UNIT STANDARD TITLE				**SAQA US ID**
Produce documentation for a computer program to agreed standards				115388
Date	**Learner Name**	**Learner Signature**	**Assessor Name**	**Assessor Signature**

DETAILED DESIGN DOCUMENT BY:

Full names:
Postal address:
TITLE:

Telephone number:
E-mail:
Date of submission:

Contents

1. Introduction

The sole purpose of this project is to promote internal mobility of the recruitment processes within the Company as well as enhance succession planning and improving employee development and engagement.

- Improve efficiency and effectiveness of internal recruitment processes.

- Facilitate better communication and engagement within the company.

- Centralize and automate job postings.

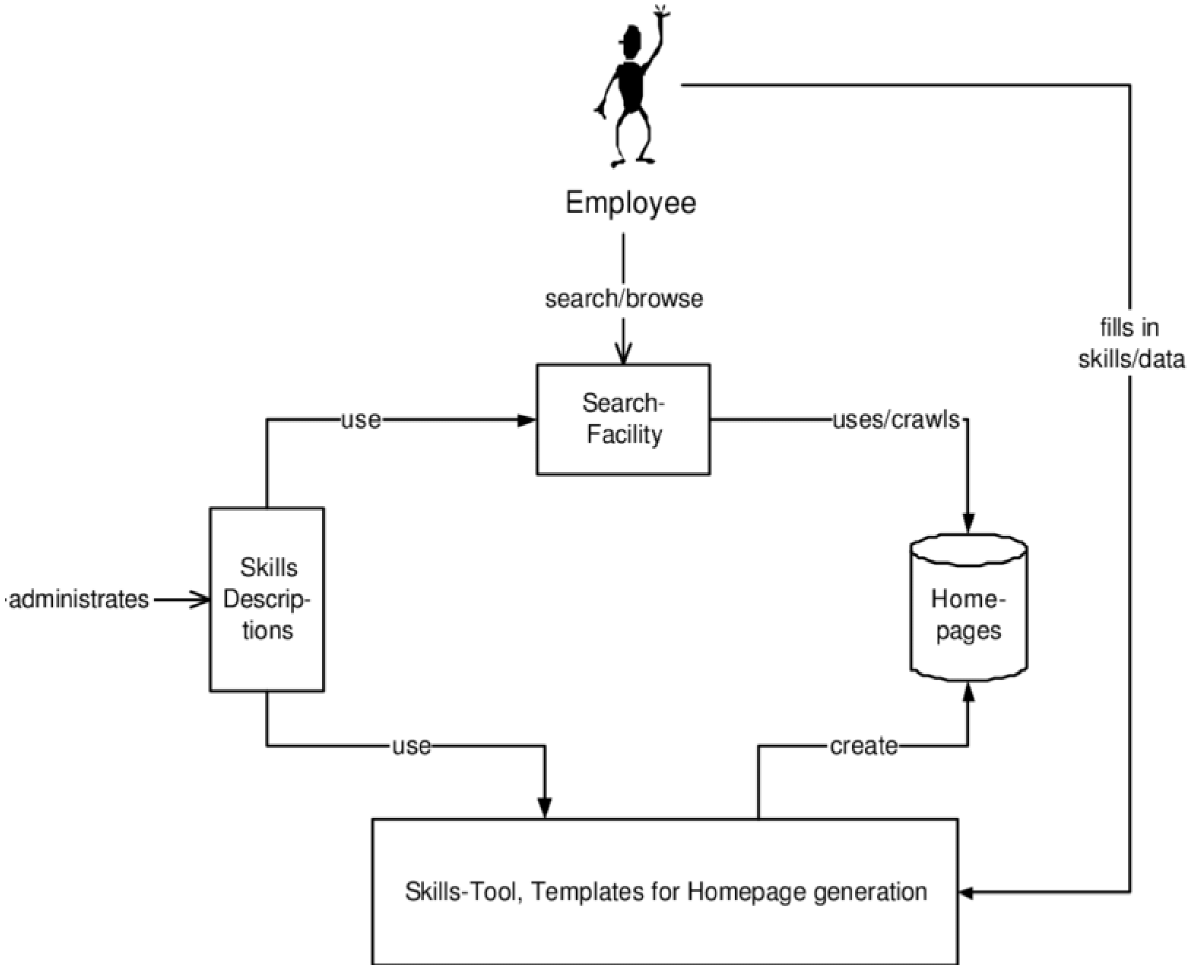
The proposed system aims to assist in the recruitment process, by creating a seamless and smooth platform in which companies can post job openings and in turn, job-seekers can flaunt their qualifications which would help assist in the unemployment rates.

The Related programs where applicable are Architecture design class diagram,
ERD

· The Related documents are research documents, research interview, cost benefits analysis and project plan.

