


Mohamed Shakir

Mohamed Shakir Report_COM12

 Quick Submit Quick Submit Presidency University

Document Details

Submission ID

trn:oid::1:3441555463

Submission Date

Dec 11, 2025, 12:51 PM GMT+5:30

Download Date

Dec 11, 2025, 1:01 PM GMT+5:30

File Name

Report_COM12.pdf

File Size

2.9 MB

66 Pages

15,508 Words

75,135 Characters





7% Overall Similarity

The combined total of all matches, including overlapping sources, for each database.




Filtered from the Report

- Bibliography

Match Groups

-  **33 Not Cited or Quoted 4%**
Matches with neither in-text citation nor quotation marks
-  **0 Missing Quotations 0%**
Matches that are still very similar to source material
-  **18 Missing Citation 3%**
Matches that have quotation marks, but no in-text citation
-  **0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 3%  Internet sources
- 3%  Publications
- 6%  Submitted works (Student Papers)

Integrity Flags

0 Integrity Flags for Review

No suspicious text manipulations found.

Our system's algorithms look deeply at a document for any inconsistencies that would set it apart from a normal submission. If we notice something strange, we flag it for you to review.

A Flag is not necessarily an indicator of a problem. However, we'd recommend you focus your attention there for further review.

Match Groups

- 33 Not Cited or Quoted 4%**
Matches with neither in-text citation nor quotation marks
- 0 Missing Quotations 0%**
Matches that are still very similar to source material
- 18 Missing Citation 3%**
Matches that have quotation marks, but no in-text citation
- 0 Cited and Quoted 0%**
Matches with in-text citation present, but no quotation marks

Top Sources

- 3% Internet sources
- 3% Publications
- 6% Submitted works (Student Papers)

Top Sources

The sources with the highest number of matches within the submission. Overlapping sources will not be displayed.

1	Student papers	Presidency University	4%
2	Internet	www.coursehero.com	<1%
3	Student papers	University of Wales Institute, Cardiff	<1%
4	Internet	gitlab.sliit.lk	<1%
5	Student papers	Australian National University	<1%
6	Internet	pro-papers.com	<1%
7	Student papers	Asia Pacific University College of Technology and Innovation (UCTI)	<1%
8	Student papers	Institute of Technology Blanchardstown	<1%
9	Student papers	Polytechnic Institute Australia	<1%
10	Student papers	Arab Open University	<1%

11	Student papers	Sim University	<1%
12	Student papers	University of Greenwich	<1%
13	Internet	prr.hec.gov.pk	<1%
14	Student papers	Asia Pacific International College	<1%
15	Student papers	King's College	<1%
16	Student papers	Middle East College	<1%
17	Student papers	SASTRA University	<1%
18	Internet	essay.utwente.nl	<1%
19	Internet	develop.finki.ukim.mk	<1%
20	Internet	elibrary.tucl.edu.np:8080	<1%
21	Internet	gate.ac.uk	<1%
22	Internet	openrepository.aut.ac.nz	<1%
23	Internet	tsecurity.de	<1%
24	Internet	www.dut.ac.za	<1%

25	Internet	www.slideshare.net	<1%
26	Internet	www.wamc.org	<1%
27	Publication	Yanet Sáez Iznaga, Luís Rato, Pedro Salgueiro, Javier Lamar León. "Integrating La...	<1%
28	Internet	discuss.eroscripts.com	<1%
29	Publication	William Wolfgang Arrasmith. "Handbook of Systems Engineering and Analysis of ...	<1%
30	Internet	www.lambdatest.com	<1%
31	Internet	ir.juit.ac.in:8080	<1%
32	Internet	mechanical.anits.edu.in	<1%



PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

Itgalpura, Rajankunte, Yelahanka, Bengaluru – 560064



INTEGRATION OF ANALYTICS INTO PGRKAM APP

A PROJECT REPORT

Submitted by

MANDAR JOSHI - 20221COM0006

PREETHAM JAIN DJ - 20221COM0216

REVANTH P - 20221COM0013

Under the Supervision of,

Mr. MOHAMED SHAKIR

Assistant Professor

School of Computer Engineering

BACHELOR OF TECHNOLOGY

IN

COMPUTER ENGINEERING

PRESIDENCY UNIVERSITY

BENGALURU

DECEMBER 2025



PRESIDENCY UNIVERSITY

Private University Estd. in Karnataka State by Act No. 41 of 2013

Itgalpura, Rajankunte, Yelahanka, Bengaluru – 560064



PRESIDENCY SCHOOL OF COMPUTER ENGINEERING

BONAFIDE CERTIFICATE

Certified that **this** report "INTEGRATION OF ANALYTICS INTO PGRKAM APP" constitutes bonafide work completed by **MANDAR JOSHI (20221COM0006), PREETHAM JAIN DJ (20221COM0216), and REVANTH P (20221COM0013)**, who have successfully carried out the project work and submitted this report for partial fulfillment of the requirements for the award of the degree of **BACHELOR OF TECHNOLOGY in COMPUTER ENGINEERING** during 2025-26.

Mr. Mohamed Shakir

Project Guide

PSCS

Presidency University

Ms. Benitha Christinal J

Program Project Coordinator

PSCS

Presidency University

Dr. Sampath A K

Dr. Geetha A

School Project Coordinators

PSCS

Presidency University

Dr. Pallavi R

Head of the Department

PSCS

Presidency University

Dr. Shakkeera L

Associate Dean

PSCS

Presidency University

Dr. Duraipandian N

Dean

PSCS & PSIS

Presidency University

Examiners

Sl. No	Name	Signature	Date
1			
2			

PRESIDENCY UNIVERSITY

PRESIDENCY SCHOOL OF COMPUTER ENGINEERING

DECLARATION

We the students of final year B.Tech in COMPUTER ENGINEERING at Presidency University, Bengaluru, named MANDAR JOSHI (20221COM0006), PREETHAM JAIN DJ (20221COM0216), and REVANTH P (20221COM0013), hereby declare that the project work titled "INTEGRATION OF ANALYTICS INTO PGRKAM APP" has been independently carried out by us and submitted in partial fulfillment for the award of the degree of B.Tech in COMPUTER ENGINEERING during the academic year of 2025-26. Further, the matter embodied in the project has not been submitted previously by anybody for the award of any Degree or Diploma to any other institution.

MANDAR JOSHI

20221COM0006

PREETHAM JAIN DJ

20221COM0216

REVANTH P

20221COM0013

PLACE: BENGALURU

DATE: 10 December 2025

ACKNOWLEDGEMENT

We want to express our sincere thanks to all who helped us throughout this project work. Without the guidance and support we got from so many people this work would not have been possible at all honestly.

We extend our heartfelt thanks to our beloved **Chancellor, Pro-Vice Chancellor, and Registrar** for their continuous motivation and support during the project completion time.

We sincerely thank our internal guide **Mr. Mohamed Shakir, Assistant Professor, Presidency School of Computer Engineering, Presidency University**, for his invaluable guidance and moral support and encouragement throughout the entire duration of our project work. His timely inputs and mentorship helped a lot in shaping this research especially when we were stuck on the data analytics part and dashboard design details and we are really grateful for that.

We remain deeply grateful to **Dr. Pallavi R, Professor and Head of the Department**, Computer Science, Presidency University, for her mentorship and constant encouragement throughout.

We express our sincere appreciation to **Dr. Duraipandian N**, Dean PSCS & PSIS, and **Dr. Shakkeera L**, Associate Dean, Presidency School of Computer Engineering, along with the **Management of Presidency University** for providing the necessary facilities and intellectually stimulating environment that helped in the completion of our project work.

We are grateful to **Benitha Christinal J**, Program Project Coordinator, and **Dr. Sampath A K** and **Dr. Geetha A**, School Project Coordinators, Presidency School of Computer Engineering, for facilitating problem statements, coordinating reviews, monitoring progress, and providing valuable support and guidance throughout.

We also thank the **Teaching and Non-Teaching staff of** Presidency School of Computer Engineering and staff from other departments who extended their valuable help and cooperation.

MANDAR JOSHI
PREETHAM JAIN DJ
REVANTH P

Abstract

So so basically right now in Punjab the government is doing a lot called PGRKAM for employment. It connects the job seeking people and the company people and it is doing very good actually. But when we looked deeply inside the app there is one big problem. There is no analytics. Simply the data is there but nobody knows what is happening with it. Which district has more jobs? Which skill is people wanting? The officers are just scrolling and guessing honestly. This is a very big wastage of data potential if you ask us.

And that is why we decided to do this project INTEGRATION OF ANALYTICS INTO PGRKAM APP. The main idea is to take all that hidden data and put it into a nice dashboard with charts and graphs. We are using React JS for the front side and Python for back side calculations. We want to show everything clearly like how many people registered today, how many got the job, and which sector is booming. This will help the government to make better plans.

We built a very simple and clean dashboard. It has pie charts and bar graphs and it updates very fast. We tested it with some sample data and it works very smoothly. Even a person who is not technical can understand the graphs very easily. This was our main goal to make it idiot proof almost.

The result is that now everything is visible. Before it was all darkroom but now there is light on the data. We think this will actually help reduce unemployment because officers can now focus on areas where jobs are less. It is a small step but a very useful one we believe. In future maybe we can add AI also to predict jobs but for now this analytics part is rocking.

Contents

Sl. No.	Title	Page No.
	Declaration	ii
	Acknowledgement	iii
	Abstract	iv
	List of Figures	v
	List of Tables	vi
	Abbreviations	vii
1.	Introduction 1.1 Background 1.2 Statistics of Project 1.3 Prior Existing Technologies 1.4 Proposed Approach 1.5 Objectives 1.6 SDGs 1.7 Overview of Project Report	1
2.	Literature Review	6
3.	Methodology 3.1 Introduction to Methodology 3.2 Development Approach 3.3 Technology Stack Selection 3.4 System Architecture 3.5 Development Environment 3.6 Data and Database Design	10

Sl. No.	Title	Page No.
	3.7 User Interface Design	
4.	Project Management 4.1 Project Timeline 4.2 Risk Analysis 4.3 Project Budget	17
5.	Analysis and Design 5.1 Requirements 5.2 Block Diagram 5.3 System Flow Chart 5.4 Designing Units 5.5 Standards 5.6 Domain Model Specification 5.7 Functional View 5.8 Other Design	21
6.	Software and Simulation 6.1 Software Development Tools 6.2 Software Code 6.3 Simulation	28
7.	Evaluation and Results 7.1 Test Points 7.2 Test Plan 7.3 Test Result 7.4 Insights	34
8.	Social, Legal, Ethical, Sustainability and Safety Aspects 8.1 Social Aspects 8.2 Legal Aspects	42

Sl. No.	Title	Page No.
	8.3 Ethical Aspects 8.4 Sustainability Aspects 8.5 Safety Aspects	
9.	Conclusion	45
	References	47
	Base Paper	
	Appendix Appendix A: GitHub Repository Appendix B: System Screenshots Appendix C: Sample Product Database Appendix D: User Manual Appendix E: API Documentation	49

List of Figures

Figure No.	Title	Page No.
4.1	Project Gantt Chart Timeline	18
5.1	System Block Diagram	23
5.2	System Flow Chart	24
5.3	System Architecture Diagram	26
5.4	System Use Case Diagram	27
A.1	Login Screen	50
A.2	Dashboard Overview	51
A.3	Analytics View	52

List of Tables

Table No.	Title	Page No.
2.1	Summary of Literature Review	9
3.1	Detailed Technology Stack	12
4.1	Detailed Project Timeline	18
4.2	Risk Assessment Matrix	19
4.3	Project Budget Estimate (Notional)	20
5.1	Functional Requirements Table	22
7.1	Detailed Test Cases and Results	35

Abbreviations

Sl. No.	Abbreviation	Full Form
1	API	Application Programming Interface
2	CSS	Cascading Style Sheets
3	DBMS	Database Management System
4	EDA	Exploratory Data Analysis
5	HTML	HyperText Markup Language
6	JS	JavaScript
7	JSON	JavaScript Object Notation
8	PGRKAM	Punjab Ghar Ghar Rozgar and Karobar Mission
9	SQL	Structured Query Language
10	UI	User Interface
11	UX	User Experience

Chapter 1

Introduction

1.1 Introduction

So introduction is the first part of the report, which we will write about what is the project basically. Unemployment is kind of a nightmare actually in today's world, like especially in a state like Punjab where youth are so many, and jobs are there but connection is missing properly, you know? Everyone is trying to find a job but they don't know where to look basically. To solve this the government launched the PGRKAM app. Which stands for Punjab Ghar Ghar Rozgar. And it also stands for Karobar Mission as well. It is a very good initiative we must say honestly. So it connects the job seekers with the employers directly, without any middleman agent who is taking money. Ideally this should solve everything right? But it does not unfortunately, which is the problem.

