PART TIME WORK PORTAL

MAIN PROJECT REPORT

Submitted in partial fulfillment of the requirements

for the award of the degree of

BACHELOR OF COMPUTER APPLICATIONS

2022 – 25

Done By,

Sam Joseph (220021085488)

Under the guidance of

Ms. Aneesha K Jose

RAJAGIRI COLLEGE OF MANAGEMENT AND APPLIED SCIENCES

(Affiliated to Mahatma Gandhi University, Kottayam)

Rajagiri Valley P.O, KERALA- 682039

RAJAGIRI COLLEGE OF MANAGEMENT AND APPLIED SCIENCES

(Affiliated to Mahatma Gandhi University, Kottayam)

Rajagiri Valley P.O, KERALA- 682039

CERTIFICATE

*This is to certify that the project work titled **"Part Time Work Portal "** submitted to Mahatma Gandhi University in partial fulfillment of the requirements for the award of the Degree of Bachelor Of Computer Applications is a record of the original work done by **Sam Joseph** under my supervision and guidance and that this project work has not formed the basis for the award of any Degree/ Diploma/ Fellowship or similar title to any candidate of this or any other University.*

Faculty In-Charge	**Coordinator**	**Head of the Department**

Submitted for viva-voce held on ……../……../………

Internal Examiner	**External Examiner**

RAJAGIRI COLLEGE OF MANAGEMENT AND APPLIED SCIENCES

(Affiliated to Mahatma Gandhi University, Kottayam)

Rajagiri Valley P.O, KERALA- 682039

DECLARATION

I **SAM JOSEPH** hereby declare that project report entitled **"PART TIME WORK PORTAL"** submitted in partial fulfillment of the requirements for the award of the Degree of Bachelor of Computer Applications is a record of original research work done by me under the supervision & guidance of **Ms. ANEESHA K JOSE** and the dissertation has not formed the basis for the award of any Degree/Diploma/ Associate ship / Fellowship or similar title to any candidate of this or any other University.

Sam Joseph

Place: Kakkanad

Date:…../……/……

ACKNOWLEDGEMENT

I consider it as a privilege to express my sincere gratitude and respect to all those who guided and inspired me in the successful completion of this project work.

I convey my reverential salutation to **Almighty God**, for enabling me to take up and complete the project successfully.

I would like to express my sincere thanks to **Rev. Dr. Mathew Vattathara CMI,** Director and **Prof. Dr. Laly Mathew,** Principal, Rajagiri College of Management and Applied Sciences for providing the necessary infrastructure and support for the completion of this project work.

I would like to express my sincere thanks to **Mr. Sijo Jacob**, HOD, Department of Computer Science, Rajagiri College of Management and Applied Sciences for his valuable advice and support which have helped me greatly in the accomplishment of the project.

I would like to express my sincere thanks to the Project Coordinator **Mr. Joby Jacob**, Assistant Professor, Department of Computer Science, Rajagiri College of Management and Applied Sciences for his consistent guidance and inspiration throughout the period for the completion of my project.

I sincerely thank my project guide **Ms. Aneesha K Jose**, Assistant Professor, Department of Computer Science, Rajagiri College of Management and Applied Sciences for her consistent guidance and inspiration throughout the period for the completion of this project.

I would like to thank all the teaching and non-teaching staff of Rajagiri College of Management and Applied Sciences, for their valuable guidance and suggestions rendered during the project.

Finally, I thank my parents and all my friends for their help, encouragement and moral support given to me during the course of this work.

Sam Joseph

TABLE OF CONTENTS

INTRODUCTION

1.1 ABOUT THE PROJECT

The Part-Time Work Portal is a web-based platform created to streamline the process of connecting employers and job seekers for part-time job opportunities. By addressing the growing demand for flexible employment, the platform allows employers to post temporary or part-time job openings and helps applicants find suitable roles based on their skills, experience, and location.

The portal offers a centralized space for job posting, application management, task allocation, and payment handling, simplifying the entire cycle of hiring and completing part-time work. It aims to create an efficient ecosystem where employers can find the right candidates quickly and where applicants can easily manage their job applications, assignments, and payments.

REQUIREMENT ANALYSIS AND SPECIFICATION

2.1 SYSTEM STUDY

2.1.1 Existing System

The current method of finding part-time jobs is highly unstructured and inefficient. Employers often rely on word-of-mouth, local advertisements, or social media to find suitable candidates, leading to a limited reach and inefficient hiring process. Similarly, students and job seekers struggle to find relevant job opportunities that match their skills and location preferences.

Most existing systems lack a centralized platform that connects employers with potential candidates in a streamlined manner. There is no proper categorization of job listings, making job searches cumbersome. Additionally, the absence of a structured application process makes it difficult for employers to track and manage job applications effectively. Payment transparency and communication between employers and applicants are also major challenges in the traditional system.

Disadvantages of existing system

- Time consuming hiring Process
- Lack of structured job listing
- Inefficient application tracking
- No integrated communication system
- Inconsistent payment processing

2.1.2 Proposed System

The Part-Time Work Portal aims to eliminate the inefficiencies of the existing system by providing a fully automated and user-friendly platform for connecting employers and job seekers. This online platform enables employers to post job openings, review applications, communicate with applicants, and track hiring status effortlessly. Similarly, students and job seekers can browse jobs, apply for relevant positions, and track their application progress in a structured manner.

The platform ensures real-time job updates and provides filtering options based on job category, location, and required skills. Secure authentication and role-based access control enhance trust and transparency between employers and applicants. The system also facilitates seamless communication and payment tracking, ensuring clarity in job agreements and compensation.

Advantages of proposed system

- Streamlined and efficient hiring process.
- Structured job listings with search and filter options.
- Automated application tracking.
- Enhanced employer-applicant communication.
- Secure and transparent payment processing
- Real-time job updates

2.1.3. Feasibility Study

A feasibility study is conducted to evaluate whether the proposed system is viable and beneficial in terms of cost, technical implementation, and operational efficiency. It helps developers anticipate the future impact of the system and determine its effectiveness. The study includes the following aspects:

- Technical Feasibility
- Economic feasibility
- Operational feasibility

2.1.3.1. Technical Feasibility

The system is assessed from a technical perspective to ensure that it meets the required functionalities and performance standards. The project utilizes the latest web development technologies, including ReactJS for the frontend and MySQL for database management. The system is designed to handle multiple users simultaneously and provide real-time job updates without performance issues. Minimal constraints exist in terms of hardware and software requirements, making the project technically feasible.

2.1.3.2. Economic Feasibility

Economic feasibility evaluates the cost of development against the expected benefits. The cost of developing the Part-Time Work Portal has been carefully analyzed, ensuring that it does not exceed the potential financial and operational benefits. The platform eliminates the need for costly manual job searches, reducing hiring expenses for employers and improving job accessibility for applicants. The financial analysis confirms that the project is economically viable and cost-effective.

2.1.3.3. Operational Feasibility

Operational feasibility determines whether the system will be effectively utilized once implemented. The portal provides an intuitive interface, ensuring ease of use for both employers and job seekers. There are no significant barriers to adoption, as the system enhances job search and hiring processes without requiring specialized training. The availability of real-time updates an structured application management further ensures the system's usability. Hence, the project is operationally feasible.

2.2 USER CHARACTERISTICS

The Part-Time Work Portal is designed to cater to the needs of different users involved in the hiring process. The system provides an efficient, secure, and user-friendly platform for employers to post jobs, applicants to find jobs, and administrators to oversee operations.

The users of this system are categorized as follows:

2.2.1 Administrator

2.2.2 Employer

2.2.3 Applicant

2.2.1 Administrator

The Administrator oversees the entire Part-Time Work Portal, ensuring smooth operations, security, and compliance. They manage user accounts, approve job postings, monitor applications, handle disputes, and oversee payment transactions. Admins also generate reports and maintain the platform to ensure a seamless experience for all users.

2.2.2 Employer

Employers can register on the portal to post job openings, review applications, and hire suitable candidates. They have access to applicant details, can communicate with potential hires, and track the status of their job listings. Employers also manage payments and job agreements through the platform, ensuring transparency and efficiency in the hiring process.

2.2.3 Applicant

Applicants (job seekers) can register on the portal to browse available job opportunities, apply for relevant positions, and track their application status. They can filter jobs based on location, category, and skill requirements. Applicants can also communicate with employers, negotiate job details, and receive payments securely through the system. The portal enhances accessibility and streamlines the job search process for students and part-time workers.

2.3 SYSTEM SPECIFICATION

2.3.1. Hardware Specification

The selection of hardware is very important for the existence and proper working of any software. When selecting the hardware, the size and capacity requirements are also noted.
Below are the hardware details required by the system.

Processor	Intel Core i5-12450H(2.00 GHz) or above
RAM	8 GB or above
Storage	512 GB and above
Other	Keyboard and Mouse

2.3.2 Software Specification

Operating System	Windows 11
Front End	React.js, CSS, JavaScript
Back End	Node.js, MySQL

2.3.3 About software tools and platform

FRONT-END SPECIFICATION: ReactJS

ReactJS is a popular JavaScript library used for building dynamic and interactive user interfaces. It was developed by Facebook and is widely used for developing modern web applications. ReactJS follows a component-based architecture, allowing developers to create reusable UI components efficiently.

Characteristics

- Uses a virtual DOM for faster rendering and improved performance.
- Follows a declarative programming approach, making the code more readable and maintainable.
- Supports one-way data binding, ensuring better control over the application state.
- Can be integrated with various state management libraries like Redux for handling complex application states.

BACK-END SPECIFICATION: Node.js

Node.js is an open-source, cross-platform JavaScript runtime environment that allows developers to run JavaScript on the server side. It is built on the V8 JavaScript engine and is widely used for developing scalable web applications.

Characteristics

- Uses an event-driven, non-blocking I/O model, making it lightweight and efficient.
- Supports asynchronous programming, allowing for high-performance web applications.
- Provides built-in modules for handling HTTP requests, file systems, and streams.
- Can be used with Express.js to simplify backend development and API creation.

DATABASE SPECIFICATION: MySQL

MySQL is a widely used relational database management system (RDBMS) known for its reliability, scalability, and performance. It is an open-source database solution that efficiently handles structured data for web applications.

Characteristics

- Uses Structured Query Language (SQL) for managing and querying data.
- Provides support for ACID (Atomicity, Consistency, Isolation, Durability) transactions.
- Ensures data integrity with features like foreign key constraints and indexing.
- Supports replication and clustering for high availability and load balancing.

SYSTEM MODELING

3.1 MODULE AND DESCRIPTION

The Part-Time Work Portal is a web-based platform created to streamline the process of connecting employers and job seekers for part-time job opportunities. There are 8 modules in this project. They are as follows:

1. Employer Registration Management
2. Applicant Registration Management
3. Job Posting Management
4. Job Application Management
5. Application Allocation Management
6. Work Execution Management
7. Remuneration Management
8. Feedback Management

1. EMPLOYER REGISTRATION MANAGEMENT

This module facilitates the registration and onboarding process for employers, such as businesses and organizations. Employers can create accounts, provide company details, and verify their identity to ensure platform authenticity. Once registered, they gain access to features like job posting, application tracking, and payment processing

2. APPLICANT REGISTRATION MANAGEMENT

This module is designed for individuals seeking part-time jobs. Applicants can register by creating profiles that include personal details, skills, experience, and preferences. It may also allow for uploading resumes and certifications. This module ensures a seamless onboarding process and helps applicants access job listings tailored to their profiles.

3. JOB POSTING MANAGEMENT

Employers can use this module to create detailed job posts that include the job title, description, required skills and experience, and the being offered. They can manage the visibility of these posts by activating or deactivating them as needed. Additionally, employers can set an application deadline to ensure timely responses from candidates.

4. JOB APPLICATION MANAGEMENT

This module allows applicants to apply for jobs posted on the platform. Job seekers can browse available positions and submit their applications directly to the employers. The system tracks each job application’s status, making it easier for both employers and applicants to monitor progress.

5. APPLICANT ALLOCATING MANAGEMENT

Employers can view, manage, and evaluate the incoming job applications using this module. The system updates the status of applications based on the employer’s actions, such as Shortlisting or Hiring. It keeps both employers and applicants informed about the status of their application.

6. WORK EXECUTION MANAGEMENT

This module manages the process of applicants accepting work, performing tasks, and marking work as completed. It ensures that applicants and employers are on the same page regarding the status of work assignments. Applicants carry out the assigned tasks based on the job requirements. Applicants mark the work as completed, prompting the employer to review the work for approval.

7. REMUNERATION MANAGEMENT

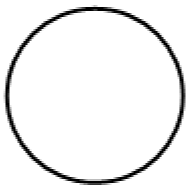
This module facilitates the payment process from employers to applicants after the work is completed. Employers can process payments based on the agreed remuneration for the job, ensuring applicants receive compensation promptly.

8. FEEDBACK MANAGEMENT

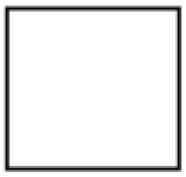
After job completion, both employers and applicants can provide feedback about their experience. This module collects and displays feedback, which helps both parties improve their future interactions. Feedback can be in the form of ratings, comments, and suggestions for improvement.

3.2 DATA FLOW DIAGRAM

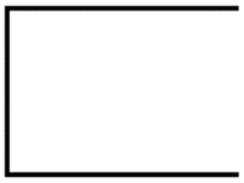
A data flow diagram (DFD) is a graphical tool used to illustrate and analyze the movement of data within a system. It serves as a fundamental tool from which other components of the system are developed. DFDs describe the logical flow of data between various elements in a system, such as people, departments, and workstations, independently of the physical components. These diagrams, often referred to as "bubble charts," are crucial for understanding system requirements and identifying key data transformations that will be translated into programs during system design. DFDs consist of interconnected bubbles representing processes, data sources or destinations, data stores, and information or data lines.
The four symbols used for drawing DFDs are:



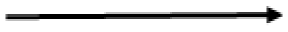
Process (or Function)



External Entity



Data Store



Data Flow

Rules for drawing data flow diagrams:

Rule 1: Define the context of the diagram by identifying all input and output data flows.

Rule 2: Choose a starting point for drawing the DFD.

Rule 3: Provide meaningful labels for all data flow lines.

Rule 4: Label processes with action verbs that relate to input and output data flows.

Rule 5: Exclude insignificant functions usually handled during programming.