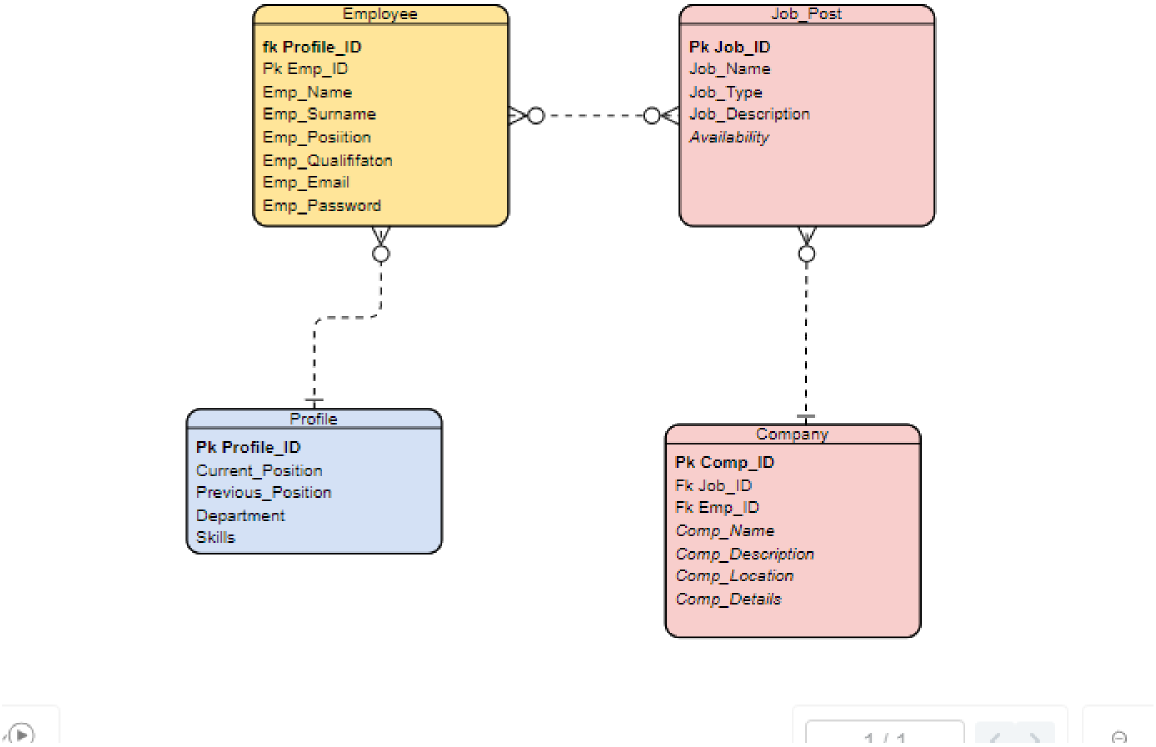
2. Architecture Design

The Employment System Architecture is designed to be scalable, adaptable, and user-centric, offering a streamlined and efficient process for user registration and job position matching within the company. By integrating advanced algorithms, communication tools, and security measures, the system aims to enhance the overall recruitment experience and contribute to the company's talent acquisition strategy. It provides information on the decomposition of the system into modules (classes), dependencies between modules, hierarchy and partitioning of the software.


Employee
search/browse
fills in skills/data
Search-Facility
use
uses/crawls
administrates
Skills Descrip-tions
Home-pages
use
create
Skills-Tool, Templates for Homepage generation

3. Database Design

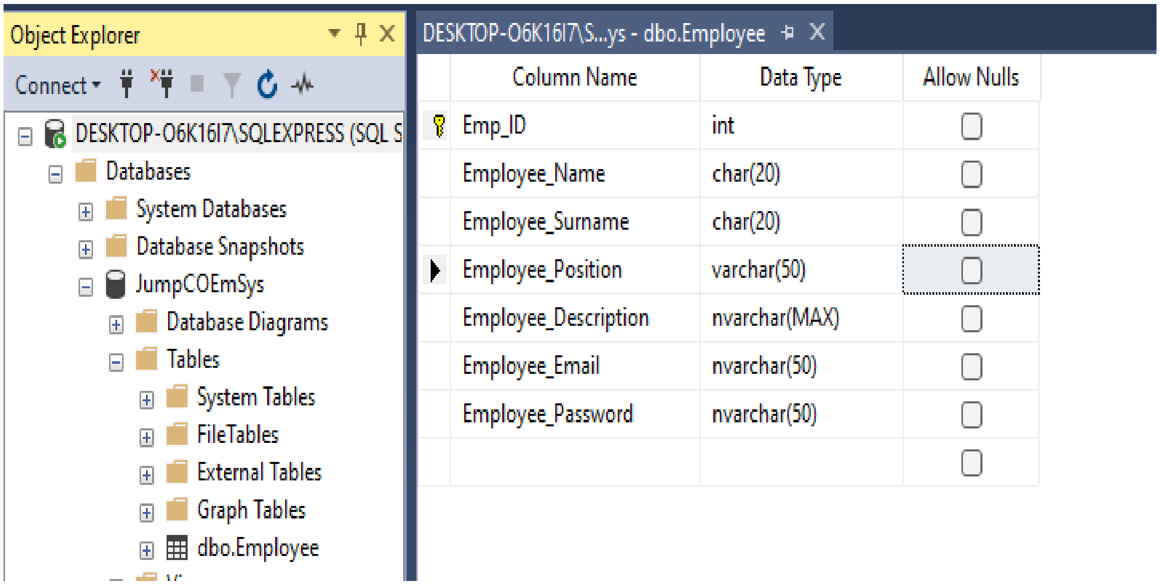
The database design specifies how the data of the software is going to be stored. Below is the JumpCo Employment System ERD, that specifies the structure of the database.



3.1 Tables data

3.1.1 Table Name: Employee Table

a) Table Structure



Column Name	Data Type	Allow Nulls
Emp_ID	int	□
Employee_Name	char(20)	□
Employee_Surname	char(20)	□
Employee_Position	varchar(50)	□
Employee_Description	nvarchar(MAX)	□
Employee_Email	nvarchar(50)	□
Employee_Password	nvarchar(50)	□
		□

