CSE2004 DATABASE MANAGEMENT SYSTEM

PROJECT REVIEW REPORT

PHASE - 3

REGISTER NO. 1: 19BCE0162

NAME 1: APURVA SHARMA

REGISTER NO. 2: 19BCE0914

NAME 2: KHUSHEE JAIN

EPJ SLOT: L13+14

PROJECT TITLE: ONLINE RECRUITMENT NETWORK

MOBILE NUMBER: 83199 43063

PROJECT TYPE: APPLICATION

APPLICATION NAME : RECRUITEASE

ACKNOWLEDGEMENT

We take this opportunity to express our profound gratitude and deep regards to our project guide **Prof. Saravanakumar K** for his exemplary guidance, monitoring and constant encouragement throughout the course of this subject **CSE2004: Database Management Systems** that helped us to complete this project .

The blessing, help and guidance given by him from time to time shall carry us a long way in the journey of life on which we are about to embark. We also take this opportunity to express a deep sense of gratitude to the management of **VIT UNIVERSITY** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

Lastly, we thank the almighty, **our parents, brothers, sisters and friends** for their constant encouragement without which this project would not be possible.

CONTENTS (Click on the contents):

INTRODUCTION:

AIM:

This project has been done by the students of B.Tech Computer Science program for the course of “Database Management Systems” with course code CSE2004. The basic approach of this project is to create a application for job seekers and companies where they can post job opportunities and look for jobs at the portal, the application will be connected to a database, and hence a fully functional application is to be made.

PROBLEM STATEMENT:

The main aim of this project is to prepare an online recruitment system where applicants and companies can find jobs and post vacancies and opportunities.

PROPOSED SYSTEM OVERVIEW

1. **Home Page:** The landing page where the user can navigate to login or signup. The job seekers can also browse for job opportunities without logging in.
2. **Finding Jobs:** Navigate through jobs posted by companies.
3. **Sign Up Page:** Signing up as either a job seeker or a company.
4. **Login:** Logging into your account
5. **Account Page:** Interface for the user to see application status as a job seeker, look for applications from the job seekers and send interview schedules and offer letters to the applicant.

PHASE 1 DOCUMENTATION:

Note: Some tables were added/changed

DATA COLLECTION STAGE:

List of Entity Sets:

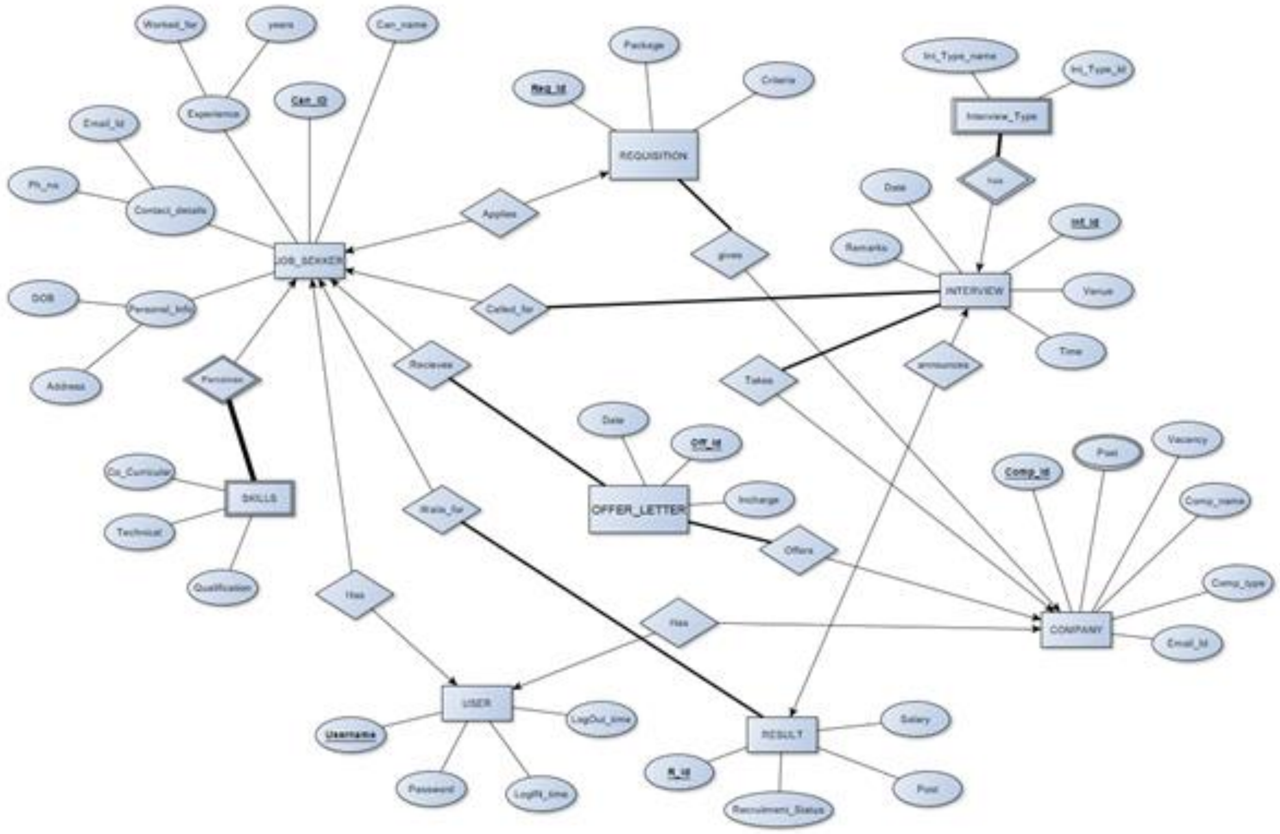
1. Company
2. Offer Letter
3. User
4. Job Seeker
5. Requisition
6. Interview
7. Skill
8. Interview_Type
9. Result

DATA IDENTIFICATION STAGE:

1. Company (Comp_id , Comp_name, Email_id, Comp_Type,Vacany,Post)
2. Offer Letter (Off_Id, Date, Incharge)
3. User(Username, Password, Login_Time, Logout_Time)
4. Job Seeker (Can_Id, Can_Name, Pesonal_Info, Experience,Contact_details)
5. Requisition (Package, Criteria, Req_Id)
6. Skill (Qualification, Co_Curricular, Technical)

7. Interview (int_Id, Remarks, Date, Time,Venue)
8. Interview_Type(int_Type_Id, int_Type_Name)
9. Result (R_Id, Recruitment_status, Post, Salary)

