle: { backgroundColor: COLORS.lightwhite },
21           headerShadowVisible: false,
22           headerLeft: () => (
23             <ScreenHeaderBtn iconUrl={icons.menu} dimension="60%" />
24           ),
25           headerRight: () => (
26             <ScreenHeaderBtn iconUrl={images.profile} dimension="100%" />
27           ),
28           headerTitle: "",
29         }} />
30     </Stack.Screen>
31     <ScrollView showsVerticalScrollIndicator={false}>
32       <View
33         style={{
34           flex: 1,
35           padding: SIZES.medium,
36         }}
37       >
38         <Welcome
39           searchTerm={searchTerm}
40           setSearchTerms={setSearchTerm}
41           handleClick={() => {
42             if (searchTerm) {
43               router.push(`/search/${searchTerm}`);
44             }
45           }} />
46         <PopularJobs />
47         <NearbyJobs />
48       </View>
49     </ScrollView>
50   </SafeAreaView>
51 );

```

Figure 3.9: Root frontend component

This component is the root component of the application we are calling every component like search component, popular job component nearby jobs component etc. here in this component, also the Api call for searching the jobs are also being called in this root component.

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS POSTMAN CONSOLE
This version of expo-cli is not supported anymore.
It's highly recommended to update to the newest version.

The API endpoints used in this version of expo-cli might not exist,
any interaction with Expo servers may result in unexpected behaviour.

This command is being executed with the global Expo CLI. Learn more: https://blog.expo.dev/the-new-expo-cli-f4250d8e3421
To use the local CLI instead (recommended in SDK 46 and higher), run:
> npx expo start

Starting project at C:\Users\Wohnish\OneDrive\Desktop\react_native_jobs
Some dependencies are incompatible with the installed expo package version:
- react-native - expected version: 0.71.14 - actual version installed: 0.71.3
Your project may not work correctly until you install the correct versions of the packages.
To install the correct versions of these packages, please run: expo doctor --fix-dependencies,
or install individual packages by running: expo install [package-name ...]
Starting Metro Bundler

[QR Code]

> Metro waiting on exp://172.16.118.129:19080
> Scan the QR code above with Expo Go (Android) or the Camera app (iOS)

> Press a | open Android
> Press w | open web

> Press r | reload app
> Press m | toggle menu

> Press ? | show all commands

Logs for your project will appear below. Press Ctrl+C to exit.

```

Figure 3.10: Metro Bundler

This command line interface is used by expo to bundle the javascript code and assets. The metro bundler starts when we write the command expo in the cli of the root directory of the project folder. There is a QR code in the metro bundler with the use of this QR code we can run the application in our own smartphone, also there are options to open the application in android/IOS emulators also in here we can reload the application if needed.

```
1 import { useState, useEffect } from "react";
2 import axios from "axios";
3
4 const useFetch = (endpoint, query) => {
5   const [data, setData] = useState([]);
6   const [isLoading, setIsLoading] = useState(false);
7   const [error, setError] = useState(null);
8
9   const options = {
10     method: "GET",
11     url: "https://web-scraper-backend-okkj.onrender.com/api/internship/findfoundit",
12     params: { ...query },
13   };
14
15   const fetchData = async () => {
16     setIsLoading(true);
17
18     try {
19       const response = await axios.request(options);
20
21       setData(response.data.data);
22       setIsLoading(false);
23     } catch (error) {
24       setError(error);
25       console.log(error);
26     } finally {
27       setIsLoading(false);
28     }
29   };
30
31   useEffect(() => {
32     fetchData();
33   }, []);
34
35   const refetch = () => {
36     setIsLoading(true);
37     fetchData();
38   };
39
40   return { data, isLoading, error, refetch };
41 };
42
43 export default useFetch;
44
```

Figure 3.11: useFetch.js

This useFetch component is the api call that we used in our application where we are connecting the server to our frontend with the use of asynchronous function calls.

TECHNOLOGIES AND LANGUAGES USED

For creating the Job Finder Application, we have used various technologies -

JAVASCRIPT

JavaScript is the language that is used for client-side programming in web development. This language helps the developer to add dynamic behavior and interactivity to the web pages that are to be designed by the user.

JavaScript is an object-oriented programming language. JavaScript provides the developer with various Application Programming Interfaces also abbreviated as APIs

to work with regular expression, some of the standard data structures, dates, texts and also with Document Object Model that is abbreviated as DOM.

JavaScript provides the developers to create interactive websites that can change the web page content as well as the styles dynamically. The best part about JavaScript is that it provides frameworks that add more functionality to the web applications.

The frameworks are ReactJS, React Native, Node.js etc. JavaScript even provides some functions so that web applications could interact with users. Some of the functions are alert, prompt and confirm. When we submit anything on the web then sometimes we get a prompt or alert from the web application. So, this is possible because of JavaScript functionality.

The arrow functions that will be used in our project are part of Javascript. The arrow functions allow the developer to write the shorter function syntax. These functions are used when developers want to access properties of this, inside the callback.

REACT NATIVE

Facebook created the open-source React Native framework to expand JavaScript's functionality for creating mobile apps. It offers a strong and effective method for creating cross-platform apps with a single codebase for both iOS and Android. JavaScript is used by React Native to bring the adaptability and excitement of web development to the world of mobile apps.

Component-based architecture is what React Native uses, much like React for web development. By building reusable and modular user interface components, developers can improve maintainability by promoting code structure. This is consistent with the object-oriented design of JavaScript, which makes it understandable to developers with a web development experience.

Rapid updating is one of React Native's best features. With the help of this feature, developers can see the immediate results of code modifications without having to recompile the whole program. The rapid development cycle that hot reloading promotes is in line with the dynamic and responsive character of JavaScript, encouraging a flexible and adaptive approach.

By serving as an interface between JavaScript and native components, React Native gives developers access to features that are unique to each platform. This guarantees that a React Native application may utilize the entire range of native capabilities, enhancing the user experience with features specific to each platform.

React Native can expose common developers' JavaScript APIs so that they share a similar development experience. Some of these APIs are useful for different functions like navigation, handling touch inputs, and device capabilities. This improves the flexibility of JavaScript for mobile software development.

React Native uses JavaScript as an asynchronous programming language that works perfectly with network requests. Hence, this makes it easy to integrate the react native application to the backend services thus guarantees efficient data sharing between the server and mobile app.