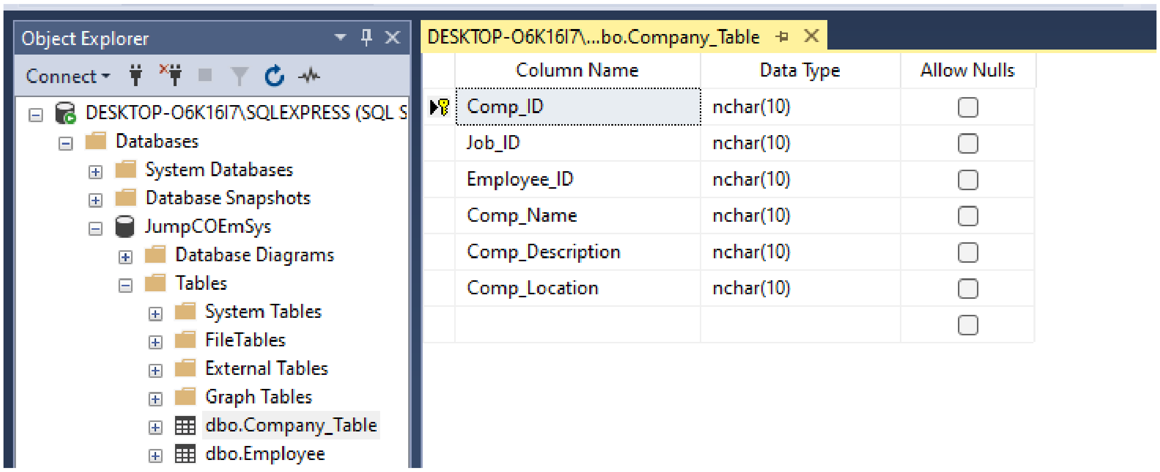
++

b) Primary and foreign keys

Reference	Control type	Data input	Data output
CR 1	Primary Key	Emp_ID	varchar
CR 2			

3.1.2 Table Name: Company

a) Table Structure



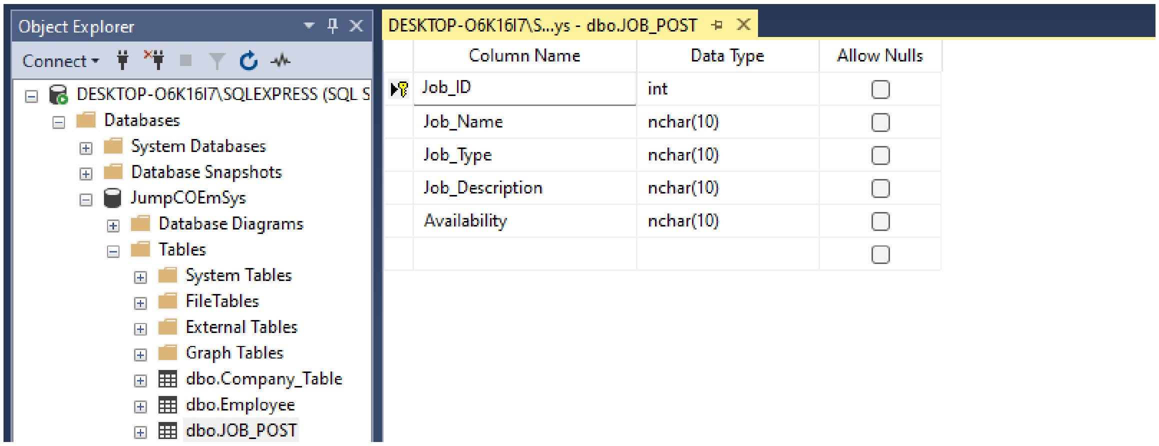
Column Name	Data Type	Allow Nulls
Comp_ID	nchar(10)	□
Job_ID	nchar(10)	□
Employee_ID	nchar(10)	□
Comp_Name	nchar(10)	□
Comp_Description	nchar(10)	□
Comp_Location	nchar(10)	□
		□

b) Primary and foreign keys

Reference	Control type	Data input	Data output
CR 1	Primary Key	Comp_ID	varchar
CR 2	Foreign Key	Job_ID	varchar

3.1.3 Table Name: Job_Post

a) Table Structure



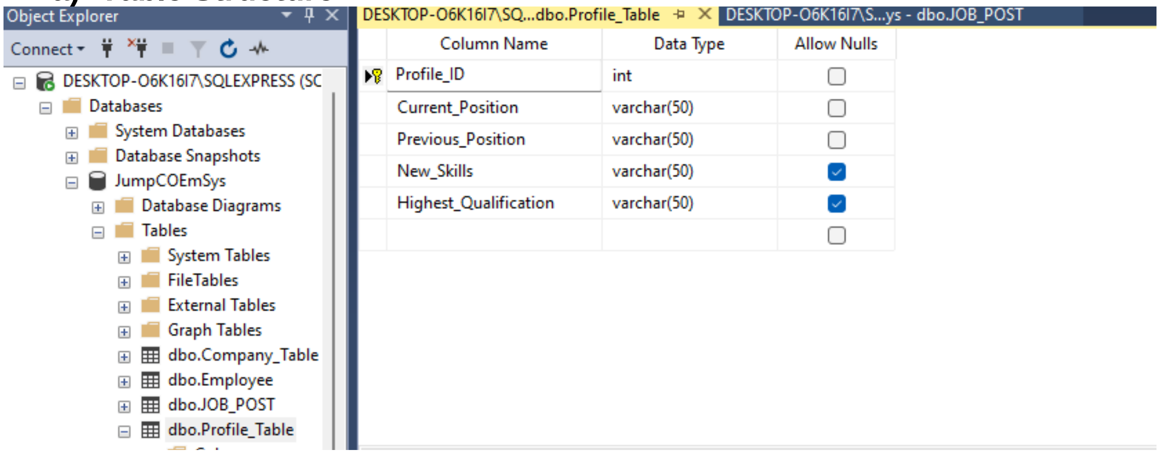
Column Name	Data Type	Allow Nulls
Job_ID	int	□
Job_Name	nchar(10)	□
Job_Type	nchar(10)	□
Job_Description	nchar(10)	□
Availability	nchar(10)	□
		□

b) Primary and foreign keys

Reference	Control type	Data input	Data output
CR 1	Primary Key	Job_ID	varchar
CR 2			

3.1.4 Table Name: Profile

a) Table Structure



Column Name	Data Type	Allow Nulls
Profile_ID	int	☐
Current_Position	varchar(50)	☐
Previous_Position	varchar(50)	☐
New_Skills	varchar(50)	☑
Highest_Qualification	varchar(50)	☑
		☐

b) Primary and foreign keys

Reference	Control type	Data input	Data output
CR 1	Primary Key	Profile_ID	varchar
CR 2			

3.3 Transactions implementation

Explain how you will implement the ACID (atomicity, consistency, isolation and durability) properties of transactions (programs that access databases.)

