COMSATS University Islamabad,
Abbottabad Campus

SOFTWARE DESIGN DESCRIPTION

(SDD DOCUMENT)

for

<Ai-Based Interviewer Facilitator Application>

Version 1.0

By

Hamdan Ali Tahir	**CIIT/FA20-BCS-008/ATD**
Danish Nasar	**CIIT/FA20-BCS-006/ATD**
Khubaib-ur-Rehman	**CIIT/FA20-BCS-141/ATD**

Supervisor

Mam Bushra Mushtaq

Bachelor of Science in Computer Science (2020-2023)

Table of Contents

Revision History

Name	Date	Reason for changes	Version

Application Evaluation History

Comments (by committee) ***include the ones given at scope time both in doc and presentation**	**Action Taken**

Supervised by

<Mam Bushra Mushtaq>

Signature_______________

Introduction

The system shall have a web-based user interface. It is an interviewer facilitator application that enhances the interview process for the candidate and interviewer both. Both interviewer and candidate will have their portal first of all they will register and after then after login candidate will upload his resume by using the form which will be given in his portal then at the interviewer's side he will create a session for an interview and add job description and to fetch resume of the candidate he will add his email to fetch his resume information then both of these things are analyzed by using gpt model and it will provide a pool of questions after first pool it will start analyzing the conversation between interview using transcription module and keep on updating the pool of question depending upon the conversation there will also an option of updating pool of question by using that interviewer can update the pool of questions and then when interview is finished a candidate evaluation report will be generated and interviewer can also generate reports of previous interviews by using there session id.

Design methodology and software process model

In an Object-Oriented approach, we can represent key entities such as candidates, interviewers, and system components as objects with associated attributes and behaviors. This facilitates a modular and organized system design. Encapsulation ensures that each module is self-contained, reducing dependencies and enhancing code maintainability. The use of inheritance allows for the creation of reusable code components, reducing redundancy and promoting a more efficient development process.

System overview

The system is designed to facilitate a seamless and intelligent interview process. Each module contributes to different aspects of the overall functionality:

Candidate Resume Analysis : Utilizes Natural Language Processing (NLP) to analyze and extract relevant information from candidate resumes.

Job Description Analysis : Applies text analysis techniques to understand and categorize job descriptions, ensuring alignment with candidate profiles.

Real-time Transcription : Enables real-time transcription of interview conversations, providing a textual record for analysis.

Conversational Analysis: Applies analysis and language processing to evaluate the tone and content of interview conversations.

Follow-up Questions: Incorporates an intelligent system for generating follow-up questions based on candidate responses.

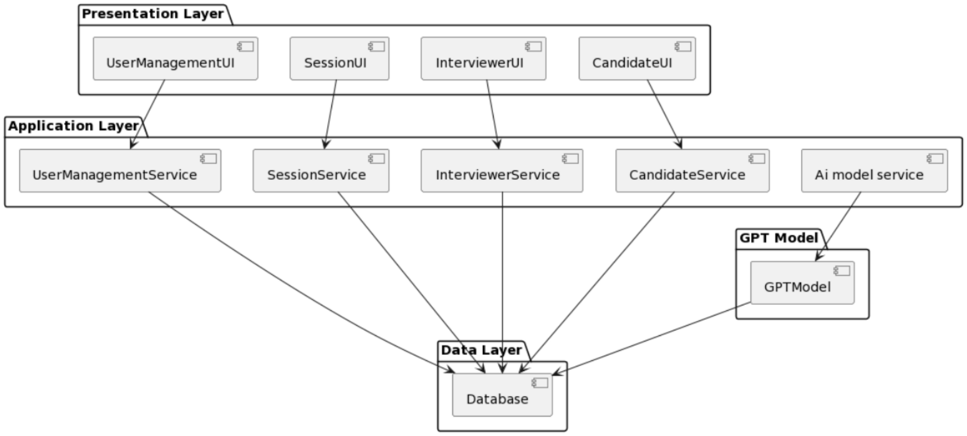
Job Compatibility Report: Generates a comprehensive report assessing the compatibility of a candidate with a given job role.

User and Candidate Management: Implements a Database Management System (DBMS) for effective user and candidate data storage, retrieval, and management.

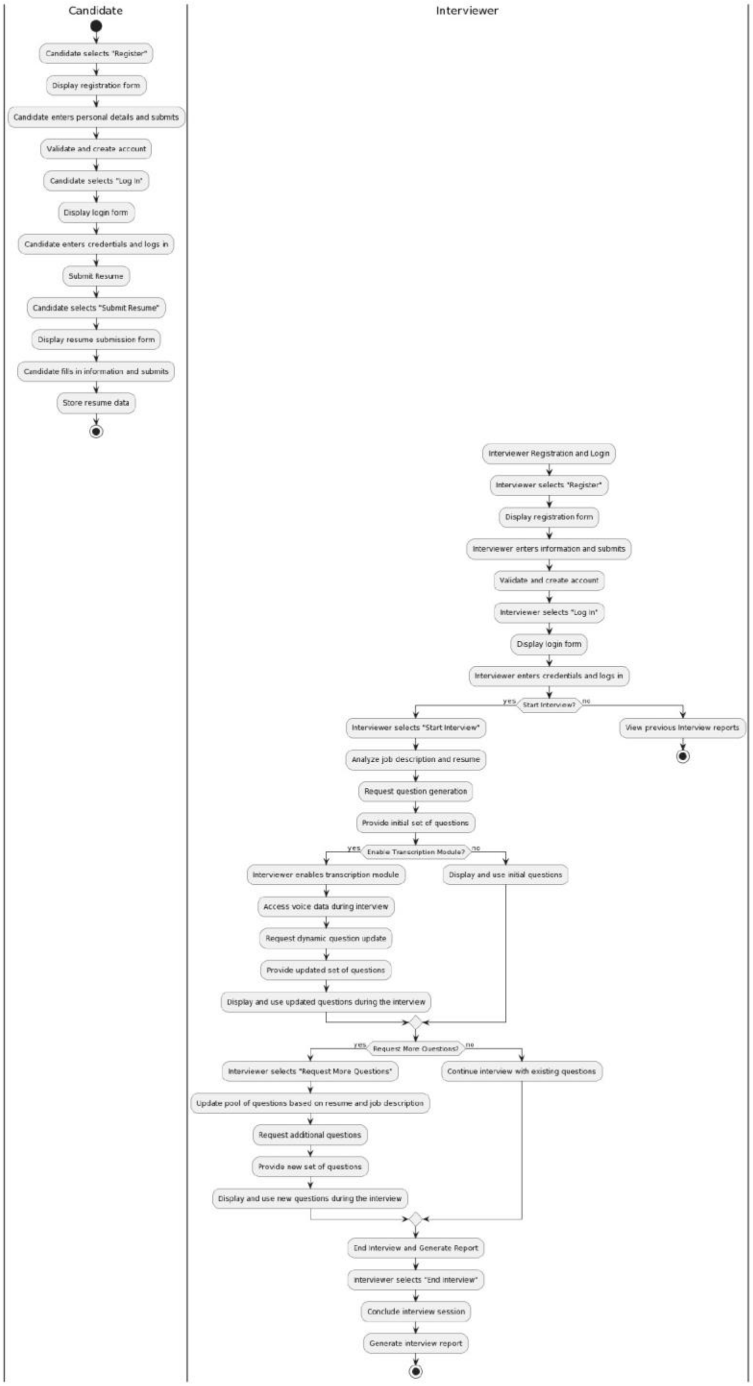
Data Security: Prioritizes the security of sensitive information, implementing encryption, access controls, and other security measures.