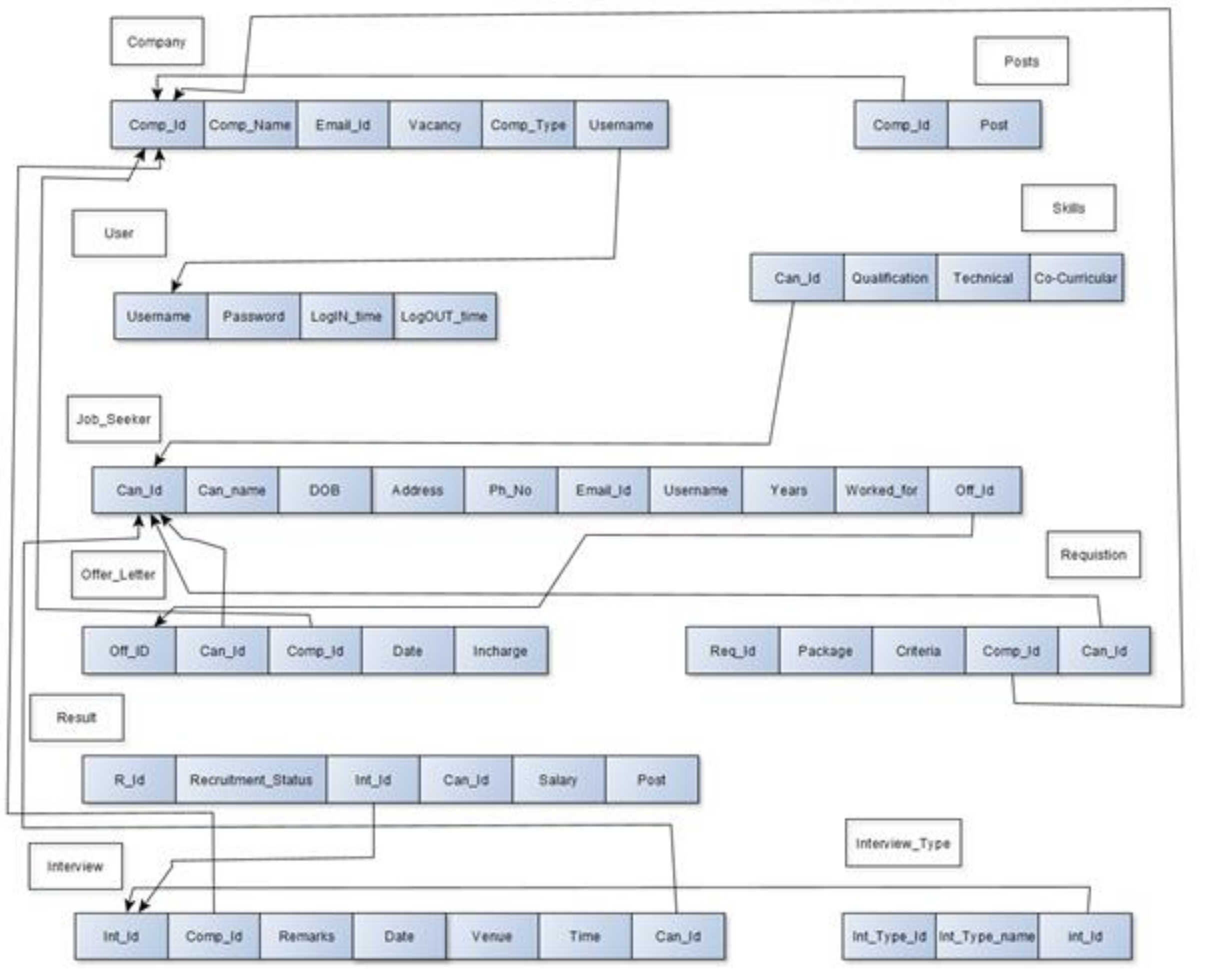
E-R DIAGRAM:



RELATIONSHIP SETS:

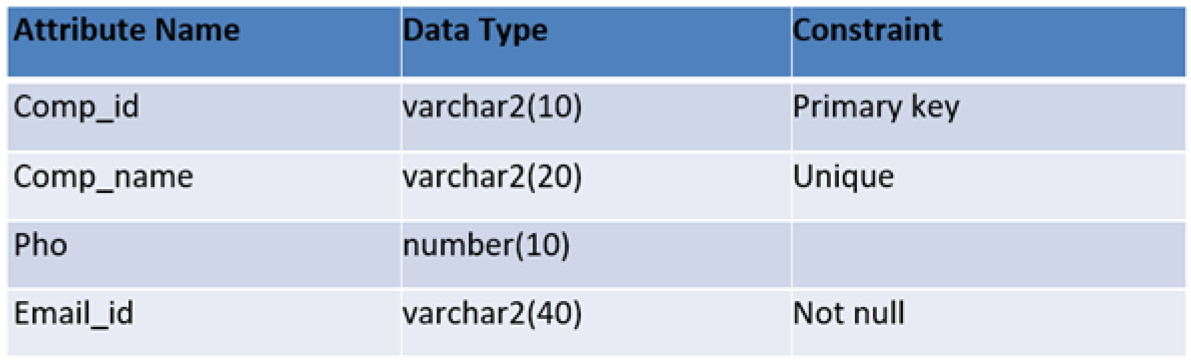
1. Company (Comp_id , Comp_name, Email_id,Comp_Type,Vacany, Username)
2. Offer Letter (Off_Id, Comp_Id, Can_Id, Date, Incharge)
3. User(Username, Password, Login_Time, Logout_Time)
4. Job Seeker (Can_Id, Can_Name, DOB, Address, Email_Id, Ph_no, Worked_for,years, Offer_Id, Username)
5. Requistion (Package , Criteria, Req_Id, Comp_Id,Can_Id)
6. Skill (Can_Id, Qualification, Co_Curricular, Technical)
7. Interview (int_Id, Comp_Id, Remarks, Date, Time, Venue, Can_Id)
8. Interview_Type(int_Type_Id, int_Type_Name, int_Id)
9. Result(R_Id,Recruitment_status, Post, Salary, Can_Id, int_Id)
10. Posts(Comp_Id,Post)

<u>SCHEMA:</u>


Company
Comp_Id
Comp_Name
Email_Id
Vacancy
Comp_Type
Username
Posts
Comp_Id
Post
User
Username
Password
LogIN_time
LogOUT_time
Skills
Can_Id
Qualification
Technical
Co-Curricular
Job_Seeker
Can_Id
Can_name
DOB
Address
Ph_No
Email_Id
Username
Years
Worked_for
Off_Id
Offer_Letter
Off_ID
Can_Id
Comp_Id
Date
Incharge
Requistion
Req_Id
Package
Criteria
Comp_Id
Can_Id
Result
R_Id
Recruitment_Status
Int_Id
Can_Id
Salary
Post
Interview
Int_Id
Comp_Id
Remarks
Date
Venue
Time
Can_Id
Interview_Type
Int_Type_Id
Int_Type_name
int_Id

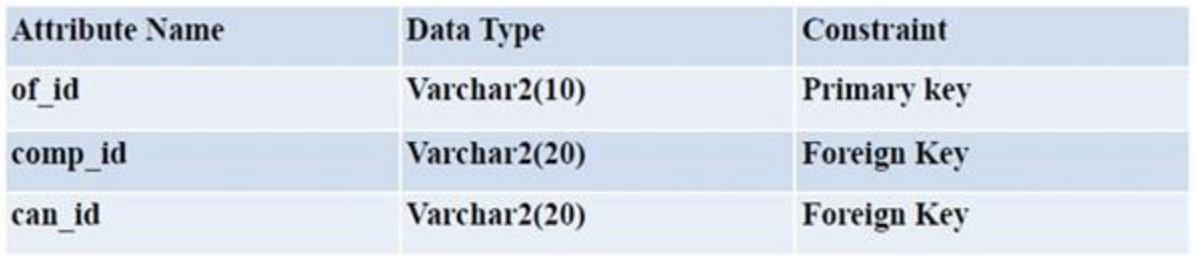
Addition of Constraint on the Conceptual Schema

COMPANY:



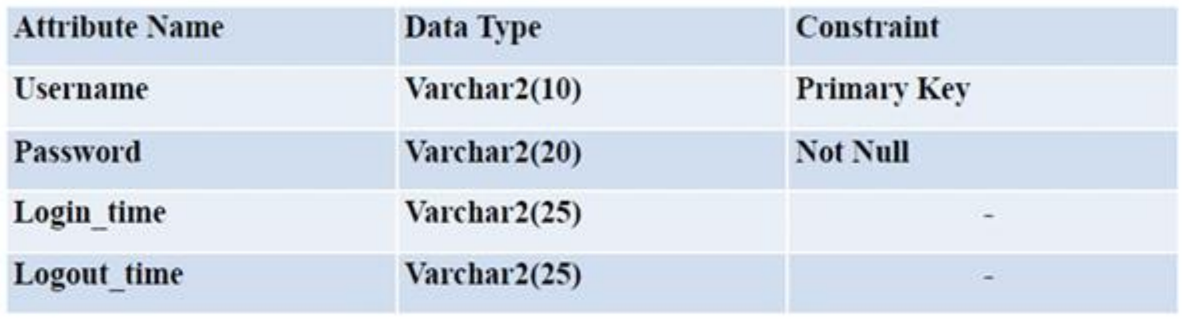
Attribute Name	Data Type	Constraint
Comp_id	varchar2(10)	Primary key
Comp_name	varchar2(20)	Unique
Pho	number(10)	
Email_id	varchar2(40)	Not null

Offer Letter:



Attribute Name	Data Type	Constraint
of_id	Varchar2(10)	Primary key
comp_id	Varchar2(20)	Foreign Key
can_id	Varchar2(20)	Foreign Key