- ***Atomicity***

All the tractions in the system that will take place in the database will be managed by the system. Before anything takes place in the system the system must approve it to take place in the database, anything transaction that fails it won't enter the database

- ***Consistency***

I designed the system in a way that by default the database must be consistency and manageable. Any transaction that will take place in the system. Whether registration, deleting or editing any transaction it won't affect another transaction of any employee or the same traction

- ***Isolation***

The system has been designed in two half's, the employee site and the admin site, the employees are restricted form the database tractions and the admin has access to the database and the system as a hole.

- ***Durability***

The system is guaranteed that anything that is wrong or the system fails it will notify the user, whether the server is down, system cannot connect, the system will always notify the user if these anything wrong.

4. Graphical User Interface

.

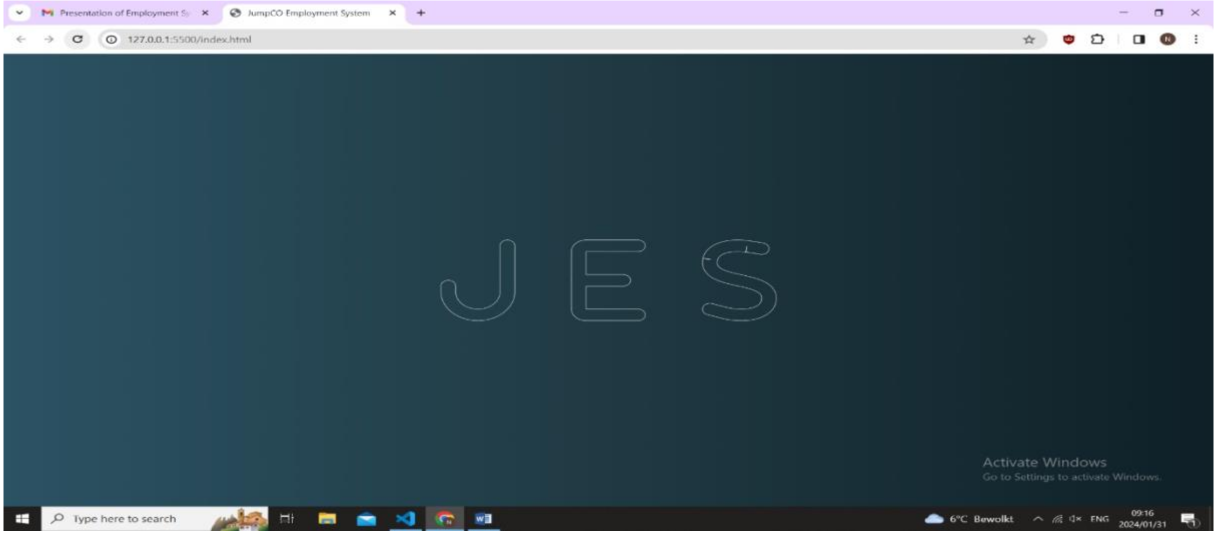
- Pictures of all the forms in the graphical user interface with a reference to the functional requirement it implements. We used html to present the graphical user interfaces.