This system aims to revolutionize the traditional interview process by introducing intelligent automation and data-driven insights. The modular design ensures that each aspect is treated independently, allowing for flexibility and scalability in future developments.

Architectural Design

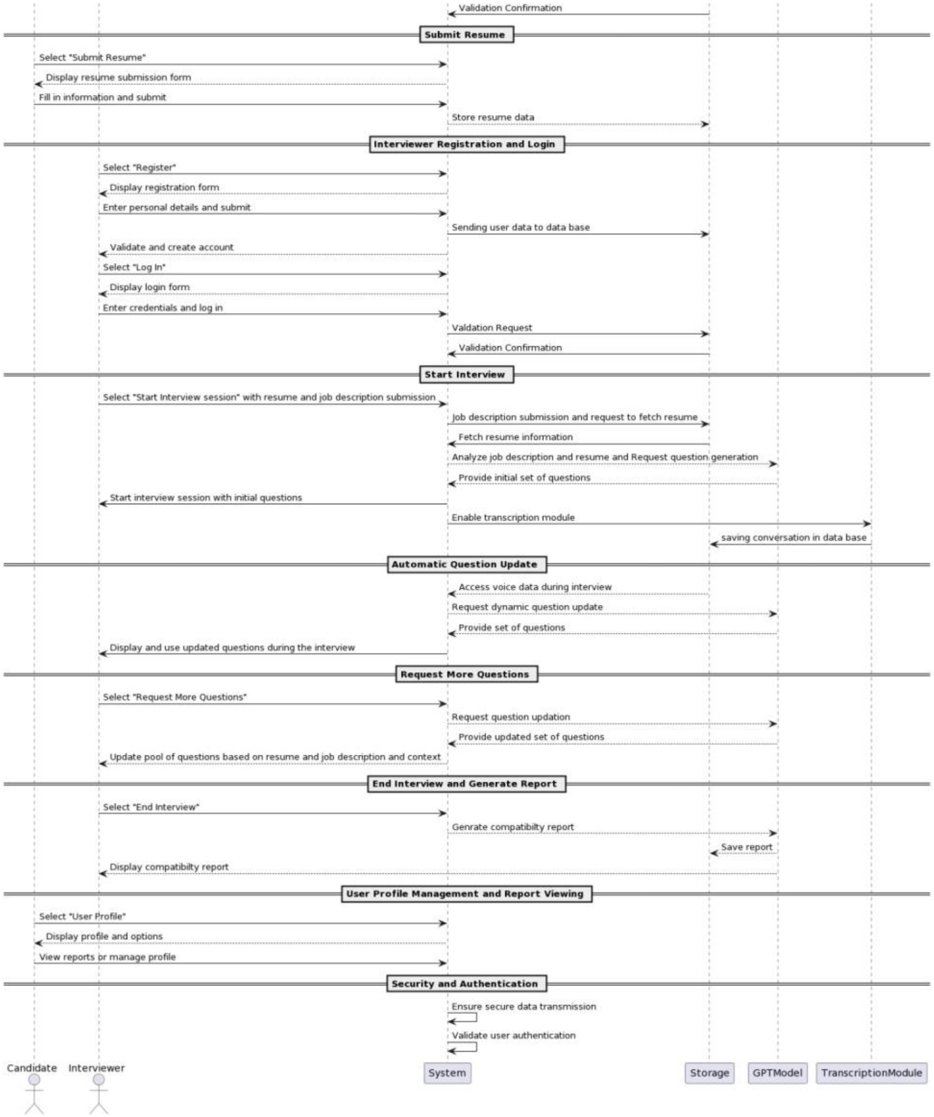


Process flow/Representation

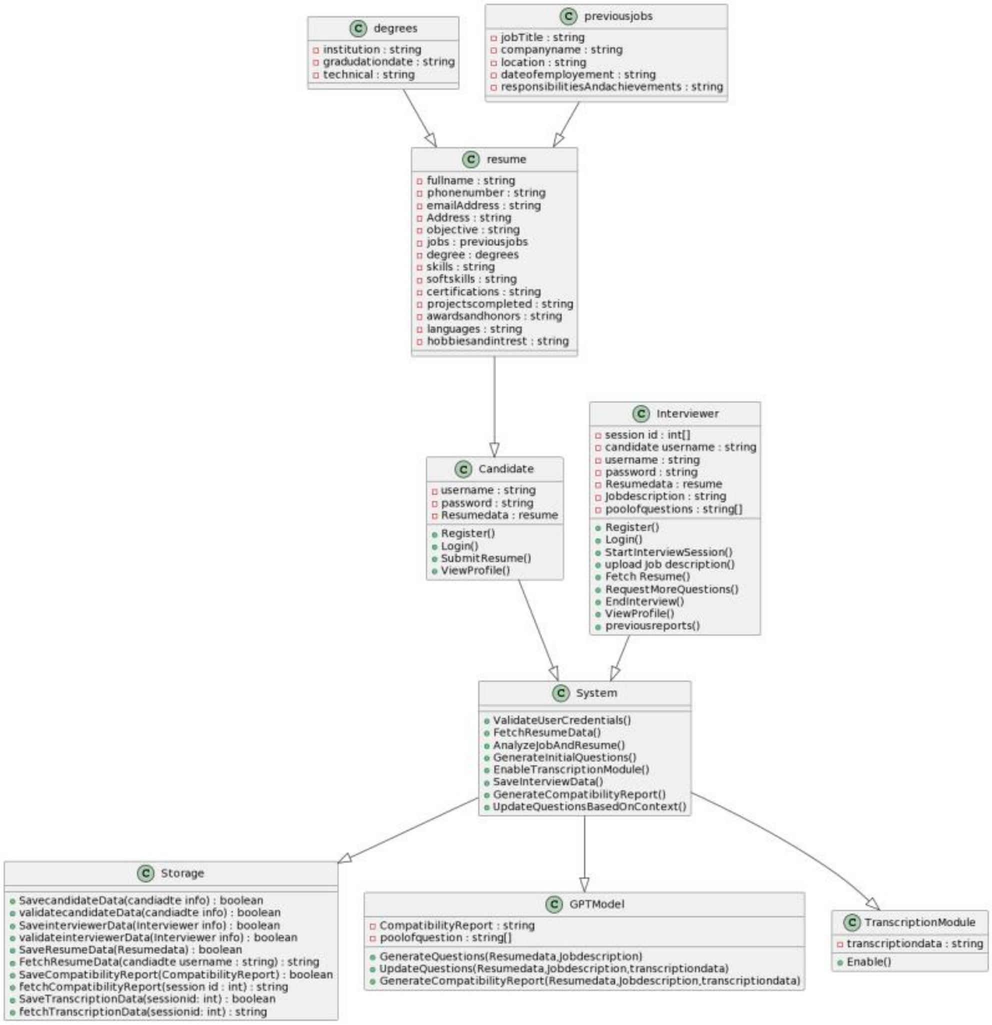

Candidate
Interviewer
Candidate selects "Register"
Display registration form
Candidate enters personal details and submits
Validate and create account
Candidate selects "Log In"
Display login form
Candidate enters credentials and logs in
Submit Resume
Candidate selects "Submit Resume"
Display resume submission form
Candidate fills in information and submits
Store resume data
Interviewer Registration and Login
Interviewer selects "Register"
Display registration form
Interviewer enters information and submits
Validate and create account
Interviewer selects "Log In"
Display login form
Interviewer enters credentials and logs in
yes
Start Interview?
no
Interviewer selects "Start Interview"
View previous Interview reports
Analyze job description and resume
Request question generation
Provide initial set of questions
yes
Enable Transcription Module?
no
Interviewer enables transcription module
Display and use initial questions
Access voice data during interview
Request dynamic question update
Provide updated set of questions
Display and use updated questions during the interview
yes
Request More Questions?
no
Interviewer selects "Request More Questions"
Continue interview with existing questions
Update pool of questions based on resume and job description
Request additional questions
Provide new set of questions
Display and use new questions during the interview
End Interview and Generate Report
Interviewer selects "End Interview"
Conclude interview session
Generate interview report