AXIOS

Such as ease of making an HTTP process when developing a React Native application is one of the popular JavaScript packages called Axios. The flexibility of JavaScript makes Axios a simple and user-friendly tool for developers that work with APIs and communicate information between the React Native applications and the back-end services.

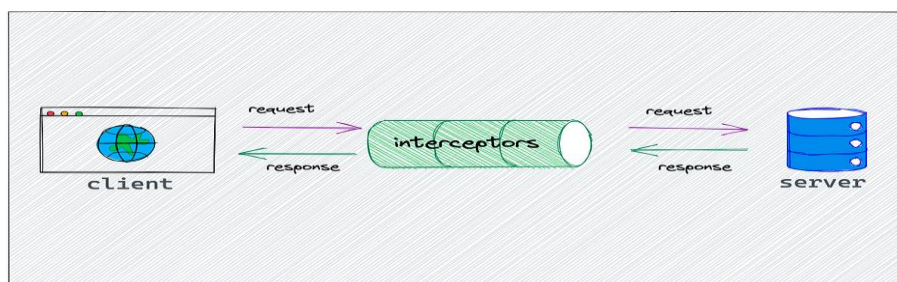


Figure 3.12: UI/Server communication

Axios uses Promises to handle asynchronous actions in accordance with the asynchronous nature of JavaScript. This fits in nicely with the React Native development approach, enabling programmers to manage asynchronous operations like data fetching without interfering with the main thread of the application.

Strong error handling features offered by Axios enable developers to handle potential HTTP request problems with grace. The library also allows for response detection, which allows responses to be customized even before they are sent to the application. A more regulated and customized integration with backend services is ensured by this flexibility.

Cross-Origin Resource Sharing (CORS) issues are a frequent concern in web and mobile development, and Axios tackles them. Axios is a dependable option for React Native developers working with APIs hosted on separate domains because it handles CORS-related concerns by default.

Axios's asynchronous design fits quite well with React Native elements. In order to ensure a dynamic and responsive user experience, developers can use Axios into their React Native applications to swiftly acquire data and update component states.

Axios makes it simple for developers to switch between web and React Native applications by maintaining a consistent API across various platforms. The consistent application of the library makes development easier, especially for developers who are already acquainted with JavaScript and Axios in web development.

RAPID API

React Native application API integration may be made easier and faster with the help of Rapid API. Rapid API uses JavaScript's flexibility to give developers a single location to find, connect, and manage a wide range of APIs. By easily integrating external services, data sources, and features, this integration improves the usefulness of React Native applications.

With the platform's single dashboard for managing APIs, developers can effectively track performance, keep an eye on usage, and manage API keys. This centralized method simplifies the process of integrating external services into React Native apps, in line with JavaScript's goal of user ease.

One of the most important parts of API integration is made easier with RapidAPI: managing API keys. Through the platform, developers can produce, track, and secure API keys with ease, improving security and giving them more control over how the API is used in React Native applications.

Code snippets for several programming languages, including JavaScript, are available at Rapid API. With ready-to-use code snippets that easily interact with the selected APIs, this functionality speeds up the construction of React Native applications. This supports quick iteration and deployment in accordance with the agile development concepts of JavaScript.

Rapid API may be used with JavaScript because of its built-in cross-platform features, which also apply to iOS and Android React Native applications. By using the platform, developers can get access to a variety of APIs that are suited to the unique requirements of cross-platform mobile development.

EXPO

With Expo, developers can now update their React Native apps over-the-air (OTA) without forcing consumers to download an update from the app store. This is consistent with the dynamic nature of JavaScript, allowing developers to easily make updates and enhancements.

Expo simplifies the process of building and deploying React Native applications. The iOS and Android platforms allow developers to create apps without having to make platform-specific changes. This streamlined approach boosts productivity and aligns with JavaScript's objective of streamlining development.

Expo allows React Native developers to access features like the camera, location, and push notifications with ease by seamlessly integrating with native device functionalities. By utilizing these integrations, JavaScript developers can produce feature-rich and complex mobile applications without having to work extensively with native code. Expo offers a variety of development tools together with a robust Command Line Interface (CLI). These tools make debugging, compiling, and initial project setup easier. Expo's CLI allows JavaScript developers to effectively handle different parts of the development lifecycle, which improves the overall development experience.

3.5 KEY CHALLENGES

Job finder faced different issues during its implementation period. Web scraping for extracting data turned out to be a multistage process with different job sites having their own peculiarities

and anti-scraping strategies. Coming up with an intuitive yet sturdy frontend was no easier task. It necessitated careful design principles and optimal UX, while also taking into account the variety of ways people are likely to interact with it. Another challenge was bridging the barrier that separated the frontend with its counterpart in the back-end systems such that the interactions that the users had should have been translated into the operations of the backend smoothly. These called for creative solutions to issues such as robust scraping for data extraction, perfecting UI for better users' experience, and coordination between the frontend and backend systems. Overcoming these barriers was important for providing an integrated and effective "Job Finder" app.

CHAPTER 4: TESTING

4.1 TESTING STRATEGY

JEST TESTING

In order to verify the correctness and operation of each module, function, and component in the React Native application, Jest testing is essential. Ensuring that every separated code unit performs as intended is the main objective, which adds to the application's overall resilience and reliability.

Unit tests are run using the Jest testing framework in conjunction with the React Testing Library. Using a comprehensive methodology, potential defects are found and fixed early in the development lifecycle by assessing React components, functions, and modules separately. A comprehensive analysis of every code unit is made easier by Jest's interaction with React Testing Library, which offers an extensive collection of tools for rendering components, interacting with them, and producing assertions.

REACT COMPONENTS TESTING

Jest allows rendering and interaction scenarios to be simulated for React components. This covers processing user interactions, validating changes to component states, and evaluating the rendering result. One of Jest's features, snapshot testing, takes a picture of the rendered output of the component. By serving as a baseline, this snapshot enables developers to identify inadvertent modifications in ensuing test runs, guaranteeing the reliability of user interface elements.

FUNCTIONS AND MODULE TESTING

Beyond components, Jest makes it easier to create focused test cases, which guarantees the accuracy of functions and modules. This entails testing the behavior of individual modules and evaluating the expected output of functions given particular inputs. Developers can prove the accuracy of their code logic by declaring expectations about the results of various scenarios with efficiency thanks to Jest's assertion features.