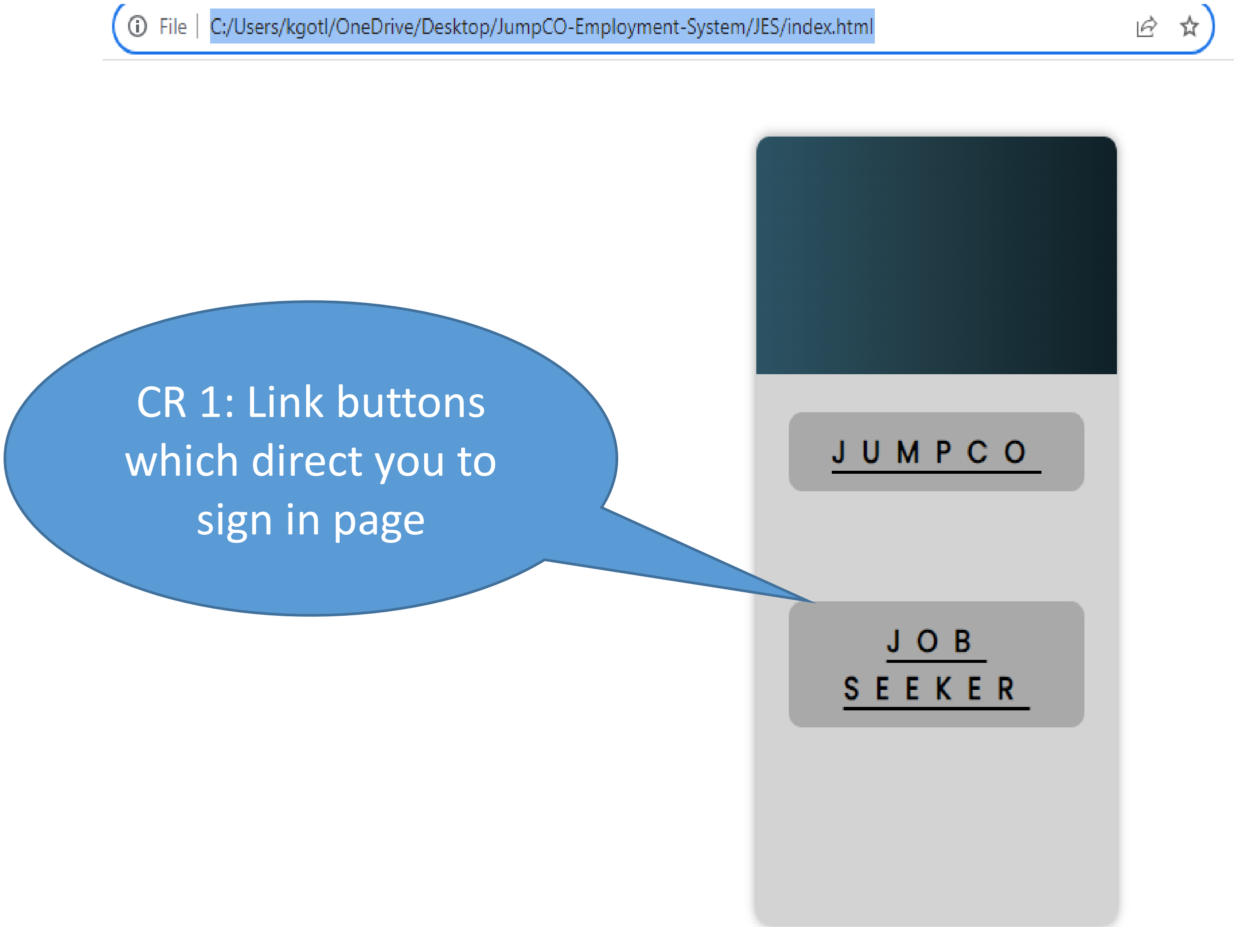
4.1 Graphical User Interface

a) JSP GUI

LandingPage.html

On this page the user has an option to click on the Job seeker link if they are registered as a job seeker else the Jumpco link if they are registered as an employer.