Design models [along with descriptions]

Sequence Diagram

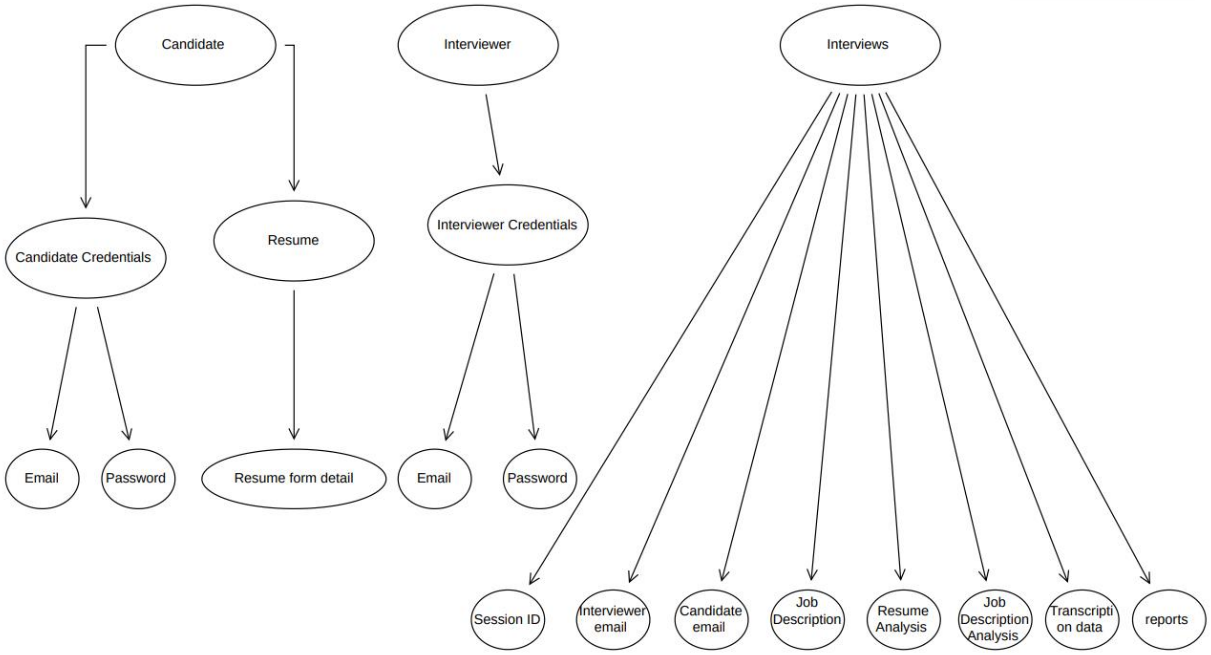

Validation Confirmation
Submit Resume
Select "Submit Resume"
Display resume submission form
Fill in information and submit
Store resume data
Interviewer Registration and Login
Select "Register"
Display registration form
Enter personal details and submit
Sending user data to data base
Validate and create account
Select "Log In"
Display login form
Enter credentials and log in
Valdation Request
Validation Confirmation
Start Interview
Select "Start Interview session" with resume and job description submission
Job description submission and request to fetch resume
Fetch resume information
Analyze job description and resume and Request question generation
Provide initial set of questions
Start interview session with initial questions
Enable transcription module
saving conversation in data base
Automatic Question Update
Access voice data during interview
Request dynamic question update
Provide set of questions
Display and use updated questions during the interview
Request More Questions
Select "Request More Questions"
Request question updation
Provide updated set of questions
Update pool of questions based on resume and job description and context
End Interview and Generate Report
Select "End Interview"
Genrate compatibilty report
Save report
Display compatibilty report
User Profile Management and Report Viewing
Select "User Profile"
Display profile and options
View reports or manage profile
Security and Authentication
Ensure secure data transmission
Validate user authentication
Candidate
Interviewer
System
Storage
GPTModel
TranscriptionModule

Class Diagram


degrees
institution : string
gradudationdate : string
technical : string
previousjobs
jobTitle : string
companyname : string
location : string
dateofemployement : string
responsibilitiesAndachievements : string
resume
fullname : string
phonenumber : string
emailAddress : string
Address : string
objective : string
jobs : previousjobs
degree : degrees
skills : string
softskills : string
certifications : string
projectscompleted : string
awardsandhonors : string
languages : string
hobbiesandintrest : string
Interviewer
session id : int[]
candidate username : string
username : string
password : string
Resumedata : resume
Jobdescription : string
poolofquestions : string[]
Register()
Login()
StartInterviewSession()
upload Job description()
Fetch Resume()
RequestMoreQuestions()
EndInterview()
ViewProfile()
previousreports()
Candidate
username : string
password : string
Resumedata : resume
Register()
Login()
SubmitResume()
ViewProfile()
System
ValidateUserCredentials()
FetchResumeData()
AnalyzeJobAndResume()
GenerateInitialQuestions()
EnableTranscriptionModule()
SaveInterviewData()
GenerateCompatibilityReport()
UpdateQuestionsBasedOnContext()
Storage
SavecandidateData(candiadte info) : boolean
validatecandidateData(candiadte info) : boolean
SaveinterviewerData(Interviewer info) : boolean
validateinterviewerData(Interviewer info) : boolean
SaveResumeData(Resumedata) : boolean
FetchResumeData(candiadte username : string) : string
SaveCompatibilityReport(CompatibilityReport) : boolean
fetchCompatibilityReport(session id : int) : string
SaveTranscriptionData(sessionid: int) : boolean
fetchTranscriptionData(sessionid: int) : string
GPTModel
CompatibilityReport : string
poolofquestion : string[]
GenerateQuestions(Resumedata,Jobdescription)
UpdateQuestions(Resumedata,Jobdescription,transcriptiondata)
GenerateCompatibilityReport(Resumedata,Jobdescription,transcriptiondata)
TranscriptionModule
transcriptiondata : string
Enable()

Data design

1. First of all, the candidate and interviewer should be registered and their credentials should be saved in the database so that when they want to login there credentials can be validated.
2. Candidates will fill the form of the resume after logging in and then that data of resume will be stored in database with the unique email address of the candidate.
3. The same is for the interviewer that he/she will login first and after getting login there will be two follows one is to get only the previous reports of interviews by just entering the unique session id of the interview present in the interview table.
4. And in the second case which is main case the interviewer will start interview and unique session id for that interview will be stored in the database after that interviewer will upload the job description and it will be saved in the database and then interviewer will fetch the resume of the candidate by adding the email of candidate which will be part of the interview table after analysis will be made for both job description and resume and there analysis will also be save in the database after that conversation will be started between the interviewer and candidate that conversation will also be stored in table of interview and at last when the interviewer ends the interview the report will be generated and saved in the database which can be accessed by using the unique id of interview session.



Data dictionary

1. In the Candidate table there will be candidate credentials (email, password) and the resume information which includes (name, degrees, jobs, hobbies etc.)
2. In the interviewer table there will be credentials of the interviewers.
3. In the interviews table there will be unique session id for the interview email of interviewer and candidate, job description, Job description and resume Analysis data of conversation between candidate and interviewer known as transcription data and the reports of the interviews.