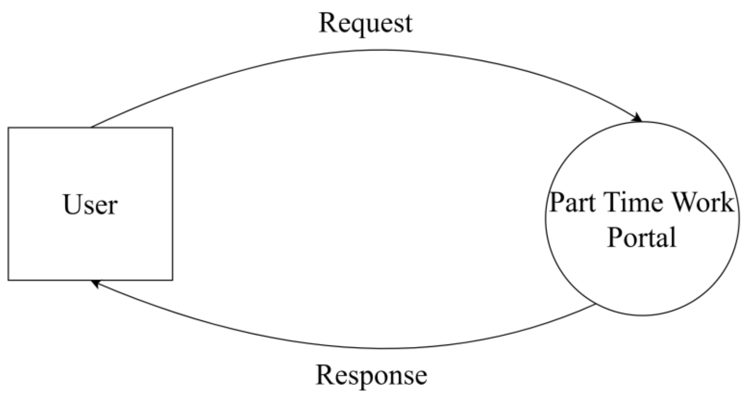
Rule 6: Avoid including control or flow of control information.

Rule 7: Avoid overcrowding the DFD with excessive information.

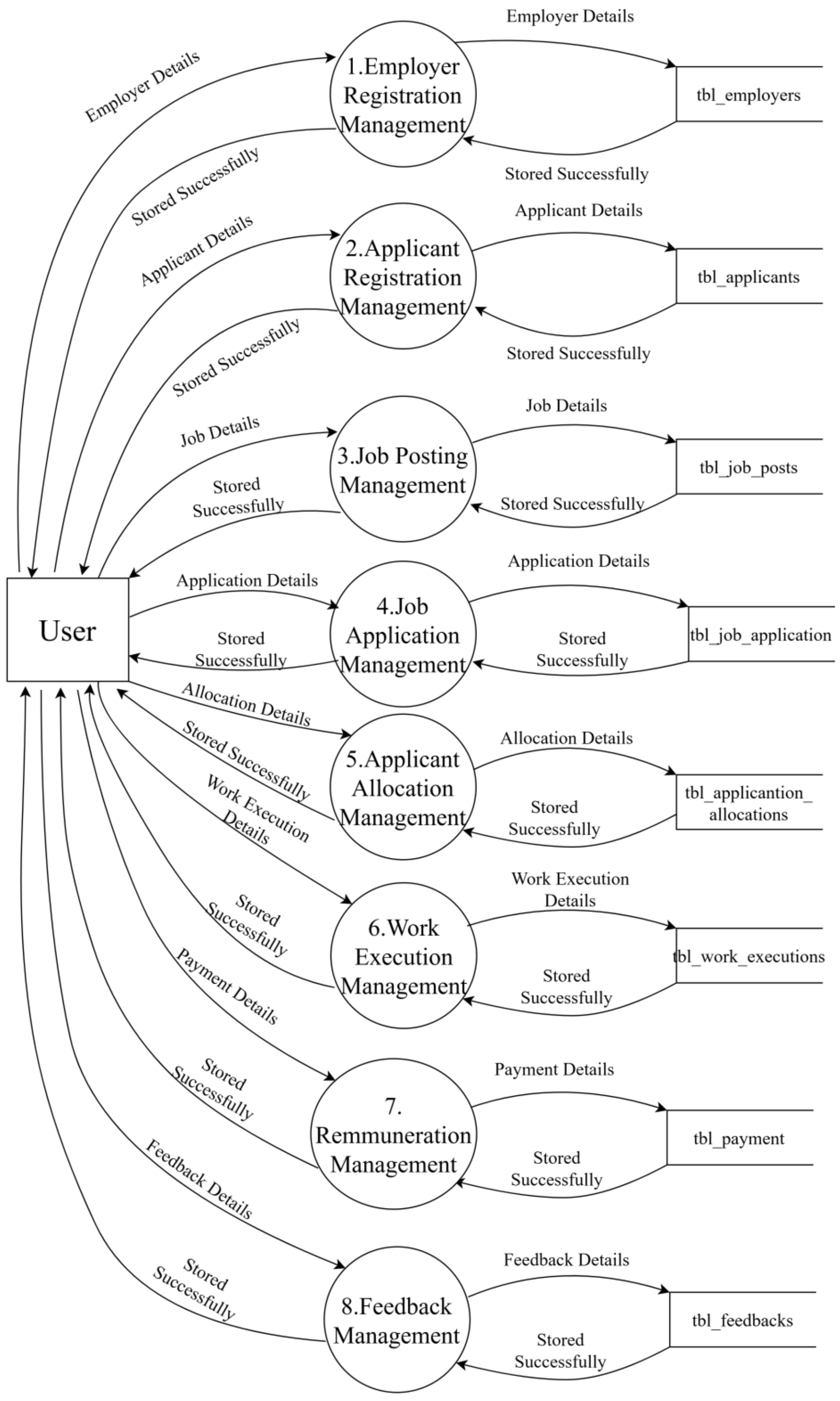
Rule 8: Be open to starting the diagram over if necessary.

DATA FLOW DIAGRAMS

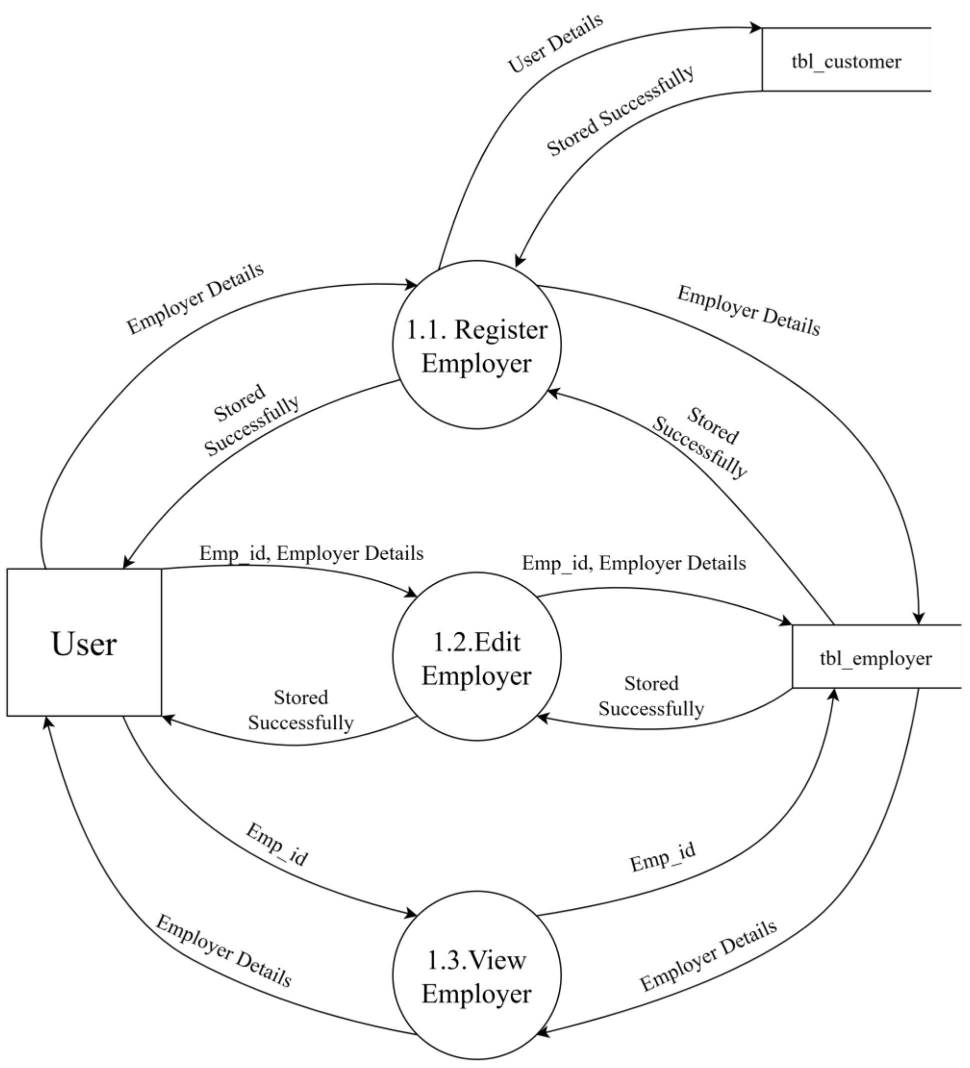
Level 0 DFD Showing Part Time Work Portal



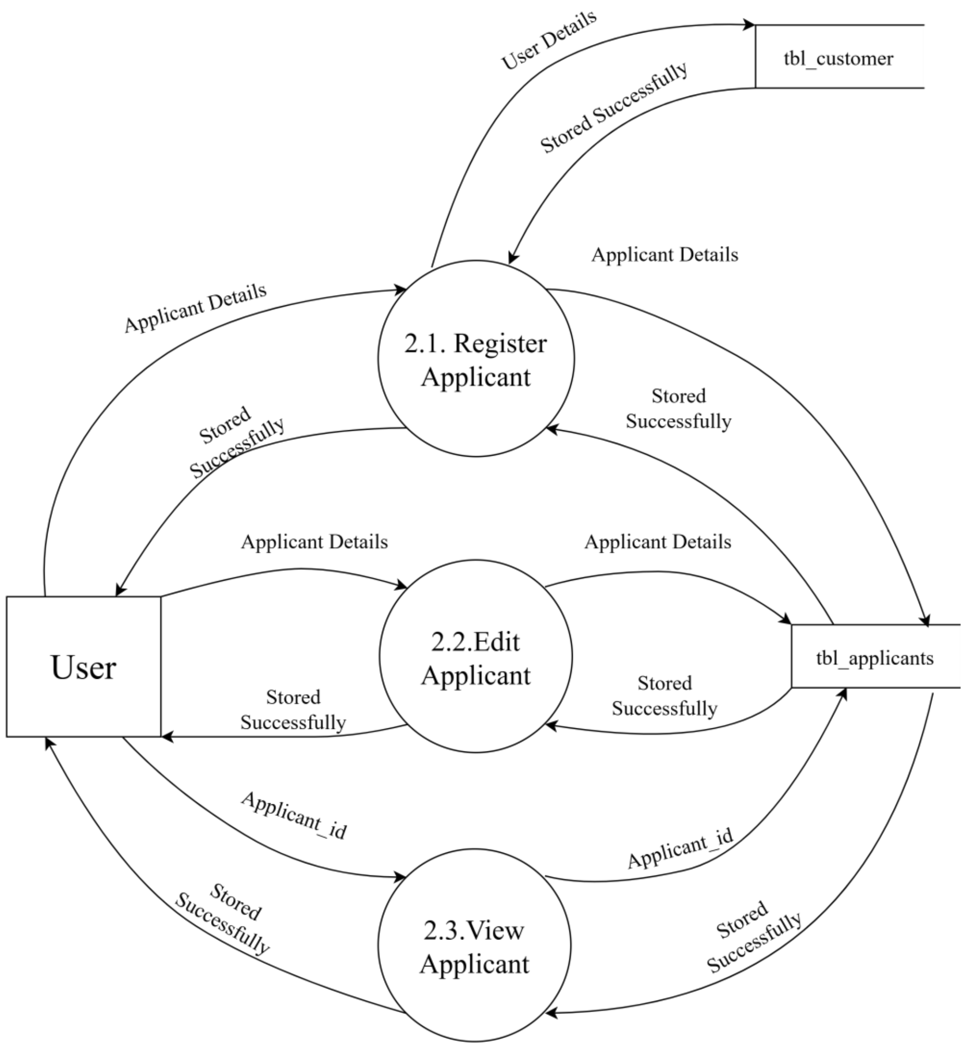
Level 1 DFD showing Part Time Work Portal



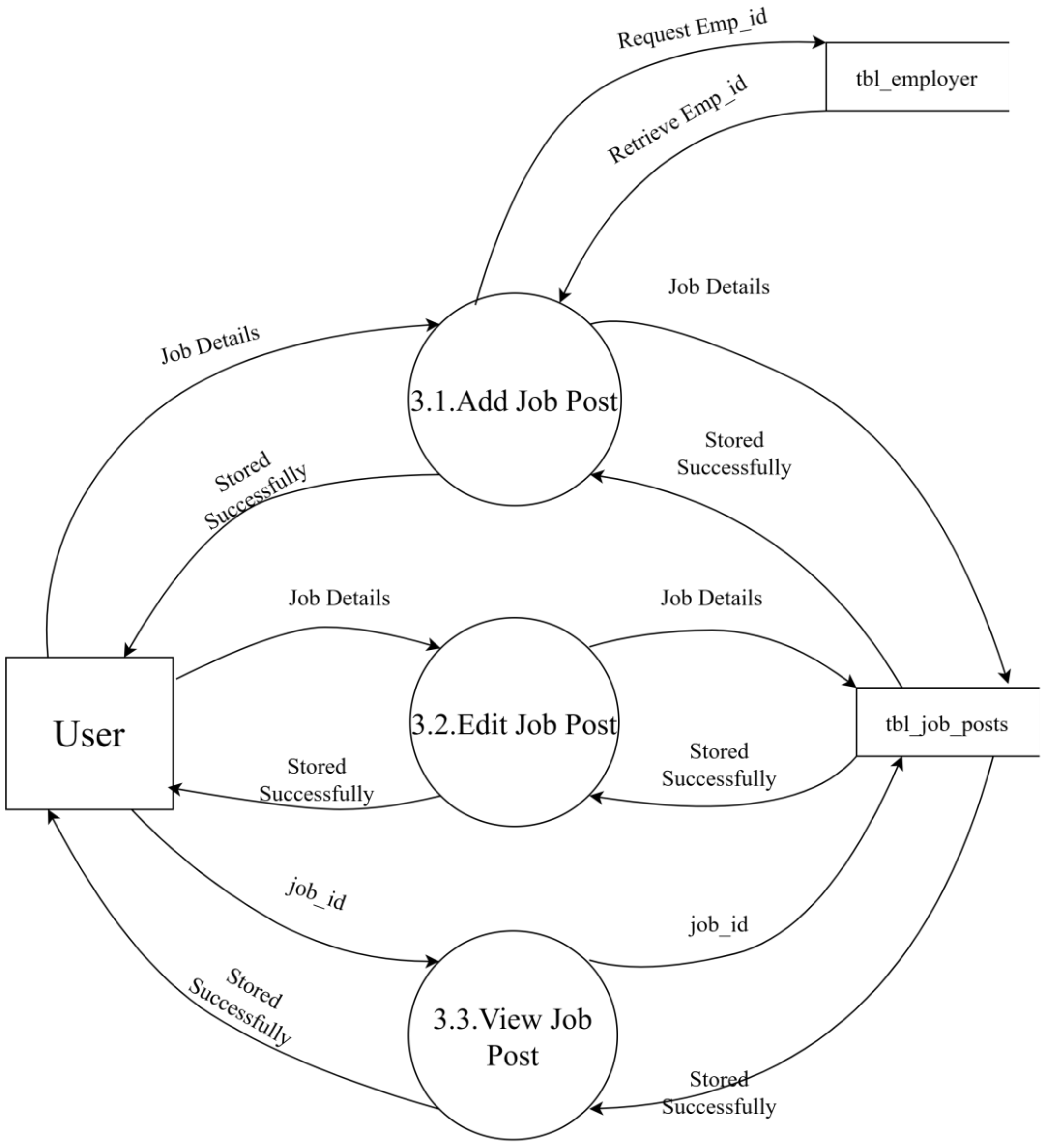
Level 2 DFD showing Employer Registration Management