POSTMAN TESTING

Validating the functionality of API endpoints and evaluating the smoothness of the backend and frontend integration are the main goals of using Postman for API testing. This testing stage is essential to ensuring correct communication and data flow within the application by making sure the backend services react appropriately to different requests made by the frontend.

With its easy-to-use interface for creating, sending, and analysing HTTP requests, Postman is a complete solution for API testing. The following are the main components of this testing procedure as they relate to the "Job Finder" application:

REQUEST CREATION AND CONFIGURATION

With Postman, testers may build and customize HTTP requests that resemble those sent by the frontend to communicate with the backend API. This entails defining the headers, request parameters, request bodies, and request methods (GET, POST, PUT, and DELETE).

ENDPOINT VALIDATION

Every API endpoint undergoes methodical testing to guarantee that it reacts suitably to diverse kinds of requests. Verifying that the endpoints deliver the anticipated status codes, headers, and response bodies is part of this validation process. It guarantees that the API specifications are followed and the backend services are implemented correctly.

DATA FLOW AND COMMUNICATION

Examining the data flow between the frontend and backend is made easier with Postman. Testers evaluate each API endpoint's data transmission, reception, and processing to make sure that the communication channels are open and operating as intended. Ensuring the integrity of data throughout the application is crucial.

4.2 TEST CASES AND OUTCOMES

TEST CASE:

Scenario: Connecting to the localhost server.

Steps:

- Run the application locally.
- Execute a test request to the API endpoint at `http://localhost:3000/api/test`.

Expected Outcome: In this case, response should denote successful communication with the correct status code like 200 OK and a message confirming the same.

Scenario: Trying accessing an non-existing API resource on localhost.

Steps:

- Run the application locally.
- Execute a test request to an invalid API endpoint (e.g., `http://localhost:3000/api/invalid`).

Expected Outcome: In case of such a situation, the response should show an error status code, for instance 404 Not Found and a message that the endpoint does not exist.

Scenario: Test API endpoint with parameters.

Steps:

- Run the application locally.
- Execute a test request to an API endpoint with parameters.

Expected Outcome: This response should consider proper treatment of parameters, returning appropriate data or response on the basis of the entered values.

Scenario: Check API endpoint response format.

Steps:

- Run the application locally.
- Execute a test request to an API endpoint (e.g., `http://localhost:3000/api/info`).

Expected Outcome: The response should also be formatted in the specified manner like JSON and provide appropriate information concerning the application's status or details.

Outcome:

- All scenarios pass successfully.
- Jesbt confirms that these are API endpoints, that they respond to different situations correctly, and accordingly give out the expected results on localhost. This enables the communication between the frontend and the localhost server to be in order when doing local development and testing.