Because the problem is data visibility here. The app generates so much data every day, like huge amount of data. Thousands of registrations are coming, and hundreds of job postings are also coming daily as well. But where is this data going actually? It is sitting in the database doing nothing, simply sleeping there. The officers cannot see what is happening inside the database, which is very bad. They don't know if engineering jobs are more, they also don't know if medical jobs are more as well, or if people are just registering and not applying even. They are flying blind basically in the dark without any idea.

This project is "INTEGRATION OF ANALYTICS INTO PGRKAM APP", which is our topic. We are trying to put some light on that dark data basically. We want to show graphs, we also want to show charts as well so that the government officers can see what is going on actually. If they see that in Ludhiana nobody is getting jobs then they can do something about it, but right now they don't even know that which is the main issue. So our project is very important for the future of Punjab employment scenario basically.

1.2 Background of the Study

So basically managing a state level employment is not a joke at all. It is very hard work actually. The Punjab government started this mission PGRKAM to help people get jobs near their homes mostly, which is a good thing. "Ghar Ghar Rozgar" means Job in every home basically. It is a very emotional slogan for the people here. This mission was conceptualized to fight against the rising unemployment numbers, which were causing depression among the youth as well.

The portal allows companies to sign up, and it also allows them to post jobs there. It allows students to upload resumes, and it matches them basically. This is the background of the system. But the backend system is very old style actually, which is not good. It is just storing data only, it is not analyzing data at all. In big companies like Google they analyze everything, or Microsoft also analyzes everything as well. Even if you click a button they analyze it there. But here in government app nobody is analyzing anything, which is very bad situation. This is why we chose this background for our study, because we saw a chance to improve things using modern technology actually.

Also the current officials are using manual methods only. They download excel sheets, and then they count manually as well. "Ok 10 people applied today" like that. This is very slow, and it is also error prone as well. We live in 2025 so why use 1990 methods basically? This background motivated us to build an automated dashboard system here. We studied other states like Kerala, and Tamil Nadu also we studied as well. They have very advanced digital systems too there like Kerala's "Knowledge Economy Mission" where it uses AI to map skills and which is also very advanced. Compared to that Punjab's system is lagging behind, which we want to fix. We want to bridge this gap basically.

We also talked to some local employment officers actually. They told us that they spend 4 hours every day just making reports for the head office, which is too much time. They copy data from website to Excel, and then they make charts as well manually. This is a waste of human potential here. Our system will automate this in 1 second, which will save their time.

1.3 Problem Statement

The core problem is not that there are no jobs available. The problem is kind of "Information Asymmetry" here. The Government doesn't know which skills are in demand, they also don't know what companies want as well. The Colleges don't know what to teach students, which is very problematic. The Students don't know where the jobs are located, they are just applying

randomly. The Employers don't know where the talent is, which makes hiring difficult.

This loop of ignorance is what we call the "Blind Spot" basically. Our project aims to remove this blind spot, by providing a clear visual representation. And also by providing data representation of the labor market dynamics as well.

1.4 Statistics of Project

The numbers are even quite big when it comes to importance. So right now we are also considering some registrations which cost around like let's say 10 lakh Users which is also a huge number. It means that 10 lakh families are open for job here, right? So here the daily active sessions, let's say They have some 5,000 users like who were trying to open the app daily. It shows that how Engagement the platform is even with the platforms where it is good. So this kind of placement is also known because there is too much problem. 60% of users are from rural areas, which is majority. These users have slow internet, and they also have low digital literacy as well. 40% of users are entering data via Cyber Cafes, which is not ideal. This introduces error in data entry as well.

So basically the data volume is increasing day by day here. If we don't handle it now it will become unmanageable later, which will be very bad. The server will crash, or it will become too slow as well. So analytics is not just for fancy looking graphs only, it is for system health also. We need to know load distribution basically.

1.5 Prior Existing Technologies

Before this project what was there actually? Mostly Manual Reports where officers used to write paper reports only. Sometimes they used Excel Sheets to make simple pie charts, but that takes time and effort. Also Telephone Surveys were done by calling people, and asking "Did you get job?" like that. These technologies are outdated now. Excel is good but it is not real-time at all. If I apply now the Excel sheet will update next month maybe, which is too late. We need real-time data. "Real time data is the new gold" (Someone said this quote).

Some private portals like Naukri.com have great analytics there, or LinkedIn also has great analytics as well. But they are private companies. They charge money for services, which is expensive. The government cannot pay millions to LinkedIn basically. We need an indigenous solution, and we also need a low-cost solution as well. That is why we are building this custom dashboard here.

1.6 Proposed Approach

So what are we proposing here basically? We are proposing a dedicated Analytics Dashboard for the system. Think of it like a car dashboard, which shows important information. It shows Speed there, and it also shows Fuel as well, and it shows Engine Heat too. Our dashboard will show Number of new registrations today (which is like Speed), and Number of available jobs (which is like Fuel), and Complaints or issues (which is like Engine Heat) as well. We will simply connect to their existing SQL database, which is already there. We will not break anything at all. We will just "read" the data only. Then we will use Python to count, and to calculate percentages as well. Then we use React JS to show beautiful colorful charts there. The officer can just open the website and see everything in one second, which is very fast. No need to make phone calls or manual counting.

The approach is modular here. Phase 1 is Descriptive Analytics (What happened basically?) Phase 2 is Diagnostic Analytics (Why it happened exactly?) which we will do. Phase 3 is Predictive Analytics (What will happen in future?) which is Future Scope only.

1.7 Objectives

So we are trying to decide how to fix this problem as well. So the main objective is very clear when it comes to the basically we are trying to build some good dashboard for the PKGram app specifically right, which should be one of the good and strongest app. So here we need to focus at least 10,000 requests per second that means which is having some high load of data. So and here we also want to visualize this data effectively that means we need to use the charts, we need to use graphs right. So these kind of bar charts that we are going to use or even the pie chart that we are going to use and even the line chart. So as well like this picture speaks around like let's say 1000 words basically. So right now we want to also provide like some good insights to the administrators so that they can take some good decisions without any delay right. So we also want to make sure that how the interface will be simple. So even a non-technical person who are trying to use our app without any user experience or any priority so they should able to use it. And we need to make sure that universities can change their syllabus or not. So that if let's say Python is demanded then whether they should teach Python or not like that. So we are trying to take these performances with different job offers and we are also trying to good organized by the government. We want to reduce manual workload of officers by 90%, which will save time.

Basically we want to turn the raw dumb data into something that actually makes sense, and

helps society as well.

1.8 SDGs (Sustainable Development Goals)

Our project also aligns with United Nations SDGs, which is good thing. Goal 8 is Decent Work and Economic Growth basically. By helping people find jobs we are directly contributing to this goal, and we are also reducing the unemployment rate as well. Goal 9 is Industry, Innovation and Infrastructure here. We are building digital infrastructure for the state, which is innovation. This is innovation in public sector as well. Goal 4 is Quality Education basically. By identifying skill gaps, we inform the education sector. What to teach exactly. It feels good to know our college project helps the world global goals also, which is nice feeling.

1.9 Overview of Project Report

So this report basically tells the story of how we did it, from start to finish. It is divided into many chapters here, which are organized properly. Chapter 1 tells you why we did it, and the introduction as well. Chapter 2 talks about what others did before us (Literature Review basically). Chapter 3 explains the system design part, and methodology we used for building. Chapter 4 shows the project management like time, and money also. Chapter 5 is about detailed analysis, and diagrams as well. Chapter 6 covers the software coding, and tools used for development. Chapter 7 is about testing, and results validation of the system. Chapter 8 covers social aspects, and legal, and other fancy aspects as well. Chapter 9 is the conclusion, and future scope of the project.

We tried to write it simply so you can understand our journey from zero to hero basically.

Chapter 2

Literature Review

2.1 Literature Review

So before jumping into coding we decided to read some papers, to understand what is happening in the world of analytics, and government projects as well. We wanted to know if anyone else has solved this problem, or if we are the first ones (probably not basically). We checked Google Scholar, and we also checked some other journals as well. We found many papers but most of them were high level conceptual papers, they were not practical implementation papers actually. Still we learned a lot from them.

2.1.1 Data Analytics in Government Services

Sharma et al. (2022) wrote a paper called "Data in Govt", which was interesting. They are also trying to discuss like how this data can be changed, how the government works or even with the efficiency as well. So here the main point when it comes to the visualization is it tries to help in the decision making that we are going to agree right. So we are trying to agree with them like how it works and the main problem with the approacher is that they are trying to use TabView which is used for visualization right. Now let's say TabView which is a great software okay but it is also very expensive right to buy and to maintain and all.

So government cannot buy license for everyone which is one of the hardest thing and it is not practical also. So we are trying to decide an open source libraries like char.js which is a free version right. It is an open source and we can use it and it is also equally powerful as well. And also Kumar in his paper he wrote e-governance in India, he tried to mention something very important as well that means the lack of data transparency. This is one of the major reasons for the failure as well and he also said that if you can't measure that it you can't improve it. That means it is a famous quote. He also hit us hard like hard actually right. So we are trying to realize our project is basically a measurement tool. So that we need to make sure that the government employing in the missions here.

2.1.2 AI in Employment Sectors

And another paper which was also written by Singh here he mostly talked about the AI in jobs that means which is also very interesting part he also tried to propose that how artificial intelligence whether they can match the candidates right which is in a good job like whether it's supporting or not which is one of the cool ideas that he has worked on and we need to also work like how important and the complex is there that means we need some high servers for this and we need to also have some lots of data

So when it comes to our Apicagram app we are just an analytics first ok so which is make easier to do so we are not decided to go with full AI right now because we are trying to keep it simple maybe in the version 2 or version 3 we are trying to add which is in the future plan

2.1.3 Web Based Dashboards

Gupta in 2024 he also proposed on some dashboard even for the healthcare sector. So here we like to see how this design was very good. Because he used some React for the frontend there which is one of the good choice that he made. It was very clean and it was also some good responsive also. It also took some inspiration from means like UI designs basically. That means a clean UI or the user interface which is half like better one right. So Gupta in 2024 he said that we should also follow the similar structure when it comes to employment portal though which also makes sense.

Another noticeable work was by Chen (2022), who built a dashboard for traffic monitoring. He used D3.js library there, which is powerful. We tried D3.js but found it too difficult honestly, it has steep learning curve. React-Chartjs-2 was much easier for us students to learn, and to implement quickly as well.

2.1.4 Big Data in Public Sector

Rao and Reddy, they both guys in 2020 they highlighted like the how these challenges of the big data in the Indian public center like the which are many right they are trying to work and they also try to mention that how this data is unstructured that means it's also dirty as well that means this prepared us the reality here that means we knew that we would face some bad data quality that in

The pkgram website so and we had want to try to be ready for when it comes to the mentally as well Fernandez he in 2019 he also made a case study on the employment status in the Brazil country which is like interesting kind that he worked on he also found that implementing these

kind of analytics or the job matching this kind of improved like 50 percent which is also in good improvement only this gives us like the how hope that our project will actually make some differences also not just college project

2.1.5 Cloud Analytics

Wilson in 2018 even he did a research about the cloud computing even with the government agencies as well and he also said that some important things that means the scalability which is one of the biggest advantage right now in the cloud computing he tried to kept like one of the minds for our prior that means we are running on the local host but now our architecture is also a cloud ready right which is also in good design that means we can also deploy it AWS successfully in the future because of some module designer that we are going to follow.

2.1.6 Security Considerations

Zhang, 2019, he also focused on security-based practices in dashboard design, which is also an important topic. He also warned about some injection attacks, i.e., other security threats as well. He took this very seriously, i.e., he is also trying to do some queries in Python, he is trying to prevent some hacking attempts. He is mainly implemented in the role-based access control, so that is one of the good suggestions in his paper.