ADMIN:

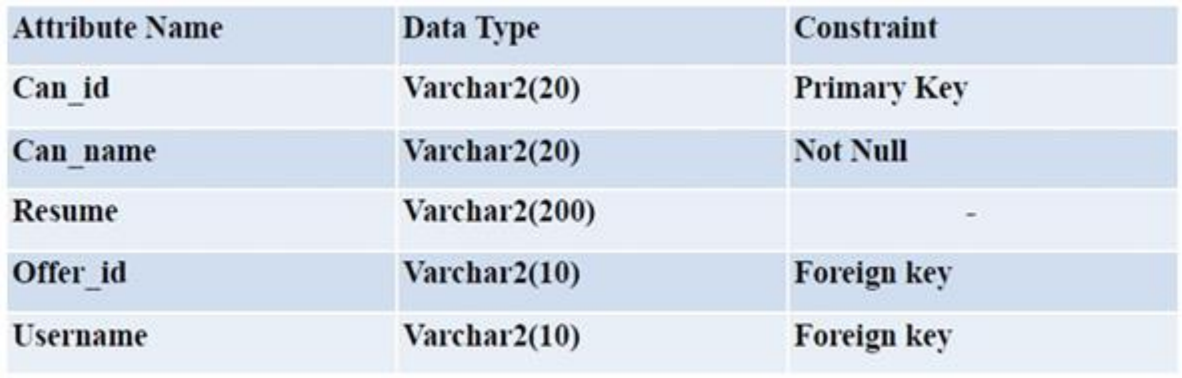


Attribute Name	Data Type	Constraint
Username	Varchar2(10)	Primary Key
Password	Varchar2(20)	Not Null
Login_time	Varchar2(25)	-
Logout_time	Varchar2(25)	-

Experience:

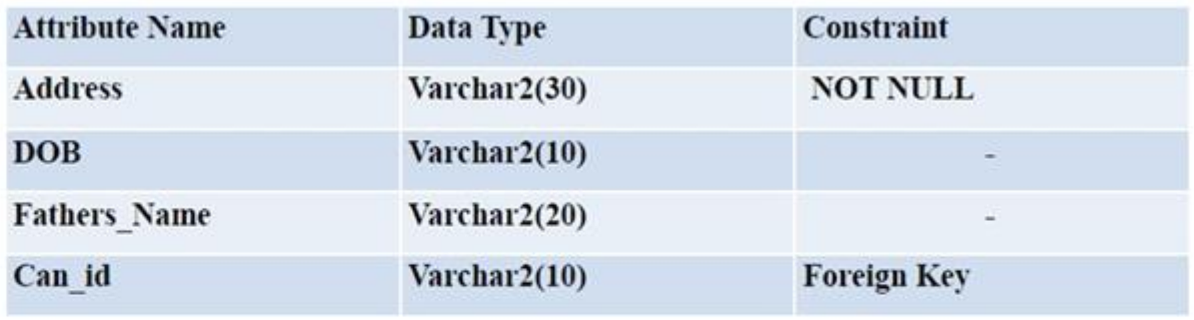
Attribute Name	Data Type	Constraint
Exp_detail	Varchar2(30)	-
Exp_org	Varchar2(10)	-
Comp_id	Varchar2(20)	Foreign key
Can_id	Varchar2(20)	Foreign key

Job Seeker:



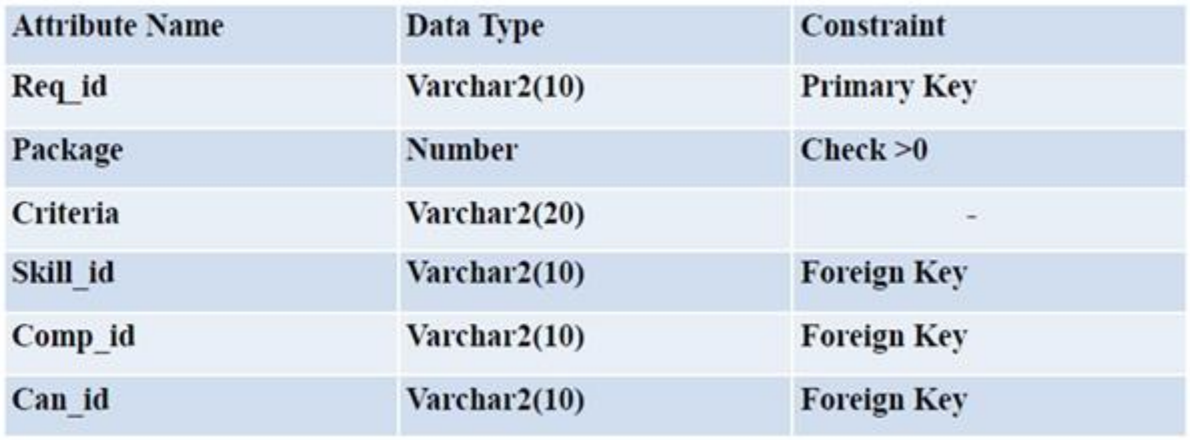
Attribute Name	Data Type	Constraint
Can_id	Varchar2(20)	Primary Key
Can_name	Varchar2(20)	Not Null
Resume	Varchar2(200)	-
Offer_id	Varchar2(10)	Foreign key
Username	Varchar2(10)	Foreign key

Personal Info:



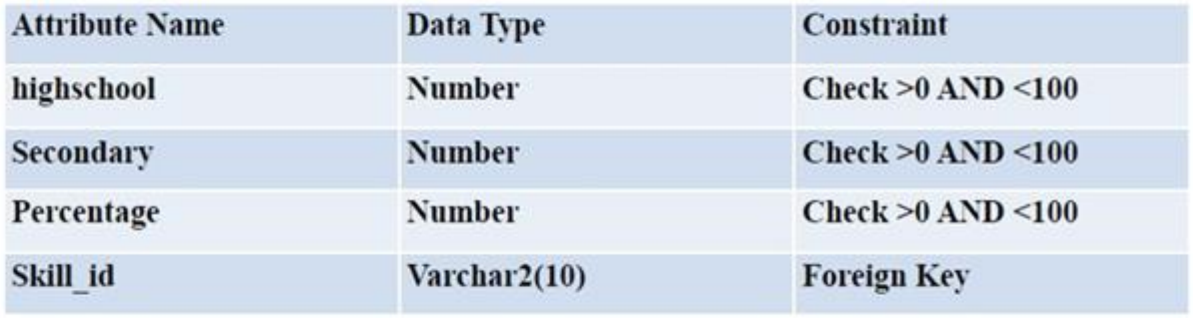
Attribute Name	Data Type	Constraint
Address	Varchar2(30)	NOT NULL
DOB	Varchar2(10)	-
Fathers_Name	Varchar2(20)	-
Can_id	Varchar2(10)	Foreign Key

Requisition:



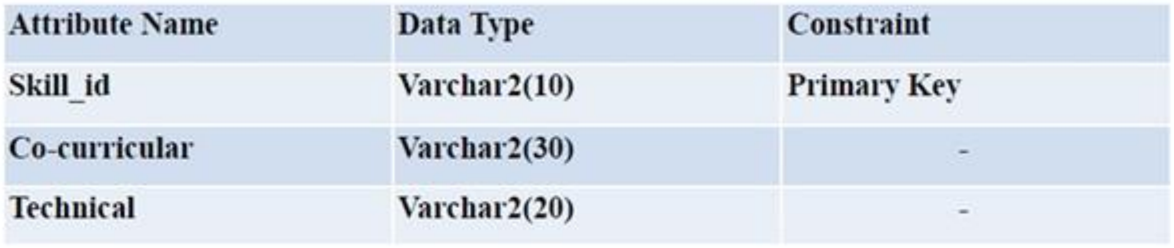
Attribute Name	Data Type	Constraint
Req_id	Varchar2(10)	Primary Key
Package	Number	Check >0
Criteria	Varchar2(20)	-
Skill_id	Varchar2(10)	Foreign Key
Comp_id	Varchar2(10)	Foreign Key
Can_id	Varchar2(10)	Foreign Key

Academics:



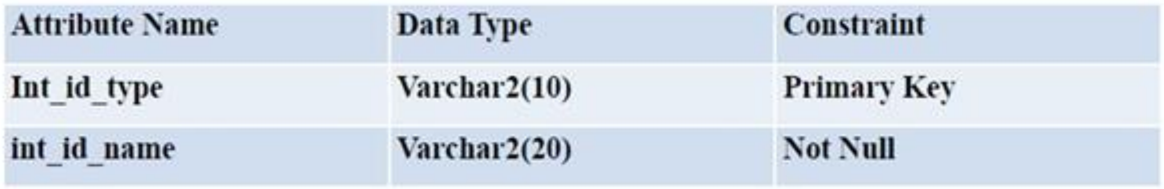
Attribute Name	Data Type	Constraint
highschool	Number	Check >0 AND <100
Secondary	Number	Check >0 AND <100
Percentage	Number	Check >0 AND <100
Skill_id	Varchar2(10)	Foreign Key

SKILL:



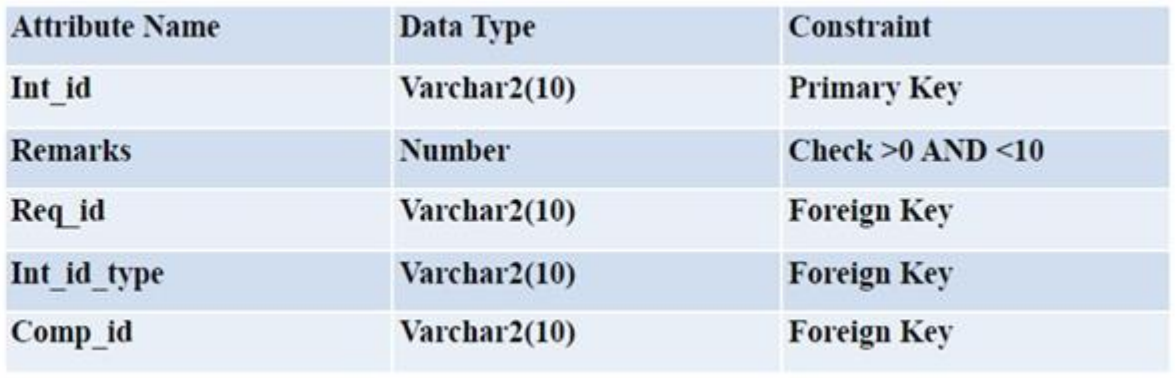
Attribute Name	Data Type	Constraint
Skill_id	Varchar2(10)	Primary Key
Co-curricular	Varchar2(30)	-
Technical	Varchar2(20)	-

Interview Type:



Attribute Name	Data Type	Constraint
Int_id_type	Varchar2(10)	Primary Key
int_id_name	Varchar2(20)	Not Null

Interview:



Attribute Name	Data Type	Constraint
Int_id	Varchar2(10)	Primary Key
Remarks	Number	Check >0 AND <10
Req_id	Varchar2(10)	Foreign Key
Int_id_type	Varchar2(10)	Foreign Key
Comp_id	Varchar2(10)	Foreign Key

PHASE 2 DOCUMENTATION:

NORMALIZATION:

Job Seeker

Can_id	Can_name	DOB	Address	Email_id	Ph_no	Worked_for	Years	Off_id	Username
J1	Flynn Rider	29-4-1990	B104 Street A, California	flyn@ymail.com	12654836	Trello	3	O1	Flynn_02

J2	Jake Harper	25-5-1995	C22 Street B, New Jersey	jke@ymail.com	31177434	Accenture	4	O2	Jake_45
J3	Rodger S	29-4-1997	D23 Street A, Tawain	rodger@ymail.com	39675369	Hopper Technologies	5	O2	Rodger_09
J4	Linda martin	30-8-1990	X3 Street A, India	linm@ymail.com	34567889	Safety Travels	2	O3	Linda_DJ02

FD: Can_id→Can_name,DOB,Address,Email_id,Ph_no,Username

Email_id → Can_id,Can_name,DOB,Address,Ph_no, Username

Username → Can_id,Can_name,DOB,Address,Email_id,Ph_no

Can_id, Worked_for → Years

Email_id, Worked_for→ Years

Username,Worked_for →Years

Candidate keys: Since Off_id is not dependent on any attribute it must be present in candidate key.

(Can_id, Worked_for,Off_id)$^+$= Can_id, Worked_for, Off_id, Can_name, DOB, Address,Username,Email_id,Ph_no,Years = R

(Email_id,Worked_for,Off_id)$^+$= Can_id, Worked_for, Off_id, Can_name, DOB, Address,Username,Email_id,Ph_no,Years = R

(Username,Worked_for,Off_id)$^+$ = Can_id, Worked_for, Off_id, Can_name, DOB, Address,Username,Email_id,Ph_no,Years = R

(Ph_no,Worked_for,Off_id)= Can_id, Worked_for, Off_id, Can_name, DOB, Address,Username,Email_id,Ph_no,Years = R

Candidate keys are : { (Can_id, Worked_for,Off_id), (Email_id,Worked_for,Off_id), (Username,Worked_for,Off_id), (Ph_no,Worked_for,Off_id) }

Prime Attributes are: {Can_id,Email_id , Ph_no,

Worked_for, Off_id, Username}

Non-prime attributes are: {Can_name, years, DOB, Address}

<u>Normalisation:</u>

1. Since all the attributes of this relation are atomic. The table is in 1 Normal Form.

2. **For 2NF:**

   1. It should be in 1NF.

   2. Elimination of partial key functional dependency.

   Minimal Cover of FD's :
   can_id → Username
   Email_id→Username
   Username→can_name,
   Username→Address
   Username→ DOB
   Username → Email_id
   Username→ph_no
   (Email_id,Worked_for) → years

Now in Username→ Can_name,DOB, Address, Ph_no partial dependency is present.

So, we will decompose this in a separate table as:

Decomposition:

R1→ Username, can_name, DOB, Address,ph_no

with FD's: Username → can_name, DOB, Address,ph_no

R2→ Can_id,Email_id,Worked_for,Years,Off_id,Username

with FD's: Can_id→ Username Email_id→Username Username→Email_id,Can_id Worked_for,Username→ Years

R1 is in 2NF as it does not contain any partial dependency and it is in 1NF.

And candidate key for R2 is Username as (Username)$^+$ = Username,can_name, DOB, Address,ph_no = R1

In R2 we have (Worked_for,Username) → Years as partial dependency.

So, we will decompose the table as

R3→ Worked_for,Username,Years
with FD's: (Worked_for,Username) → Years

R4→ Can_id,Email_id, Worked_for,Off_id, Username
with FD's: Cab_id→Username, Email_id→Username, Username→Email_id,Can_id

Here both R3 and R4 do not contain any partial dependency and are in 1NF thus they are in 2NF.

Candidate key of R3 is Worked_for,Username as (Worked_for,Username)$^+$= Worked_for,Username,Years = R3

Candidate keys of R4 are (Can_id, Worked_for,Off_id), (Email_id,Worked_for,Off_id), (Username,Worked_for,Off_id) as their closure gives R4.

Final Tables are

R1→ Username, can_name, DOB, Address,ph_no
R3 → Worked_for,Username,Years

R4→ Can_id,Email_id, Worked_for,Off_id, Username

Checking lossless decomposition:

- R1 ∩ R3 = Username

And Username is the candidate key for R1. Hence, Decomposition into R1 and R3 is lossless.

- R3 ∩ R4 = Worked_for, Username

And Worked_for, Username is the candidate key for R3. Hence, Decomposition into R3 and R4 is lossless.

- R1 ∩ R4 = Username

And Username is the candidate key for R1. Hence, Decomposition into R1 and R4

is lossless.

3. **For 3NF:**

    1. It should be in 2NF.

    2. It should not contain any transitive dependency.

R1 do not contain any Transitive Dependency

Thus it is in 3NF.

R3 do not contain any Transitive Dependency.

Thus it is in 3NF.

R4 do not contain any Transitive Dependency.

Thus it is in 3NF.