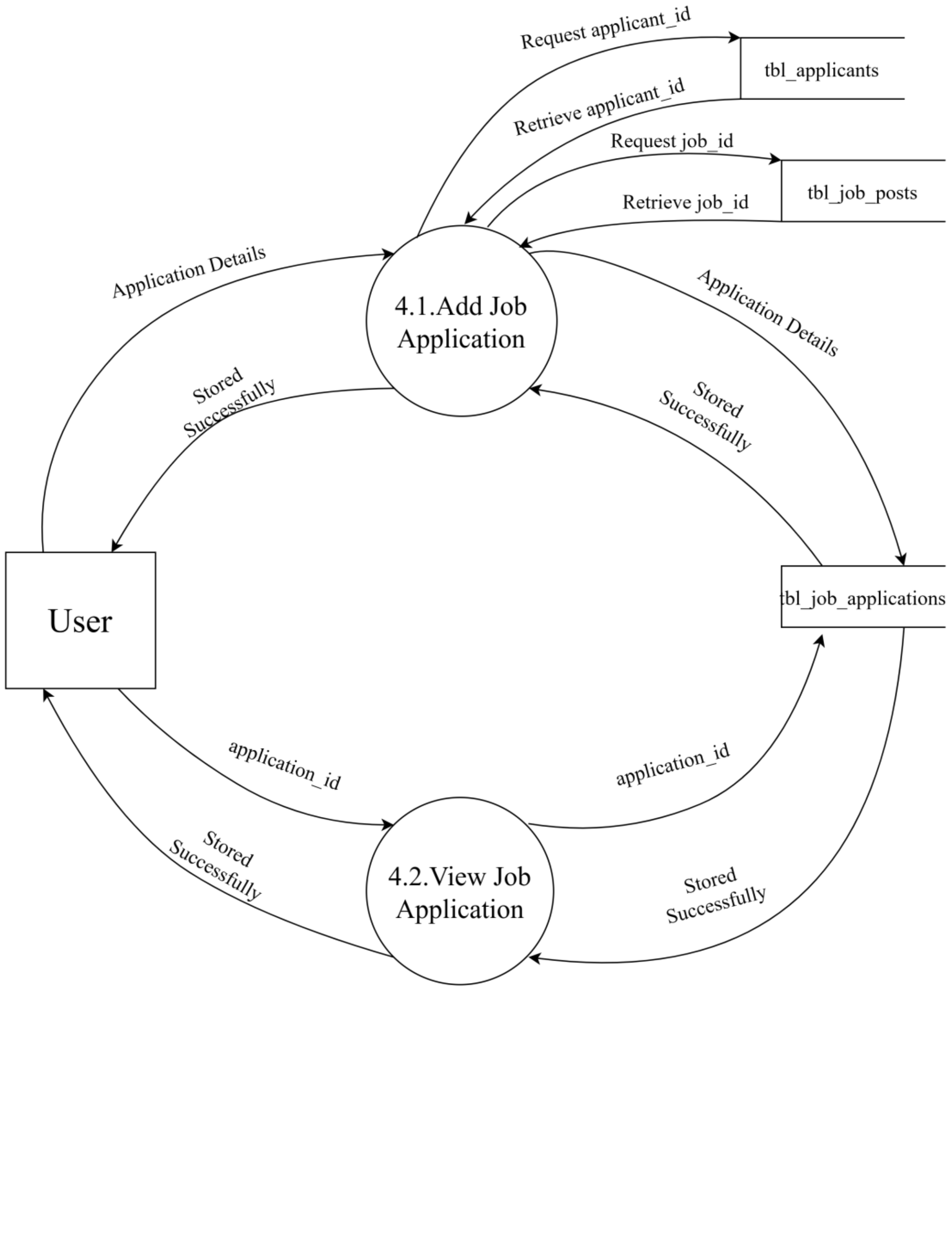
Level 2 DFD showing Applicant Management



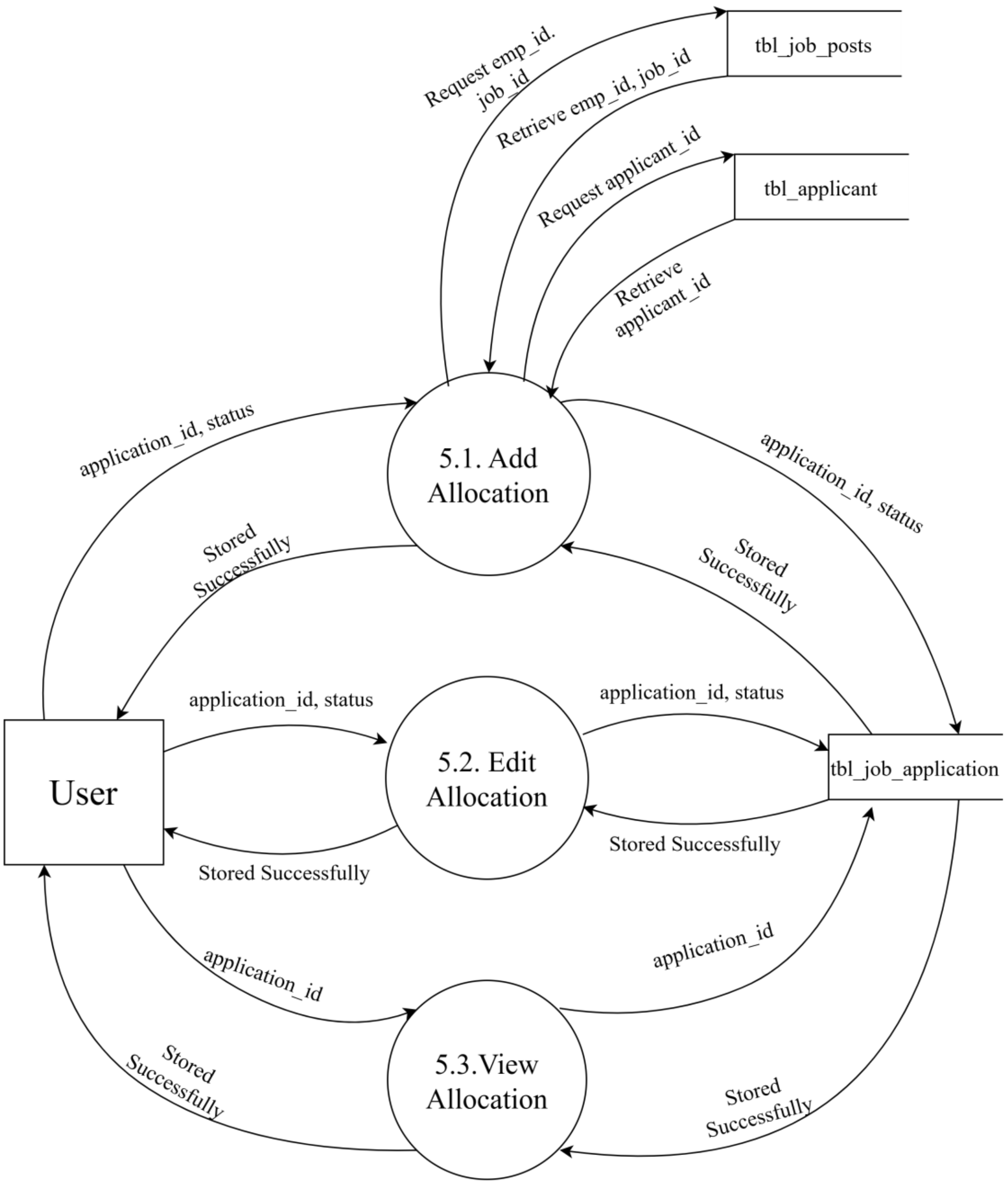
Level 2 DFD showing Job Posting Management



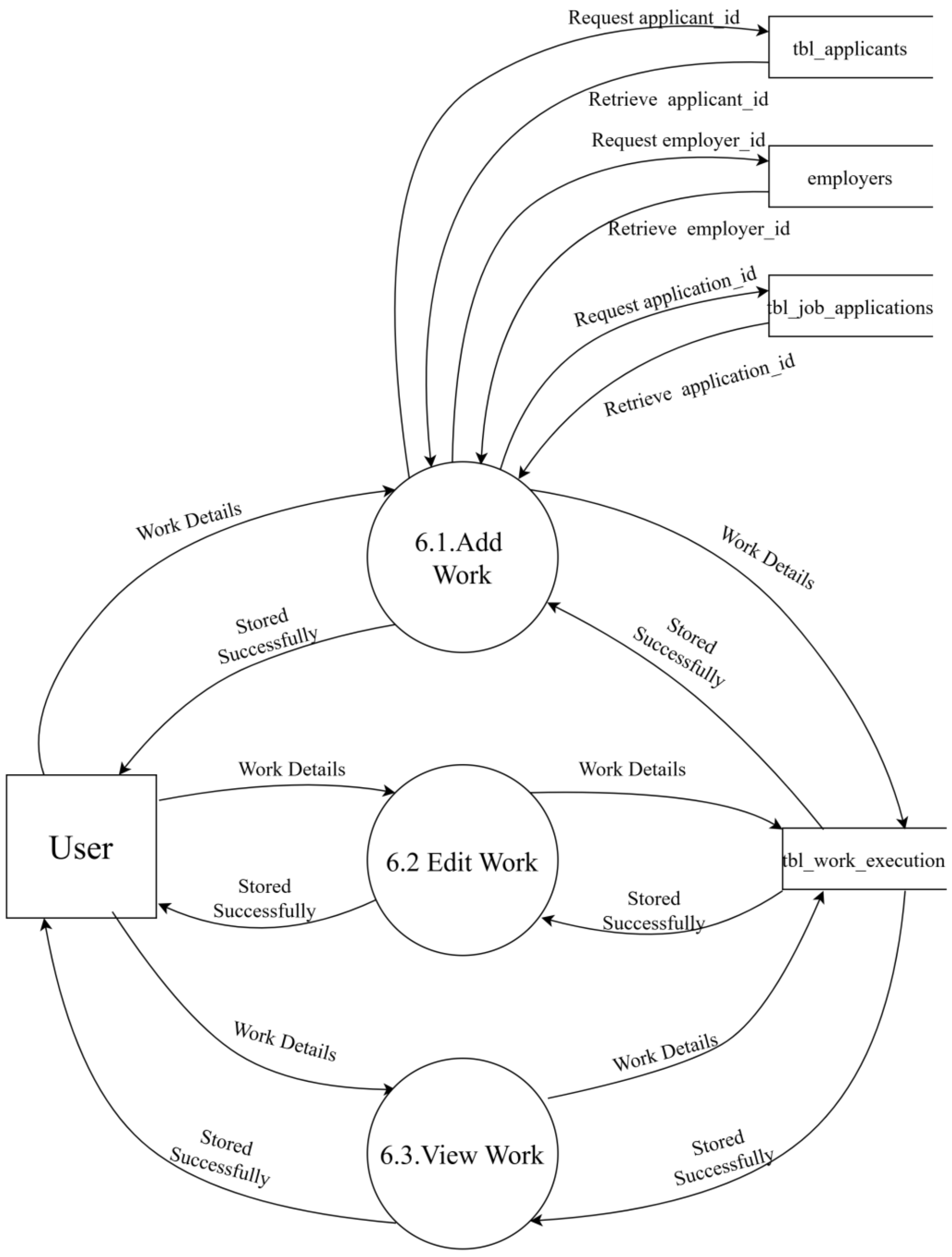
Level 2 DFD showing Job Application Management



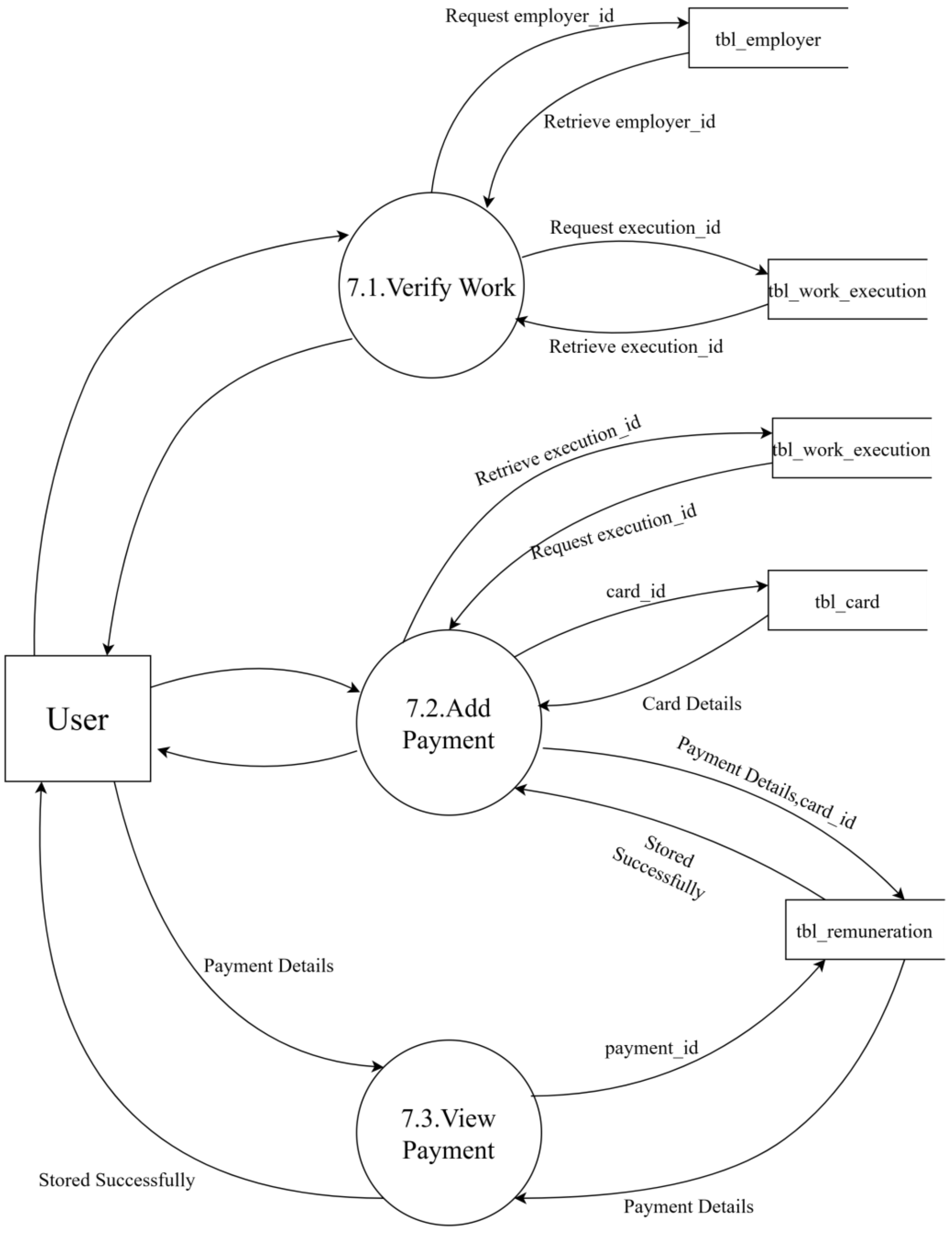
Level 2 DFD showing Application Allocation Management