CHAPTER 5: RESULT & FINDINGS

5.1 RESULTS

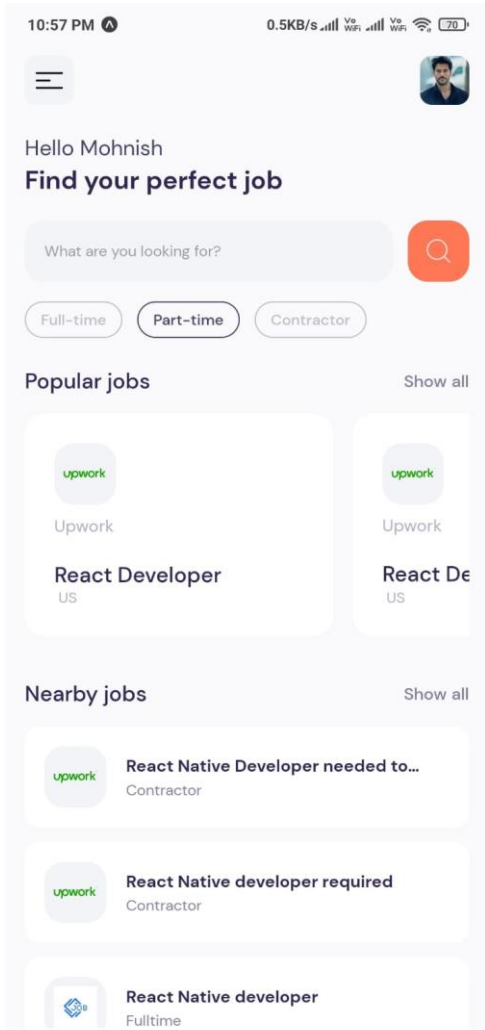


Figure 5.1: Home Screen

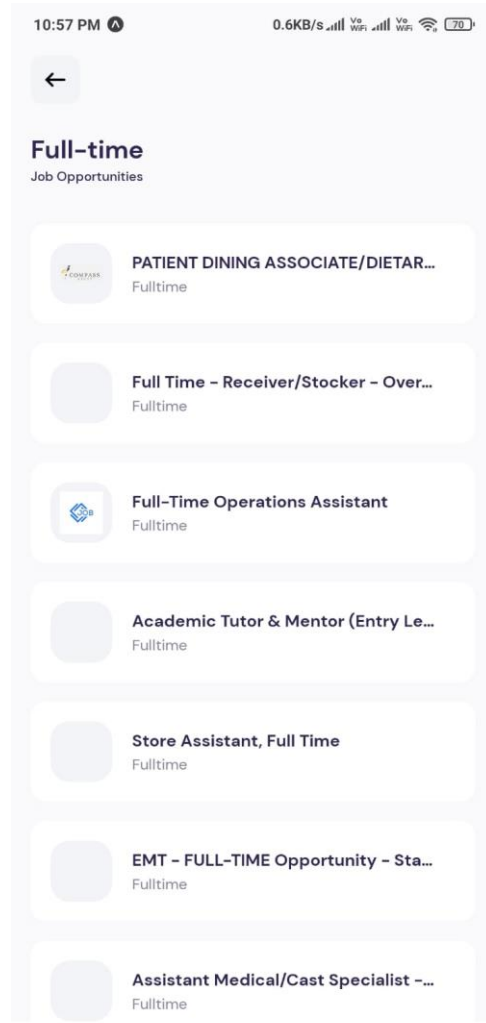


Figure 5.2: Full time category screen

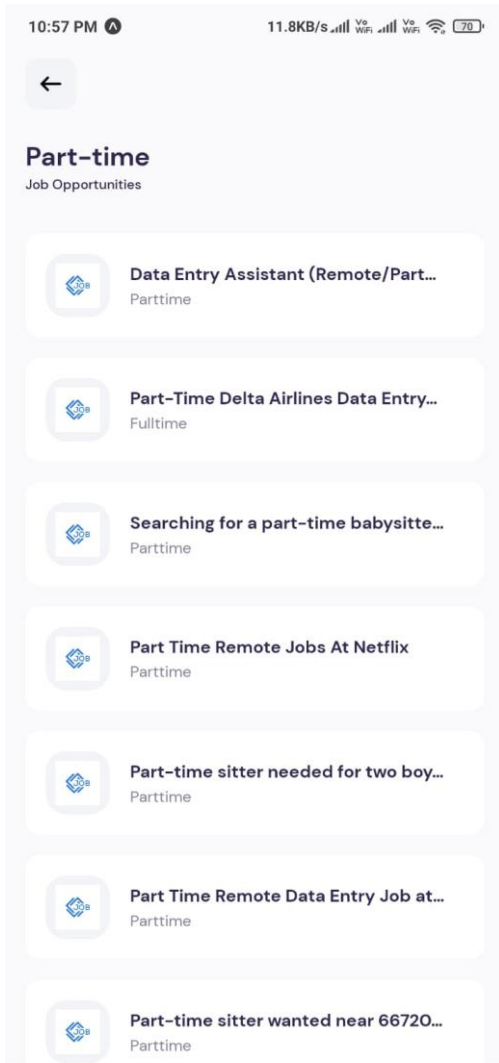


Figure 5.3: Part time category screen

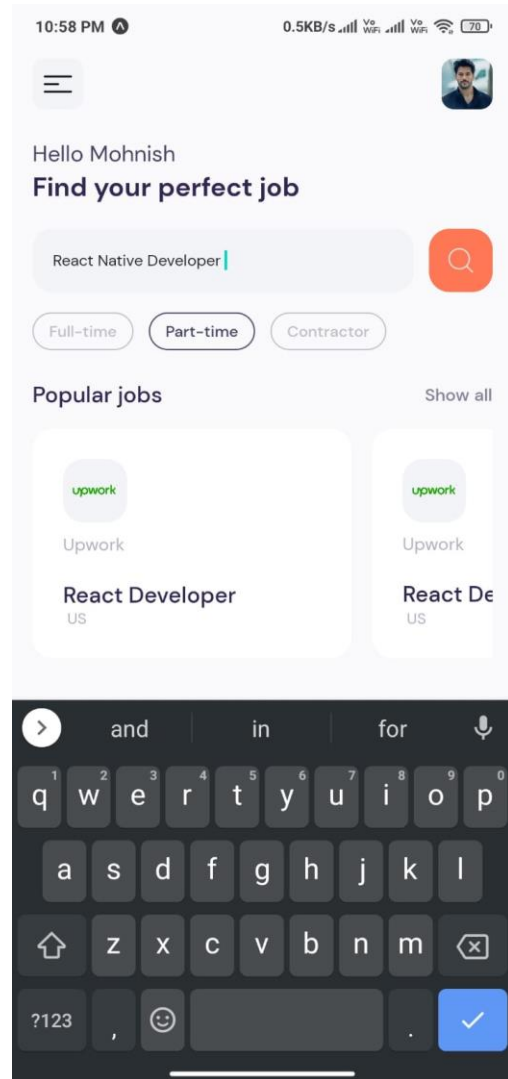


Figure 5.4: Search Functionality

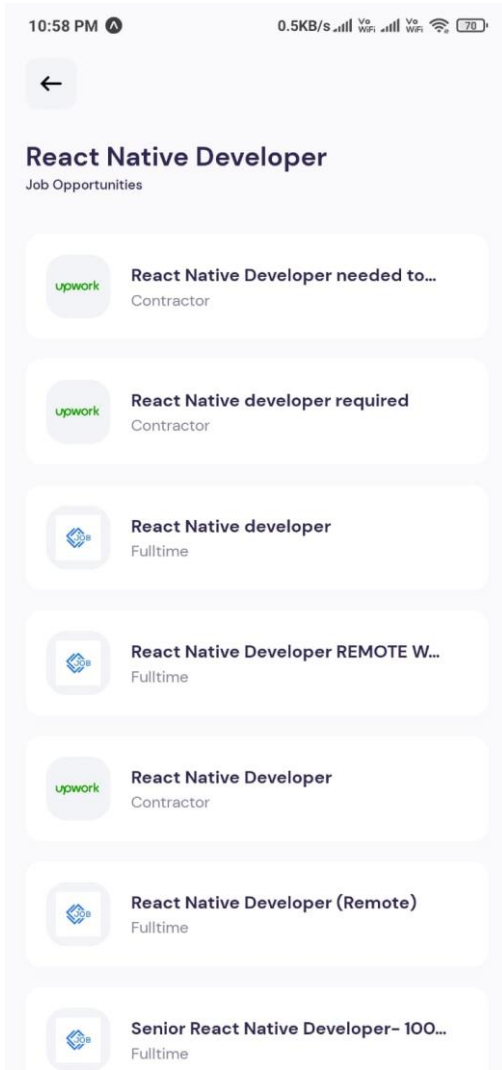


Figure 5.5: Search Results

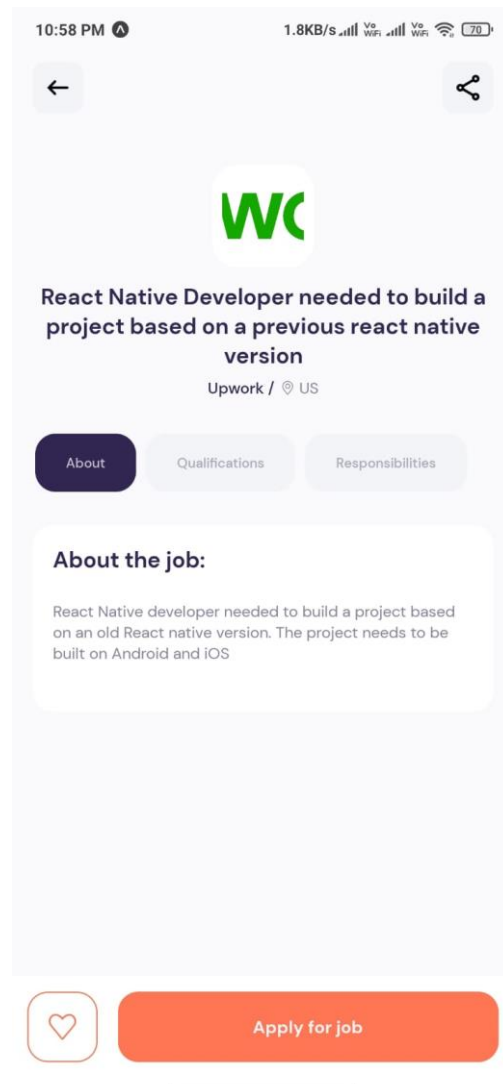


Figure 5.6: Job Detail Screen

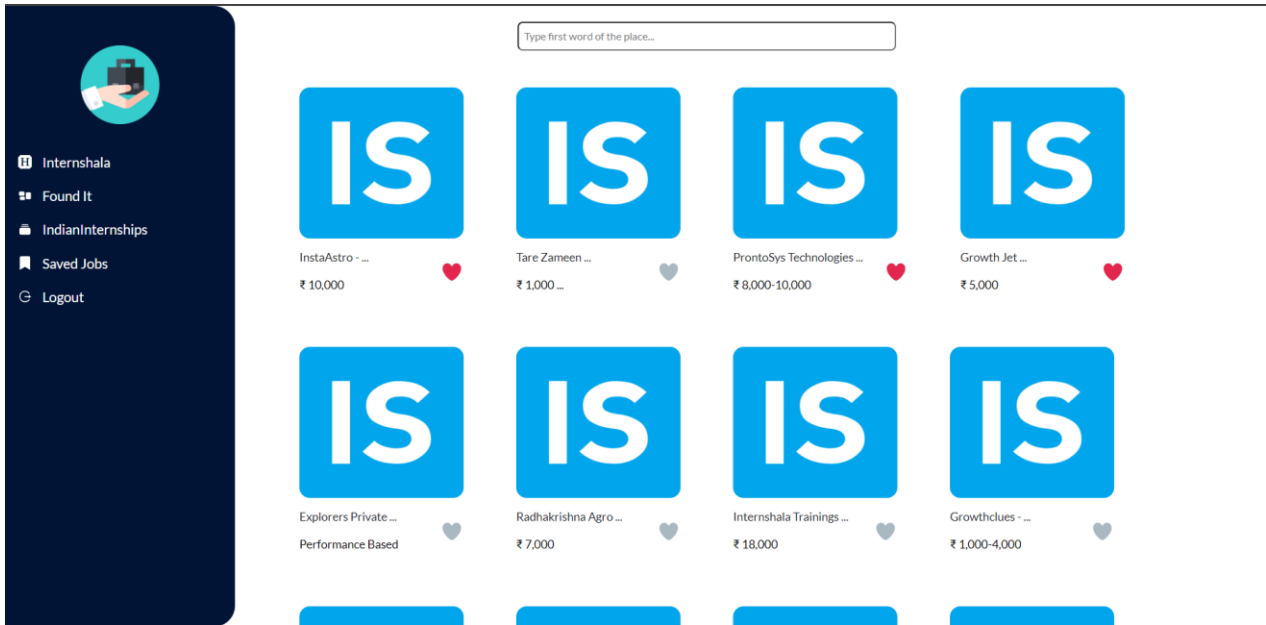


Figure 5.7: Website User Interface

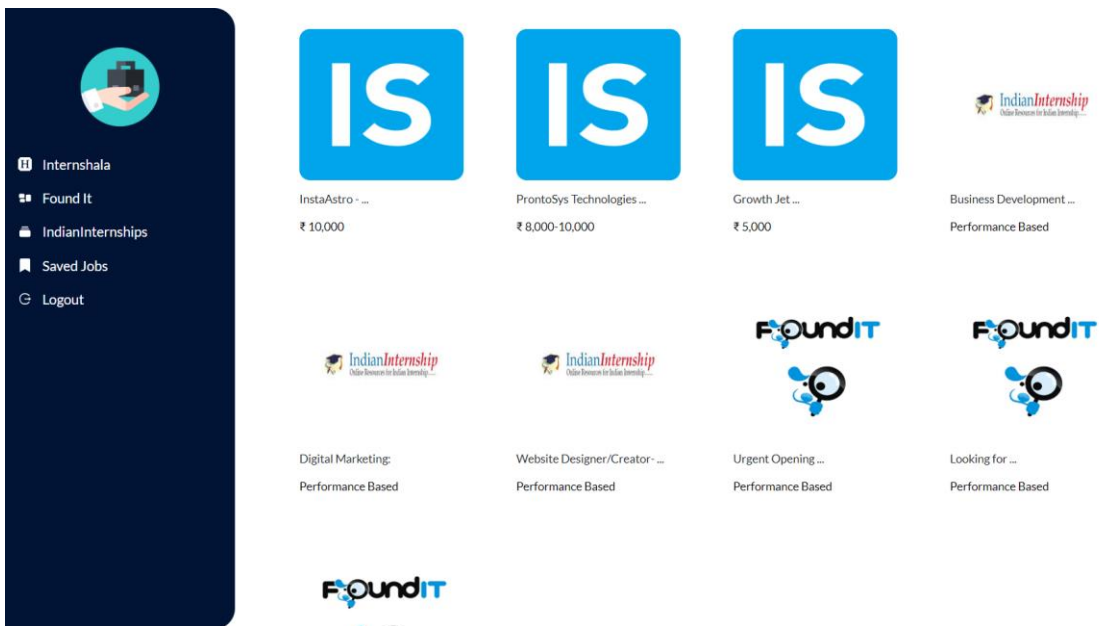


Figure 5.8: Saved Jobs

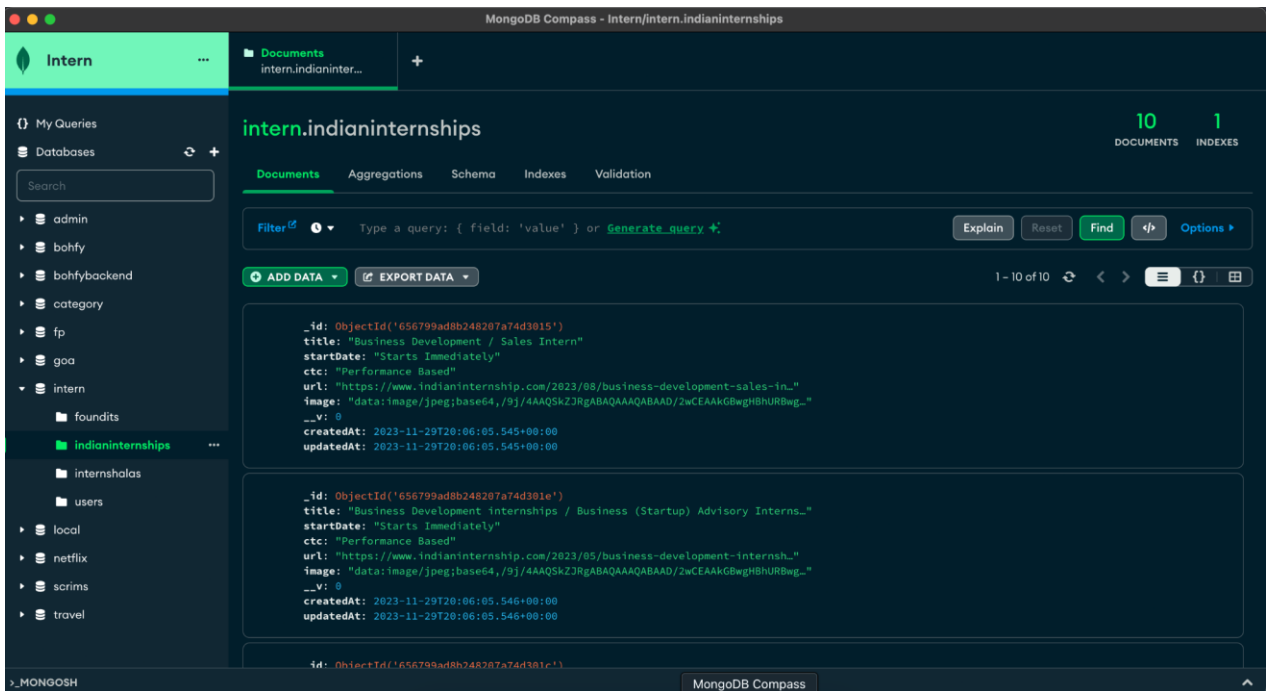


Figure 5.9: Indian internships database

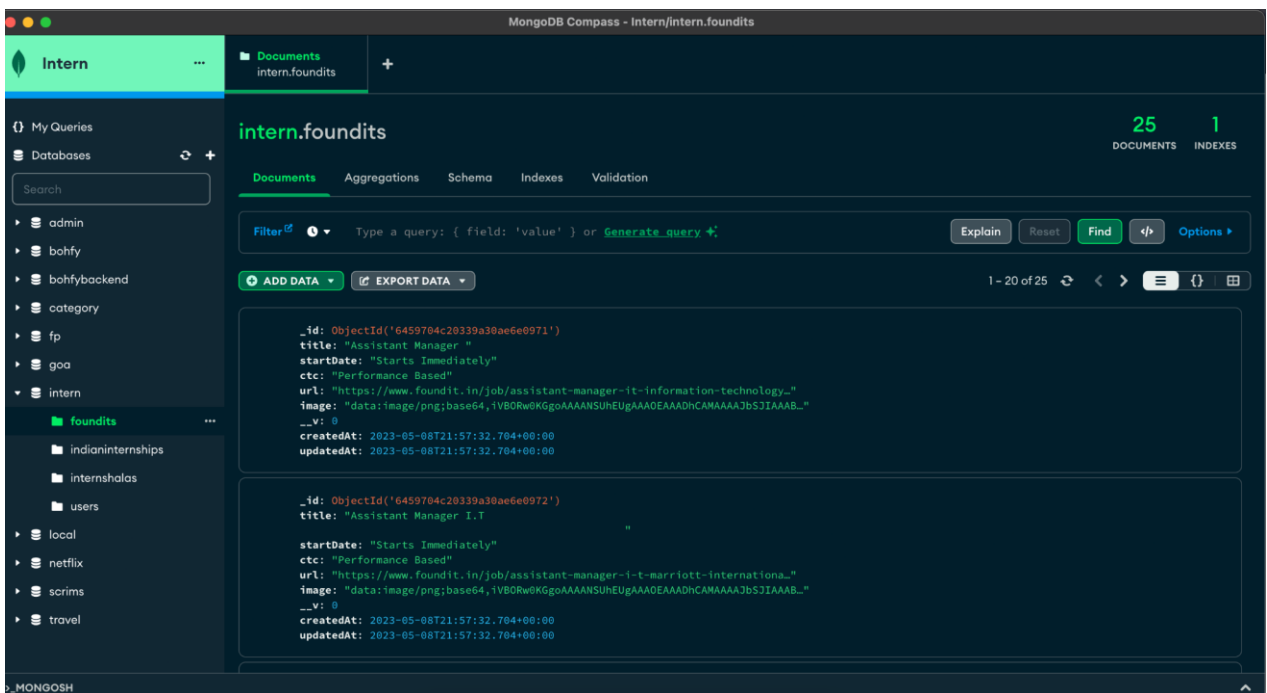


Figure 5.10: FoundIt database

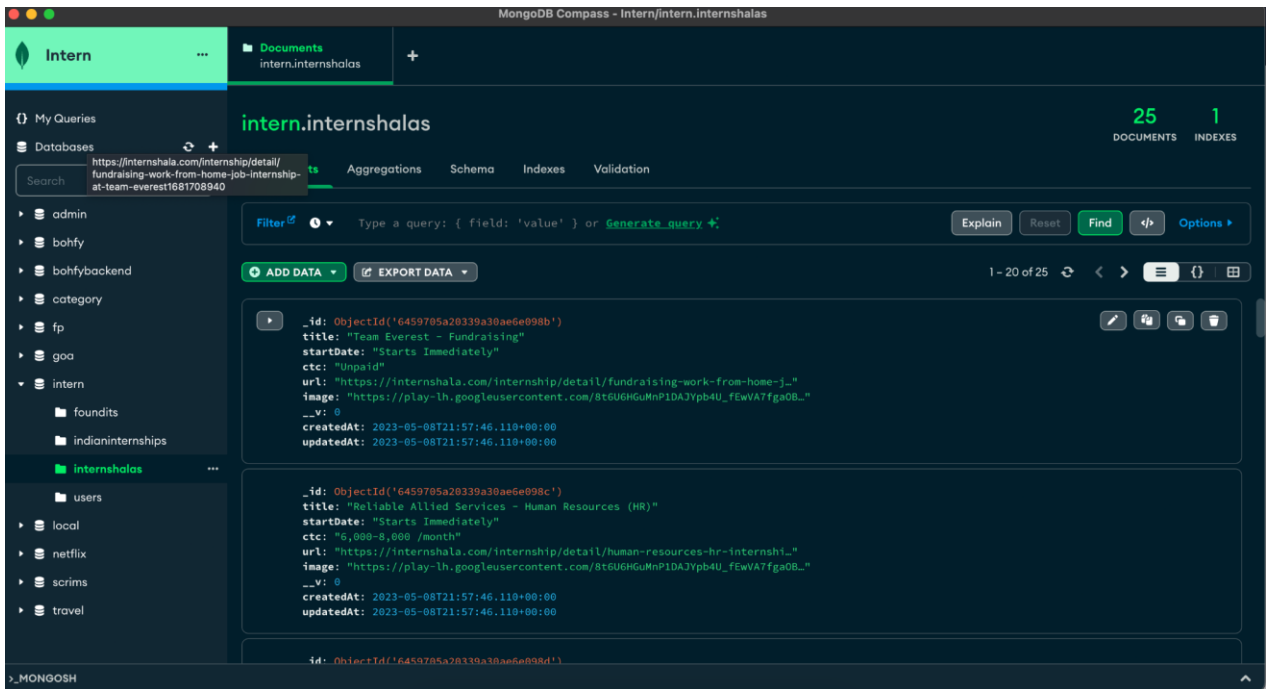


Figure 5.11: Internshala database

CHAPTER 6: CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

The project's journey has resulted in a summary of important discoveries, an open admission of its flaws and a study of significant advances in the field. This conclusion acts as a compass, pointing out the importance of our effort and laying out the plan for further advancements.

KEY FINDINGS

Our technique has produced several important conclusions. It was carefully developed for the accurate summary of job profiles. Above all, the system extracts relevant information from a variety of user profiles with remarkable accuracy and precision. By use of an extensive testing plan, we have verified its effectiveness in achieving the main goals of the project.