4. For BCNF:

    1. It should be in 3NF.

2. LHS of each FD should be candidate key or super key.

R1→ Username, can_name, DOB, Address,ph_no

It is in BCNF as LHS is a candidate key in the
FD: Username → can_name, DOB, Address,ph_no

R3 → Worked_for,Username,Years

It is in BCNF as LHS is a candidate key in the
FD: Worked_for,Username → Years

R4 → Can_id,Email_id, Worked_for,Off_id, Username

with FD's: Cab_id→Username, Email_id→Username, Username→Email_id,Can_id

It is not in BCNF as LHS do not contain super keys.

So decompose the table into

Decomposition:

R5 → Worked_for , Username, Off_id

all the three attributes together forms a candidate key.

Therefore it is in BCNF.

R6→Username, Email_id, Can_id with FD's: Username→ Can_id, (Email_id Can_id)→ Email_id, (Username,Email_id) → Username,Can_id

Candidate keys are Username , Can_id, Email_id as

$(\text{Username})^+$ =Username,Can_id,Email_id

$(\text{Can\_id})^+$ =Username,Can_id,Email_id

$(\text{Email\_id})^+$ =Username,Can_id,Email_id

Since every dependency has LHS as a candidate key it is in BCNF.

Final Tables are: R1→ Username, can_name, DOB, Address,ph_no
R3 → Worked_for,Username,Years

R5 → Worked_for,Off_id,Username

R6→Username,Email_id,Can_id

Interview_Type

Int_type_id	Int_type_name	Int_id
IT1	Technical	I1
IT2	HR	I1
IT2	HR	I3

IT1	Technical	I4

FD: Int_type_id → Int_type_name

Int_type_name → Int_type_id

Normalization :

1. Candidate keys: {(Int_id Int_type_id), (Int_id int_type_name) }
2. It's already in 1 NF, 2 NF, 3 NF
3. For BCNF decompose into two tables: Int_type_id ,Int_id and Int_type_id ,Int_type_name

(Int_type_id, Int_id) and (int_type_id and int_type_name)

Interview

Int_id	Comp_id	Can_id	Remarks	Date	Time	Venue
I1	C1	J1	Good	28-Jun-2020	9: 00	MB
I2	C1	J2	Excellent	28-Jun-2020	8: 00	SJT

I3	C2	J3	NI	1-Jul-2020	9: 00	SJT
I3	C3	J4	Good	4-Jul-2020	9: 30	SJT

FD: Int_id → Comp_id , Remarks , Can_id , Remarks , Time , Venue

Comp_id , Can_id → Int_id , Remarks ,Remarks , Time , Venue

Remarks ,Time , → Int_id , Comp_id ,Can_id ,Remarks

Normalization :

1. Candidate keys: {Int_id , (Comp_id Can_id) , (Remarks Time Venue**)**}
2. It's already in 1 NF, 2 NF, 3 NF and BCNF
3. Hence final table is: Interview(Int_id, Comp_id, Can_id, Remarks, Date, Time, Venue)

With minimal functional dependencies:

Int_Id → Date, Time, Venue

Comp_id Can_id → Date, Time, Venue

Date Time Venue → Int_id, Comp_id, Can_id, Remarks

Result

Result_id	Post	Salary	Can_id	Int_id	Recruitement_status
RID1	Developer	2000000pa	J1	I1	Waiting
RID2	Product Manager	2000000pa	J2	I2	Recruited
RID3	Product Designer	-	J3	I3	Rejected
RID4	Consultant	1000000pa	J4	I4	Recruited

FD: Result_id → Post , Salary , Can_id , Int_id , Recruitement_status

Int_id → Result_id, Post , Salary , Recruitement_status , Can_id

Can_id → Result_id Post , Salary

Normalization :

1. Candidate keys: Result_id ,Can_id , Int_id

2. It's already in 1 NF, 2 NF, 3 NF and BCNF

3. Hence final table is: Skill(Result_id, Post, Salary , Can_id, Int_id, Recruitement_status) where the functional dependency is -

Result_id → Post, Salary , Can_id, Int_id, Recruitement_status

Int_id → Result_id, Post, Salary , Can_id , Recruitement_status

Can_id → Result_id, Post, Salary

Posts

Comp_Id	**Post**
C1	Consultant
C1	Developer
C2	Consultant
C2	Manager

No Functional Dependency hence the only candidate key is (Comp_Id, Post).

Thus it is 1NF, 2NF, 3NF and BCNF.

Skill

Can_id	Qualification	Co_Curricular	Technical
J1	M S	Design	C1
J2	M Tech	Creative Writing	C2
J3	B Tech	Societies	C3
J4	Ph. D	Sports	C4

FD: Can_id → Qualification ,Co_Curricular ,Technical

Normalization :

4. Candidate keys: Can_id
5. It's already in 1 NF, 2 NF, 3 NF and BCNF
6. Hence final table is: Skill(Can_id, Qualification, Co_Curricular, Technical) where the functional dependency is -

   Can_id → Qualification, Co_Curricular, Technical

Requisition

Req_id	Package	Criteria	Comp_id	Can_id
R1	600000	2	C1	J1
R2	700000	2	C2	J2
R3	900000	1	C3	J3
R4	500000	1	C4	J4

FD: Req_id → Package **,** Comp_id **,** Criteria **,** Can_id

Comp_id **,** Can_id → Req_id **,** Package **,** Criteria

Normalization :

1. Candidate keys: Req_id , Comp_id, Can_id
2. It's already in 1 NF, 2 NF, 3 NF and BCNF
3. Hence final table is: Requisition(Req_id, Package, Criteria, Comp_id, Can_id) where the functional dependency is -

   Req_id → Package, Criteria, Comp_id, Can_id

   Comp_id, Can_Id → Req_id, Package, Criteria

User

Username	Password	Login_time	Logout_time
Flynn	Hfsjcavakl	22:00 UTC	23:00 UTC
Jake	Vahjdjac	20:00 IST	21:00 IST
Rodger	Hwgyufe	21:00 UTC	23:00 UTC
Linda	Ahlbuciac	19:00 IST	20:00 IST

FD: Username → Password ,LogIn_time, LogOUT_time

Normalization :

1. Candidate keys: **Username**
2. It's already in 1 NF, 2 NF, 3 NF and BCNF
3. Hence final table is: User(Password, Username, LogIn_time, LogOUT_time)

where the functional dependency is –

Username → Password LogIn_time, LogOUT_time

Offer Letter

Off_id	Comp_id	Can_Id	Date	Incharge
O1	C1	J1	2 – 3 – 2001	J . Murugan
O2	C1	J2	12 – 4 – 2001	J . Murugan
O3	C2	J3	17 – 5 – 2001	Dev Mehta
O4	C3	J2	2 – 3 – 2001	Riya S

FD: Off_id → Comp_id, Can_Id , Date , Incharge

Comp_id, Can_Id → Off_id , Date , Incharge

Normalization :

1. Candidate keys: Off_id , Comp_id, Can_Id
2. It's already in 1 NF, 2 NF, 3 NF and BCNF
3. Hence final table is: Offer Letter(Off_id, Comp_id, Can_Id, Date, Incharge) where the functional dependency is -

Off_id → Comp_id, Can_Id, Date, Incharge

Comp_id, Can_Id → Off_id, Date, Incharge

Company

Comp_id	Comp_name	Email_id	Comp_Type	Vacancy	Username
C1	Campp	campp@ymail.com	Travel	3	Campp012
C2	Trello	trello@ymail.com	Software	2	Trello56
C3	Embibe	embibe@ymail.com	Education	2	Embibie09
C4	TheTribe	thetribe@ymail.com	Software	1	TheTribe34

FD : Comp_id → Comp_name , Email_id , Comp_Type , Vacancy, Username

Email_id → Comp_id ,Comp_name ,Comp_Type ,Vacancy, Username

Username → Comp_name , Email_id , Comp_Type , Vacancy,Comp_id

Comp_name Comp_Type → Comp_id ,Email_id ,Vacancy, Username

Normalization :

1. Candidate keys: {Comp_id, Email_id , Username, (Comp_name Comp_Type) }
2. It's already in 1 NF, 2 NF, 3 NF and BCNF

Hence final table is: Company (Comp_id , Comp_name, Email_id, Comp_Type, Vacany, Username)