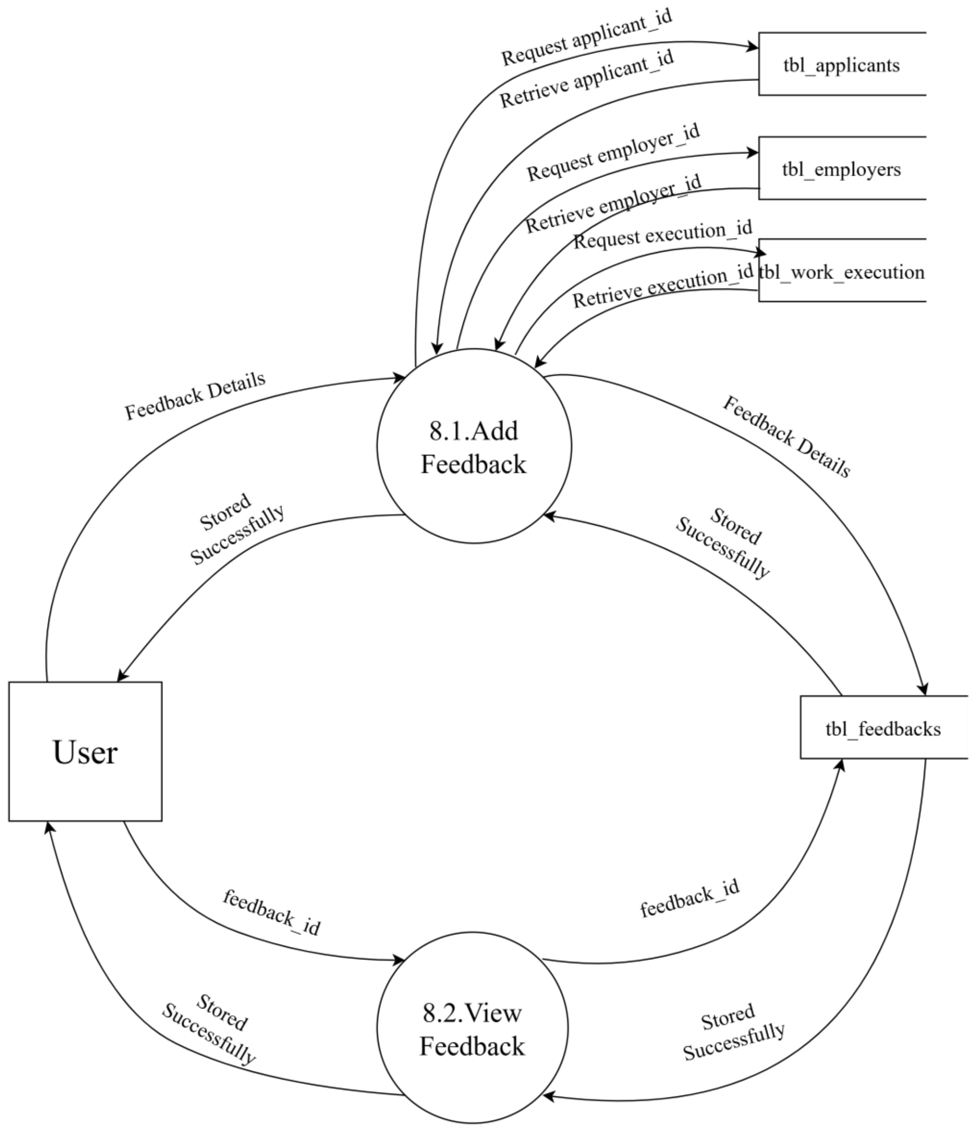
Level 2 DFD showing Work Execution Management



Level 2 DFD showing Remuneration Management

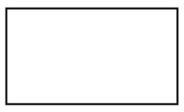


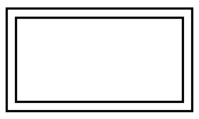
Level 2 DFD showing Feedback Management

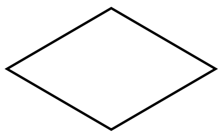


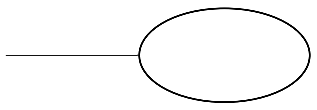
3.3 ENTITY RELATIONSHIP DIAGRAM

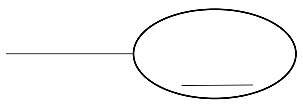
The ER model provides a conceptual way to perceive the real world, representing it as a collection of entities and the relationships between them. Central to this model is the Entity-Relationship diagram, a visual tool used to illustrate data components. ER modelling is extensively applied in the initial design phase of database applications, and its concepts are fundamental to various database design tools.

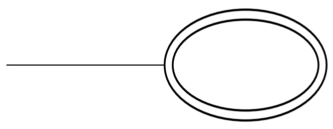

Entity Type


Weak Entity Type

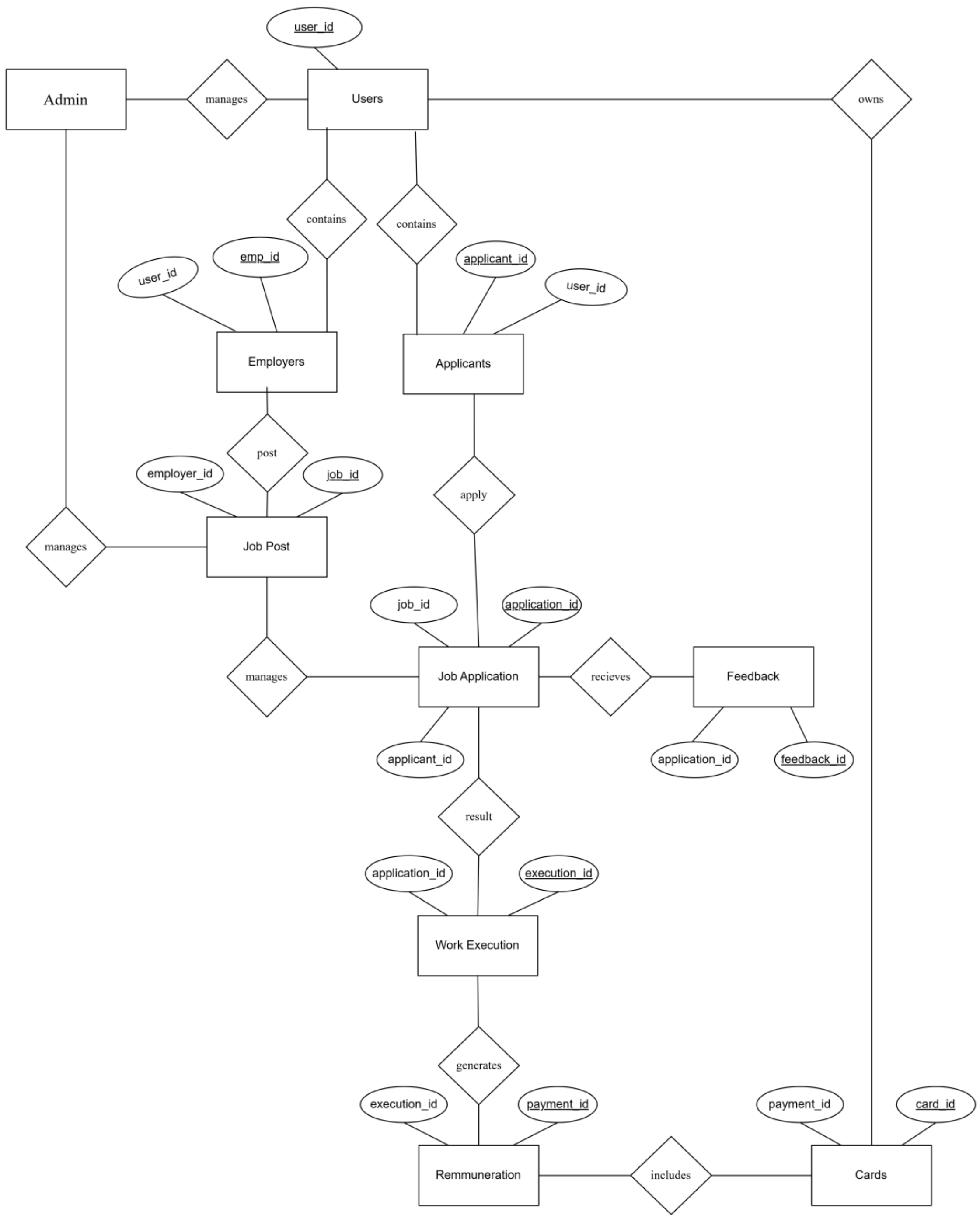

Relationship Type


Attribute


Key attribute


Multivalued Attribute

Entity Relationship diagram for Part Time Work Portal



SYSTEM DESIGN

4.1 INPUT DESIGN

Input design is the process of converting a user-oriented description of the inputs to a computer-based system into a programmer-oriented specification. The quality of system input determines the quality of system output. Input specification describes the manner in which data enter the system for processing. Input design features can ensure the reliability of the system and produce result from accurate data or they can result in the production of errors. The input design also determines whether the user can interact efficiently withthe system.

Input design requires consideration of the needs of the data entry operator. Three data entry considerations are:

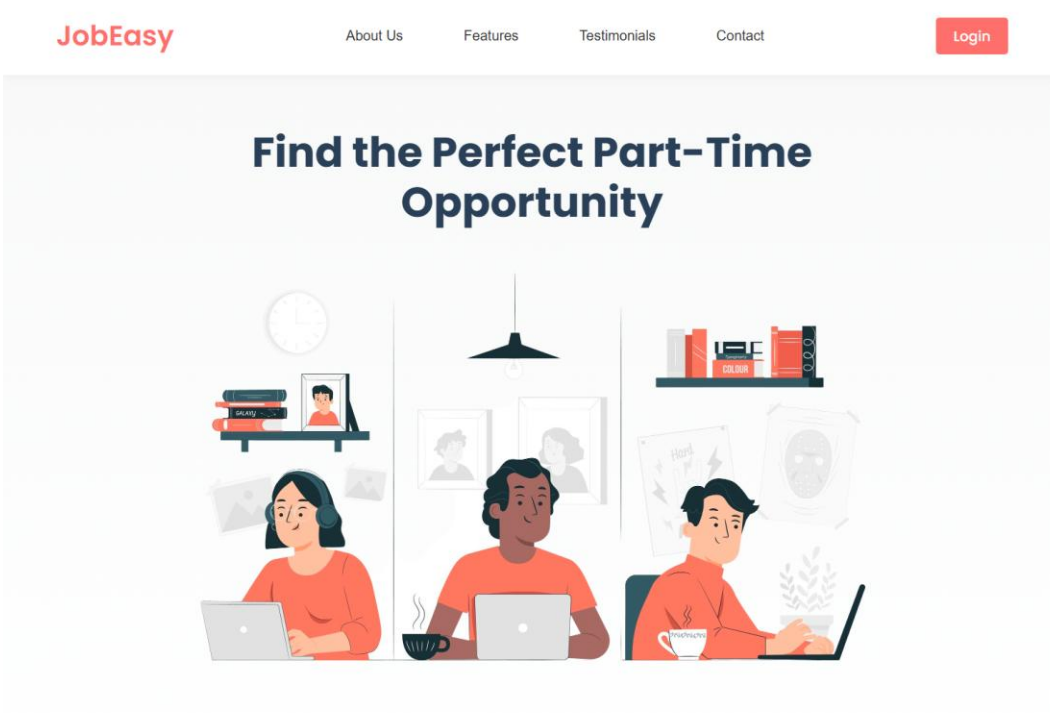
- The field length must be documented
- The sequence of fields must match the sequence of the fields on the source document.
- The data format must be identified to the data entry operator.

In our system almost all inputs are being taken from the databases. To provide adequate inputs we have to select necessary values from the databases and arrange it to the appropriate controls.

Inaccurate input data are the most common cause of errors in data processing. Errors entered by data entry can be controlled by input design. Input design isthe process of converting user-oriented inputs to a computer-based format. There are three major approaches for entering data into the computer. They are menus, formatted forms and prompts. A menu is a selection list that simplifies computer data access or entry. Instead of remembering what to enter, the user choices from the list of option. A formatted form is a preprinted form or a template that request the user to enter data in appropriate location. Itis a fill-in-the-blank type form. The form is flashed on the screen as a unit. Inprompt the system displays one enquiry at a time, asking the user for a response.