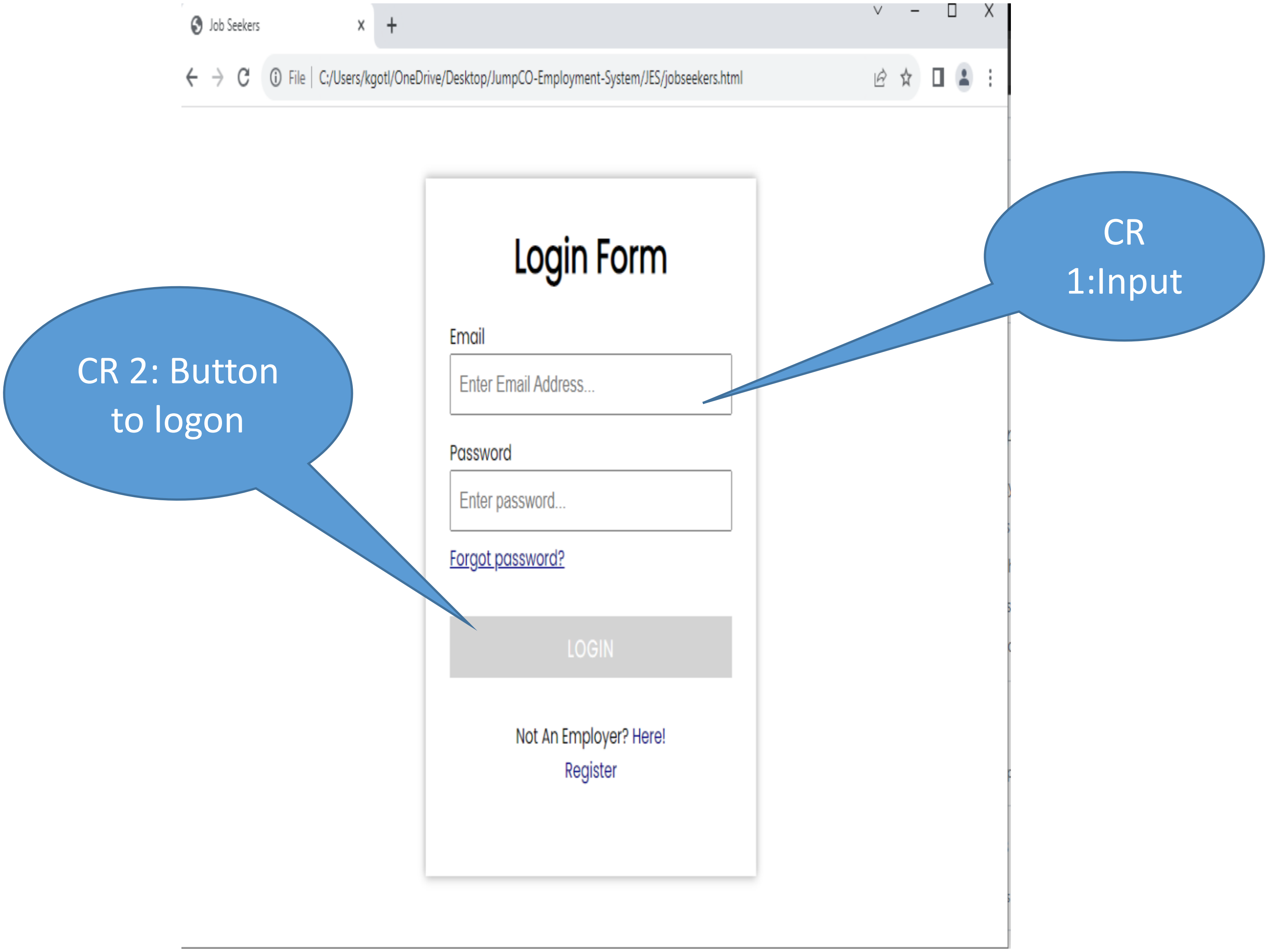


File | C:/Users/kgotl/OneDrive/Desktop/JumpCO-Employment-System/JES/index.html

CR 1: Link buttons which direct you to sign in page

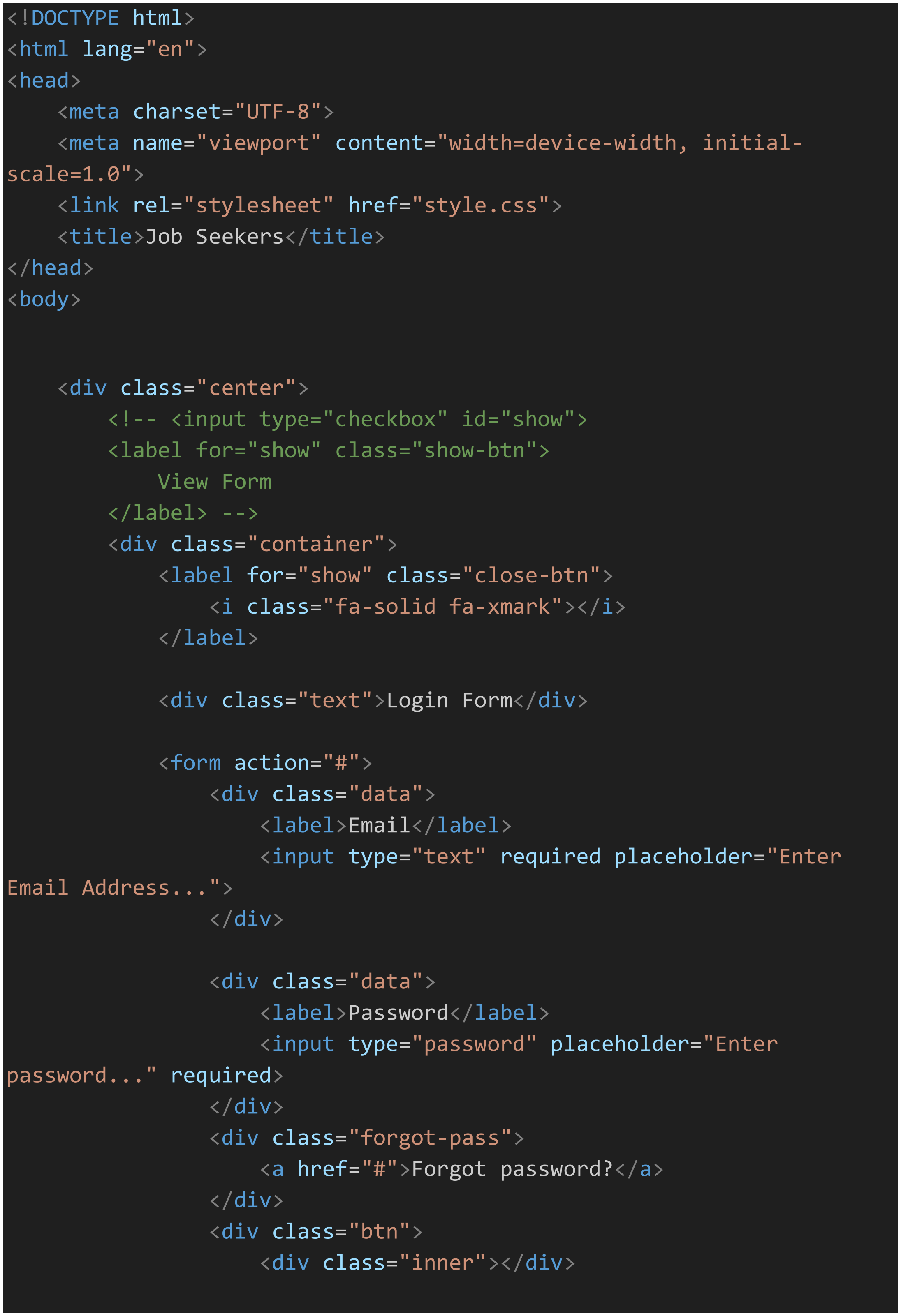
Login.html

The employee/employer will be required to fill in their login credentials on this login page e.g., enter required fields and click on the sign in button to enter.