Algorithm & Implementation

Pseudo code

Candidate Registration

```
function registerCandidate(name, email, password):
    if email is not already in use:
        createCandidateAccount(name, email, password)
        return "Registration successful"
    else:
        return "Email is already in use"
function createCandidateAccount(name, email, password):
    // Create a new candidate account
    // Set candidate details such as name, email, and password
    // Store candidate details in the database
```

Candidate Login

```
function loginCandidate(email, password):
    if credentials are valid:
        // Log in the candidate
        return "Login successful"
    else:
        return "Invalid credentials"
```

Submit Resume

```
function submitResume(candidateID, registrationNumber, all other resume details):
    // Submit a resume for the candidate
    // Store resume details in the database linked to the candidate
```

Interviewer Registration

```
function registerInterviewer(name, email, password):
    if email is not already in use:
        createInterviewerAccount(name, email, password)
        return "Registration successful"
    else:
        return "Email is already in use"


function createInterviewerAccount(name, email, password):
    // Create a new interviewer account
    // Set interviewer details such as name, email, and password
    // Store interviewer details in the database
```

Interviewer Login

```
function loginInterviewer(email, password):
    if credentials are valid:
        // Log in the interviewer
        return "Login successful"
    else:
        return "Invalid credentials"
```

Start Session

```
function startInterviewSession(interviewerID, sessionName, candidateInfo):
    // Start a new interview session
    // Create a session in the system and link it to the interviewer
    // Store session details in the database
```

Submit Job Description

```
function submitJobDescription(interviewerID, jobDetails):
    // Submit a job description
    // Store job details in the system linked to the interviewer
```

Fetch Resume

```
function fetchResume(interviewerID, candidateEmail):
    // Fetch the candidate's resume
    // Retrieve resume details from the system based on the candidate's email
```

Start Interview

```
function startInterview(interviewerID, sessionName, candidateInfo):
    // Start an interview session
    // Retrieve session details from the system and initiate the interview
```

Refresh Question Pool (Automated)

```
function refreshQuestionPool(interviewerID, sessionName):
    // Refresh the question pool during an interview session
    // Update the pool of interview questions based on the current context
```

End Interview and Generate Report (Automated)

```
function endInterviewAndGenerateReport(interviewerID, sessionName):
    // Conclude an interview session and generate a report
    // Generate a summary report of the interview details
```

User Profile Management

```
function updateUserProfile(userID, newDetails):
    // Update user profile details
    // Update user information in the system
```

View Previous Interview Reports

```
function viewPreviousInterviewReports(interviewerID):
    // Retrieve and display previous interview reports
    // Fetch and display reports based on the interviewer's ID
```

Software requirements traceability matrix

This section should contain a table that summarizes how each software requirement has been met in this document. The tabular format permits one-to-one and one-to-many relationships to be shown.

Table 1 Requirements Traceability Matrix

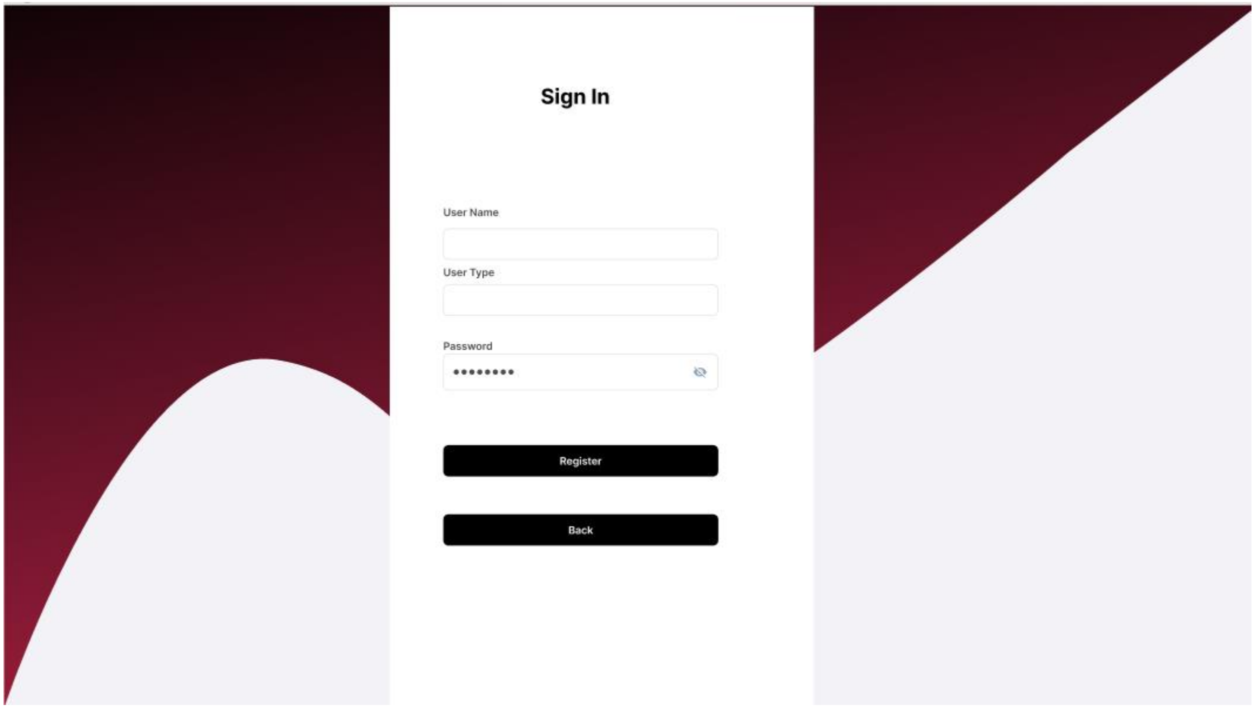
Req. Number	Ref. Item	Design Component	Component Items
FR01	Class Diagram	Candidate, Storage, System	Candidate Registration
FR02	Class Diagram	Candidate, Storage, System	Candidate Login
FR03	Class Diagram	Candidate, Previous job, Degree, storage	Submit Resume
FR04	Class Diagram	Interviewer, Storage, System	Interviewer Register
FR05	Class Diagram	Interviewer, Storage, System	Interviewer Login
FR06	Class Diagram	Interviewer	Start Interview session
FR07	Class Diagram	Interviewer, System, Storage	Upload Job description
FR08	Class Diagram	Interviewer, Storage,	Fetch Job Description