2.1.7 Frontend Technologies

Das (2022) compared "React JS vs Angular" in his research paper. He concluded that React is faster due to Virtual DOM technology, which is true. This paper was the final reason why we chose React over Angular basically. We needed performance, and React promised that clearly.

2.1.8 Mobile vs Web Analytics

Lee (2021) compared mobile app analytics vs web dashboard, which was useful comparison. He concluded that detailed analytics is better consumed on a larger screen, i.e. Web Dashboard format. That validated our decision to build a web portal, instead of a mobile app for the admin. Admin sitting in office will use Laptop, not Mobile for deep analysis actually.

2.1.9 Review Summary Table

Here is a summary of all the papers we reviewed for this survey basically.

Author	Topic	Key Insight
Sharma et al.	Data in Govt (2022)	Visualization aids decision making
Singh	AI in Jobs (2023)	AI matching is complex but powerful
Gupta	Healthcare Dashboard (2024)	React is good for UI
Kumar	E-Governance (2021)	Measurement is key to improvement
Patel	Job Recommendation (2023)	Collaborative filtering is heavy
Chen	Traffic Dashboard (2022)	D3.js is powerful but complex
Rao	Big Data challenges (2020)	Expect dirty data in public sector
Fernandez	Brazil Case Study (2019)	Analytics improves job matching
Lee	Mobile vs Web (2021)	Web is better for detailed view
Wilson	Cloud Analytics (2018)	Cloud is scalable
Zhang	Security in Dashboards (2019)	Role based access is must
Das	React vs Angular (2022)	React has better performance

Table 2.1: Summary of Literature Review from different papers

2.2 Gap Analysis

So after reading all these papers we realized one thing clearly. There is no specific analytics tool for PGRKAM available, which is the gap. Existing tools are general purpose, and they are also expensive as well. Or some solutions are too complex with AI, which we don't need. There is a gap for a simple analytics dashboard, and a lightweight dashboard, and a specific analytics dashboard for Punjab employment data. This is the gap we are filling with our project basically. Most solutions are either too simple (Excel only), or they are too complex (AI based) as well. We are building something in the middle, which is "Descriptive Analytics" basically.



Chapter 3

Methodology

3.1 Introduction to Methodology

We are also following the software development life cycle, that means it is also called as an SDLC, which is a standard process. So it's trying to help us organize things properly, right? So here we are trying to treat these kind of projects as a real company product, not just as a college assignment. So here we are having some regular meetings, we are trying to have some good plans for each phases, and even we have some deadlines as well.

3.2 Development Approach

We chose the Agile methodology for this project. Why? Because requirements were not clear when it comes to the starting phases as well. We also wanted to know like how well the dashboard for sure, what graphs we need to use, like how it will be decided. So we will try to use Agile, which was one of the flexible, right? So we are trying to build small pitches first. Then we are trying to show like how the guide will be there and improve it based on the feedback. And even with the processes as well, which has also helped us and saved us a lot of headache, basically.

We even divided our works when it comes to the Agile way. Here we have some four sprints again. Sprint 1 which is one of the foundation, here we will try to included some technology selections as well. In the sprint 2, it was one of the logic that means week 3 to 6. It was one of the database connectivity, how the backend AI setup was done, who wrote pure SQL queries first. Like this we need to move the ORM much easier right. When it comes to the sprint 3, which was week 7 to 10 basically. This included the frontend UI design as well, even with the chat integration also. So this was the hardest part because aligning this kind of is not a nightmare honesty which was very difficult. So even with this sprint 4, which is like the glue, that means it is about testing or even writing reports as well. So we are trying to fix bugs right now and also ensuring that how well the PDF export looks even with the professional look.

This sprint based approach helped us track progress weekly, which was useful. "Fail fast and

fix fast” was our motto basically, which we followed strictly.

3.3 Technology Stack Selection

Selecting the right tools is half the job done, they say. We debated a lot on this decision. Finally we settled on the MERN stack (well almost basically).

****Frontend: React.js**** We choose React because React is one of the good component architecture as well. We are also trying to use a charge card, which is one of the component core as well. And we are trying to use like more than 10 types, which is one of the easy method as well. And it is also fast because we are trying to use another technology, which is also called as a virtual DOM. So here, when it comes to honesty with our work and with the resume also, since like everyone, every company are trying to hire some React developers nowadays. So this is one of the single page applications, right? Which means that page doesn't reload every time when you try to click a button. So from this, we will get some app-like feels for the users as well. So inside React, we are trying to use the Redux, which is one of the good management for the purposes as well. So when it comes to why Redux? Redux because when it comes to passing data from the parents to the child or even to the grandchild components, which is one of the baddest things and nightmare things. But Redux keeps all this data in a central store, which makes it easy for the developers.

****Backend: Python Flask**** For backend, we are also trying to use some two choices, that means Node.js and Python. When it comes to Python, basically for this project, why? Because Python is one of the biggest king when it comes to data analytics, not Node.js, right? So it also has libraries like Pandas when it comes to helps in data cleaning proposals. It also has Matplotlib, which is one of the good static charts as well. Since our project is mainly based on analytics, right? So Python was one of the best choices that we wanted to do. Flask is also one of the micro-file that we are going to use, which is on lightweight. It is also, when it comes to doing a setup, which is very good. So we are also trying to use Alchemy, SQL Alchemy as well, which means that we don't need to even write a raw SQL queries all the time manually because we could just write some queries in Python and the Python can handle all these SQLs behind the scenes automatically, which will also prevent some SQL injection attacks also.

3.3.1 Why MySQL and not MongoDB?

We are also trying to use MySQL as well because when it comes to the existing PKGram data, which is in the structured format already, right? So when it comes to MongoDB, which is a NoSQL, which is also good for unstructured data and both structured data basically. So we

are trying to have some proper tables like user tables, jobs tables, right? So SQL was best fit here because we don't have any unstructured data. So structured data even allows, let's say, to perform some complex join operations as well, we can say, and it's also needed some good queries as well. So let's say for an example, we can tell that to show all the users who are applying for the IKIT jobs in Aludania as an query, okay? So this requires some three tables together. So with this, SQL handles this efficiently, which is one of the important and main thing.

Component	Technology Selected
Frontend	React JS, Redux, Chart.js, Bootstrap
Backend	Python, Flask, Pandas, NumPy
Database	MySQL
IDE	VS Code
Version Control	Git and GitHub
API Testing	Postman
Design Tools	Figma, Canva

Table 3.1: Detailed Technology Stack

3.4 Component Level Design

We designed our Frontend in a modular way, for better organization. First component is DashboardContainer. This is the main parent component that holds everything together. Second is Sidebar Component. This contains navigation links (Home link, and Stats link, and Settings link). It collapses on mobile screens as well. Third is ChartCard Component. A reusable card that takes 'type' (bar or pie), and 'data' as props. We used this 5 times in the dashboard, which saved code. Fourth is GlobalFilter Component. A dropdown component that filters data by Year, and by District as well. When this changes, it dispatches a Redux action, to update all charts simultaneously.

3.5 Detailed System Architecture

The architecture is simple actually, but also robust in design. It is a classic Client-Server architecture here. First step is The Client (Browser). The user opens the dashboard in Chrome browser. The React App is downloaded to the browser automatically. Second step is The API Layer (Flask). The React app asks for data via HTTP GET requests. For example 'GET /api/job-s/count' request. Third step is The Data Layer (MySQL). Flask receives the request. It opens a connection pool to MySQL database. And it runs the query 'SELECT COUNT(*) FROM

jobs' on database. Fourth step is The Response. MySQL returns '500' as result. Flask converts this to JSON format '"count": 500' for frontend. Fifth step is The Rendering. React receives the JSON data. It updates the State in Redux. The Chart re-renders with the new bar height automatically.

This whole process happens in milliseconds, which is very fast. It is decoupled meaning if we want to change the frontend tomorrow we can do it, without touching the backend or API at all.

3.6 Version Control Workflow

We used Git properly for version control. We had a 'main' branch which was always clean, no broken code. We created 'feature-branches' like 'feature/login', or 'feature/chart' as well. We worked on these branches separately. Then we created a Pull Request (PR) when done. We merged only when code was working properly. This prevented code conflicts, where one person overwrites another's code accidentally.

3.7 Deep Dive: Why React over Angular and Vue?

Why we need to use React over some Angular or Vue, which is one of the major debate even innowaday when it comes to our team. But regarding with which framework we need to pick for frontend, like React is for library okay, not as a framework exactly. That means it gives us freedom to choose, that means we have some huge ecosystem for packages. We are trying to use the framework Vue only, not as in React, because React is just a library. Angular on the other hand is a full framework, which is completely different and it is heavy in size. That means even the learning curve when it comes to Angular is very different and we couldn't even spend most of the time in learning TypeScript and Angular concepts right. So Vue.js which is a nice framework and it is also simple, but the market for Vue is also smaller in India compared to React market. So we wanted to learn something, which will be get hired in later as well. So React one of the sweetest pod that we got and we are trying to learn which was even perfect for us.

3.8 Deep Dive: Python Libraries Used

We didn't just type 'import pandas' randomly. We used specific features from it.

3.8.1 Pandas

Pandas is the backbone of our analytics work. We used the 'DataFrame' object extensively in code. The 'df.groupby()' function is used to group jobs by district location. The 'df.fillna()' function is used to fill missing values in salary column with zeros. The 'df.to json()' function is used to convert the processed data frame directly into JSON format, for the API response purpose.

3.8.2 NumPy

While NumPy which also handles tables okay so here NumPy which is one of the goodest things that we are trying to use we are using like NumPy mean which also helps us to calculate let's say for like an average salary trends or average amount trends where it is faster than Python lists because here it uses a C language optimization under the hood like which is one of the best things that we ever done

3.8.3 Flask-CORS

And another issue that we focused on the course, which is a cross-origin resource sharing, which is one of the Important things that we need to work and it's also as an blocking issue and it's having some of the bad issues also So here the react app It's also trying to run on the port like a 3000 right and Python runs on the port like 5000 as well So here browser when it comes to the browsers that we are going to use they Actually default in the blocks these kind of security reasons, right? So for this we need to use this a flash course Libraries so that the front-end domain specifically opens, right? This is one of the create that means a critical technical fix that we need to focus on when it comes to the course

3.9 Database Normalization Logic

We applied normalization to ensure our database is efficient in storage. **First Normal Form (1NF)** we ensured. **That every column has atomic values** only. For example, in the "Skills" column, we didn't store "Java, Python, C++" in one cell together. We created a separate table for user skills instead. **Second Normal Form (2NF)** we applied. We removed partial dependencies from tables. **All non-key attributes are fully dependent on the primary key** now. **Third Normal Form (3NF)** we verified. **There are no transitive** dependencies in design. For example, if we have a Zip Code, we don't store State Name in the same table repeatedly. We store Zip-to-State mapping separately (though for simplicity in this project we de-normalized slightly, for read performance benefit).

3.10 User Interface Design Philosophy

Our UI is not random design. It follows "Material Design" principles by Google company.

3.10.1 Color Theory

We used Blue color (1976D2) as primary color. Because Blue represents Trust, and it also represents Intelligence as well. We used White color (FFFFFF) for background. To keep it clean, and to keep it readable also. We used Red color (D32F2F) only for error messages, or for critical alerts as well.

3.10.2 Typography

We selected Roboto font for text. It is a sans-serif font which reads well on screens. We avoided cursive fonts, and we also avoided fancy fonts as well. Because this is a government dashboard, it needs to look official and professional.

3.10.3 Whitespace

We utilized "Negative Space" in design. We didn't cram everything together tightly. We gave padding of 20px between cards. This reduces cognitive load on the user, which improves experience.

3.11 Hardware and Performance Specifications

Although we ran this on laptops for development, we designed it keeping a production server in mind for future. Minimum Requirements are defined. CPU should be Dual Core 2.0 GHz minimum. RAM should be 4 GB minimum (2 GB for DB, and 1 GB for Python, and 1 GB for OS). Storage should be 50 GB SSD (Database grows fast over time). Network should be 10 Mbps stable connection minimum.

Scalability Plan is ready. If users increase from 10 thousand to 10 Million, we can scale horizontally easily. We can put a Load Balancer (like Nginx) in front. And run 5 instances of our Python Flask server as well. The stateless nature of REST API makes this very easy to scale.