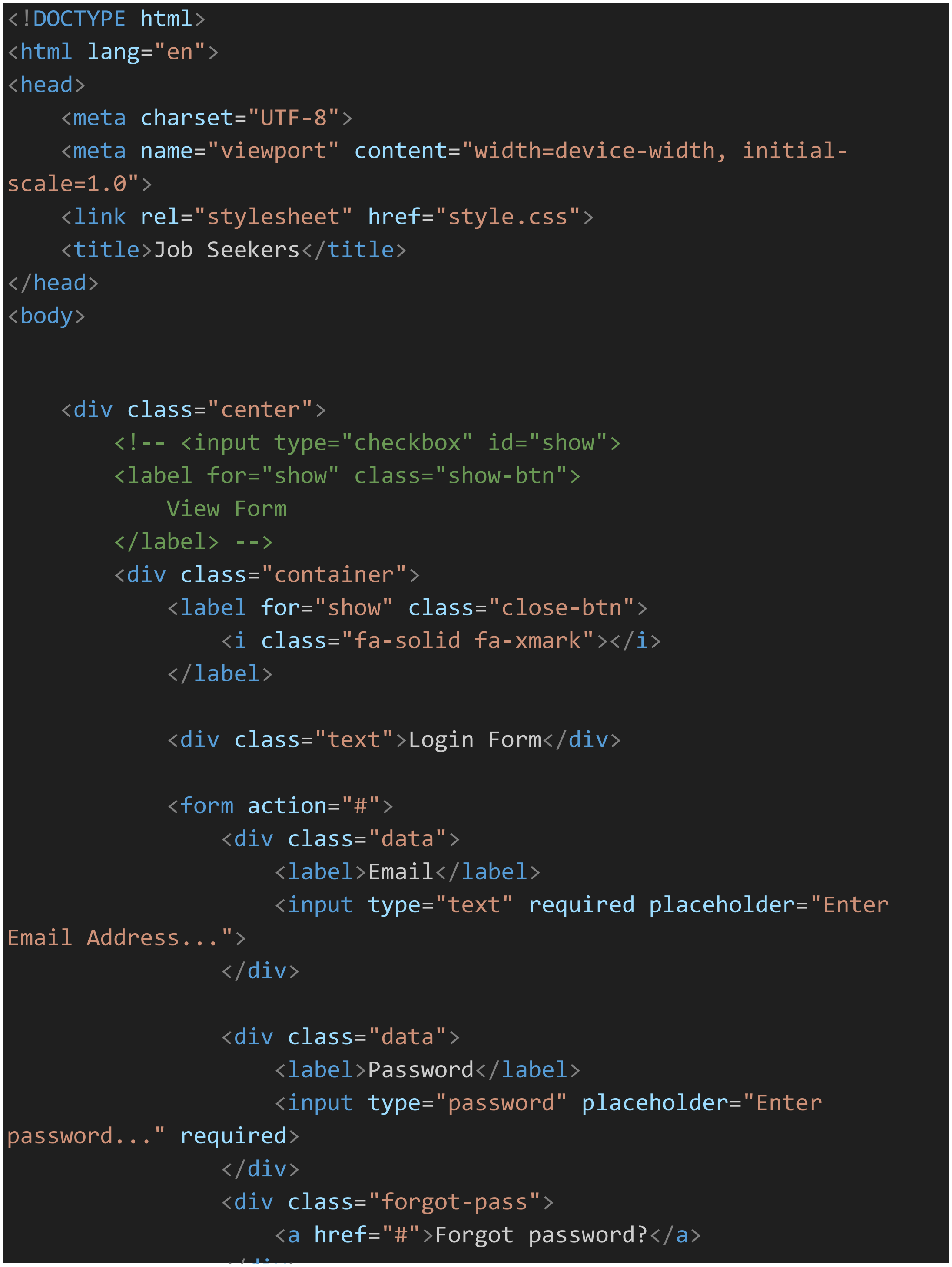
b) HTML script

LandingPage.html



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Job Seekers</title>
</head>
<body>


    <div class="center">
        <!-- <input type="checkbox" id="show">
        <label for="show" class="show-btn">
            View Form
        </label> -->
        <div class="container">
            <label for="show" class="close-btn">
                <i class="fa-solid fa-xmark"></i>
            </label>

            <div class="text">Login Form</div>

            <form action="#">
                <div class="data">
                    <label>Email</label>
                    <input type="text" required placeholder="Enter
Email Address...">
                </div>

                <div class="data">
                    <label>Password</label>
                    <input type="password" placeholder="Enter
password..." required>
                </div>
                <div class="forgot-pass">
                    <a href="#">Forgot password?</a>
                </div>
                <div class="btn">
                    <div class="inner"></div>
```

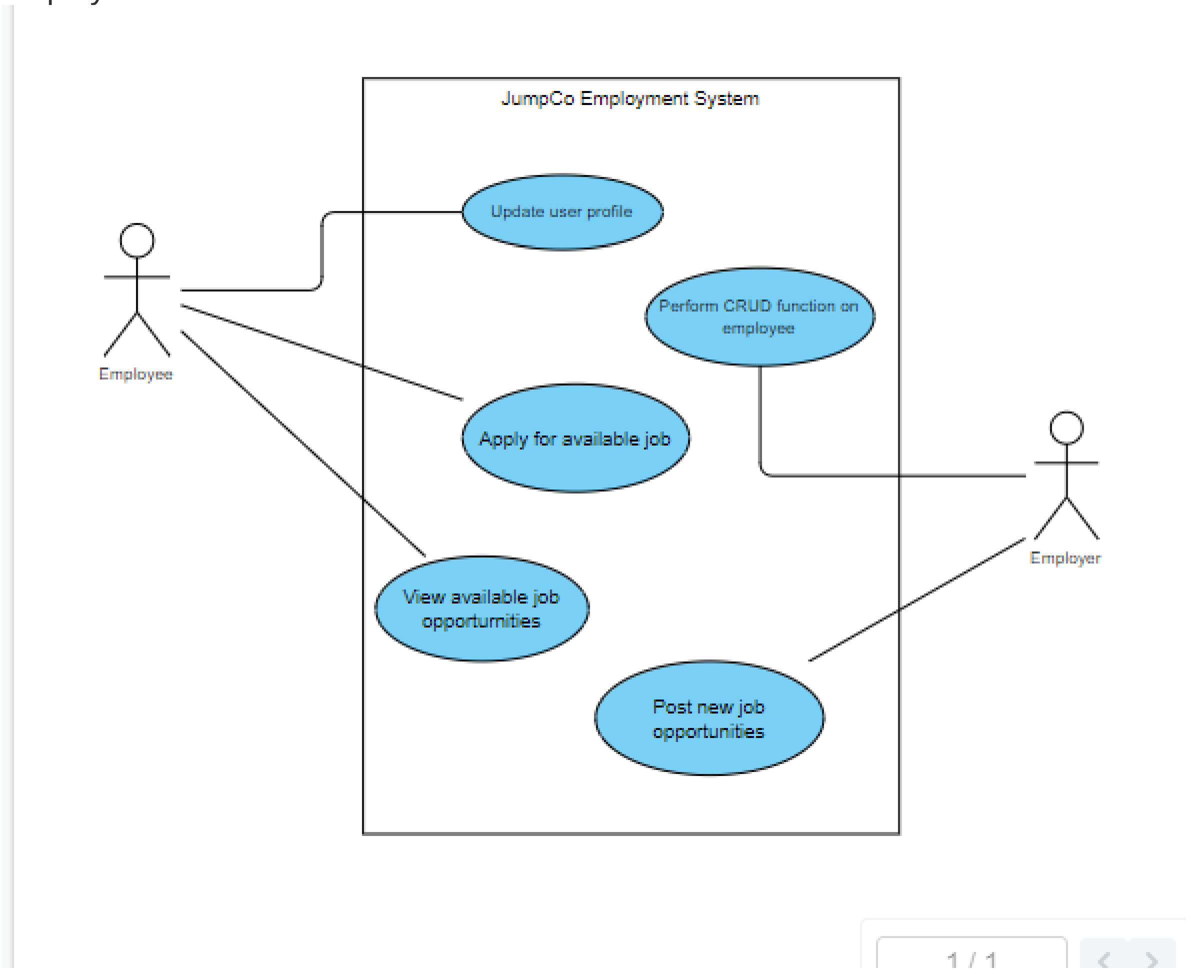
LoginPage.html