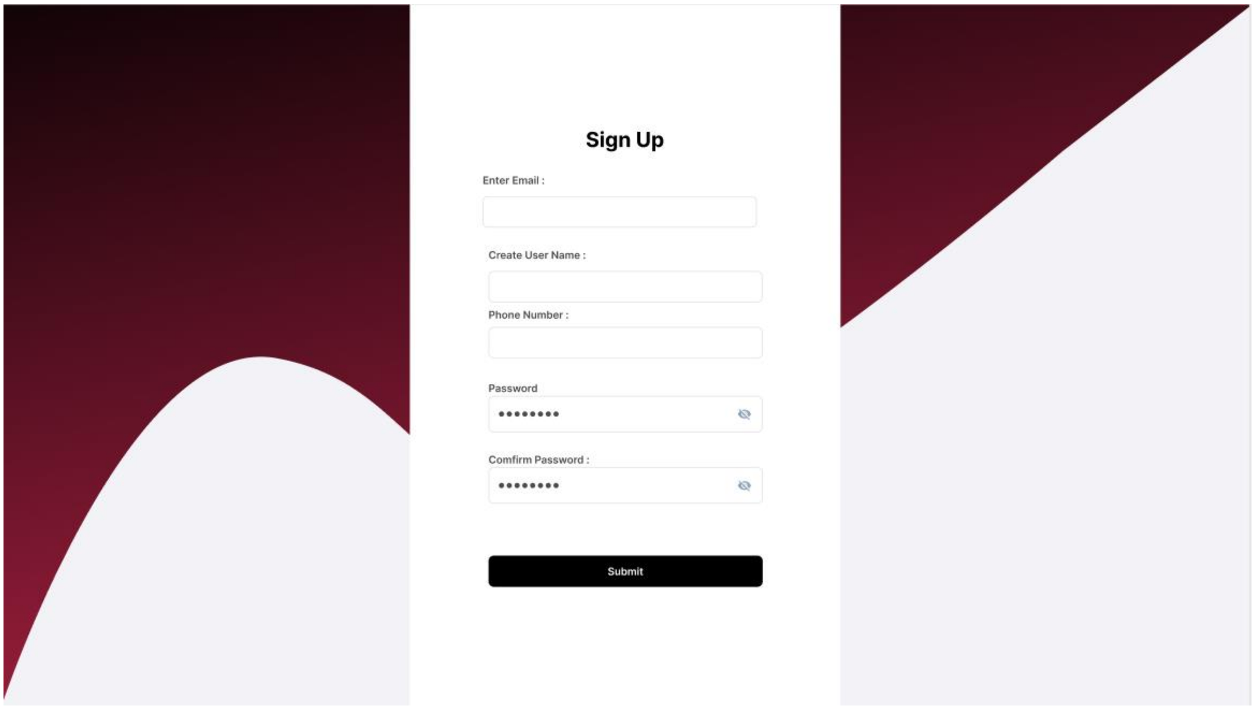
FR09	Class Diagram	Interviewer, Storage, System, GPT Model	Start Interview
FR10	Class Diagram	System, GPT model	Initial Question Generation
FR11	Class Diagram	Interviewer, Storage, System, GPT model	Refresh Question Pool
FR12	Class Diagram	Interviewer, System, Storage, GPT model	End interview and Generate report.
FR13	Class Diagram	Interviewer, Storage	Profile Management
FR14	Class Diagram	Interviewer, Storage, System	View Previous Reports
FR15	Class Diagram	Transcription Module	Transcription from voice to text

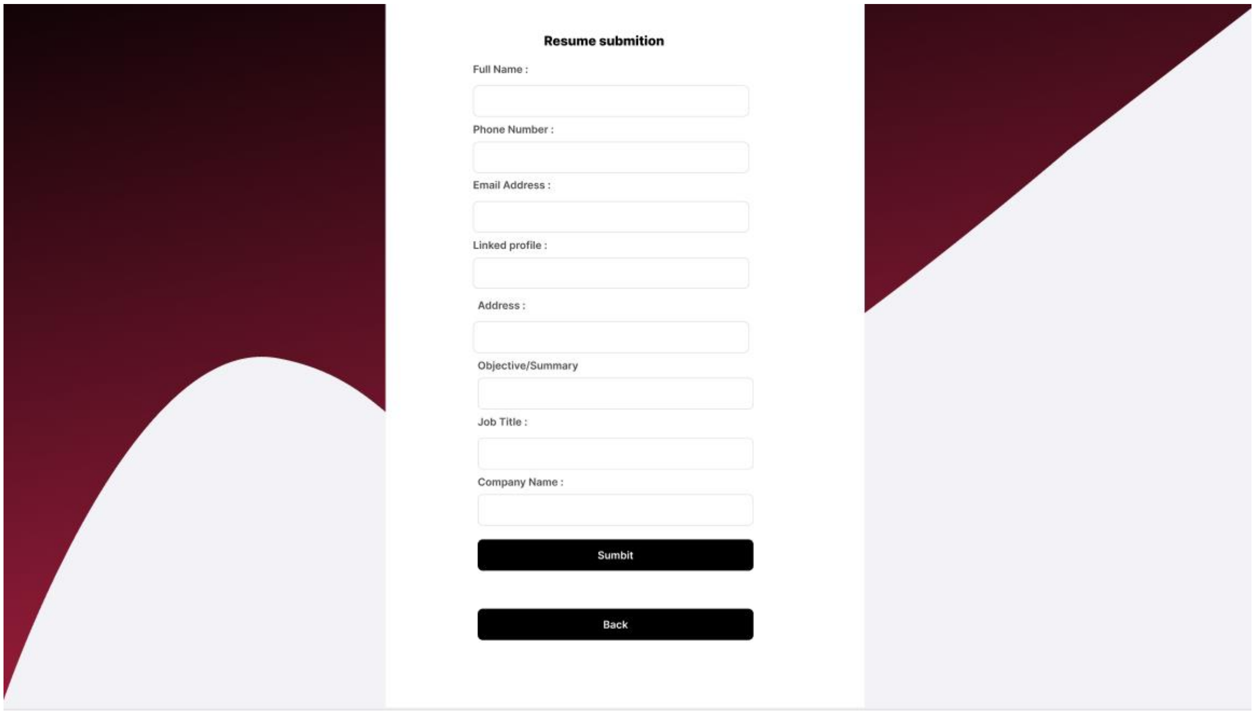
Human interface design

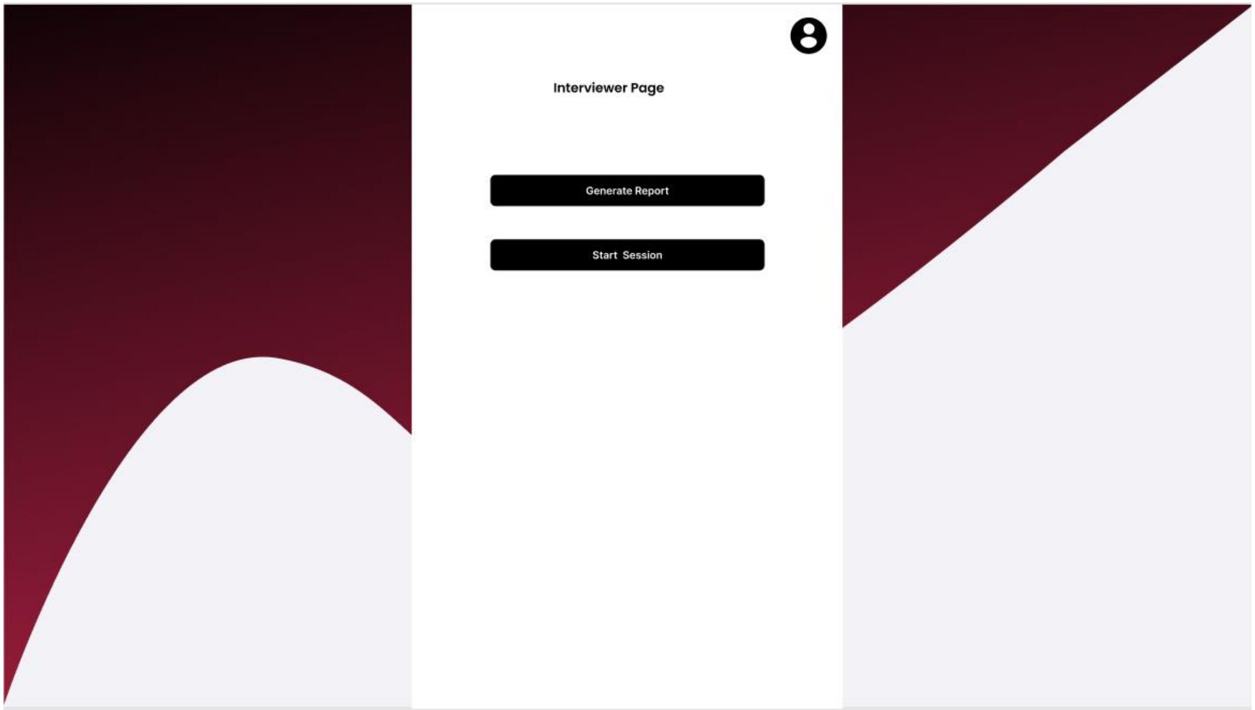
1. The candidate and interviewer can create account and login to there account using the credentials such as email and password.
2. Candidates have option of upload resume when he clicks on upload resume button a form opens the candidate will fill that form and will submit it by clicking the submit button and the data from that form will be stored in the data base.
3. Interviewer can start a new interview session or generate previous interview report.
4. BY clicking on button if previous interview reports a page will open where interviewer will add the unique session id of interview in text box and the report will be generated against that session id by fetching it from the database.
5. If the interviewer clicks on the start session button a page will open which will have three options upload job description , resume fetch and start interview.
6. First of all interviewers will click on the upload job description button after that a page will open where interviewer can select the file to upload a pdf file by using link and after reading that file form pc interviewer will click on submit button and it will be stored in database then interviewer will go back to the session page
7. Now interviewer will click on the fetch resume button and page will open where interviewer will add the email of candidate in the text box and then click on fetch button the resume details which are submitted by the candidate will be fetch from the database.
8. Again, the interviewer will come on the session page and click on the start interview button. When the interviewer clicks on the button the job description and resume analysis are done by sending data to the gpt model .
9. A new screen will be open which will consist of 6 initial set of questions generated by gpt model in this page interviewer will have two button one for updating questions and one for end interview.
10. When the interviewer clicks on the update pool the set of questions will be updated.
11. When the interviewer clicks on the end interview button the interview will end with the automatic generation of compatibility report.