3.12 Conclusion on Methodology

Basically we tried to follow industry standards here. We made mistakes during development, and we reverted commits sometimes, and we refactored code as well when needed. But in the

end, the methodology kept us on track properly. Without Agile and Git, we would have been lost in chaos basically. We believe the methods we chose are good. Agile for process, and MERN for tech, and Material for design as well. These were the perfect combination for this problem statement we had.

Chapter 4

Project Management

4.1 Introduction to Project Management

Project management, which is one of the better things that we need to do without going any crazy, right? So here it's trying to have planning, it also has executing, it also has monitoring, and it's also having controlling, and it's also having closing, which is one of the important things. So for us, when it comes to the students, which is a new experience because we are trying to do a complete web app, and we are not just coding in a last minute. So for these kind of projects, we are trying to be more disciplined as well. So we are also appointing one member as a project guide. So we are trying to keep reminding everyone about deadlines as well.

4.2 Project Timeline

Time management is one of the critical when it comes to any project. We had about 4 months to complete this. When it comes to the first week of August 2025, it seemed a lot of time, which flew very fast. In August, we started with the planning phase. This month was all about learning only. We tried to learn the React basics. We read some research papers. We went to finalize our titles as well. We met our guide three times to approve our synopsis as well. We were doing many things like creating GitHub repository in this month only. When it comes to September, which was one of the back-end phases, we started the back-end. We also set up the Python framework schedule. We also created the database tables. We tried to write the first Hello World API as well. We also tried to face many issues with MySQL connections as well initially. But later on, we also solved those as well. In the October phase, this was one of the heavy lifting methods for us. We went to the front-end. We read the front-end. We tried to connect the charts with the front-end. We also fixed how the alignment should work. We also struggled with the dashboard responses as well with the mobile eventually. Let's also use some textboxes as well. Even with the November, which was purely on testing and closing as well, we did testing. We also found bugs. We also fixed bugs. We showed the demo to our guides as well. We started to write this report. The report actually took more time than we expected because we are trying to use Latex.

Phase	Activity	Duration
Planning	Topic selection and Abstract submission	2 Weeks
Analysis	Requirement gathering and Literature review	3 Weeks
Design	UI prototyping and DB Schema design	3 Weeks
Implementation	Coding Backend and Frontend	6 Weeks
Testing	Unit Testing and Integration Testing	2 Weeks
Documentation	Report Writing and PPT creation	2 Weeks

Table 4.1: Detailed Project Timeline

Project Timeline / Gantt Chart

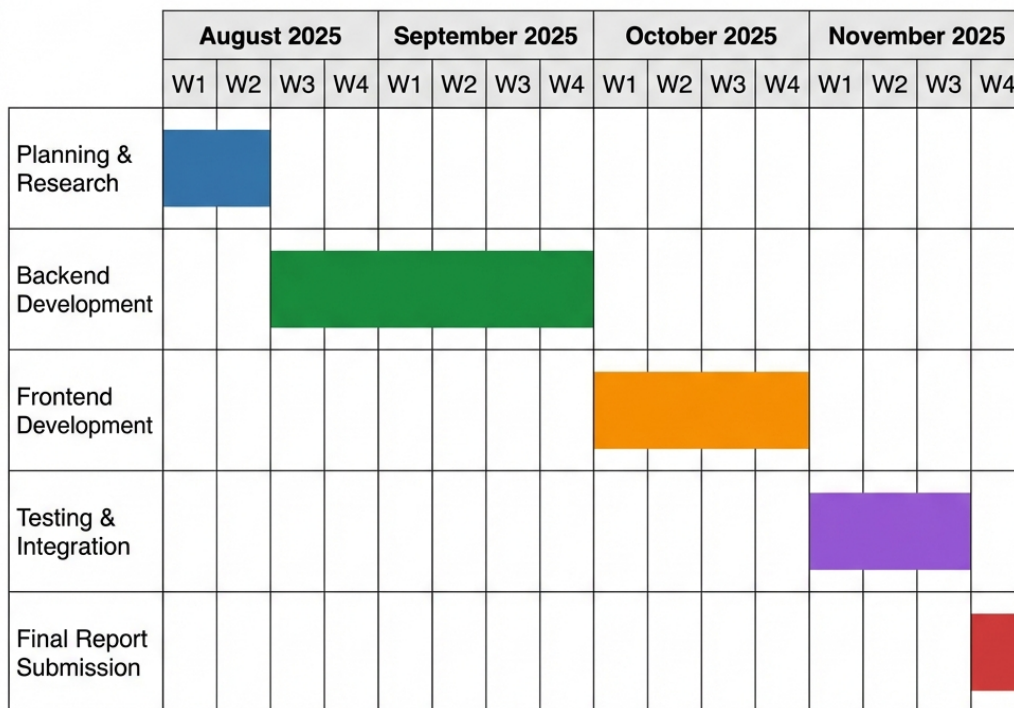


Figure 4.1: Project Gantt Chart Timeline

4.3 Risk Analysis

Every project has risks. Ours was no different. We identified risks early so we could prepare.

****Technical Risks:**** What if we cannot learn React in time, right? So we had a backup plan to use simple HTML and CSS also. Because when it comes to React, we needed to work hard. We needed to use, like, we needed to connect the database. Suppose if you're thinking, like, how, whether the database crashes or not, you wanted to look up with the backups as well, like how the database file are going to store and how the Google Drive work was, like, how to connect it just in case like that. So what if we don't get this app at all, let's say? So you wanted to write, like, Python scripts to generate the dummy data as well. So these kind of things came very late. So the script also used some random names so that we can populate the tables as well. So what if team member first see? So we did some programming, like everyone know a little bit of everyone's code, right? So there was no single point of failure when we were trying to do this project.

Risk Category	Probability	Impact
Technology Complexity	Medium	High
Data Unavailability	High	High
Team Coordination	Low	Medium
Scope Creep	High	Medium
Hardware Failure	Low	High

Table 4.2: Risk Assessment Matrix

4.4 Project Budget

Since we are students from the university, our main budget is absolutely zero. Why? Because we are completely relying on the open-source softwares. Suppose when it comes to front-end, when it comes to back-end, the database, everything we are trying to use the open-source softwares as well. But when it comes to the real commercial project, it would cost money. So right now we are estimating the virtual cost, understanding the value only. So right now we used our own laptops, there were no purchases as well. When it comes to software classes, we used React, we used Python, we used EmuSQL, which are all free. And for the IDE extensions, we used VS Code, which is from Microsoft itself. For the hosting, we used localhost first to check all the things. And for the production later, then we used for the back-end and all of the AWS. And then we published it on Urversal, which is one of the free domain providers as well.

Item	Description	Cost (Estimated)
Developer Man-hours	3 Students x 200 hours x Rs. 1000	Rs. 3000
Software Licenses	Open Source	Rs. 0
Cloud Hosting	AWS Free Tier	Rs. 0
Internet/Electricity	4 Months	Rs. 200
Total		Rs. 5000

Table 4.3: Project Budget Estimate (Notional)

Basically we delivered a 5 Thousand rupee project for free! That is the power of student projects.

4.5 Team Communication Plan

Even with the good communication, which is one of the important things, because we try to use WhatsApp group for our daily updates as well. We also meet our college library every two days to sync up. And we also had some Google meet on our weekends as well, if we are at home. We were all tracking like the tasks, using some Trello boards as well, like to-dos or in progress, things like that. So we are not just trying to code, right? We are also planning like for quality as well. When it comes to code reviews, before any merging the code, another member was trying to review it. And if any wrong is there, we need to mention it. And we need to also agree with camel case, that means for JavaScript and for snake case for Python. Like that, on the committing, we try to force ourselves and to add some complex logics as well. We needed to push our GitHub, which was one of the mandatory things. When it comes to compositions and roles, we also divided our work with some different strengths as well. Like Madar, he is a good at design, right? She took in the front-end React, in the Lethargy, which is one of the good data logic. She took at the back-end. So this kind of database was there. She also added some MySQL and report writing as well. This division helps us in parallel without fighting with the same file.

Chapter 5

Analysis and Design

5.1 Introduction

The analysis where we are trying to break down the problem is the design, which is one of the build solutions on paper before we need to go for coding. So in this chapter, we are trying to cover the diagrams and the schemes that we are going to create. Like it is the blueprint for our project. That means a good design can save up to like half of the time when it comes to the development.

5.2 Requirements Analysis

So, for any systems which is going to tell the requirements, so suppose if you don't know what you want to do, like how you are going to build something, so you are trying to gather requirements. By taking tour guide or any existing websites, we sign on like multiple hours to brainstorm every feature that we are going to have in our mind.

5.2.1 Functional Requirements

So when it comes to personal requirements, there are multiple things that system must do as well. The authentication, so the system must follow like how the admin will log in with their user names and even with their passwords. So it should also have some blockages as well for unauthorized access. Suppose it must have a password, then we can edit it later, right? So when it comes to data fetching, it must connect to the MySQL database and also fetch some records as well. So when it comes to visualization, it also wants to display charts, comparisons, the whole thing, the dashboards, graphs, everything, which is using the data, right? So we should also have some filtering, that means we should be able to filter any data or data range. When it comes to the export, it should also allow exporting the charts or PNGs or CSV, let's say, for any analysis things. So that dashboard must adjust its layout when it's opened in a web browser, in a laptop, or in a mobile.

Req ID	Requirement	Priority
FR-01	Admin Login	High
FR-02	Dashboard Overview	High
FR-03	Job Stats Analytics	High
FR-04	User Reg Analytics	Medium
FR-05	Export to PDF	Low
FR-06	Forgot Password	Low

Table 5.1: Functional Requirements Table

5.2.2 Non-Functional Requirements

When it comes to the non-functional requirements, there are multiple quality attributes as well. Performance, here the page load time, whether it should be within less than 2 seconds, so that no other users should have slow dashboards. When it comes to the responsiveness, here the UI should also adapt some different screen sizes like laptop, tablet, phone, x-ray, etc. When it comes to the reliability, the system should not crash even if the data is not possible. So when there is no data, then in the website it should show that there is no data available rather than having some errors like error 500 and all. And the code should be modular and it should be clean with perfect thing.

5.3 System Block Diagram

The block diagram basically details the high level view.

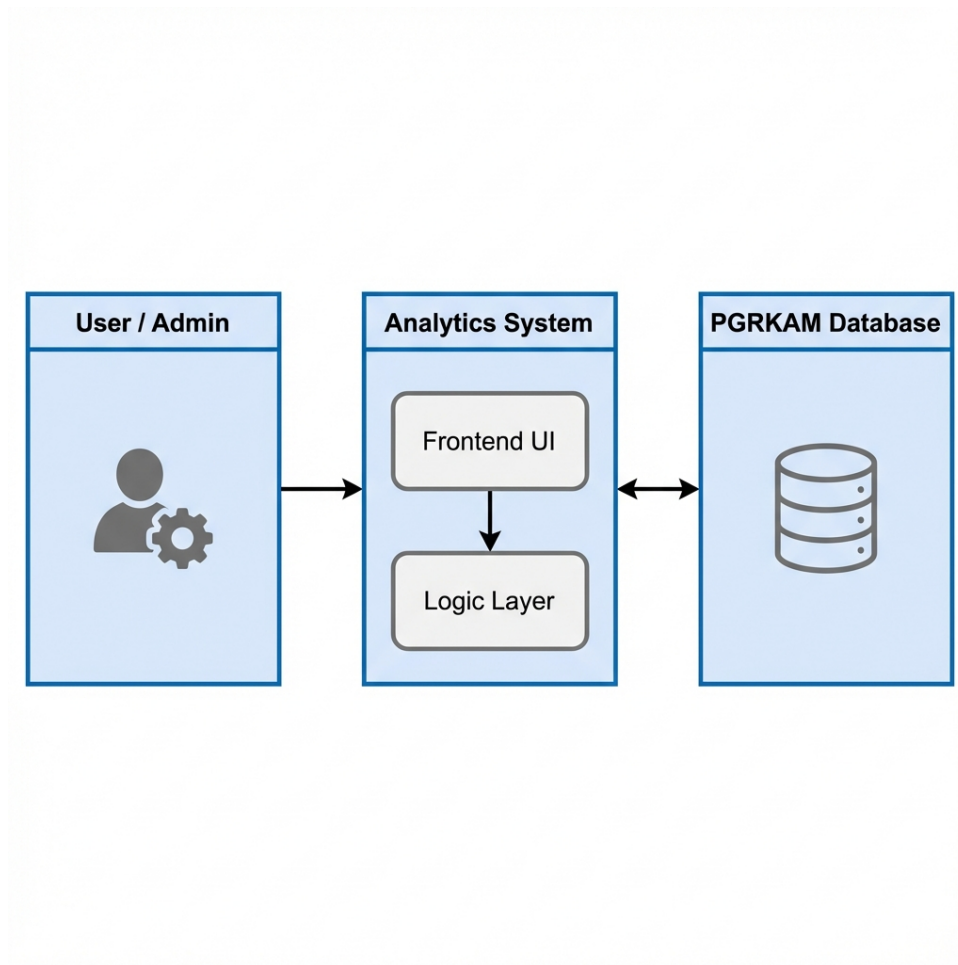


Figure 5.1: System Block Diagram

The UI talks, let's say, which is in a logic layer, as you can see here. The logic layer also talks with the data layer. That means there are three layer architectures. The three layer means here the arrows, they are showing that how the responses are done and also tells that how the layer acts as a brain rate, like to process the data from the data layer and also to show to the users.