```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1.0">
    <link rel="stylesheet" href="style.css">
    <title>Job Seekers</title>
</head>
<body>


    <div class="center">
        <!-- <input type="checkbox" id="show">
        <label for="show" class="show-btn">
            View Form
        </label> -->
        <div class="container">
            <label for="show" class="close-btn">
                <i class="fa-solid fa-xmark"></i>
            </label>

            <div class="text">Login Form</div>

            <form action="#">
                <div class="data">
                    <label>Email</label>
                    <input type="text" required placeholder="Enter
Email Address...">
                </div>

                <div class="data">
                    <label>Password</label>
                    <input type="password" placeholder="Enter
password..." required>
                </div>
                <div class="forgot-pass">
                    <a href="#">Forgot password?</a>
```

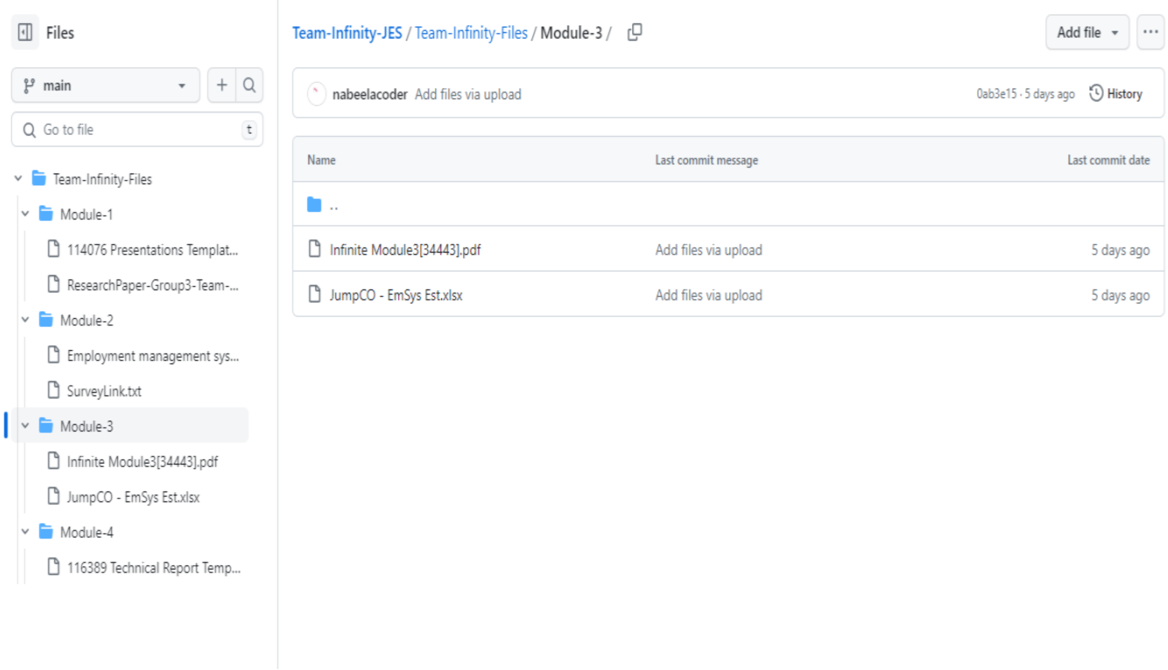
5. Use Case Diagram

The following use case diagram shows the functions between the employer and employee.


JumpCo Employment System
Update user profile
Perform CRUD function on employee
Apply for available job
View available job opportunities
Post new job opportunities
Employee
Employer
1 / 1

6. Online Help

Fig 1: Help Maker Document Designer



Files

main

Go to file

- Team-Infinity-Files
  - Module-1
    - 114076 Presentations Templat...
    - ResearchPaper-Group3-Team-...
  - Module-2
    - Employment management sys...
    - SurveyLink.txt
  - Module-3
    - Infinite Module3[34443].pdf
    - JumpCO - EmSys Est.xlsx
  - Module-4
    - 116389 Technical Report Temp...

Team-Infinity-JES / Team-Infinity-Files / Module-3 /

Add file

nabeelacoder Add files via upload 0ab3e15 · 5 days ago History

Name	Last commit message	Last commit date
..		
Infinite Module3[34443].pdf	Add files via upload	5 days ago
JumpCO - EmSys Est.xlsx	Add files via upload	5 days ago

7. References

Petri Ihantola, Arto Vihavainen, Alireza Ahad, Matthew Butler, Jürgen Börstler,

Stephen H. Edwards, Essi Isohanni, Ari Korhonen, Andrew Petersen, Kelly Rivers., 2015. Educational Data Mining and Learning Analytics in Programming: Literature Review and Case Studies. *Digital Library,* pp. 41-63.

Cooper, J. W., n.d. *Java Design Patterns.* s.l.:Addison Wesly Longman.

Nur Atikah Arbain, A. R. H. H. S. A., 2023. Implementation of Advanced SQL Using Java Server Pages as Frontend. 3(1).

8. Appendix

Petri Ihantola, Arto Vihavainen, Alireza Ahad, Matthew Butler, Jürgen Börstler, Stephen H. Edwards, Essi Isohanni, Ari Korhonen, Andrew Petersen, Kelly Rivers., 2015. Educational Data Mining and Learning Analytics in Programming: Literature Review and Case Studies. *Digital Library,* pp. 41-63.

Cooper, J. W., n.d. *Java Design Patterns.* s.l.:Addison Wesly Longman.

Nur Atikah Arbain, A. R. H. H. S. A., 2023. Implementation of Advanced SQL Using Java Server Pages as Frontend. 3(1).