Home Page

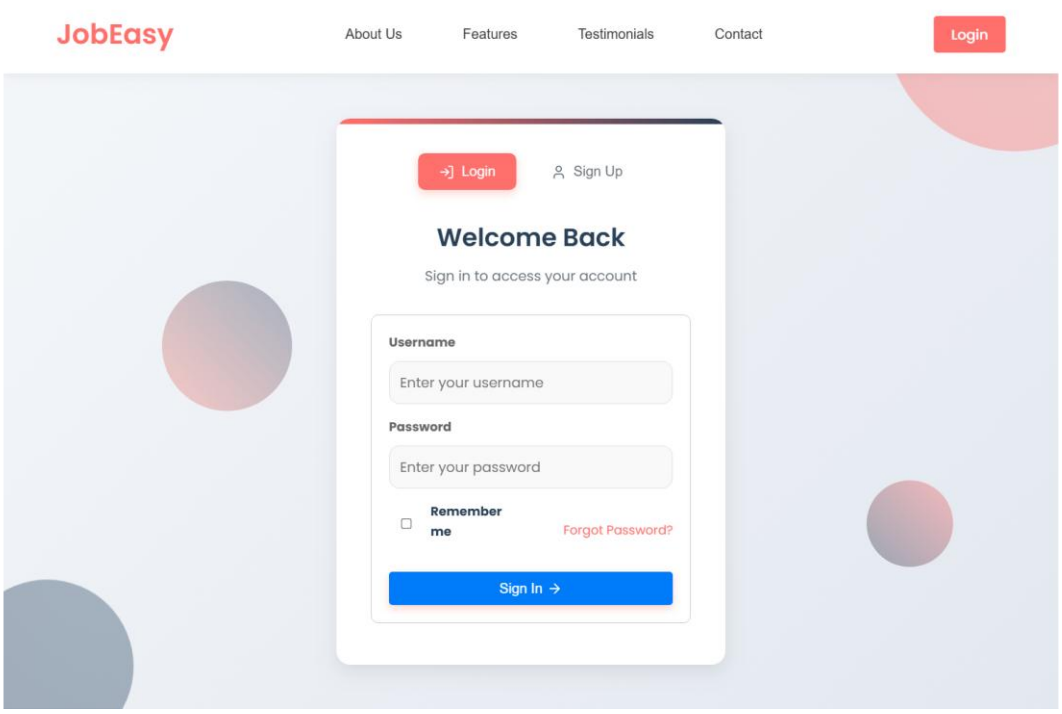
Description: This is the homepage for all users.



Find the Perfect Part-Time Opportunity

Login Form

Description: This is the login page for all users.



Welcome Back

Sign in to access your account

Username

Enter your username

Password

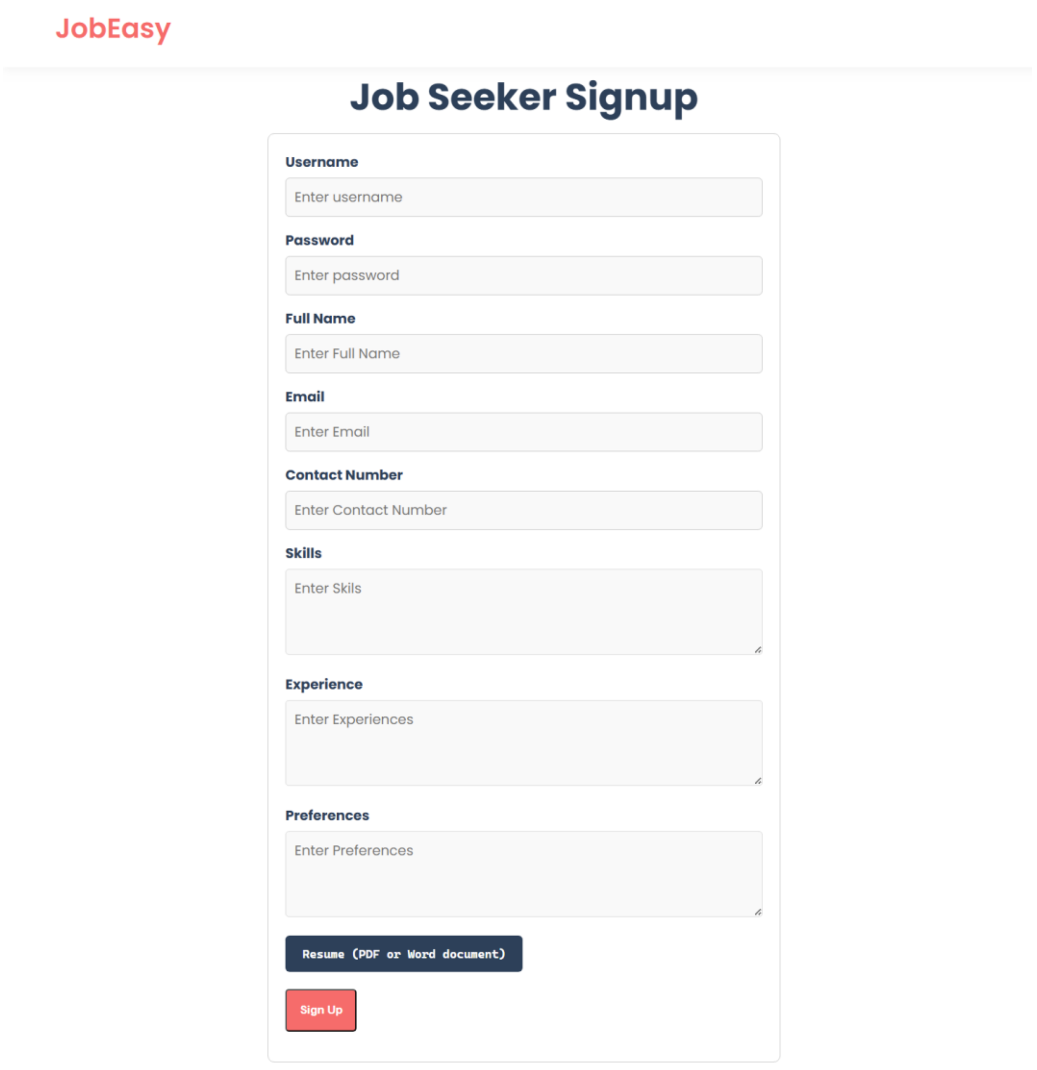
Enter your password

Remember me

Forgot Password?

Employer Management

Description: Register Employer

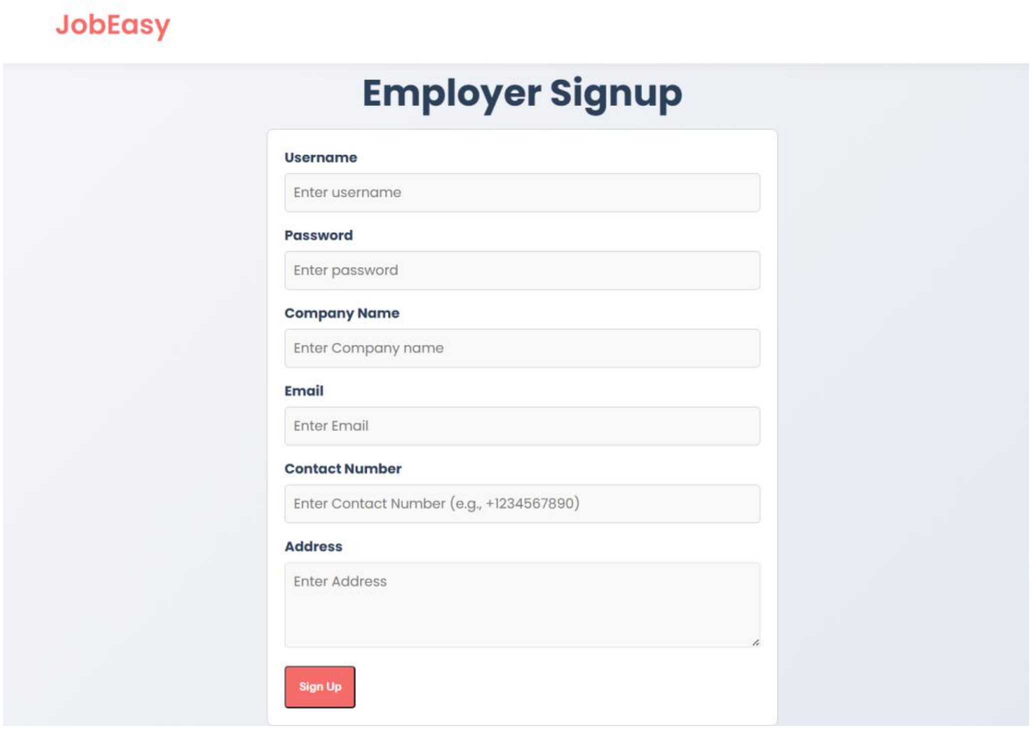

JobEasy

Job Seeker Signup

Username	Enter username
Password	Enter password
Full Name	Enter Full Name
Email	Enter Email
Contact Number	Enter Contact Number
Skills	Enter Skils
Experience	Enter Experiences
Preferences	Enter Preferences

Applicant Management

Description: Register Applicant



JobEasy

Employer Signup

Username

Enter username

Password

Enter password

Company Name

Enter Company name

Email

Enter Email

Contact Number

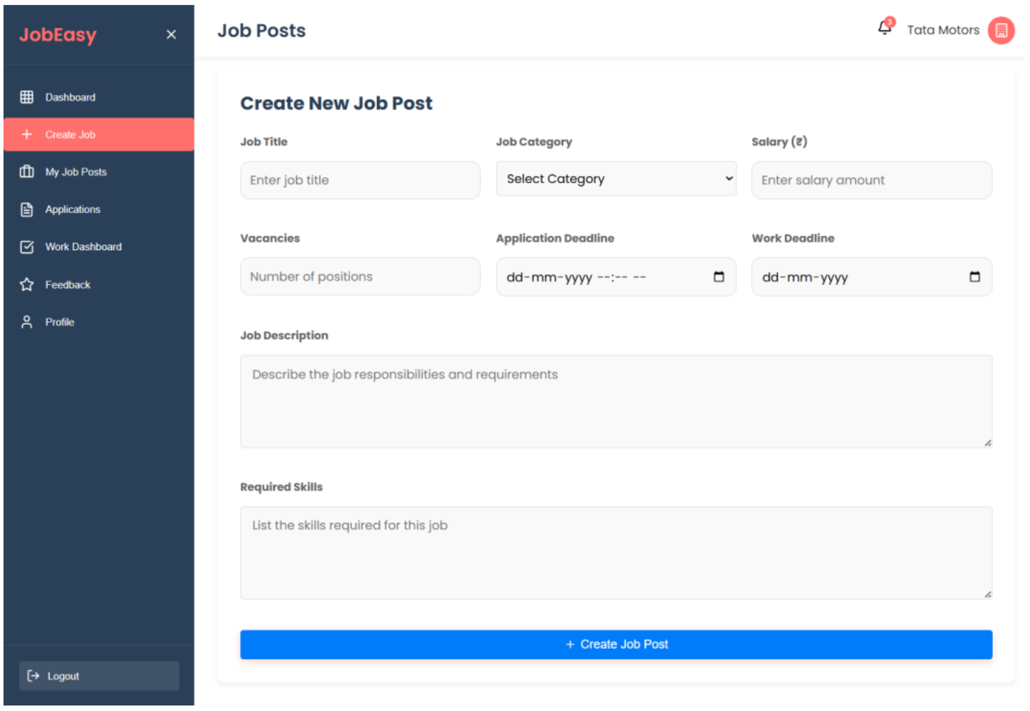
Enter Contact Number (e.g., +1234567890)

Address

Enter Address

Job Management

Description: Add Job Posts



Create New Job Post

Job Title

Enter job title

Job Category

Select Category

Salary (₹)

Enter salary amount

Vacancies

Number of positions

Application Deadline

dd-mm-yyyy --:-- --

Work Deadline

dd-mm-yyyy

Job Description

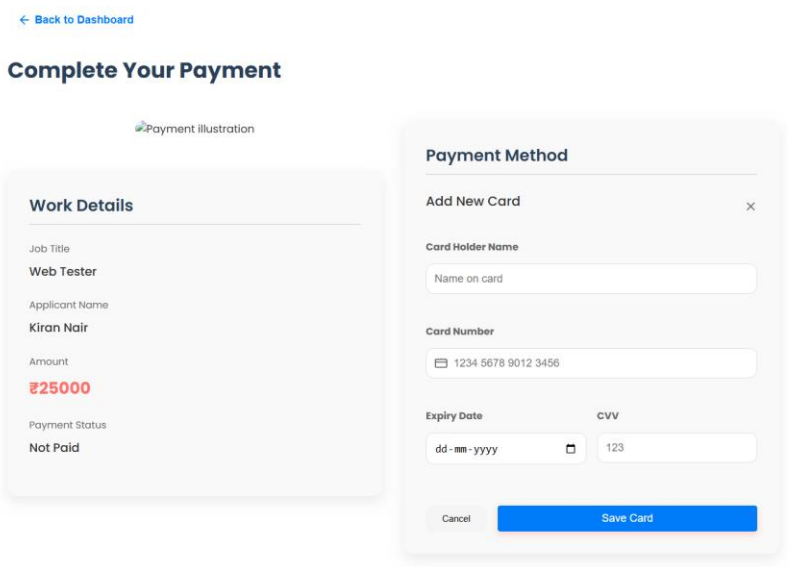
Describe the job responsibilities and requirements

Required Skills

List the skills required for this job

Card Management

Description: Add card details



← Back to Dashboard

Complete Your Payment

Payment illustration

Work Details

Job Title
Web Tester

Applicant Name
Kiran Nair

Amount
₹25000

Payment Status
Not Paid

Payment Method

Add New Card ×

Card Holder Name
Name on card

Card Number
1234 5678 9012 3456

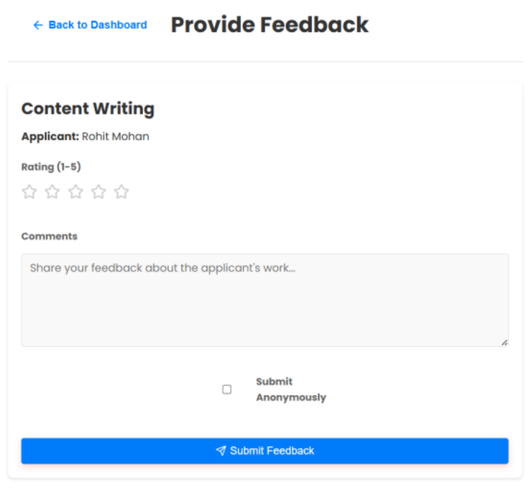
Expiry Date
dd-mm-yyyy

CVV
123

Cancel | Save Card

Feedback Management

Description: Add Feedback



← Back to Dashboard   Provide Feedback

Content Writing

Applicant: Rohit Mohan

Rating (1-5)
☆ ☆ ☆ ☆ ☆

Comments
Share your feedback about the applicant's work...