The chapter's comparative findings shed light on how the system stacks up against other efforts in the field of profile summarization and job search applications. The findings support an ongoing cycle of innovation and development by offering an in-depth understanding of the system's advantages and shortcomings.

User experience and satisfaction being crucial parts of an application's success, user testing and feedback have evaluated them. Responses that are positive show the ease of use of the system, and those that are constructive in nature help improve the user interface as well as the end user experience.

LIMITATIONS

Just like in any other project, we are fully aware that there were some constraints that defined and molded our findings. Such limitations include particular functions, data source boundaries, and some aspects of performance. These limitations need to be seen as opportunities that will lead to future enhancement of the system.

CONTRIBUTIONS TO THE FIELD

Therefore, our project will be taken as a significant enhancement for profiling of summary and search applications regarding jobs. However, it presents the new method of job search, like we are taking the job data from various sites like Indian Internship, Foundit and Internshala etc.

This makes use of technologies like React Native, Expo, Axios and Rapid API that are shaping an ever-advancing toolkit in cross-platform development of cellphones and application programming interface (API) interoperability. Such a systematic approach for data preparation, database storage and application integration is useful to developers and researchers who are in the same domain.

6.2 FUTURE WORK PLAN

INCREASE DATA SOURCE DIVERSITY

Extending the variety of databases will also enable provision of more diverse job listings for the users. It, therefore, involves adding other platforms or APIs that serve the various industries, career positions, and localities that are not provided for by a single platform. The inclusion of multiple sources of data will enable the application to present a more expansive set of employment positions that will appeal to different categories of users

CONDUCT DATA CLEANING

Data cleaning should be effective since it enables presenting accurate and reliable information to the end-users. In this case, the identification of errors, inconsistencies, and outliers in the dataset will be addressed. It has to perform strong data cleaning like identification of outliers, removal of missing values, check on dataset consistency so that people can get high quality job listing.

ADDING SECURITY MEASURES IN THE BACKEND

Above all, data of users should be safe, at any time. To strengthen the backend with more security the following steps are used; encryption protocol, secure data transmission using HTTPS and strong authentication. The security in the system should

consist of regular security audits and updates which could enable the identification of the vulnerabilities that put at risk the confidentiality and integrity of user information.

IMPROVING UI OF THE APPLICATION

The process of enhancing UI is a step towards making the entire experience of users better. Such changes might include reconstructing parts for easy accessibility, creating better organization structure, adjusting to user feedback to improve the interface's simplicity and attractiveness. User oriented interface plays an important role in engaging the users' interest as well as enhancing their contentment with the product.

REFERENCES

- [1] <http://docs.spring.io/spring/docs/current/spring-framework-reference/html/index.html>[Sep.10, 2016].