With functional dependencies-

Comp_Id → Username

Username→Email_Id

Comp_name Comp_Type→ Username

Email_id → Comp_id , Comp_name, Comp_Type, Vacancy

Total number of Tables in the final schema: 14

FINAL SCHEMAS:

1. Company (Comp_id , Comp_name, Email_id,Comp_Type,Vacany, Username)
2. Offer Letter (Off_Id, Comp_Id, Can_Id, Date, Incharge)
3. User(Username, Password, Login_Time, Logout_Time)
4. R1JobSeeker(Can_id, DOB, Address, ph_no)
5. R3JobSeeker(Worked_for, Years, Off_id)

6. R5JobSeeker(Username, Worked_for, Off_id)
7. R6JobSeeker(Username, Email_id, Can_id)
8. Requistion (Package , Criteria, Req_Id, Comp_Id,Can_Id)
9. Skill (Can_Id, Qualification, Co_Curricular, Technical)
10. Interview (int_Id, Comp_Id, Remarks, Date, Time, Venue, Can_Id)
11. R1Interview_type(Int_type_id, int_id)
12. R2Interview_type(int_type_id, int_type_name)
13. Result(R_Id,Recruitment_status, Post, Salary, Can_Id, int_Id)
14. Posts(Comp_Id, Post)

PHASE 3 DOCUMENTATION:

HARDWARE/SOFTWARE REQUIREMENTS:

Hardware requirements:

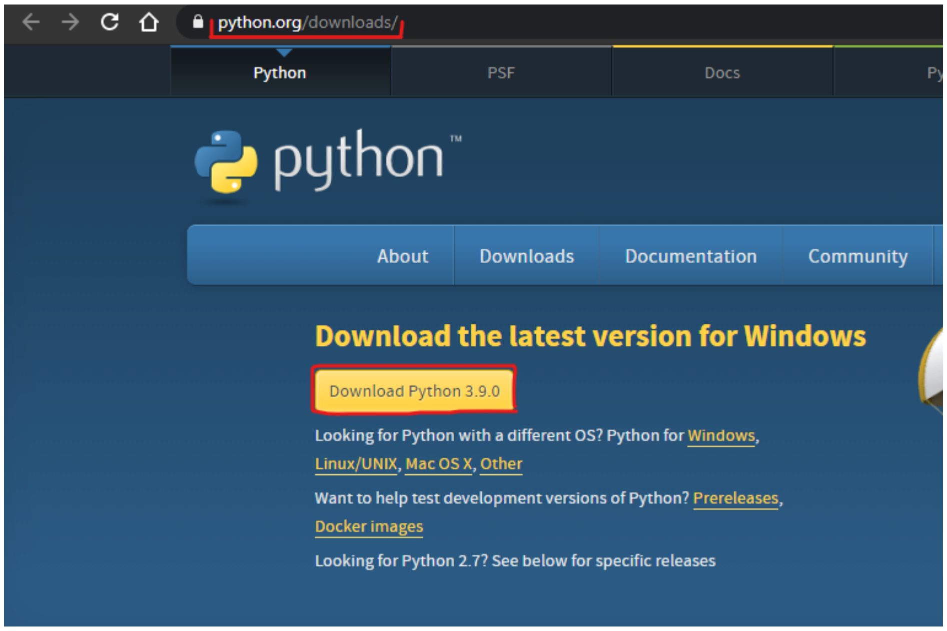
1. 2 GB RAM
2. 1.6 GHz CPU
3. Space: 45 MB

Software requirements:

1. Python 3.6+
    - Flask (Back-end)
    - SQLAlchemy (Database connection)
2. HTML/CSS

3. Modern Web Browser
4. Database used: SQLite

HELP FILE:

1. Unzip the Application File.
2. Install Python(3.6+).
   - https://www.python.org/downloads/
   - 

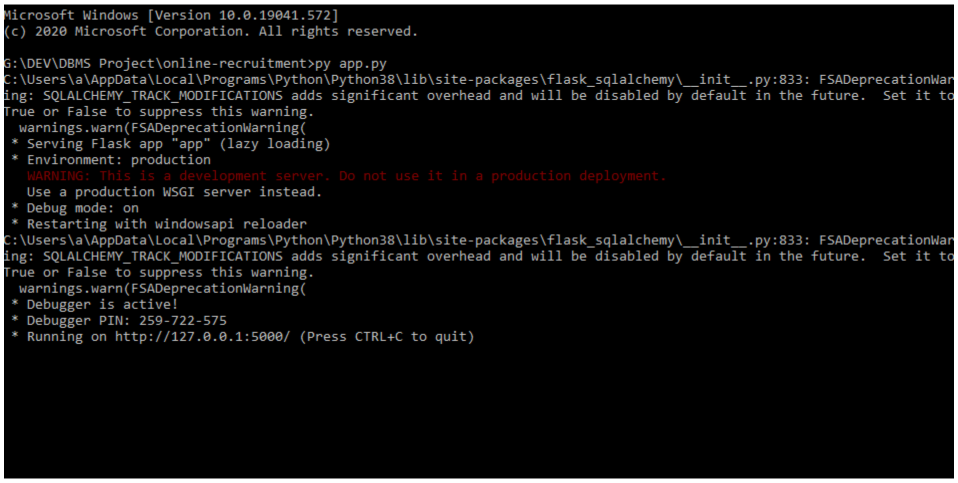
     Download the latest version for Windows

     Looking for Python with a different OS? Python for Windows, Linux/UNIX, Mac OS X, Other

     Want to help test development versions of Python? Prereleases, Docker images

     Looking for Python 2.7? See below for specific releases
3. Install pip to install python the packages.
   - `python -m pip install -U pip`
4. Create a virtual environment for the Folder using Command Line and install the required packages.
   - `pip install virtualenv`
   - `virtualenv env`
   - `env\Scripts\activate`
   - `pip install -r requirements.txt`

5. Run the app using the following command
   - `python3 app.py`

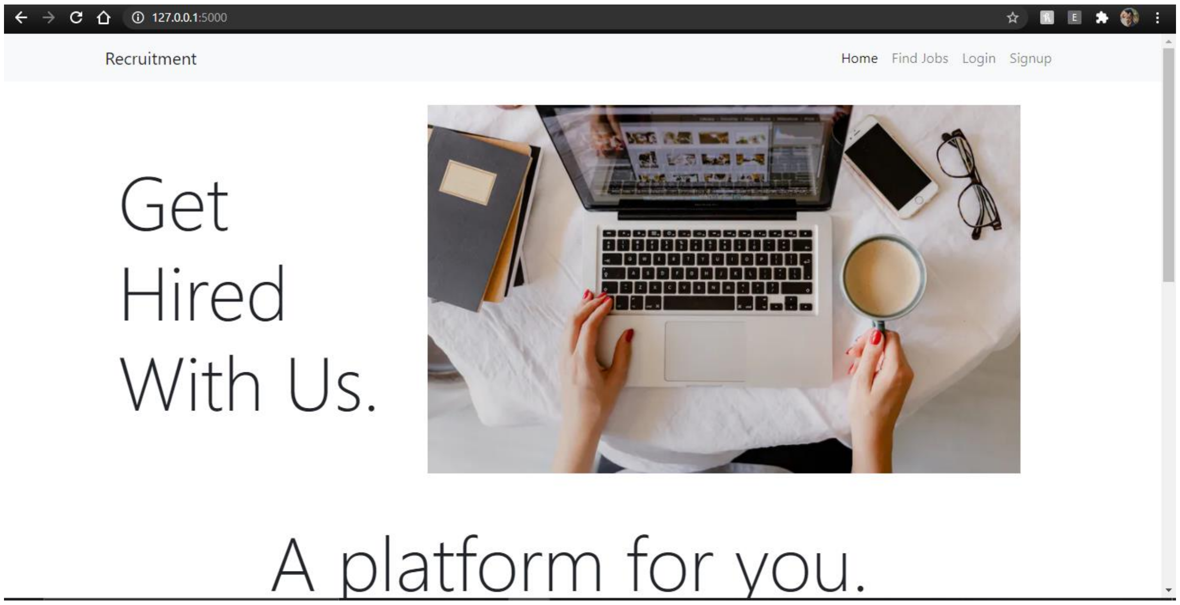
6. Open your browser window, and enter fire up the local development server at http://localhost:5000 (alternatively http://127.0.0.1:5000)
   - 