☐ Submit Anonymously

Submit Feedback

4.2 OUTPUT DESIGN

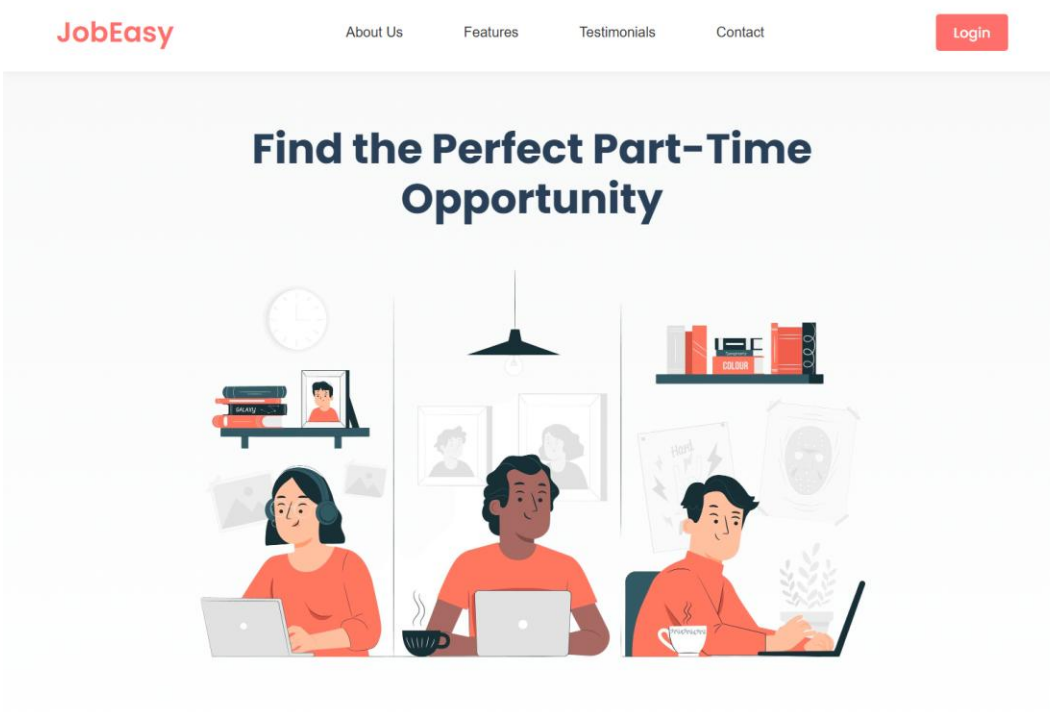
One of the important features of an information system for users is the output it produces. Output is the information delivered to users through the information system. Without quality output, the entire system appears to be unnecessary that users will avoid using it. Uses generally merit the system solely by its output. In order to create the most useful output possible. One works closely with the user through an interactive process, until the result is considered to be satisfactory.

Output design has been an ongoing activity almost from the beginning of the project. In the study phase, outputs were identified and described general in the project directive. A tentative output medium was then selected and sketches made for each output. In the feasibility analysis, a “best” new system was selected; its description identified the input and output media. In the design phase the system has included an evaluation and selection of specific equipment for the system.

Outputs from computer systems are required primarily to communicate the results of processing to the user. They are also used to provide a permanent copy of these results for later consultation.

Home Page

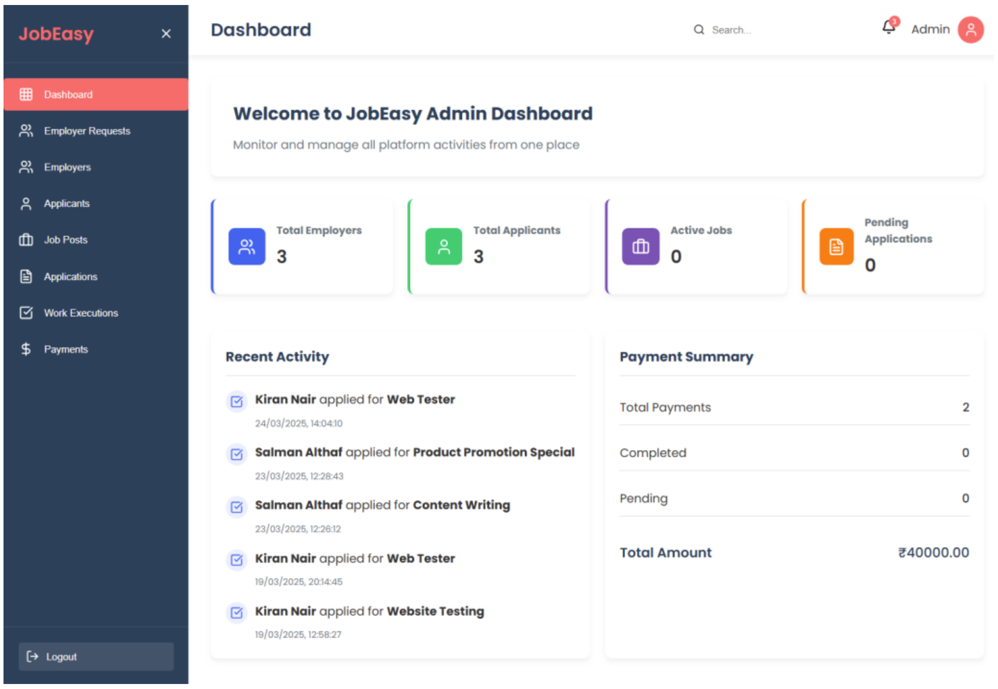
Description: This is the home page for all users



Find the Perfect Part-Time Opportunity

Admin Panel

Description: Admin panel where admin can manage website



Dashboard

Welcome to JobEasy Admin Dashboard

Monitor and manage all platform activities from one place

Total Employers 3

Total Applicants 3

Active Jobs 0

Pending Applications 0

Recent Activity

Kiran Nair applied for **Web Tester**
24/03/2025, 14:04:10

Salman Althaf applied for **Product Promotion Special**
23/03/2025, 12:28:43

Salman Althaf applied for **Content Writing**
23/03/2025, 12:26:12

Kiran Nair applied for **Web Tester**
19/03/2025, 20:14:45

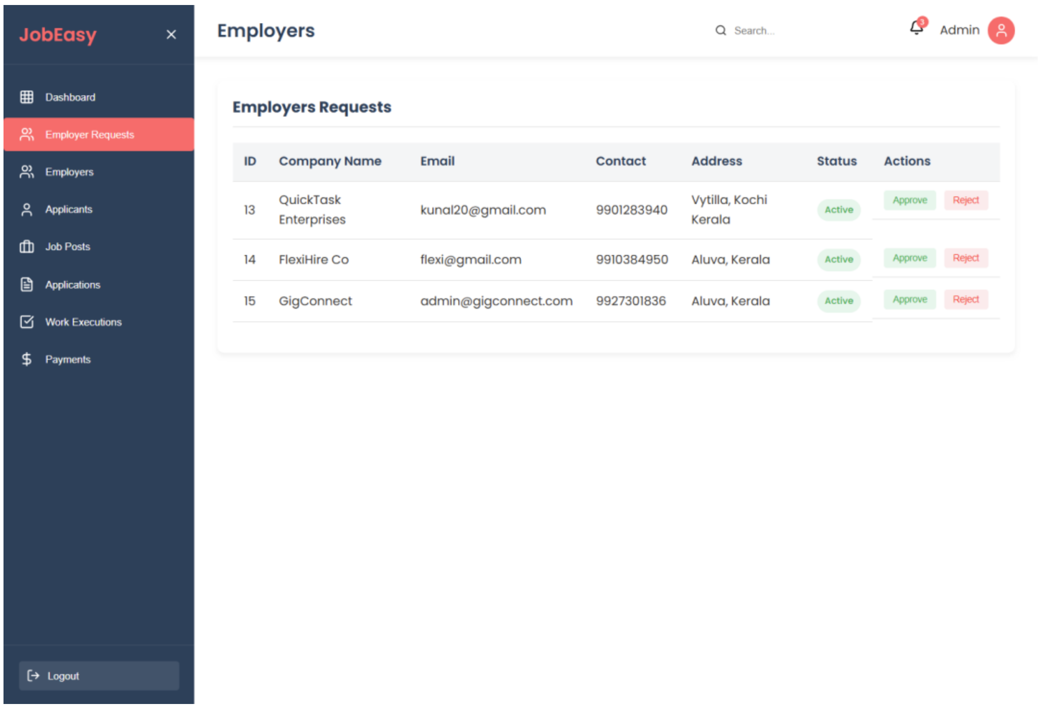
Kiran Nair applied for **Website Testing**
19/03/2025, 12:58:27

Payment Summary

Total Payments	2
Completed	0
Pending	0
Total Amount	₹40000.00

Admin User Management

Description: Admin panel where admin can manage users.



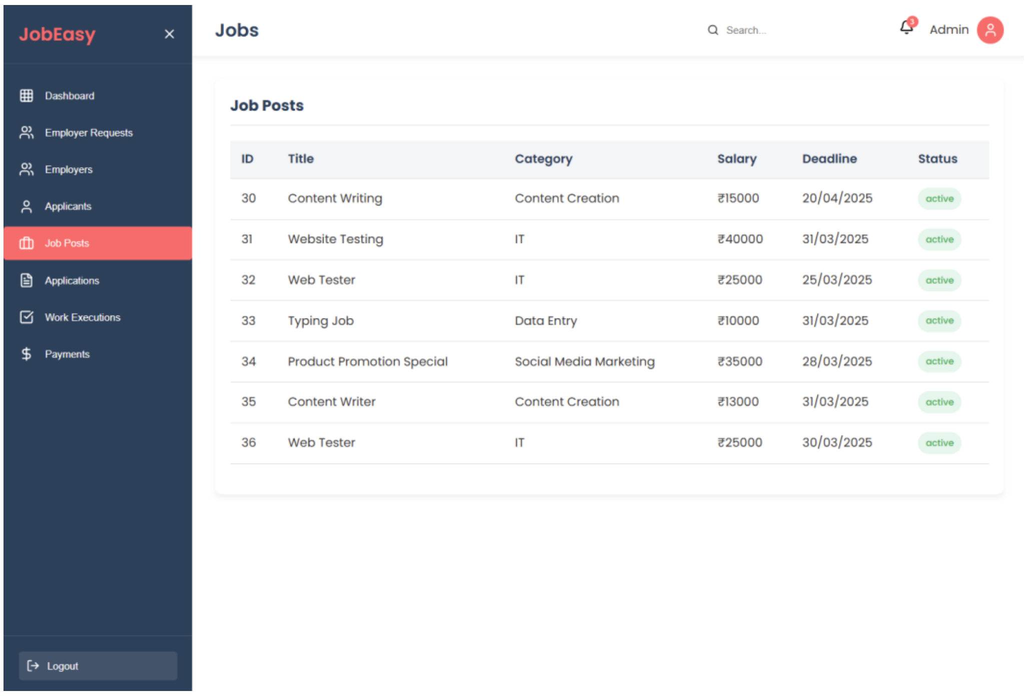
Employers

Employers Requests

ID	Company Name	Email	Contact	Address	Status	Actions
13	QuickTask Enterprises	kunal20@gmail.com	9901283940	Vytilla, Kochi Kerala	Active	Approve Reject
14	FlexiHire Co	flexi@gmail.com	9910384950	Aluva, Kerala	Active	Approve Reject
15	GigConnect	admin@gigconnect.com	9927301836	Aluva, Kerala	Active	Approve Reject

Admin Job Management

Description: Admin panel where admin can view Jobs.



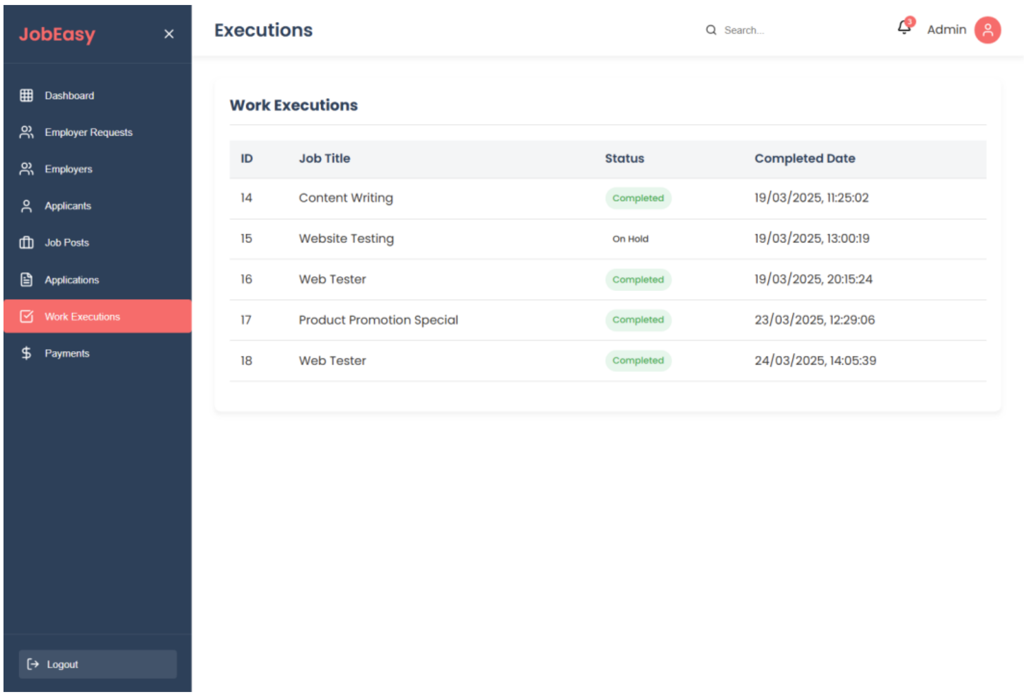
Jobs

Job Posts

ID	Title	Category	Salary	Deadline	Status
30	Content Writing	Content Creation	₹15000	20/04/2025	active
31	Website Testing	IT	₹40000	31/03/2025	active
32	Web Tester	IT	₹25000	25/03/2025	active
33	Typing Job	Data Entry	₹10000	31/03/2025	active
34	Product Promotion Special	Social Media Marketing	₹35000	28/03/2025	active
35	Content Writer	Content Creation	₹13000	31/03/2025	active
36	Web Tester	IT	₹25000	30/03/2025	active

Admin Work Execution Management

Description: Admin panel where admin can view Work Executions.