- [2] Saurabh Shukla, Saif Ali Khan, Harsh Kumar Singh, Manmohan Sharma, "Online Job Search Application", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 7 Issue 2, pp. 520-525, March-April 2021. Available at <https://ijsrcseit.com/CSEIT217296>
- [3] Wadhawan, Seema, and Smrita Sinha. "Factors Influencing Young Job Seekers Perception towards Job Portals." AIMS International Journal of Management 12.3 (2018): 199-212
- [4] Adkins, C. L., Werbel, J. D., & Farh, J. L. (2001). A field study of job insecurity during a financial crisis. *Group & Organization Management*, 26, 463– 483. <http://dx.doi.org/10.1177/1059601101264004> Ajzen, I. (1991). The theory of planned behavior. *Organizational Behavior and Human Decision Processes*, 50, 179 –211. [http://dx.doi.org/10.1016/0749-5978\(91\)90020-T](http://dx.doi.org/10.1016/0749-5978(91)90020-T).
- [5] J. Dorn and T. Naz , “Integration of Job portals by Meta- search,” in Proc. 3rd International Conf. on Interoperability for EnterpriseSoftware and Applications, Funchal, Portugal,2007, pp. 401-412.
- [6] Mochol, Malgorzata, Holger Wache, and Lyndon Nixon. "Improving the accuracy of job search with semantic techniques." Berlin, Germany, 2007.
- [7] Jang, J. S. R. (1993). ANFIS: adaptive-network-based fuzzy inference system. *IEEE Transactions on System, Man, and Cybernetics*, 23(5/6), 665 – 685.
- [8] <https://www.scribd.com> for taking different references.
- [10] S. Bsiri, M. Geierhos, and C. Ringlstetter, “Structuring job search via local grammars,” *Advances in Natural Language Processing andApplications*, pp. 201, 2008.
- [10] A. Doyle, *Internet Your Way to a New Job:How to Really Find a Job Online*, Happy about, 2008.