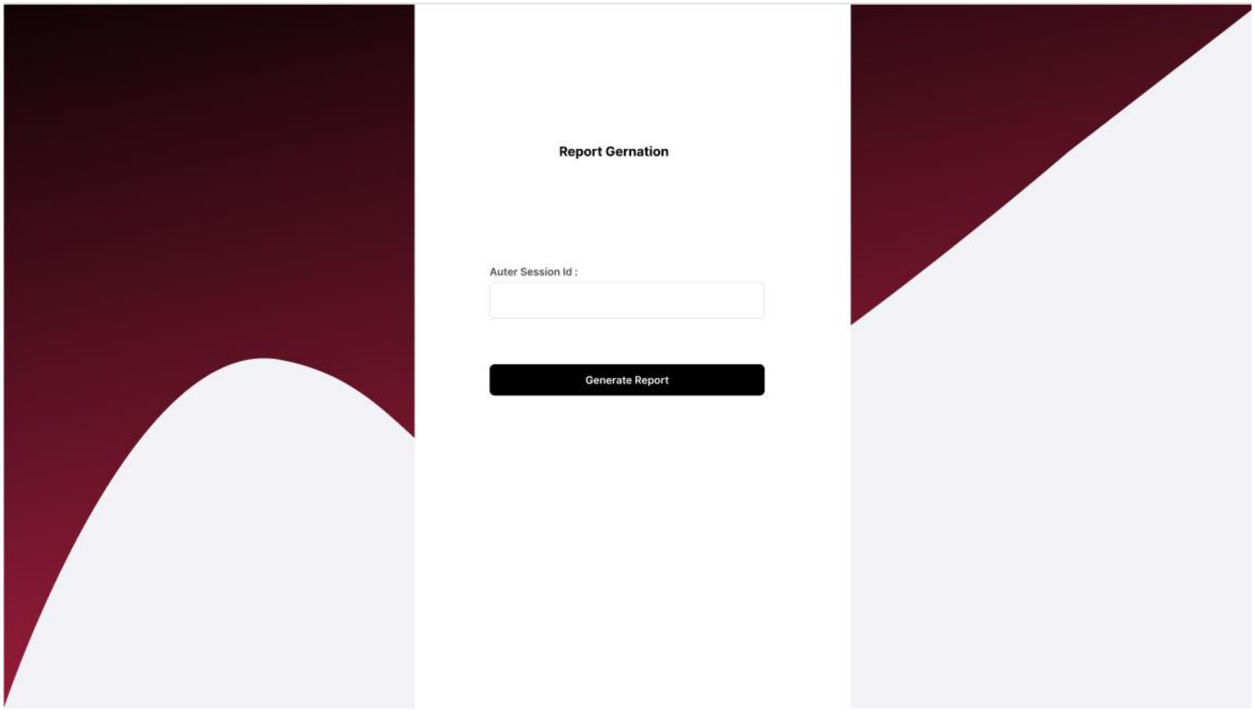
Screen images

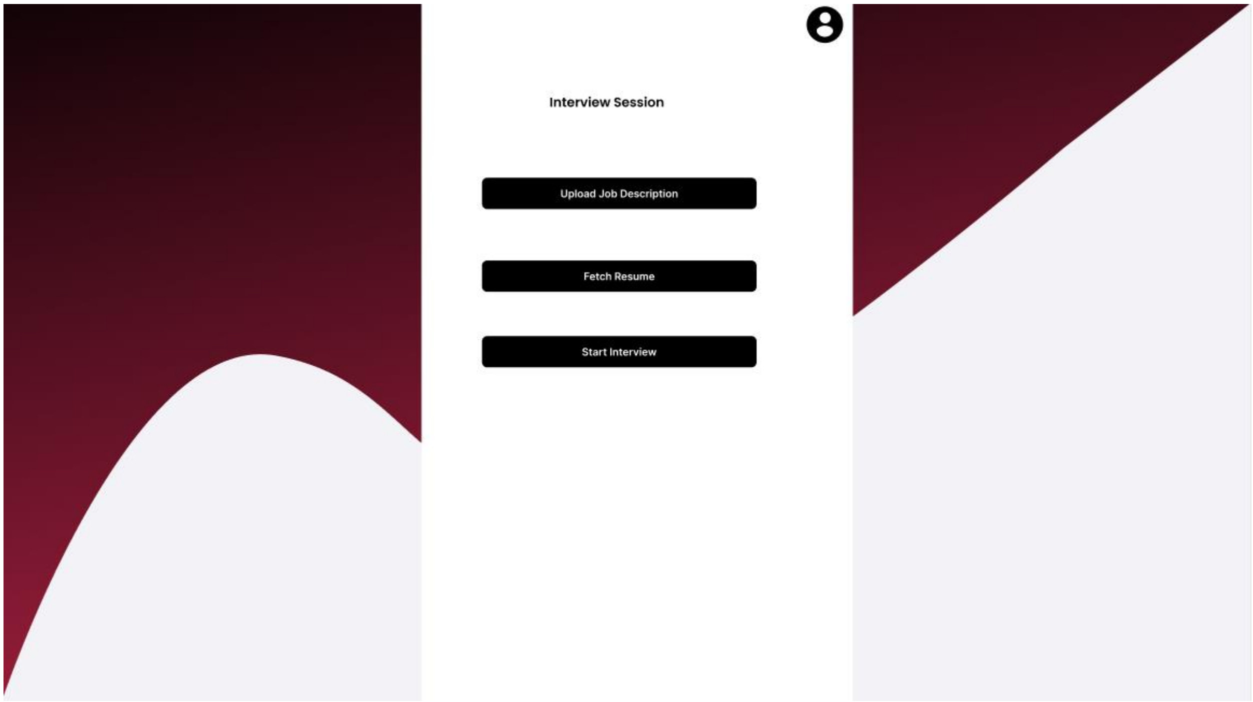


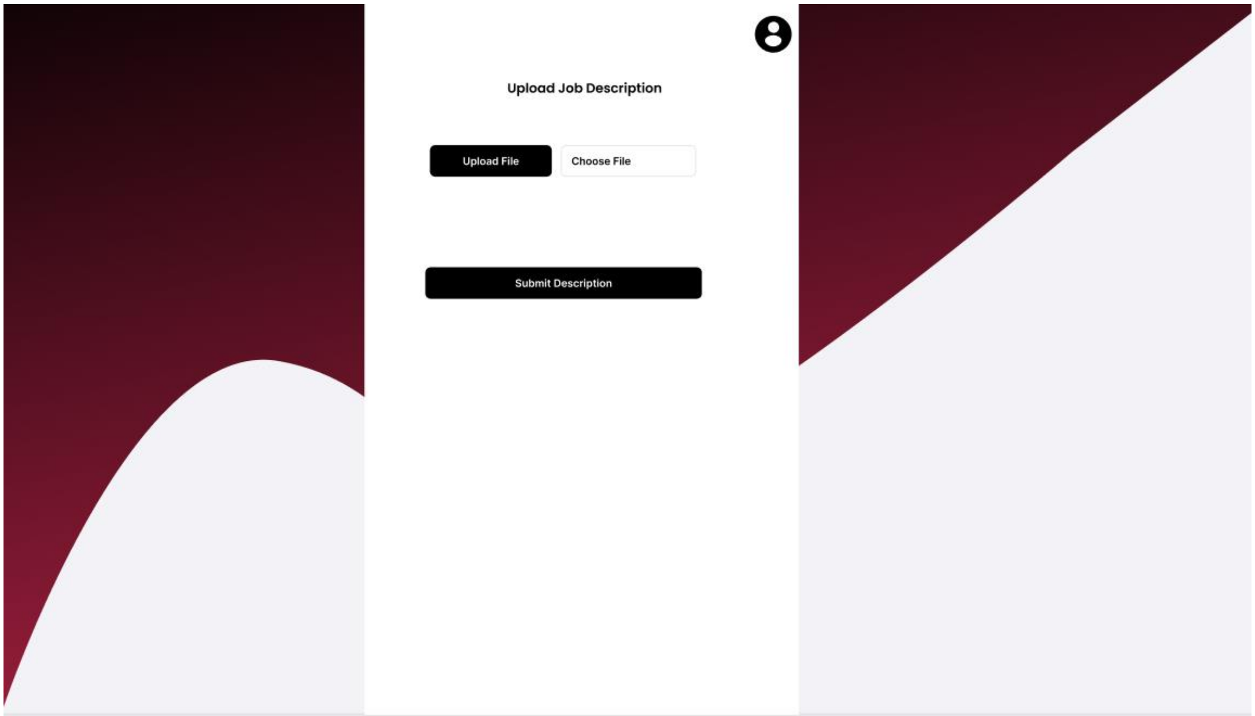


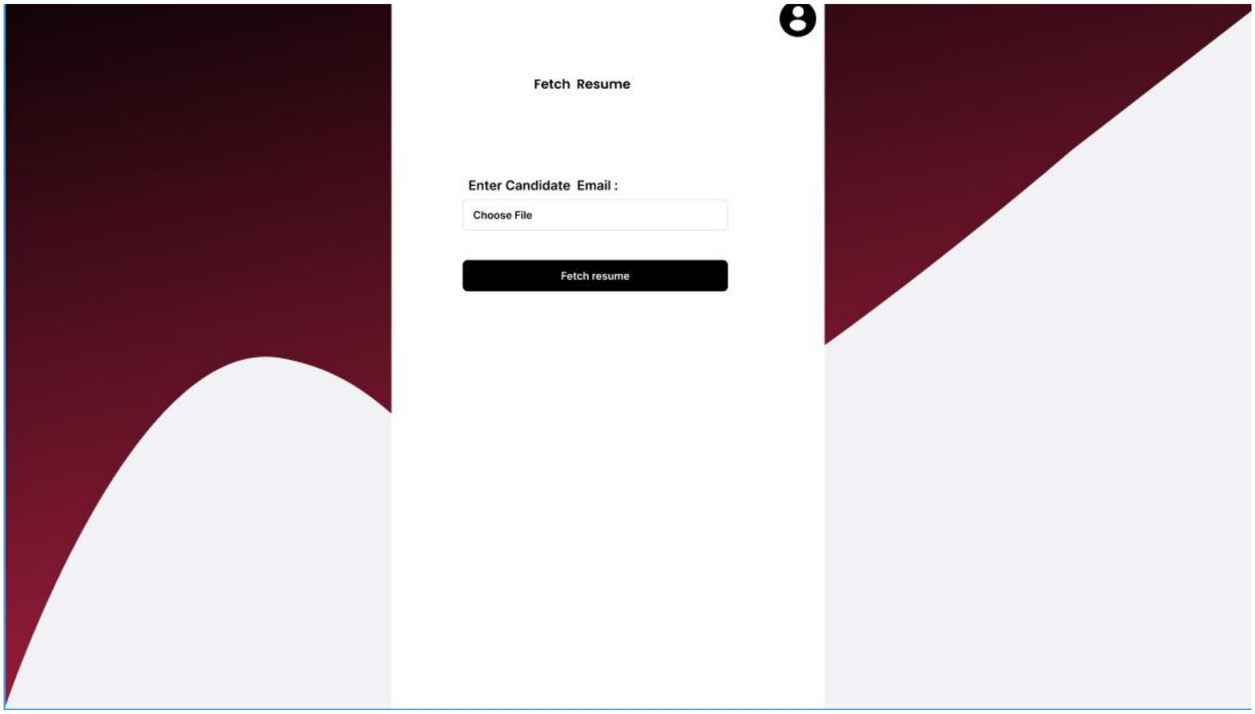

Resume submition
Full Name :
Phone Number :
Email Address :
Linked profile :
Address :
Objective/Summary
Job Title :
Company Name :
Sumbit
Back