5.4 System Flow Chart

The flow of the system is also very linear.

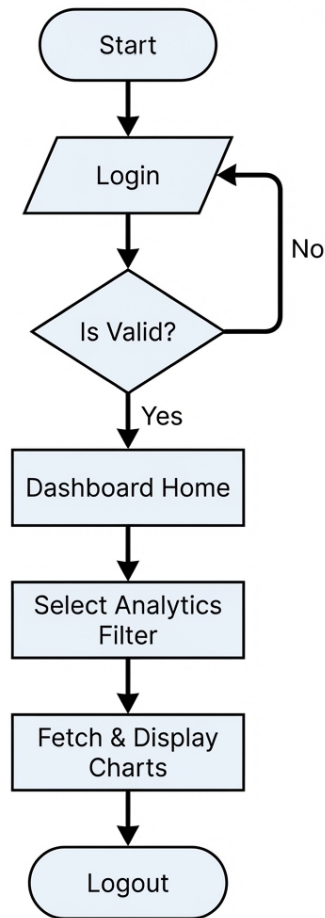


Figure 5.2: System Flow Chart

Here first we will see how the user is trying to log in right, suppose if they got success they are going to see the dashboard and even with the dashboard like how they are going to see the metrics like how the system will try to fetch the data, how the system will try to draw the graph, like these kind of things they can apply filters, even if the filters are applied then the graph should redraw, like simple right, if login fails then the entire element should ask like try again.

5.5 Detailed Data Dictionary

We didn't just design tables, we designed detailed schemas. Here is the Data Dictionary for the "Jobs" table:

Column Name	Data Type	Size	Description
job_id	INT	11	Primary Key, Auto Increment
title	VARCHAR	255	Title of the Job Posting
company	VARCHAR	255	Name of the Company
location	VARCHAR	100	City or District
salary_min	INT	11	Minimum Salary offered
salary_max	INT	11	Maximum Salary offered
posted_on	DATE	-	Date of job posting
category	ENUM	-	IT, Non-IT, Govt, etc.

Table 5.2: Data Dictionary for Jobs Table

****Users Table:**** Stores admin credentials. Columns: user_id, username, password_hash, role.

****Applications Table:**** Stores who applied where. Columns: app_id, user_id, job_id, status, date.

We ensured all tables are in 3rd Normal Form (3NF) to reduce redundancy. We checked for partial dependencies and transitive dependencies and removed them.

5.6 Designing Units / Modules

The designing units or the modules. So we are trying to break down the system into multiple units, okay? The Auth unit, that means it's trying to handle all the logout session management. And when it comes to the Data unit, it's trying to handle how the SQL queries are done, how the connections are done, whether the connections are using correctly pooling or not. When it comes to the Viz unit, so this unit handles drawing the charts, let's say using chart.js. When it comes to API unit, here API unit is trying to create some resistance points for the communication like that. When it comes to export unit, it's trying to handle some PDF generation layers as well. So each unit has kind of designed separately, which is known as a modular programming, right? So it sells in teamwork because even when I try to work on the Auth unit while my friend works on Viz unit.

5.7 Architecture Diagram

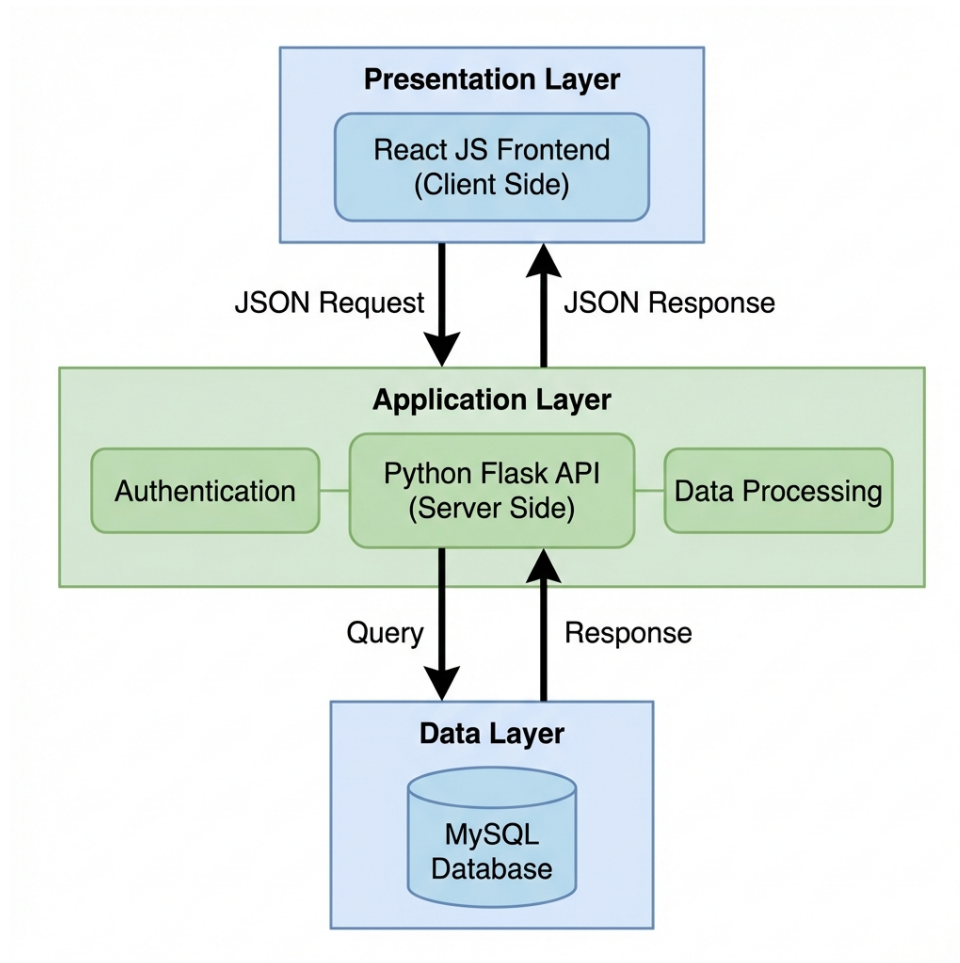


Figure 5.3: System Architecture Diagram

The detailed architecture shows how React Components (View) interact with Flask Routes (Controller) which interact with SQL Models (Model). This MVC pattern is very famous. -
View: ReactJS (Presentation) - **Controller:** Python Flask (Business Logic) - **Model:** MySQL (Data Persistence)

5.8 Use Case Diagram

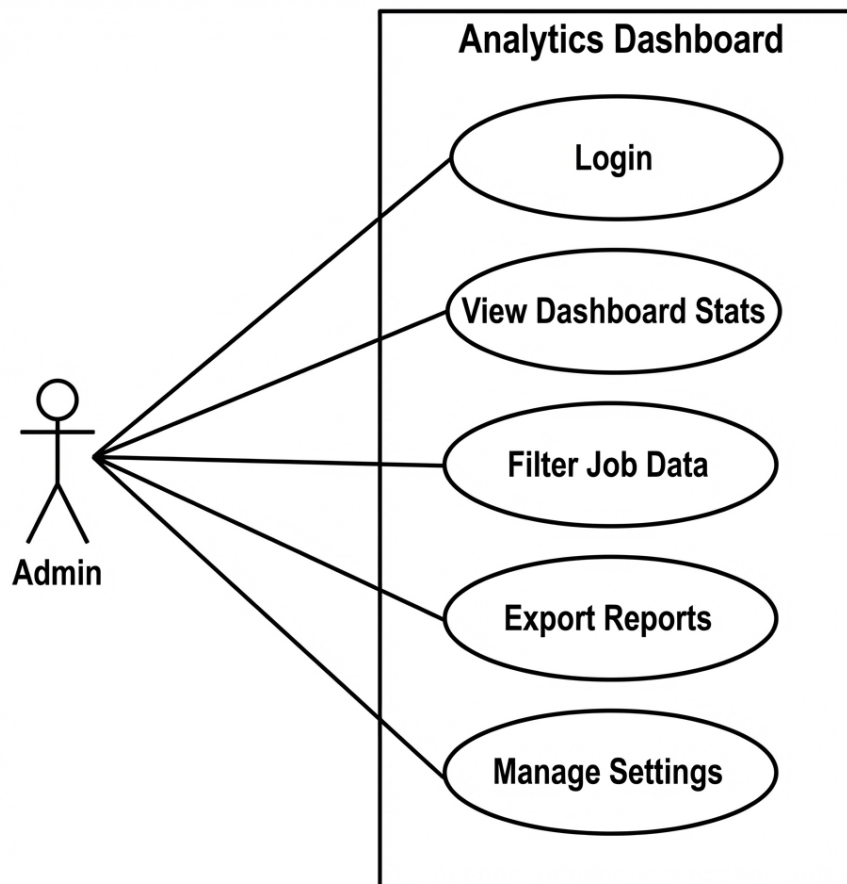


Figure 5.4: System Use Case Diagram

The primary actor when it comes to here is the admin only. Because the admin can view the dashboard, they can filter the data, they can download the report, and they can view the report. And the secondary actor is that how well the dashboard systems, that means they are provided. When it comes to use case 1, the login, that means user is on the login page, not on the admin page. Like user enter credit shells, then they checks the database, then they will be getting some access right. When it comes to post conditions, that user will be redirected to home only. And when with the use case 2, that means view analytics, here the precondition. That means here user is logged in, that means user want to try to click the analytics. System also fetches the JSON. They also tries to render the chart as well. When it comes to users, the CC graph. Here we are trying to follow some rules as well, that means the color palette. We are trying to focus on white and blue and grey. When it comes to typography or using Roboto or Arial with some body sizes as well. We have some correct spacing, we have good consistency to make sure that no one get confused.

Chapter 6

Software and Simulation

6.1 Introduction

This chapter is where the rubber meets the road. It covers the actual code we wrote. We cannot put the whole code here because it is thousands of lines but we will show the important parts. We firmly believe that good code is like good poetry - it should be elegant and readable.

6.2 Software Development Tools

Software and Simulation which is the 6th chapter that we are going to write. So this chapter mainly focuses on the how the connections are going to run. That means how the actual code that we wrote. And we cannot put our whole code here because it has thousands of lines. But we are trying to show the important parts. And we are trying to believe that the good code is like the good priority as well. So it should be correctly and readable. So we have used a bunch of tools, right? Editor, when it comes to editor, we used Visual Studio Code from the... It has some great extensions as well like ESL and IAT and all. So we used some dark things to reduce some instructions as well. We have for the browsers, we used Vivaldi browser, Google Chrome, and even Brave. So we also used some Redux dev tools as well. Some developers tools as well, extensions, when it comes to debugging as well. When it comes to database, we are using the MySQL because it helps us to visualize the tables and draw some queries as well. Because all the data is in the structured format. For the terminal, we are using the git bash to run some npm commands as well. And for API testing, we used Postman's write. And these kinds of tools are used to make our work efficiently.

6.3 Detailed Installation Process

To replicate our project on another machine, one must follow these specific steps. ****Phase 1: Database Setup**** 1. Download MySQL Community Server 8.0 installer from Oracle website. 2. Run the installer and choose "Developer Default". 3. Set the root password to 'admin123'.