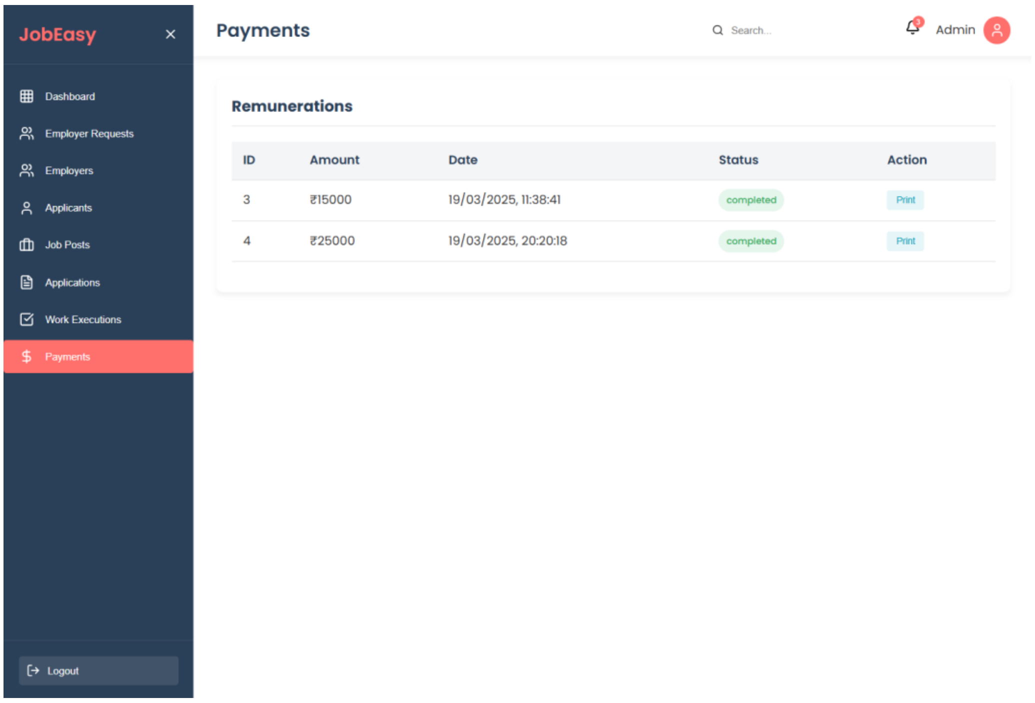


Work Executions

ID	Job Title	Status	Completed Date
14	Content Writing	Completed	19/03/2025, 11:25:02
15	Website Testing	On Hold	19/03/2025, 13:00:19
16	Web Tester	Completed	19/03/2025, 20:15:24
17	Product Promotion Special	Completed	23/03/2025, 12:29:06
18	Web Tester	Completed	24/03/2025, 14:05:39

Admin Payment Management

Description: Admin panel where admin can view Payment history.



Remunerations

ID	Amount	Date	Status	Action
3	₹15000	19/03/2025, 11:38:41	completed	Print
4	₹25000	19/03/2025, 20:20:18	completed	Print

4.3 DATABASE DESIGN

4.3.1 Normalization

Designing a database is a complex task and the normalization theory is a useful aid in this design process. The process of normalization is concerned with transformation of conceptual schema into computer representation form.

A bad database design may lead to certain undesirable situations such as,

• Repetition of information

• Inability to represent certain information

• Loss of information

To minimize these anomalies, normalization may be used. If the database is in a normalized form, the data can be restructured and can maintain it easily. This is important that the databases using that we are using may free from data redundancy and inconsistency. For this need we maintain the tables in a normalized manner.

First Normal Form

A relation is in first Normal Form (1NF), if and only if all its attributes are based on single domain. The objective of normalizing a table is in to remove its repeating groups and ensure that all entries of the resulting table have at most single value.

Second Normal Form

A table is said to be in second Normal Form (2NF), when it is in 1 NF and every attribute in the record is functionally dependent upon the whole key, and not just a part of the key.

Third Normal Form

A relation that is in First and Second Normal Form and in which no non-primary-key attribute is transitively dependent on the primary key, then it is in Third Normal Form (3NF).

TABLE DESIGN

Table No: 1
Table Name: tbl_customer
Table Description: Customer Details

FIELD	DATATYPE	CONSTRAINTS	DESCRIPTION
user_id	INT(5)	PRIMARY KEY	User id
username	VARCHAR(25)	NOT NULL	User Name
password	VARCHAR(16)	NOT NULL	Password of the user
user_type	VARCHAR(8)	NOT NULL	Type of the user
status	BIT(1)	NOT NULL	Active/ De-active Status of the user

Table No: 2
Table Name: tbl_Employers
Table Description: Employer Details

FIELD	DATATYPE	CONSTRAINTS	DESCRIPTION
emp_id	INT(5)	PRIMARY KEY	Employee ID
user_id	INT(5)	FOREIGN KEY(user)	Reference from the user table
company_name	VARCHAR(255)	NOT NULL	Company Name
email	VARCHAR(25)	NOT NULL	Email
contact_number	VARCHAR(15)	NOT NULL	Contact Number
address	VARCHAR(255)	NOT NULL	Address
reg_date	DATETIME	NOT NULL	Registration date

Table No: 3
Table Name: tbl_Applicants
Table Description: Applicant Details

FIELD	DATATYPE	CONSTRAINTS	DESCRIPTION
applicant_id	INT(5)	PRIMARY KEY	Applicant ID
user_id	INT(5)	FOREIGN KEY(user)	Reference from the user table
name	VARCHAR(15)	NOT NULL	Name of the applicant
email	VARCHAR(25)	NOT NULL	Email
contact_number	VARCHAR(10)	NOT NULL	Contact Number
skills	VARCHAR(15)	NOT NULL	Skills
experience	VARCHAR(15)	NOT NULL	Experience
preference	VARCHAR(15)	NOT NULL	Preferences
resume_file	IMAGEFIELD	NOT NULL	Path to the resume file
reg_date	DATETIME	NOT NULL	Registration Date

Table No: 4
Table Name: tbl_job_posts
Table Description: Job Post Details

FIELD NAME	DATATYPE	CONSTRAINT	DESCRIPTION
job_id	INT	PRIMARY KEY	Unique Id of Job
employer_id	INT	FOREIGN KEY (employer)	References from Employer table
job_title	VARCHAR(25)	NOT NULL	Job Title
job_description	TEXT	NOT NULL	Job Description
job_category	VARCHAR(25)	NOT NULL	Category of the job (e.g., IT, Marketing, etc.)
required_skills	TEXT	NOT NULL	Required Skills
salary	DECIMAL(10, 2)	NOT NULL	Remuneration
application_deadline	DATETIME	NOT NULL	Application Deadline
status	VARCHAR(10)	DEFAULT 'active'	Status (active or inactive)
posted_date	DATETIME	NOT NULL	Date when the job was posted

Table No: 5
Table Name: tbl_job_applications
Table Description: Job Application Details

FIELD NAME	DATATYPE	CONSTRAINT	DESCRIPTION
application_id	INT	PRIMARY KEY, AUTO_INCREMENT	Application ID
job_id	INT	FOREIGN KEY (job_posts)	Reference from the job_posts table
applicant_id	INT	FOREIGN KEY (applicants)	Reference from the applicants table
application_date	DATETIME	NOT NULL	Application Date
status	VARCHAR(12)	DEFAULT 'applied'	Status of Application ('applied', 'shortlisted', 'hired', 'rejected')
action_taken	VARCHAR(255)	NOT NULL	Description about the specific action taken on the application
action_date	DATETIME	NOT NULL	Date and Time of last action

Table No: 6
Table Name: tbl_work_execution
Table Description: Work Execution Details

FIELD NAME	DATATYPE	CONSTRAINT	DESCRIPTION
execution_id	INT	PRIMARY KEY	Execution ID
application_id	INT	FOREIGN KEY (job_application)	References from the job application
work_status	VARCHAR(12)	DEFAULT 'assigned'	Status of the work process. ('assigned', 'in_progress', 'completed', 'a pproved')
work_completed_date	DATETIME	Auto-generated	Date and time of work completion.
employer_review_date	DATETIME	Auto-generated	Date and time when the employer reviewed the work.
deliverables_file_path	VARCHAR(30)	NOT NULL	Path to the uploaded deliverables file.

Table No: 7
Table Name: tbl_cards
Table Description: Card Details

FIELD NAME	DATATYPE	CONSTRAINT	DESCRIPTION
id	INT(5)	PRIMARY KEY	Unique identifier for each card
customer_id	INT(5)	FOREIGN KEY	References
card_number	CHAR(16)	UNIQUE	Card Number
expiry_date	DATE	NOT NULL	Expiry date of the card
card_holder_name	VARCHAR(25)	NOT NULL	Name of Cardholder
created_at	DATETIME	Auto-generated	Timestamp when record was created
updated_at	DATETIME	Auto-generated	Timestamp when record was updated

Table No: 8
Table Name: tbl_remunerations
Table Description: Payment Details

FIELD NAME	DATA TYPE	CONSTRAINT	DESCRIPTION
payment_id	INT	PRIMARY KEY, AUTO_INCREMENT	Payment ID
execution_id	INT	FOREIGN KEY (execution_id)	Reference to work_execution table
payment_amount	DECIMAL(10, 2)	NOT NULL	Payment Amount
payment_date	DATETIME	Auto-generated	Date of Payment
payment_status	VARCHAR(10)	DEFAULT 'pending'	Status of Payment ('pending', 'completed')

Table No: 9
Table Name: tbl_feedback
Table Description: Feedback Details

FIELD NAME	DATA TYPE	CONSTRAINT	DESCRIPTION
feedback_id	INT	PRIMARY KEY, AUTO_INCREMENT	Feedback ID
application_id	INT	FOREIGN KEY (job_application)	Reference from job_application table
feedback_by	VARCHAR(10)	NOT NULL	User type(employer/ applicant)
feedback_text	TEXT	NOT NULL	Content of the feedback.
rating	INT	NOT NULL	Rating associated with the feedback
feedback_date	DATETIME	Auto-generated	Timestamp of when the feedback was submitted.

TESTING

5.1 INTRODUCTION

Testing is the process of examining the software to compare the actual behaviour with that of the excepted behaviour. The major goal of software testing is to demonstrate that faults are not present. In order to achieve this goal, the tester executes the program with the intent of finding errors. Though testing cannot show absence of errors but by not showing their presence it is considered that these are not present.

System testing is the first Stage of implementation, which is aimed at ensuring that the system works accurately and efficiently before live operations commences. Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct and the goal will be successfully achieved. A series of testing are performed for the proposed system before the proposed system is ready for user acceptance testing.

Levels of Testing

5.1.1 Unit Testing

5.1.2 Integration Testing

5.1.3 Output Testing

5.1.4 Validation Testing

5.1.1 Unit Testing

In this each module is tested individually before integrating it to the final system. Unit test focuses verification in the smallest unit of software design in each module. This is also known as module testing as here each module is tested to check whether it is producing the desired output and to see if any error occurs.

5.1.2 Integration Testing

Integration testing (sometimes called integration and testing, abbreviated I&T) is the phase in software testing in which individual software modules are combined and tested as a group. It occurs after unit testing and before validation testing.

Integration testing takes as its input modules that have been unit tested, groups them in larger aggregates, applies tests defined in an integration test plan to those aggregates, and delivers as its output the integrated system ready for system testing. The purpose of integration testing is to verify functional, performance, and reliability requirements placed on major design items.

5.1.3 Output Testing

No system could be useful if it does not produce the required output in the specific format. Output testing is performed to ensure the correctness of the output and its format. The output generated or displayed by the system is tested asking the users about the format required by them.

5.1.4 Validation Testing

In software project management, software testing, and software engineering, validation is the process of checking that a software system meets specifications and that it fulfills its intended purpose. The errors which are uncovered during integration testing are connected during this phase.

5.2 TEST CASES

The test plan documents the strategy that will be used to verify and ensure that product or a system meets its design specification and other requirements. A test plan is usually prepared by or with significant input from Test Engineers.

Unit Testing

Form	Procedure	Expected Result	Actual Result	Status
Entry Form	Choose whether to Login or Register	Should display Login or Registration form based on user selection	Login/Registration form displayed	Pass
Login Form	Enter valid email and password	Should validate user credentials and redirect to user dashboard	Redirected to dashboard	Pass
Job Seeker Registration Form	Enter all mandatory fields (e.g., name, email, phone)	Should validate fields and display "Registration successful" message	Success message shown	Pass
Employer Registration Form	Enter all mandatory fields (e.g., company name, email)	Should validate fields and display "Employer registered successfully" message	Success message shown	Pass
Job Posting Form	Enter all mandatory fields	Should validate fields	Success message	Pass

	(e.g., job title, description)	and display "Job posted successfully" message	shown	
Job Application Form	Select a job and submit application	Should validate submission and display "Application submitted" message	Success message shown	Pass
Payment Form	Enter payment details (e.g., amount, method)	Should validate details and display "Payment successful" message	Success message shown	Pass
Profile Update Form	Edit profile details (e.g., skills, bio)	Should validate updates and display "Profile updated successfully" message	Success message shown	Pass
Feedback Form	Enter feedback details (e.g., rating, comments)	Should validate fields and display "Feedback submitted successfully" message	Success message shown	Pass

Job Report Generation	Click "Generate Report" for job listings	Should generate and display a report of job postings or applications	Report displayed	Pass

Integration Testing

Form/Module	Expected Result	Actual Result	Status
Login and Dashboard	Successful login redirects to appropriate user dashboard	Dashboard displayed	Pass
Job Seeker Registration & Profile	Registration creates a profile accessible in dashboard	Profile accessible	Pass
Employer Registration & Job Posting	Employer can post a job after registration	Job posted successfully	Pass
Job Posting & Application	Job posted by employer appears in seeker’s job list; application updates status	Job listed, status updated	Pass

Payment & Job Posting	Payment completion enables job posting	Job posted after payment	Pass
Profile Update & Application	Updated seeker profile reflects in applications	Profile updates reflected	Pass
Feedback & Dashboard	Submitted feedback appears in employer/seeker dashboard	Feedback visible	Pass
Job Report & Dashboard	Generated report is accessible from dashboard	Report accessible	Pass

Validation Testing

Form/Scenario	Expected Result	Actual Result	Status
Create Job Seeker Account	Check mandatory fields; if valid, save record and confirm; else display error	Record saved or error shown	Pass
Create Employer Account	Check mandatory fields; if valid, save record and confirm; else display error	Record saved or error shown	Pass
Edit Job Posting	Edit job details; if valid, update record; else show error	Record updated or error shown	Pass

Apply to Job	Validate application; if valid, submit; else show error	Application submitted or error	Pass
Delete Job Posting	Delete selected job; if invalid ID, show error; else confirm	Job deleted or error shown	Pass

IMPLEMENTATION

6.1 INTRODUCTION

Implementation is the stage in the project where theoretical design is turned into a working system and is giving confidence on the new system for the users which will work efficiently and effectively. It involves careful planning, investigation of the current system and its constraints on implementations, design of methods to achieve the changeover, an evaluation, of change over methods. Apart from planning major tasks of preparing the implementation are education and training of users. The major complex system being implemented the more evolved will be the system analysis and the design effort required just for implementation. An implementation coordination committee based on policies of individual organisation has been appointed. The implementation process begins with preparing plan for implementation of the system. According to this plan the activities are to be carried out discussions made regarding the equipment and resources and the additional equipment has to be acquired to implement the new system.