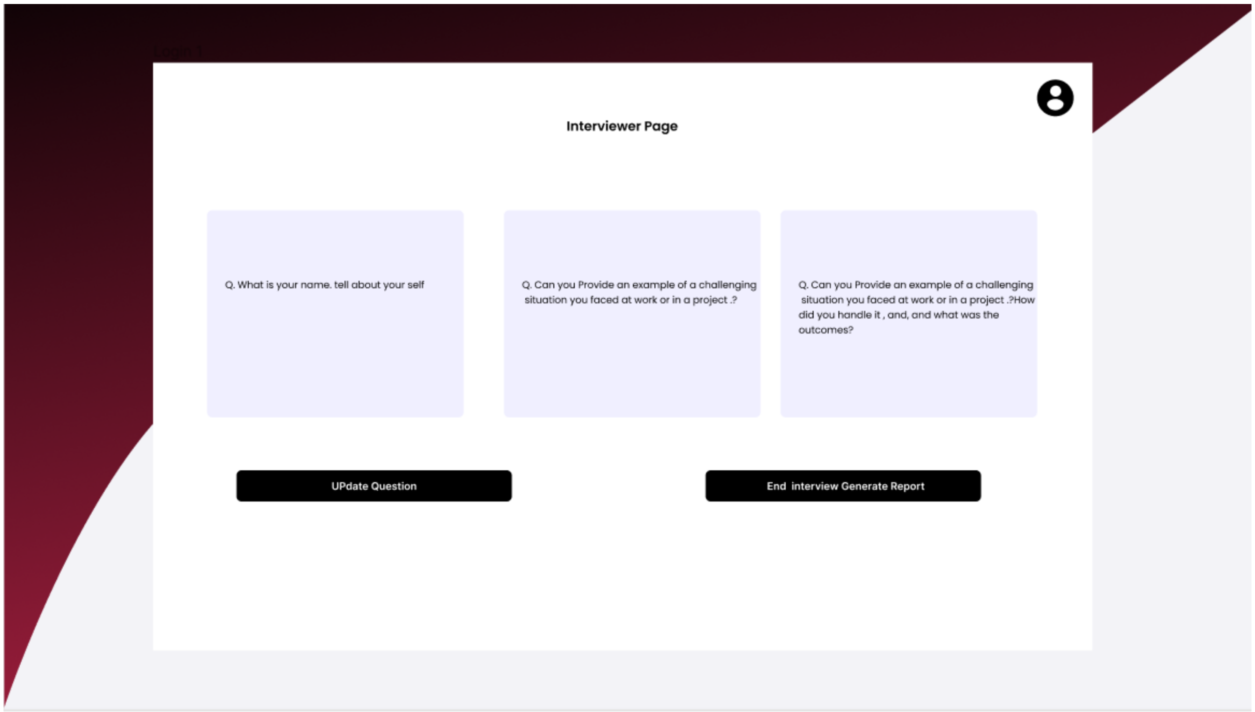

Interviewer Page
Generate Report
Start Session


Report Gernation
Auter Session Id :
Generate Report


Interview Session
Upload Job Description
Fetch Resume
Start Interview


Upload Job Description
Upload File
Choose File
Submit Description


Fetch Resume
Enter Candidate Email :
Choose File
Fetch resume

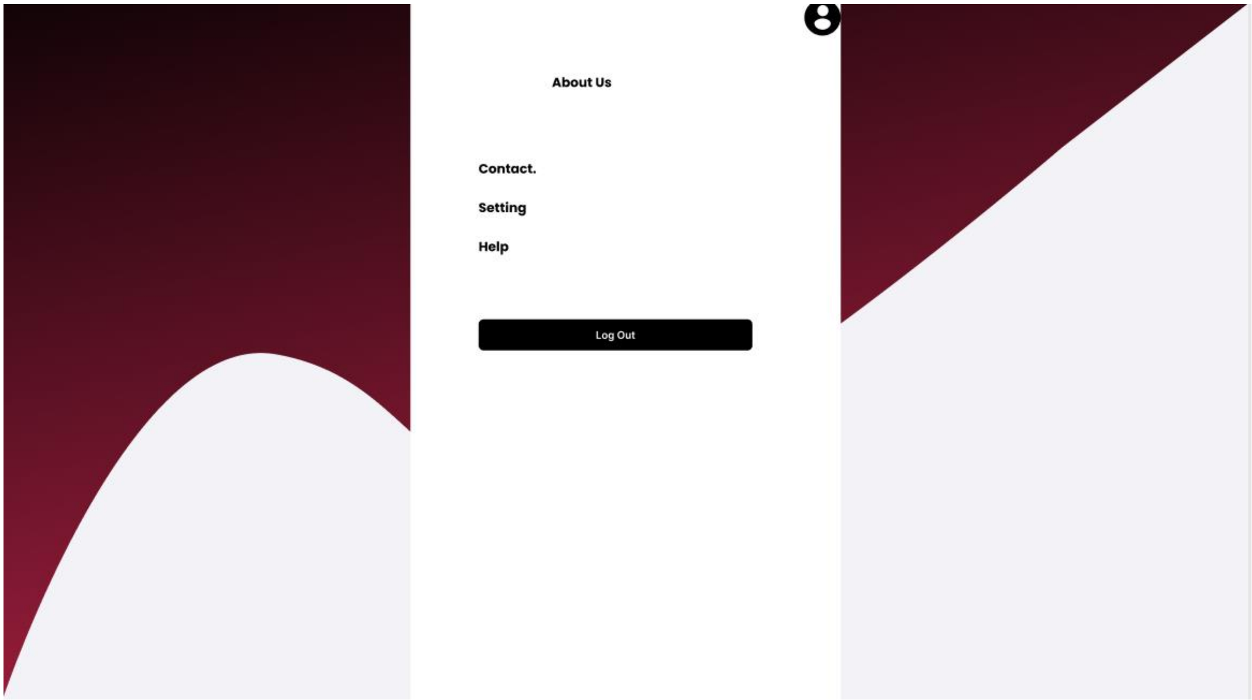


Interviewer Page

Q. What is your name. tell about your self

Q. Can you Provide an example of a challenging situation you faced at work or in a project .?

Q. Can you Provide an example of a challenging situation you faced at work or in a project .?How did you handle it , and, and what was the outcomes?



8.2 Screen objects and actions

Screen 1:

Email (Text Field): Allow users to type their Email.
Password (Text Field): Allow users to type their password.
Login (Button): By clicking login button leads to login the user.
Register (Button): Lead to sign up Screen.

Screen 2:

First Name (Text Field): Allow user to type first name.
Last Name (Text Field): Allow user to type last name.
Email (Text Field): Allow user to type email.
Password (Text Field): Allow user to type password.
Sign up: Lead to sign up the user.

Screen 3:

Resume form (with number of text fields) : Allow users to add data of resume.
Submit (button) : Allow user to submit the information of the resume._

Screen 4:

Generate previous report : Allows user to go to page of generate previous interview report
Start session : Allows user to go to the page of interview session.

Screen 5:

Session id (text field): Allow user to enter the session id of interview whose report interviewer wants to fetch.
Fetch report (button) : Allows interviewer to fetch report of interview whose session id is added in text box.

Screen 6:

Upload job description (Button): Allows interviewer to go to the page of job description submission.
Fetch resume (Button): Allow interviewer to go to the page of resume fetch.
Start Interview(Button): Allow interviewer to start interview after submitting job description and fetching resume.

Screen 7:

Choose File(Button) : Allows interviewer to Choose job description file from computer.
Submit Job description(Button): Allow interviewer to upload selected file and then go back to the Session page.

Screen 8:

Email of candidate(Text Box): Allows interviewer to add email of candidate whose resume interviewer wants to fetch.
Fetch resume (Button): Allow interviewer to Fetch resume from the database and to go back to the session page.

Screen 9:

Questions(Text Boxes): Provides interviewer six questions.
Update pool (Button): Allow interviewer to update the pool of questions.
End interview(Button): Allow interviewer to end interview and compatibility report will be generated Automatically.

Screen 10:
Logout(Button):Allows user to logout.
Profile setting (Button): Allow interviewer to do profile management.

Appendix I

- How to design using UML (OOP): For guidance please follow the instructions mentioned in the link: http://agilemodeling.com/artifacts/

- How and when to design ER diagrams: For guidance please follow the instructions mentioned in the link: http://people.inf.elte.hu/nikovits/DB2/Ullman_The_Complete_Book.pdf

- Data flow diagrams: For guidance please follow the instructions mentioned in the link and book:
    - http://www.agilemodeling.com/artifacts/dataFlowDiagram.htm
    - Software Engineering –A Practitioner's approach by Roger Pressman
- Architecture diagram: For guidance please follow the instructions mentioned in the link and book:
    - Ian Sommerville – Software Engineering 9th Edition– Chapter 6