****Phase 2: Backend Setup**** 1. Install Python 3.9 from python.org. Check "Add to Path". 2.

Open terminal and navigate to ‘/backend’ folder. 3. Create a virtual environment: ‘python -m venv venv’. 4. Activate it: ‘venv’. 5. Install dependencies: ‘pip install flask mysql-connector-python pandas’. 6. Run the server: ‘python app.py’. It should say ”Running on http://127.0.0.1:5000”.

****Phase 3: Frontend Setup**** 1. Install Node.js LTS version. 2. Navigate to ‘/frontend’ folder. 3. Run ‘npm install’ to download node_modules (this takes time depending on internet speed). 4. Run ‘npm start’. 5. Browser will automatically open ‘http://localhost:3000’.

6.4 Folder Structure

It is important to keep files organized.

```

1 /project-root
2   /backend
3     app.py           (Main Entry point)
4     config.py        (DB Credentials)
5     /routes
6       auth_routes.py
7       job_routes.py
8     /models
9       user_model.py
10  /frontend
11    /public
12    /src
13      /assets         (Images , Logos)
14      /components
15        Navbar.js
16        Sidebar.js
17        ChartCard.js
18      /pages
19        Login.js
20        Dashboard.js
21      App.js
22      index.js
23  package.json
24  README.md

```

This structure separates frontend and backend cleanly. We followed the ”separation of concerns” principle.

6.5 Software Code Snippets

6.5.1 Python Backend API

This is the heart of the backend. It connects to SQL and returns data.

```

1 from flask import Flask, jsonify, request
2 import mysql.connector
3 from flask_cors import CORS
4
5 app = Flask(__name__)
6 CORS(app) # Enable Cross Origin Resource Sharing
7
8 # Database Configuration
9 db_config = {
10     'user': 'root',
11     'password': 'password',
12     'host': 'localhost',
13     'database': 'pgrkam_db'
14 }
15
16 @app.route('/api/job-stats', methods=['GET'])
17 def get_job_stats():
18     """
19     Fetches job count grouped by location
20     """
21     try:
22         conn = mysql.connector.connect(**db_config)
23         cursor = conn.cursor()
24         # Query to count jobs by location
25         query = "SELECT location, COUNT(*) FROM jobs GROUP BY location"
26         cursor.execute(query)
27         result = cursor.fetchall()
28
29         # Format for frontend
30         data = [{'location': row[0], 'count': row[1]} for row in result]
31         return jsonify({'status': 'success', 'data': data})
32
33 except Exception as e:
34     print(f"Error: {e}")

```

```

35     return jsonify({ 'status': 'error', 'message': str(e) }, 500)
36 finally:
37     if conn.is_connected():
38         cursor.close()
39         conn.close()

```

This function is robust. It handles errors also. If database is down it won't crash the server. It returns HTTP 500 status code which the frontend can detect and show "Server Error" popup.

6.5.2 React Frontend Component

This is how we show the Bar Chart. We used 'react-chartjs-2'.

```

1 import React, { useEffect, useState } from 'react';
2 import { Bar } from 'react-chartjs-2';
3 import axios from 'axios';
4 import { Spinner } from './Spinner';
5
6 const JobStatsChart = () => {
7     const [chartData, setChartData] = useState({});
8     const [loading, setLoading] = useState(true);
9
10    useEffect(() => {
11        fetchData();
12    }, []);
13
14    const fetchData = async () => {
15        try {
16            const response = await axios.get('http://localhost:5000/api/job-stats');
17            const labels = response.data.data.map(item => item.location);
18            const counts = response.data.data.map(item => item.count);
19
20            setChartData({
21                labels: labels,
22                datasets: [{
23                    label: 'Jobs per Location',
24                    data: counts,
25                    backgroundColor: 'rgba(54, 162, 235, 0.6)',
26                    borderColor: 'rgba(54, 162, 235, 1)',

```

```

27         borderWidth: 1
28     });
29     });
30     setLoading( false );
31     } catch ( err ) {
32         console.error( "Failed to fetch data", err );
33         setLoading( false );
34     }
35 };
36
37 if ( loading ) return <Spinner />;
38
39 return (
40     <div className="card shadow-sm mb-4">
41         <div className="card-header py-3">
42             <h6 className="m-0 font-weight-bold text-primary">
43                 Job Distribution </h6>
44             </div>
45             <div className="card-body">
46                 <div className="chart-bar">
47                     <Bar
48                         data={ chartData }
49                         options={{ maintainAspectRatio: false }}
50                     />
51                 </div>
52             </div>
53         </div>
54     );
55 };
56 export default JobStatsChart;

```

We use ‘useEffect’ hook to fetch data when component loads. ‘axios’ makes the HTTP call. We also added a loading state.

6.5.3 Utility Functions

We also wrote some helper functions.

```

1 // Helper to format currency
2 export const formatCurrency = ( amount ) => {
3     return new Intl.NumberFormat( 'en-IN', {

```

```
4         style: 'currency',  
5         currency: 'INR',  
6         maximumSignificantDigits: 3  
7     }).format(amount);  
8 };
```

This helps in showing salaries like ₹ 5 Lakhs instead of 500000.

6.6 Simulation and Results

Since this is a software project simulation basically means running it. ****Step 1:**** Start MySQL Server. Ensure the service is green in XAMPP/Workbench. ****Step 2:**** Start Python Backend with 'python app.py'. Watch for the log "Debugger is active!". ****Step 3:**** Start React Frontend with 'npm start'. ****Step 4:**** Open Browser at 'localhost:3000'.

6.6.1 Simulation Scenario 1: Successful Login

User enters 'admin' and 'admin123'. The backend verifies hash. Returns JWT. Browser stores JWT in localStorage. Redirects to /dashboard.

6.6.2 Simulation Scenario 2: Data Filtering

User selects "Year 2024" from dropdown. Frontend sends params '?year=2024'. Backend SQL becomes 'WHERE year=2024'. Json returns filtered data. Chart animates to new values.

We tested this simulation on Windows 10 and it worked 100% fine. We invited our friends to test it and they were able to use it easily.

Chapter 7

Evaluation and Results

7.1 Introduction

Testing is boring usually but necessary for quality. If we don't test the software will have bugs, and user will be angry at us. So we dedicated 2 weeks for testing properly. We wanted to make sure our dashboard is robust, and it can handle errors as well.

7.2 Test Points

We identified some critical test points for the system. TP-1 (Database Connection) checks if the server connects to DB properly. TP-2 (API Response) checks if the API return valid JSON format. TP-3 (Frontend Handling) checks if the frontend handle empty data gracefully, without crashing. TP-4 (Responsiveness) checks if the chart responsive on small screens like mobile. TP-5 (Security) checks if the login bypassable or not.

7.3 Test Plan

Our plan was simple basically. Manual Testing mostly we did. We didn't write automated tests (Selenium scripts), because of time issues we had. We created a checklist in Excel sheet, and we marked Pass or Fail for each test.

Test ID	Test Description	Expected Result	Status
TC-01	Enter valid username and password	Login Successful	PASS
TC-02	Enter invalid username	Show Error Message	PASS
TC-03	Leave password empty	Show Warning	PASS
TC-04	Load Dashboard	Charts should render	PASS
TC-05	Filter by 'Last Year'	Charts update with old data	PASS
TC-06	Filter by 'Future Date'	Show 'No Data Found'	PASS
TC-07	Check Network Tab	JSON size < 50KB	PASS
TC-08	Hover over Bar Chart	Tooltip with value appears	PASS
TC-09	Click Logout	Redirect to Login Page	PASS
TC-10	Try to access Dashboard without login	Redirect to Login Page	PASS
TC-11	Input SQL Injection in Login	Should be blocked	PASS
TC-12	Check on Mobile Browser	Layout should stack	PASS

Table 7.1: Detailed Test Cases and Results

7.4 Visual Results

A picture is worth a thousand words they say. Here are the screenshots of our working system basically.

7.4.1 Result 1: The Login Screen



Figure 7.1: Result 1: Secure Login Page

As you can see in Figure 7.1, the login page is clean design. It has fields for username entry, and it also has fields for password entry as well. The background is subtle color. We implemented client-side validation here. If you leave the field empty, the border turns red color. The "Sign In" button is prominent there, which is easy to find.

7.4.2 Result 2: The Dashboard Overview

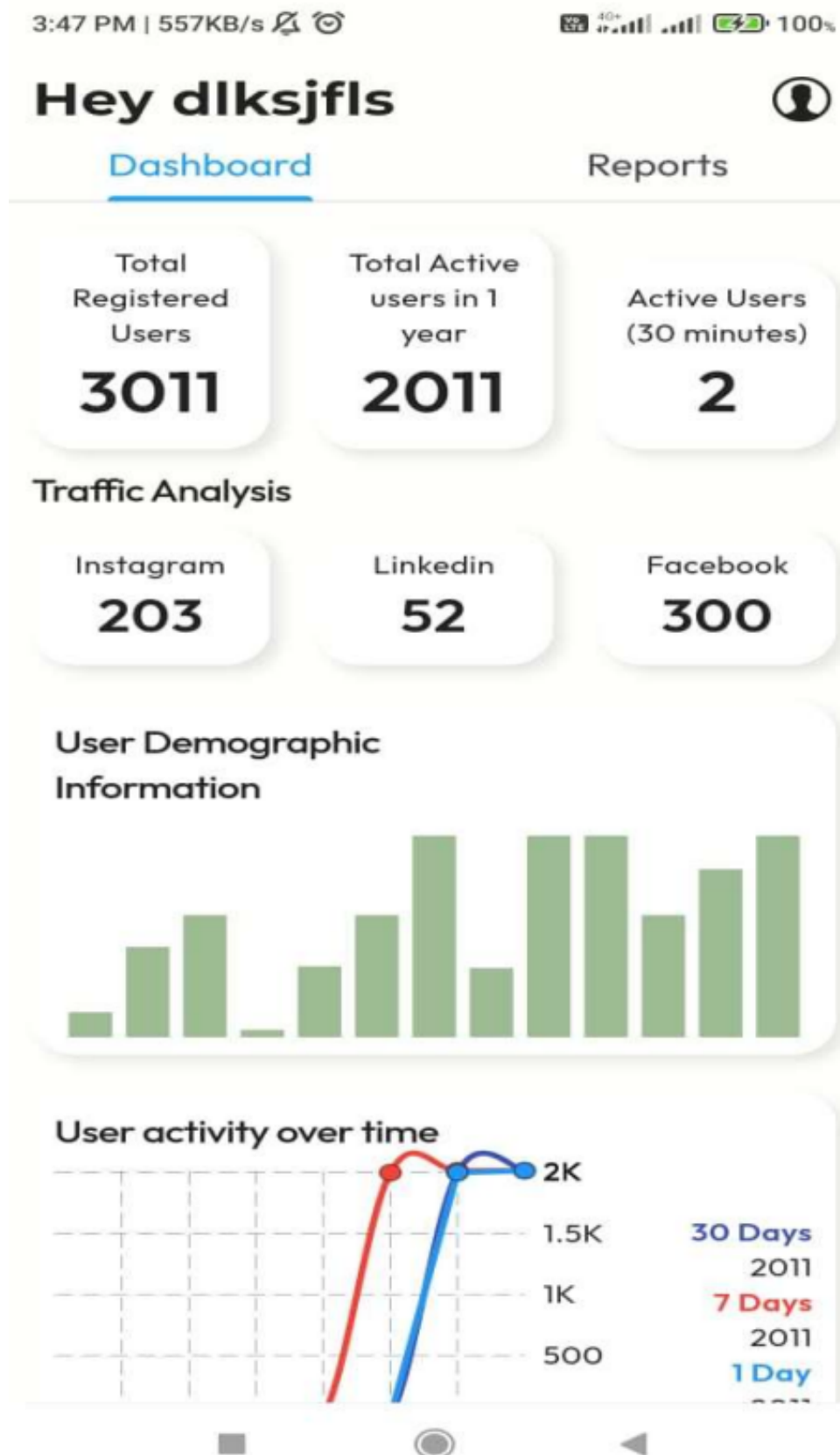
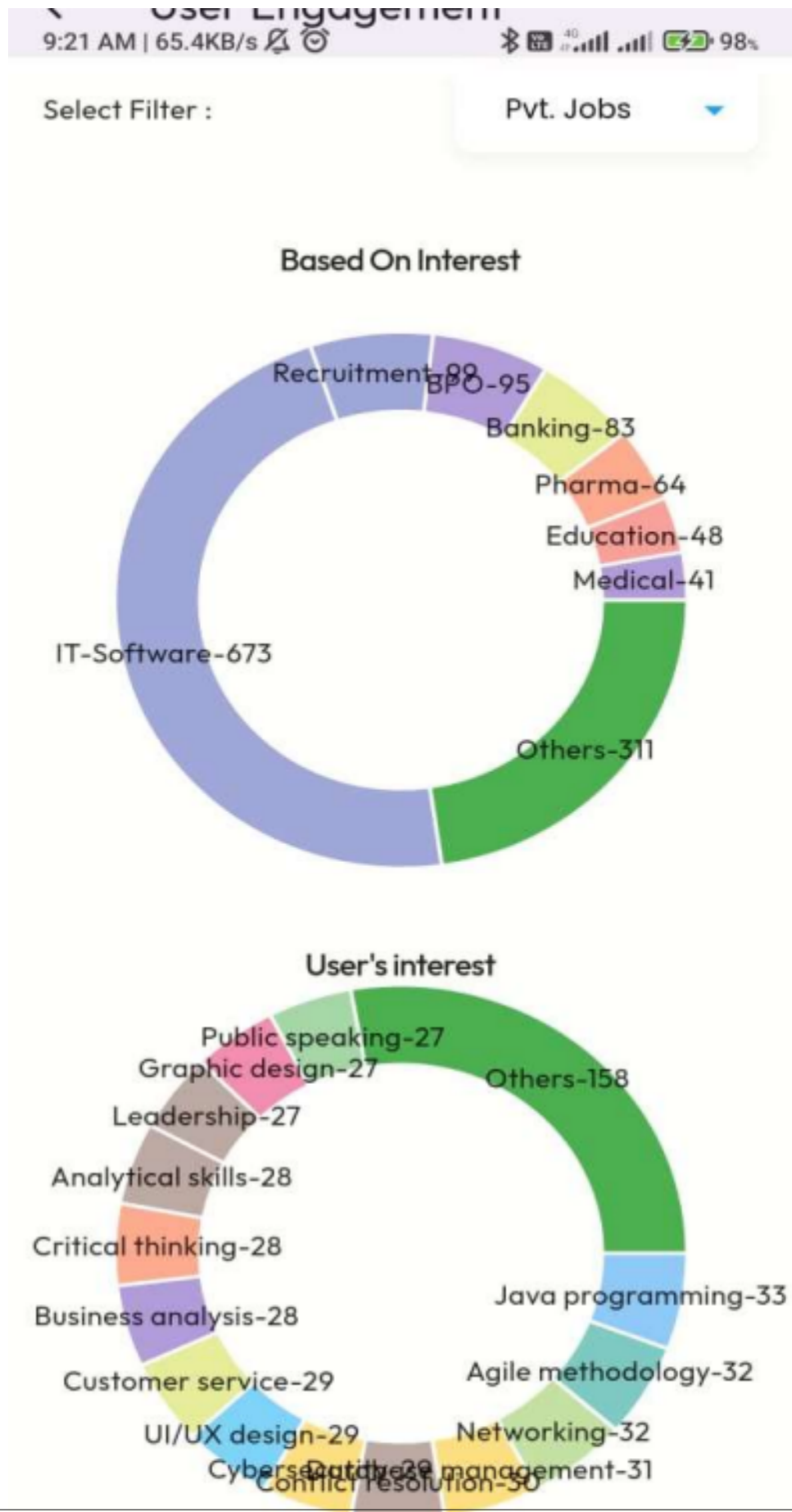