Implementation is the final and important phase. The most critical stage in achieving a successful new system and in giving the users confidence that the new system will work and be effective. The system can be implemented only after thorough testing is done and if found to working according to the specification.

6.2 INSTALLATION PROCEDURE

Installation of software refers to the final installation of the package in the real environment, to the satisfaction of the intended users and the successful operation of the system. In many organizations, those who commission the software development project will not be the one to operate them. In the initial stage, the person who is not sure that the software will make the jobs easier will doubt about the software.

Implementation is the stage of the project where the theoretical design is turned into a working system. At this stage, the main work load, the greatest upheaval and the major impact on the existing system shifts to the user department. If the implementation is not carefully planned and controlled, it can cause confusion.

Implementation includes all those activities that take place to convert from the old system to new one. Proper implementation is essential to provide a reliable system to meet the organizational requirements. Successful implementation may guarantee improvement in the organization using the new system, but improper installation will prevent it. The process of putting the developed system in to actual use is called system implementation.

6.3 IMPLEMENTATION PLAN

The Implementation Plan describes how the information system will be deployed, installed and transitioned into an operational system. The plan contains an overview of the system, a brief description of the major tasks involved in the implementation, the overall resources needed to support the implementation effort, and any site-specific implementation requirements. The plan is developed during the Design Phase and is updated during the Development Phase the final version is provided in the Integration and Test Phase and is used for guidance during the Implementation Phase.

CONCLUSION

7.1 FUTURE ENHANCEMENT

I have made every effort to design and develop the **Part-Time Work Portal** effectively, ensuring that all critical aspects are well-addressed. However, with the rapid advancements in technology and changing user demands, the system may require future improvements to maintain its relevance and efficiency.

Although the current system provides a solid foundation for improving the job search and recruitment process, there is always room for enhancements. Provisions have been made within the system to accommodate future modifications and updates without disrupting its core functionality.

The system is designed to be interactive and user-friendly, allowing for flexibility in future expansions. Some potential enhancements include:

- Enhanced Security Measures – Implement advanced authentication mechanisms and encryption techniques to protect user data.
- Performance Optimization – Improve system efficiency by optimizing database queries, caching frequently accessed data, and enhancing server response times.

These enhancements will ensure that the Part-Time Work Portal remains efficient, secure, and adaptable to future technological advancements.

BIBLIOGRAPHY

BIBLIOGRAPHY

Websites References

Following websites are referred to create this project reports.

- www.stackoverflow.com
- www.mysqltutorial.com
- www.geeksforgeeks.org
- www.w3schools.com
- www. getbootstrap.com

APPENDICES

APPENDICES

APPENDIX A

Sample Source Code/Pseudo Code

Login Page Code

```
"use client"

import { useState, useEffect } from "react"
import { useNavigate } from "react-router-dom"
import { Link } from "react-router-dom"
import { ArrowRight, Mail, Lock, User, Briefcase, LogIn, Github, Twitter }
from "lucide-react"
import "./Login.css"

const Login = () => {
  const [username, setUsername] = useState("")
  const [password, setPassword] = useState("")
  const [errorMessage, setErrorMessage] = useState("")
  const [activeTab, setActiveTab] = useState("login")
  const [isLoading, setIsLoading] = useState(false)
  const navigate = useNavigate()

  // Animation effect when component mounts
  useEffect(() => {
    document.querySelector(".auth-card").classList.add("fade-in")
  }, [])

  const handleTabChange = (tab) => {
    setActiveTab(tab)
    // Reset form values and errors when switching tabs
    setUsername("")
    setPassword("")
    setErrorMessage("")
  }

  const handleLogin = async (e) => {
    e.preventDefault()
    setIsLoading(true)

    try {
      const response = await fetch("http://127.0.0.1:8000/api/login", {
        method: "POST",
        headers: { "Content-Type": "application/json" },
        body: JSON.stringify({ username, password }),
      })

      const data = await response.json()

      if (response.ok) {
        localStorage.setItem("token", data.token)
        localStorage.setItem(
          "user",
          JSON.stringify({
            user_id: data.user_id,
            username: data.username,
            user_type: data.user_type,
          }),
        )

        // Redirect based on user type
        if (data.user_type === "employer") {
          navigate("/employer-dashboard")
        } else if (data.user_type === "applicant") {
          navigate("/applicant-dashboard")
        } else if (data.user_type === "admin") {
          navigate("/admin-dashboard")
        } else {
          navigate("/")
        }
      } else {
        setErrorMessage(data.detail || "Invalid login credentials")
      }
    } catch (error) {
      setErrorMessage("Something went wrong. Please try again later.")
    } finally {
      setIsLoading(false)
    }
  }


  return (
    <div className="login-page">
      <div className="background-shapes">
        <div className="shape shape-1"></div>
        <div className="shape shape-2"></div>
        <div className="shape shape-3"></div>
        <div className="shape shape-4"></div>
      </div>

      <header className="home-header">
        <Link to="/" className="logo">
          <h1 style={{ color: "#FF6B6B" }}>JobEasy</h1>
        </Link>
        <nav className="nav-menu">
          <ul>
            <li>
              <button className="nav-tab"><a href="#about">About
Us</a></button>
            </li>
            <li>
              <button className="nav-tab"><a
href="#features">Features</a></button>
            </li>
            <li>
              <button className="nav-tab"><a
href="#testimonials">Testimonials</a></button>
            </li>
            <li>
              <button className="nav-tab"><a
href="#contact">Contact</a></button>
            </li>
          </ul>
        </nav>
        <div className="login-container">
          <Link to="/login" className="login-button">
            Login
          </Link>
        </div>
      </header>

      <div className="login-content">
        <div className="auth-card">
          <div className="auth-tabs">
            <button
              className={`auth-tab ${activeTab === "login" ? "active" : ""}`}
              onClick={() => handleTabChange("login")}
            >
              <LogIn size={18} />
              <span>Login</span>
            </button>
            <button
              className={`auth-tab ${activeTab === "signup" ? "active" : ""}`}
              onClick={() => handleTabChange("signup")}
            >
```

```
            <User size={18} />
            <span>Sign Up</span>
          </button>
        </div>

        <div className={`auth-content ${activeTab === "login" ? "show" :
""}`}>
          {activeTab === "login" ? (
            <>
              <h2 className="auth-title">Welcome Back</h2>
              <p className="auth-subtitle">Sign in to access your account</p>

              <form onSubmit={handleLogin} className="login-form">
                <div className="form-group">
                  <label htmlFor="username">

                    <span>Username</span>
                  </label>
                  <input
                    type="text"
                    id="username"
                    name="username"
                    value={username}
                    onChange={(e) => setUsername(e.target.value)}
                    placeholder="Enter your username"
                    required
                    className="form-input"
                  />
                </div>

                <div className="form-group">
                  <label htmlFor="password">

                    <span>Password</span>
                  </label>
                  <input
                    type="password"
                    id="password"
                    name="password"
                    value={password}
                    onChange={(e) => setPassword(e.target.value)}
                    placeholder="Enter your password"
                    required
                    className="form-input"
                  />
                </div>

                <div className="form-options">
                  <div className="remember-me">
                    <input type="checkbox" id="remember" />
                    <label htmlFor="remember">Remember me </label>
                  </div>
                  <a href="#" className="forgot-password">
                    Forgot Password?
                  </a>
                </div>

                {errorMessage && <p className="error-
message">{errorMessage}</p>}

                <button type="submit" className={`submit-button ${isLoading ?
"loading" : ""}`} disabled={isLoading}>
                  {isLoading ? "Signing in..." : "Sign In"}
                  {!isLoading && <ArrowRight size={18} />}
                </button>
              </form>

            </>
```

```
          ) : (
            <>
              <h2 className="auth-title">Create Account</h2>
              <p className="auth-subtitle">Choose your account type</p>

              <div className="signup-options">
                <button className="signup-option applicant" onClick={() =>
navigate("/job-seeker-signup")}>
                  <User size={24} />
                  <div>
                    <h3>Applicant</h3>
                    <p>Find jobs and opportunities</p>
                  </div>
                  <ArrowRight size={18} />
                </button>

                <button className="signup-option employer" onClick={() =>
navigate("/employer-signup")}>
                  <Briefcase size={24} />
                  <div>
                    <h3>Employer</h3>
                    <p>Post jobs and hire talent</p>
                  </div>
                  <ArrowRight size={18} />
                </button>
              </div>

              <div className="signup-note">
                <p>
                  By signing up, you agree to our <a href="#">Terms</a> and <a
href="#">Privacy Policy</a>
                </p>
              </div>
            </>
          )}
        </div>
      </div>
    </div>
  </div>
 )
}

export default Login
```

Applicant Sign Up Page code

```
"use client";

import React from "react";
import { useState } from "react";
import { useNavigate } from "react-router-dom";
import { Link } from "react-router-dom";
import "./signup.css";

const JobSeekerSignup = () => {
  const navigate = useNavigate();

  const [formData, setFormData] = useState({
    username: "",
    password: "",
    name: "",
    email: "",
    contact_number: "",
    skills: "",
    experience: "",
    preference: "",
  });
  const [resume, setResume] = useState(null);
  const [resumeError, setResumeError] = useState("");
  const [errorMessage, setErrorMessage] = useState("");

  const handleChange = (e) => {
    const { name, value } = e.target;
    setFormData((prevState) => ({
      ...prevState,
      [name]: value,
    }));
  };

  const handleFileChange = (e) => {
    const file = e.target.files[0];

    // Validate file type
    if (file) {
      const allowedTypes = ['application/pdf', 'application/msword',
'application/vnd.openxmlformats-
officedocument.wordprocessingml.document'];
      if (!allowedTypes.includes(file.type)) {
        setResumeError("Please upload PDF or Word document only");
        setResume(null);
        return;
      }

      // Validate file size (5MB max)
      if (file.size > 5 * 1024 * 1024) {
        setResumeError("File size should be less than 5MB");
        setResume(null);
        return;
      }

      setResumeError("");
      setResume(file);
    }
  };

  const handleSubmit = async (e) => {
    e.preventDefault();

    if (!formData || typeof formData !== "object") {
      setErrorMessage("Form data is invalid.");
      return;
    }

    if (!resume) {
      setResumeError("Please upload your resume");
      return;
    }

    try {
      // First, create the customer record
      const customerResponse = await
fetch("http://127.0.0.1:8000/api/customer/", {
        method: "POST",
        headers: {
          "Content-Type": "application/json",
        },
        body: JSON.stringify({
          username: formData.username,
          password: formData.password,
          user_type: "applicant",
          status: 1
        }),
      });

      if (!customerResponse.ok) {
        const errorData = await customerResponse.json();
        setErrorMessage(errorData.detail || "Customer creation failed");
        return;
      }

      const customerData = await customerResponse.json();

      // Create FormData for multipart/form-data submission (for file upload)
      const formDataObj = new FormData();
      formDataObj.append("user_id", customerData.user_id);
      formDataObj.append("name", formData.name);
      formDataObj.append("email", formData.email);
      formDataObj.append("contact_number", formData.contact_number);
      formDataObj.append("skills", formData.skills);
      formDataObj.append("experience", formData.experience);
      formDataObj.append("preference", formData.preference);
      formDataObj.append("resume", resume);

      // Then create the applicant record with the user_id and resume
      const applicantResponse = await
fetch("http://127.0.0.1:8000/api/applicant/", {
        method: "POST",
        body: formDataObj, // No Content-Type header, browser sets it with
boundary
      });

      if (applicantResponse.ok) {
        navigate("/login");
      } else {
        const errorData = await applicantResponse.json();
        setErrorMessage(errorData.detail || "Applicant creation failed");
      }
    } catch (error) {
      setErrorMessage("Something went wrong. Please try again later.");
    }
  };
  return (

      <div className="home-container">
        <header className="home-header">
          <Link to="/login" className="logo">
            <h1 style={{ color: "#FF6B6B" }}>JobEasy</h1>
          </Link>
        </header>
        <div className="auth-title">
          <h2>Job Seeker Signup</h2>
        </div>

        <form onSubmit={handleSubmit} className="login-form">
          <div>
            <label htmlFor="username">Username</label>
            <input type="text" id="username" name="username"
value={formData.username} onChange={handleChange} placeholder="Enter
username" className="form-input" required />
          </div>
          <div>
            <label htmlFor="password">Password</label>
            <input type="password" id="password" name="password"
value={formData.password} onChange={handleChange} placeholder="Enter
password" className="form-input" required />
          </div>
          <div>
            <label htmlFor="name">Full Name</label>
            <input type="text" id="name" name="name" value={formData.name}
onChange={handleChange} placeholder="Enter Full Name" className="form-
input" required />
          </div>
          <div>
            <label htmlFor="email">Email</label>
            <input type="email" id="email" name="email" value={formData.email}
onChange={handleChange} placeholder="Enter Email" className="form-
input" required />
          </div>
          <div>
            <label htmlFor="contact_number">Contact Number</label>
            <input type="tel" id="contact_number" name="contact_number"
value={formData.contact_number} onChange={handleChange}
placeholder="Enter Contact Number" className="form-input" required />
          </div>
          <div>
            <label htmlFor="skills">Skills</label>
            <textarea id="skills" name="skills" value={formData.skills}
onChange={handleChange} placeholder="Enter Skils" className="form-
input" required></textarea>
          </div>
          <div>
            <label htmlFor="experience">Experience</label>
            <textarea id="experience" name="experience"
value={formData.experience} onChange={handleChange} placeholder="Enter
Experiences" className="form-input" required></textarea>
          </div>
          <div>
            <label htmlFor="preference">Preferences</label>
            <textarea id="preference" name="preference"
value={formData.preference} onChange={handleChange} placeholder="Enter
Preferences" className="form-input" required></textarea>
          </div>
          <div class="file-upload">
            <label htmlFor="resume">Resume (PDF or Word document)</label>
            <input
              type="file"
              id="resume"
              name="resume"
```

```
accept=".pdf,.doc,.docx,application/pdf,application/msword,application/vnd.ope
nxmlformats-officedocument.wordprocessingml.document"
            onChange={handleFileChange}
            required
          />
          {resumeError && <p className="error">{resumeError}</p>}
        </div>
        <div>
        {errorMessage && <p className="error">{errorMessage}</p>}
        <button type="submit" className="cta-button primary">Sign
Up</button>
        </div>
      </form>
    </div>
  );
};

export default JobSeekerSignup;
```

APPENDIX B

Acronyms

ER- Entity relation

DFD-Data Flow Diagram

1NF-First Normal Form

2NF-Second Normal Form

3NF-Third Normal Form