- [11] S. Mauno, U. Kinnunen, and M. Ruokolainen, "Job demands and resources as antecedents of work engagement: A longitudinal study," *Journal of Vocational Behavior*, vol. 70, 2007, pp. 149-171.
- [12] M. Mansourvar, *Development of a Job Web Portal to Capture Industry's Needs*, 2011.
- [13] A. Weber and H. Mahringer, "Choice and success of job search methods," *Empirical Economics*, vol. 35, no. 1, pp. 153-178, 2008
- [14] M. Mochol, H. Wache, and L. Nixon, "Improving the accuracy of job search with semantic techniques," in *Proc. Business Information Systems, 10th International Conf., BIS 2007*, Poznan, Poland, April 25-27, 2007, Springer, pp. 301-313
- [15] S. Mauno, U. Kinnunen, and M. Ruokolainen, "Job demands and resources as antecedents of work engagement: A longitudinal study," *Journal of Vocational Behavior*, vol. 70, 2007, pp. 149-171
- [16] E. Galanki, "The decision to recruit online: a descriptive study," *Career Development International*, vol. 7, pp. 243-251, 2002.
- [17] M. Gangle, "The only way is up? Employment protection and job mobility among recent entrants to European labor markets," *European Sociological Review*, vol. 19, pp. 429, 2007.
- [18] M. Mansourvar and N. B. M. Yasin, "Knowledge portal: a tool to capture university requirements," in *Proc. 2011 International Conf. on Graphic and Image Processing, International Society for Optics and Photonics*, October 2011, pp. 82850F-82850F.
- [19] Dorn, Jürgen, and Tabbasum Naz. "Integration of Job portals by Meta-search." *Enterprise Interoperability II*. Springer, London, 2007. 401- 412.
- [20] Kapse, Avinash S., Vishal S. Patil, and Nickil V. Patil. "E-recruitment." *International Journal of Engineering and Advanced Technology* 1.4 (2012): 82-86.
- [21] Mansourvar, Marjan, and Norizan Binti Mohd Yasin. "Development of a job web portal to improve education quality." *International Journal of Computer Theory and Engineering* 6.1 (2014): 43.

[22] Faliagka, Evanthia, et al. "Application of machine learning algorithms to an online recruitment system." Proc. International Conference on Internet and Web Applications and Services. 2012.

[23] Kelley, Erin M., Christopher Ksoll, and Jeremy Magruder. "How do Online Job Portals affect Employment and Job Search? Evidence from India." (2020).

[24] Agerström J, Björklund F, Carlsson R, Rooth DO. 2012. Warm and competent Hassan = cold and incompetent Eric: a harsh equation of real-life hiring discrimination. *Bas. Appl. Soc. Psychol.* 34:359–66.

[25] Okediran, O. O., Arulogun, O. T., Ganiyu, R. A., & Oyeleye, C. A. (2014). Mobile operating systems and application development platforms: A survey. *International Journal of Advanced Networking and Applications*, 6(1), 2195.

APPENDIX

(include additional material to support your chapters, like project plan, code snippets,

additional data, glossary of terms, user manual, business plan (for entrepreneurship project),

plagiarism certificate, etc. Plagiarism certificate should be the last page of your project report.

ORIGINALITY REPORT

10%

SIMILARITY INDEX

8%

INTERNET SOURCES

6%

PUBLICATIONS

7%

STUDENT PAPERS

PRIMARY SOURCES

1	ijsrcseit.com Internet Source	1%
2	5wwwwww.easychair.org Internet Source	1%
3	Submitted to University of Wales, Bangor Student Paper	1%
4	github.com Internet Source	1%
5	carlsonschool.umn.edu Internet Source	1%
6	imm.demokritos.gr Internet Source	1%
7	www.researchgate.net Internet Source	<1%
8	Submitted to University of Essex Student Paper	<1%
9	Drigas, A.. "An expert system for job matching of the unemployed", Expert Systems With Applications, 200402	<1%

10

Submitted to The University of the West of Scotland

Student Paper

<1 %

11

Zheng Siting, Hong Wenxing, Zhang Ning, Yang Fan. "Job recommender systems: A survey", 2012 7th International Conference on Computer Science & Education (ICCSE), 2012

Publication

<1 %

12

media.neliti.com

Internet Source

<1 %

13

Submitted to Africa Nazarene University

Student Paper

<1 %

14

Submitted to University of Nevada Reno

Student Paper

<1 %

15

Submitted to Management Development Institute Of Singapore

Student Paper

<1 %

16

Submitted to Koforidua Technical University

Student Paper

<1 %

17

rapidapi.com

Internet Source

<1 %

18

www.slideshare.net

Internet Source

<1 %

19	ijarcce.com Internet Source	<1 %
20	fdocuments.us Internet Source	<1 %
21	Submitted to Educational Service District 105 Student Paper	<1 %
22	experts.umn.edu Internet Source	<1 %
23	Submitted to Universiteit van Amsterdam Student Paper	<1 %
24	Submitted to Westcliff University Student Paper	<1 %
25	vdocuments.mx Internet Source	<1 %
26	hdl.handle.net Internet Source	<1 %
27	ir.lib.uwo.ca Internet Source	<1 %
28	www.aims-international.org Internet Source	<1 %
29	www.ir.juit.ac.in:8080 Internet Source	<1 %
30	Ruth Cortez, Alexander Vazhenin. "Virtual model-view-controller design pattern:	<1 %

Extended MVC for service-oriented architecture", IEEJ Transactions on Electrical and Electronic Engineering, 2015

Publication

Exclude quotes Off

Exclude matches Off

Exclude bibliography Off