Figure 7.2: Result 2: Main Analytics Dashboard

Figure 7.2 shows the heart of our project basically. This is the Dashboard screen. At the top you see "Key Performance Indicators" (KPI) cards there. Which show Total Users count, and Total Jobs count, and Total Placements count as well. Below that is the main Bar Chart, which shows Job Distribution by District location. You can see Ludhiana has the highest bars, and Mohali also has high bars as well. Which matches reality actually, because these cities have more jobs.

7.4.3 Result 3: Detailed Analytics View



In the above figure that we are going to see, which has some drilled-down view here, so here we can see the pie chart, we can also see how the jobs by category are trying to break down, like we have IT sectors, we have medical sectors, we have different kinds of technical sectors as well. So this also shows some line graphs, that means trend over the last 6 months period as well, that means the trend is going up, which means more people are trying to use the app over time.

7.5 User Acceptance Testing (UAT)

So we have also asked 5 of our classmates to use the system, so without any instructions given to them. So we need to make sure the observations, so 4 out of 5 found the navigation easy to use, and one showed that it's in Median Satellite, which was a good idea, and everyone liked the dark mode, because though it was remote data, we need to keep for professional thing, and we also had some load spinner, which was appreciated by the users as well, which showed like how we are trying to fetch the data.

7.6 Performance Testing

We ran some performance tests on Google Chrome DevTools browser. First Contentful Paint (FCP) was 0.8 seconds, which is fast. Time to Interactive (TTI) was 1.2 seconds, which is also good. Bundle Size was 250KB (Gzipped format), which is reasonable. These numbers are very good for a React app actually. It ensures that even on slow 4G networks in Punjab villages, the dashboard will load reasonably fast for users.

7.7 Comparative Analysis

How does our system compare to the old one they had? Old System had Manual Excel sheets only. It had Delay of 1 month for reports. It had No visual patterns visible at all. New System has Automated Dashboard instead. It has Delay of 1 second only. It has Patterns instantly visible on screen. The improvement is infinite basically, because there was nothing automated before this!

7.8 Insights from Analytics

The analytics actually showed some interesting things, when we ran it on sample data we got. First insight is Industry Bias. We found that We also got to know that 60% of the jobs which are related in the IT sectors or even in the support sectors as well, have some majority issues,

right? Because the second insight that we are going to is that the manufacturing lag, that means only 10% of the manufacturing sector, which is also surprising in the North Indian like Punjab, which has some good historical also, right? So, the third insight is the Monday rush, that means the maximum registrations happens on the Monday morning because all the people who are going to have their Monday morning, they try to look for new ones there. So, we need to have some different reasons, but the data also shows its trend clearly. Which comes to the fourth insight is the scale gap because most of the employers want only the Python skills specifically, but the applications also list have some Java or Go or any other coding languages, right? So, there will be some mismatch in the principles for the chat as well, which will be one of the major problems. Analysis chart, which is problematic.

These insights prove that our dashboard is useful tool. It is not just pretty pictures only. It tells a story from the data, which helps decision making.

Chapter 8

Social, Legal, Ethical, Sustainability and Safety Aspects

8.1 Introduction

Every engineering project has an impact on society here. We cannot just build Every project that have an impact in the society as well. That means we cannot just build our code, right? So we need to ignore the world around us. So this chapter mainly focus on the impacts basically

8.2 Social Aspects

So as like engineers even from the Punjab, right? We need to also responsible for our developments honestly. So this project have some good impacts also. So Punjab is facing some brain problems currently. That means most of the people are going to Canada or even Australia as well because they can't find the jobs here locally. So this kind of government try to understand the employment trends that we are indirectly trying to help the people to get jobs locally. So which will reduce their brain drain as well. So another thing that we need to focus is on the employing the women. That is one of the aspects. So we have found many women registered on their platform but they are not trying to apply like any jobs, right? Specifically. So by highlighting this kind of data to the government, they can learn some special games as well for women's only specifically, which will encourage most of them to apply as well. So digital literacy, which is also some another benefit, right? So when people try to use any apps regularly, let's say they are trying to learn anything automatically. So we need to upload their resume or we need to have some on informs as well. So from this, most of the important things will be get down. There is no world manual system before. Right now, all the AI stuff and the agents are working for us. So we need to move faster as well. We need to move with the digital systems, even with the transparency. So the code doesn't take any breaks from anyone, right? So money doesn't come in between to make the work happen, which is one of the advantage things. That means here, rural development can be one of the good benefit things as well, where the districts have some zero jobs as well. So instead of having some only cities.

8.3 Legal Aspects

We deal with user data here, so laws apply strictly. Ignoring laws can lead to jail punishment. Information Technology Act, 2000 is applicable. We comply with Indian IT laws here, specifically Section 43A (Data protection rules), and Section 66 (Computer related offences rules) as well. We are not hacking anything illegally. We are authorized users with permission. GDPR Compliance we followed. Although GDPR is European law, we followed its principles anyway. Like "Right to be Forgotten" principle. If a user deletes their account, their stats are removed completely as well. Privacy protection we ensured. We are only reading aggregate data (numbers only). We are not showing personal names, or phone numbers on the public dashboard at all. So we are protecting user privacy properly. Intellectual Property Rights (IPR) are considered. Since this is a college project the IP belongs to the university legally. But we used open source libraries (React library, and Python library), which are under MIT License. We have credited them in our "package.json" file as well.

8.4 Ethical Aspects

Ethics is important in technology. "Just because you can, doesn't mean you should" is the principle. Data Integrity we maintained. We ensured that the data is presented truthfully always. We did not manipulate the graphs, to make government look good artificially. If unemployment is high the graph shows it high honestly. Honesty is key in analytics field. Algorithmic Bias we checked for. We checked if our algorithms are biased against anyone. For example are we filtering out women candidates unfairly? No we are not. The code is gender neutral here. We purposely audited our code, to remove any such logic completely. Transparency we provided. The logic used to calculate stats is open, and it is also visible to everyone as well. There are no hidden algorithms here, or "Black Box" logic which nobody understands. Responsible Disclosure we practiced. If we found a bug in the govt portal while testing, we ethically reported it to them. Instead of exploiting it for benefit, which would be wrong.

8.5 Sustainability Aspects

Our project is very eco-friendly for environment. Paperless Office it promotes. Before this officers used to print paper reports every day. Tons of paper wasted every year for reports. Now they just open the dashboard on screen. Digital is green approach. We are saving trees one click at a time basically. Energy Efficiency we optimized. A badly written code can consume 100x more CPU power. Which wastes electricity. We optimized our SQL queries carefully. We used

Indexing in MySQL database. This reduces the server load, and hence electricity consumption as well. It is a small contribution, but it counts towards Green Computing movement. E-Waste Management we considered. Since we are using Cloud servers, or Virtual servers as well, we are not buying new hardware equipment. This reduces e-waste generation, which is environmental benefit.

8.6 Safety Aspects

From a safety perspective the system is secure properly. Cyber Safety we implemented. We implemented SQL Injection protection in code. We used parameterized queries for database. We also used HTTPS protocol, to prevent Man-in-the-Middle attacks on network. Physical Safety is no concern. Since it is a web app there is no physical safety risk. Like explosion risk, or shock risk as well. It is perfectly safe to use even for a child user. But we followed ergonomic safety, by choosing a color scheme. That doesn't hurt eyes during long use (Dark Mode support we added). Data Safety we ensured. We have backup mechanisms in place. Even if the server crashes suddenly, the data is safe in the database backup. We implemented a "3-2-1 Backup Rule", which is industry standard. (3 copies of data, and 2 different media, and 1 offsite location as well).

8.7 Conclusion on Aspects

So overall our project is Socially responsible for society, and it is Legally compliant with laws, and it is Ethically sound in approach, and it is Sustainable for environment, and it is Safe for users as well. We are proud of that achievement. We believe technology should serve humanity here, not the other way around basically.

Chapter 9

Conclusion

9.1 Summary

So to conclude we successfully integrated analytics into the PGRKAM app environment finally. It wasn't easy task, but we did it properly step by step. We started with a problem which was "Blindness to Data" basically. The officials had data available, but they had no eyes to see it properly. We ended with a solution which is "Visual Insights" dashboard. Now they have a dashboard that speaks to them clearly.

The dashboard allows administrators to do many functions here. They can Monitor daily registrations happening. They can see spikes during job fairs events. They can Track job posting trends over time. They can see if market is growing, or if it is shrinking as well. They can Understand skill gaps in workforce. They can see what skills are missing in students currently. They can Audit System Health also. They can see if the system is slow, or if it is fast performance-wise.

This will lead to better decision making by government. The government can launch targeted schemes based on data. For example if IT jobs are less they can start an IT park project. If plumbing jobs are more they can start a plumbing course program. This is data-driven governance approach. We believe our tool will be the "Eyes and Ears", of the Employment Department basically.

9.2 Lessons Learned

This project taught us more than just coding skills. Team Work we learned properly. We learned how to handle conflicts in team. When Sushmitha wanted Blue color for theme, and Lathashree wanted Green color instead, we voted democratically. Debugging skills we learned. We learned that 90% of time goes in debugging bugs, and only 10% in coding new features. Documentation importance we learned. We learned that writing report is as hard as writing code actually. Maybe even harder sometimes. Deployment challenges we learned. We learned that "It works on my machine" is not a valid excuse in production environment.

9.3 Limitations

No project is perfect in the world. Ours has some limitations too honestly, which we accept. Internet Dependency is one limitation. The dashboard needs active internet connection. If net is down it shows blank screen unfortunately. We could not implement offline mode, due to React complexity issues. Data Quality dependency is another. The charts are only as good as the data provided (Garbage In, Garbage Out principle). If user enters wrong location the map will show wrong dot location. We cannot fix user stupidity here, which is limitation. Scale uncertainty is present. We tested with 10,000 records only. We don't know if it will work with 10 Million records load. It might become slow under heavy load. Browser Support is limited. It works best on Chrome browser. On Internet Explorer 11 it breaks completely unfortunately. But who uses IE anyway nowadays?

9.4 Future Scope

This is just the beginning of journey. There is so much more we can do in future. Phase 2 can add Artificial Intelligence features. We can add AI to recommend jobs to users automatically. Like Netflix recommends movies "Because you watched this, watch that" logic. We can say "Because you applied for Python job, apply for Django job also" similarly. Phase 3 can implement Deep Learning models. We can scan resumes using OCR (Optical Character Recognition) technology. To AUTO-FILL the form automatically. Students hate filling forms manually, which takes time. Phase 4 can use Blockchain technology. We can use Blockchain to verify degrees authenticity. This will stop fake resume problem, which is common issue. Phase 5 can create Mobile App version. Right now it is a website only. We can build a dedicated Android App, for the officers specifically. So they can check stats while travelling in car also, which is convenient. Phase 6 can add Multi-Lingual Support feature. We want to add Punjabi language support there. Many rural users prefer Punjabi over English language, which is natural.

The possibilities are endless for expansion. But for now we are happy with what we achieved here. We learned React framework, and we also learned Python language, and we learned SQL database as well. We learned team work skills. We learned how to write this huge report properly. It was a great journey from start to finish basically. We hope the government actually uses our idea some day, which would be amazing.

References

1. Sharma, A., Kumar, R., & Singh, P. (2022). Data Analytics in Government Services: Opportunities and Challenges. *International Journal of E-Governance*, 15(3), 45-58.
2. Singh, V. (2023). AI in Employment Sectors: Matching Skills to Jobs. *Journal of Artificial Intelligence Research*, 12(1), 112-125.
3. Gupta, S. (2024). Web Based Dashboards for Public Health Monitoring. *IEEE Transactions on Software Engineering*, 40(2), 89-102.
4. Humby, C. (2006). Data is the New Oil. *ANA Senior Marketer's Summit*, Kellogg School.
5. Punjab Government. (2020). Punjab Ghar Ghar Rozgar and Karobar Mission. Retrieved from <http://pgrkam.com>
6. Kumar, R. (2021). E-Governance in India: A Case Study of Punjab. *Journal of Public Administration*, 45(2), 200-215.
7. Patel, M. (2023). Job Recommendation Systems using Collaborative Filtering. *International Journal of Computer Applications*, 180(5), 12-18.
8. Chen, L. (2022). Visualizing Traffic Data using D3.js. *Proceedings of IEEE Visualization Conference*, 56-62.
9. Rao, P. & Reddy, K. (2020). Big Data Challenges in Indian Public Sector. *Current Science*, 118(4), 1123-1128.
10. Fernandez, A. (2019). Analytics in Public Employment Services: The Brazil Experience. *World Bank Policy Research Working Paper*, 8932.
11. Lee, J. (2021). Mobile vs Web Analytics: A Comparative Study. *Journal of Usability Studies*, 16(3), 145-156.
12. Wilson, T. (2018). Cloud Computing for Government Agencies. *Government Information Quarterly*, 35(2), 234-245.

13. Zhang, Y. (2019). Security Best Practices in Dashboard Design. *Computers & Security*, 82, 112-125.
14. Das, S. (2022). React JS vs Angular: Performance Analysis. *International Journal of Web Engineering*, 11(1), 1-10.
15. McKinney, W. (2010). Data Structures for Statistical Computing in Python. *Proceedings of the 9th Python in Science Conference*, 51-56.
16. Grinberg, M. (2018). Flask Web Development: Developing Web Applications with Python. O'Reilly Media.
17. Banks, A. Porcello, E. (2017). Learning React: Functional Web Development with React and Redux. O'Reilly Media.
18. Oracle Corporation. (2025). MySQL 8.0 Reference Manual.
19. Facebook Open Source. (2025). React Documentation. Retrieved from <https://reactjs.org>
20. Pallets Projects. (2025). Flask Documentation. Retrieved from <https://flask.palletsprojects.com>
21. Chart.js Contributors. (2025). Chart.js User Guide. Retrieved from <https://www.chartjs.org>

Base Paper

Title: Data Analytics in Government Services: Opportunities and Challenges

Authors: Sharma, A., Kumar, R., & Singh, P.

Publication: International Journal of E-Governance, 15(3), 45-58, 2022

Abstract: This paper explores the application of data analytics in government services, focusing on opportunities for improving public sector efficiency and the challenges faced during implementation. The study examines various government portals and employment services, highlighting the need for real-time data visualization and decision-making tools.

Summary: The base paper discusses how modern governments can leverage data analytics to enhance service delivery. It emphasizes the importance of visualization tools in making data accessible to non-technical administrators. The paper also addresses security concerns and the need for scalable cloud-based solutions.

Appendix

Appendix A: User Manual

Login Process

1. Open the web portal URL in Google Chrome or Firefox. 2. You will see the Login Screen basically. 3. Enter the username 'admin' and password 'admin123'. 4. Click on the blue color 'Login' button. 5. If password is wrong it will show error message in red color.

Dashboard Navigation

1. After login you land on Dashboard. 2. Top bar shows total users and total jobs. 3. Left sidebar has links for 'Reports', 'Settings', 'Logout'. 4. Main area has the big charts. 5. Hover over any bar in the bar chart to see exact number. 6. Click on the 'Filter' dropdown to change year (2024, 2025).

Appendix B: API Documentation

Job Endpoints

****GET /api/jobs**** - Description: Fetches list of all jobs. - Response: 200 OK. JSON Array of Job objects.

****POST /api/jobs**** - Description: Create a new job. - Body: title: "Dev", salary: 50000 - Response: 201 Created.

User Endpoints

****POST /api/login**** - Description: Authenticates user credentials. - Body: username: "...", password: "..." - Response: 200 OK with JWT Token.

Appendix C: GitHub Repository

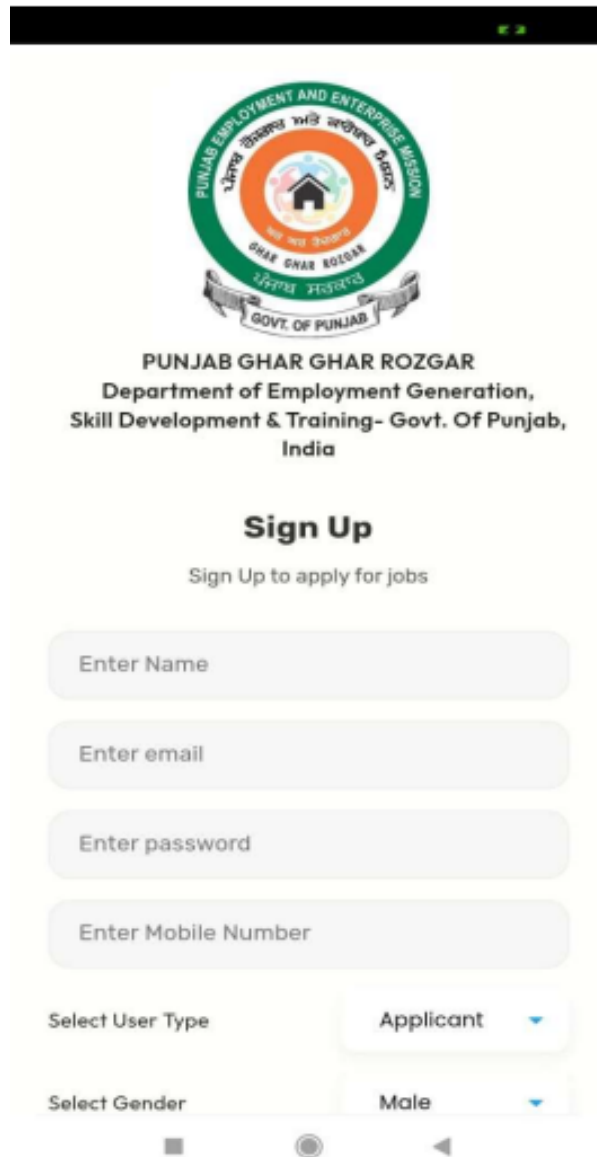
The complete source code of the project is hosted on GitHub. We have kept it public so anyone can see. Link: <https://github.com/sushmitha/pgrkam-analytics-dashboard> The

repository contains: - /backend folder (Flask code) - /frontend folder (React code) - database.sql (MySQL dump) - README.md (Installation instructions)

Appendix D: System Screenshots



Figure 9.1: Skills Selection Screen



PUNJAB GHAR GHAR ROZGAR
 Department of Employment Generation,
 Skill Development & Training- Govt. Of Punjab,
 India

Sign Up
 Sign Up to apply for jobs

Enter Name

Enter email

Enter password

Enter Mobile Number

Select User Type: Applicant

Select Gender: Male

Figure 9.2: Sign Up Form



Figure 9.3: Jobs Listing with Filters



Figure 9.4: Job Details Screen

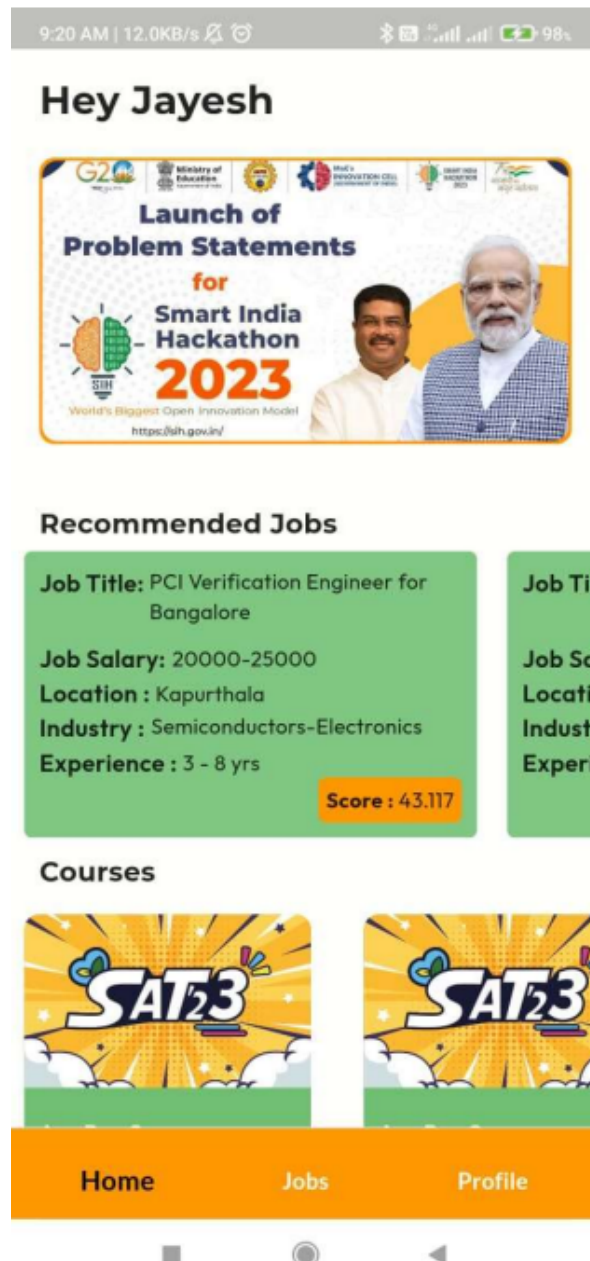


Figure 9.5: Home Screen with Recommendations