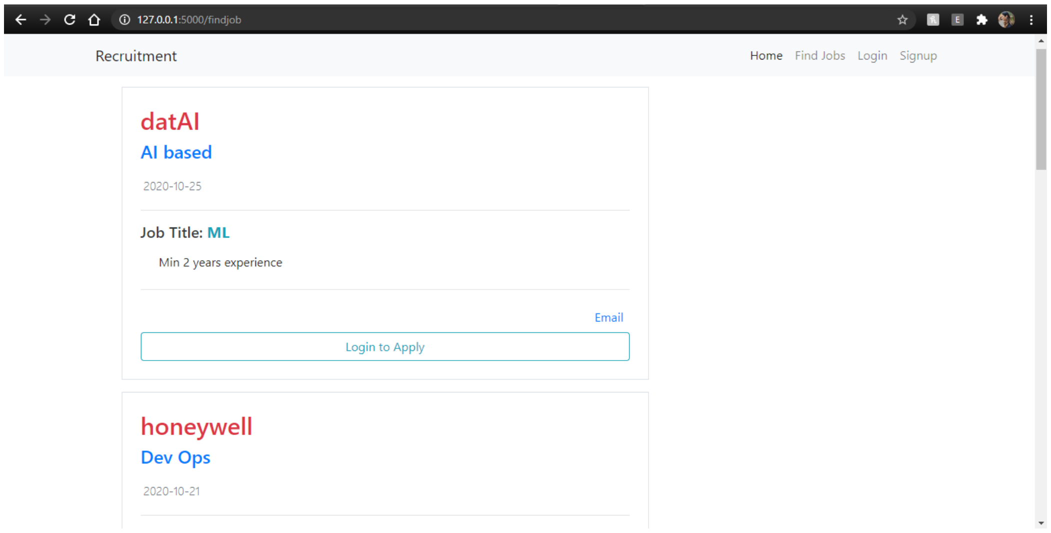
```
Microsoft Windows [Version 10.0.19041.572]
(c) 2020 Microsoft Corporation. All rights reserved.

G:\DEV\DBMS Project\online-recruitment>py app.py
C:\Users\a\AppData\Local\Programs\Python\Python38\lib\site-packages\flask_sqlalchemy\__init__.py:833: FSADeprecationWar
ing: SQLALCHEMY_TRACK_MODIFICATIONS adds significant overhead and will be disabled by default in the future.  Set it to
True or False to suppress this warning.
  warnings.warn(FSADeprecationWarning(
 * Serving Flask app "app" (lazy loading)
 * Environment: production
   WARNING: This is a development server. Do not use it in a production deployment.
   Use a production WSGI server instead.
 * Debug mode: on
 * Restarting with windowsapi reloader
C:\Users\a\AppData\Local\Programs\Python\Python38\lib\site-packages\flask_sqlalchemy\__init__.py:833: FSADeprecationWar
ing: SQLALCHEMY_TRACK_MODIFICATIONS adds significant overhead and will be disabled by default in the future.  Set it to
True or False to suppress this warning.
  warnings.warn(FSADeprecationWarning(
 * Debugger is active!
 * Debugger PIN: 259-722-575
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

7. The application starts at port 5000, now you can use the app in your browser.

Front-end Implementation:

For Front-End Implementation we have made a Home Page which has been designed using **HTML,CSS and Bootstrap.** It has three functions: Find Jobs, Login or Sign up. Using Find Jobs, we can browse through job posts by companies.

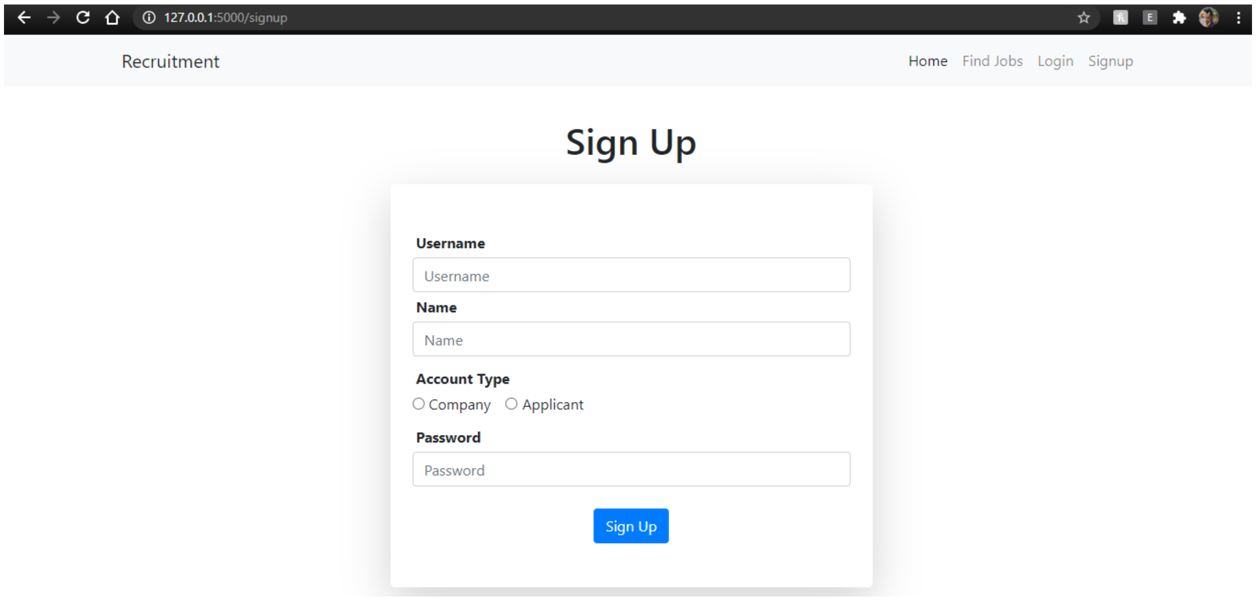




datAI
AI based
2020-10-25
Job Title: ML
Min 2 years experience

honeywell
Dev Ops
2020-10-21

In the Sign up Page you can sign up as an applicant or a company.


127.0.0.1:5000/signup

Recruitment

Home Find Jobs Login Signup

Sign Up

Username

Name

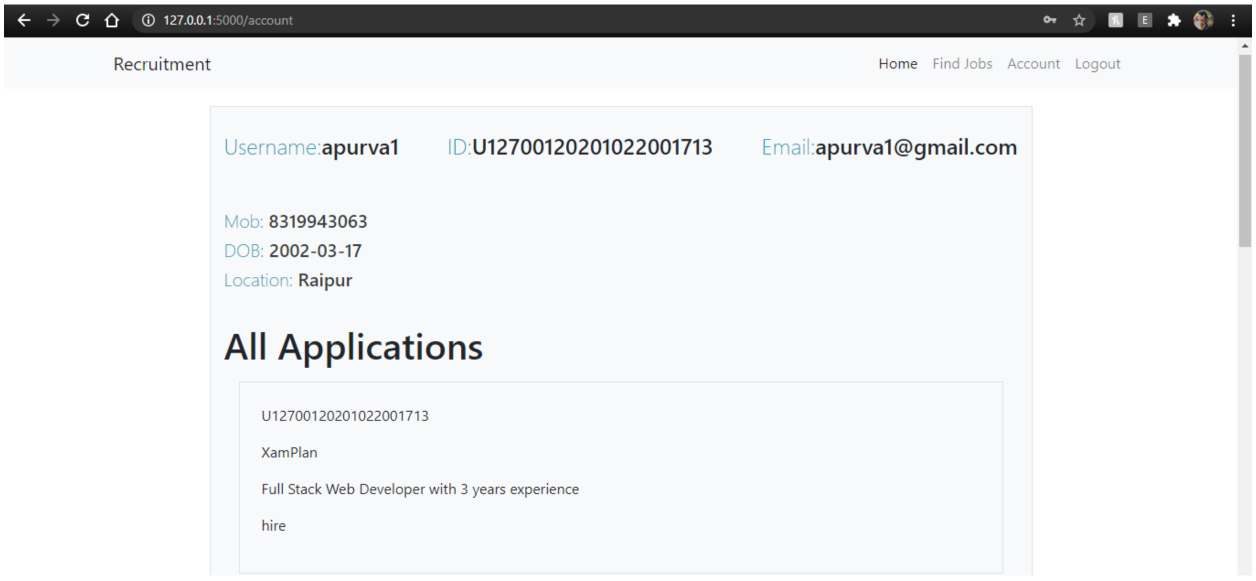
Account Type

Company Applicant

Password

Sign Up

THE ACCOUNT PAGE DISPLAYS DETAILS OF THE USER.


127.0.0.1:5000/account

Recruitment

Home Find Jobs Account Logout

Username:apurva1 ID:U12700120201022001713 Email:apurva1@gmail.com

Mob: 8319943063

DOB: 2002-03-17

Location: Raipur

All Applications

U12700120201022001713

XamPlan

Full Stack Web Developer with 3 years experience

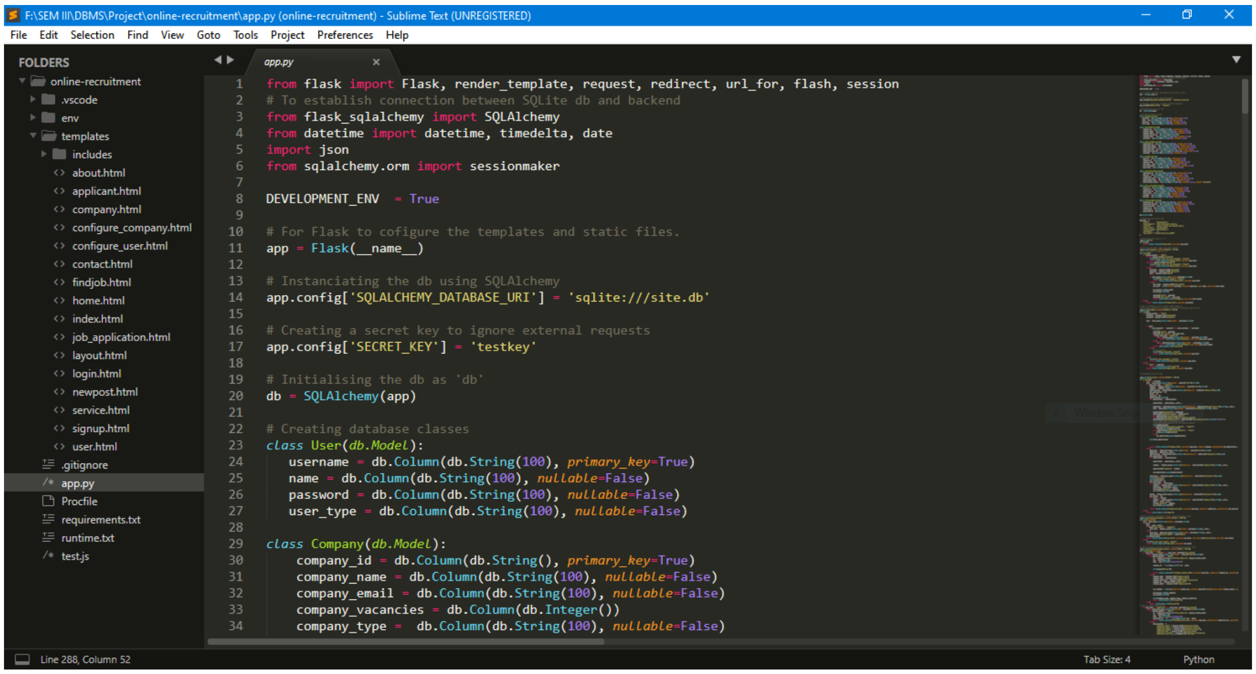
hire

BACK-END IMPLEMENTATION:

Backend implemented using **Flask** microframework for routing, backend logic, business logic, etc.

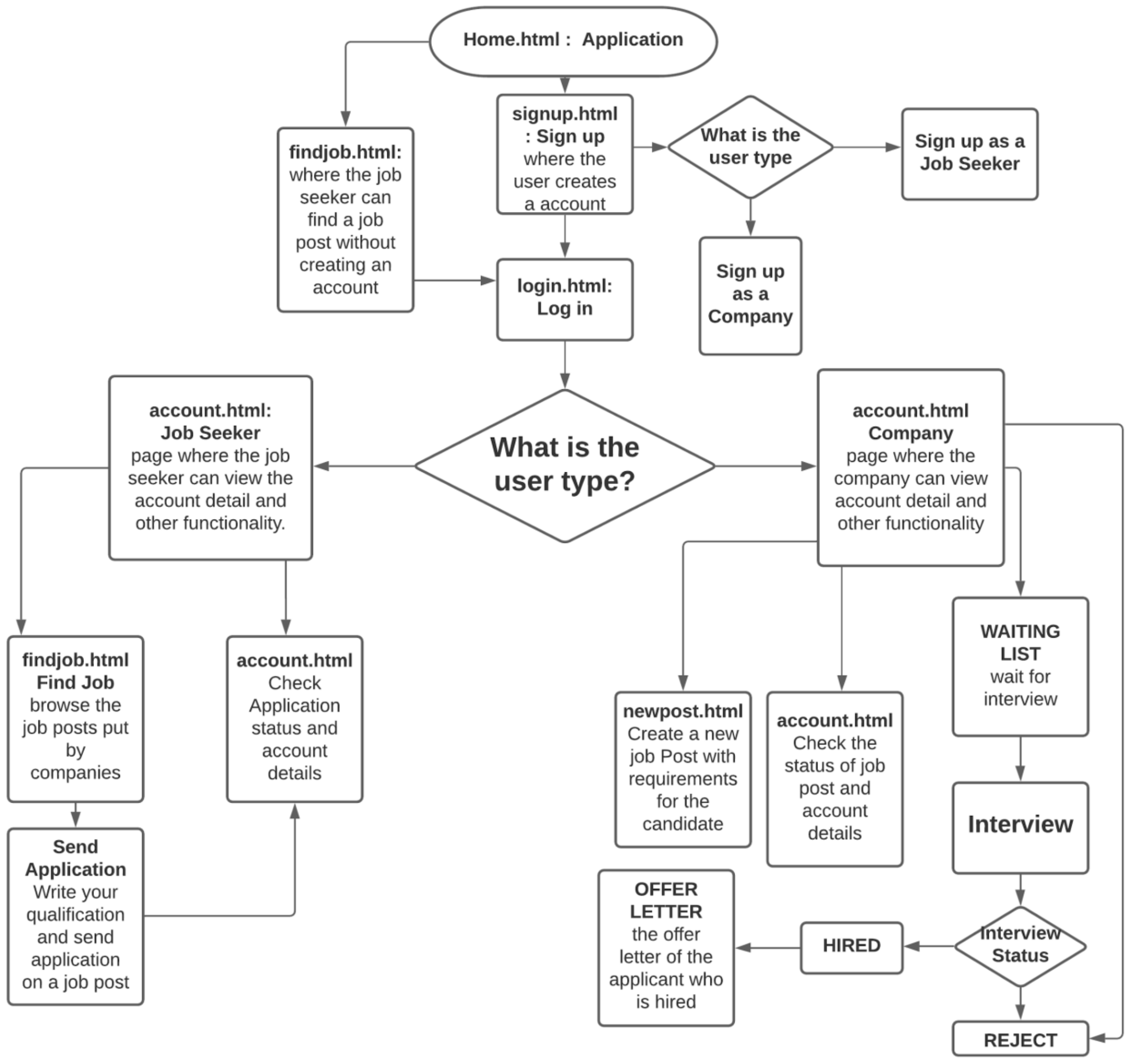
Database is implemented using **SQLAlchemy ORM** (Object Relational Mapping model) for DDL, DML, and querying data.

HTML pages use **Jinja2** Templating engine to display content dynamically which is sent through the database via the backend.



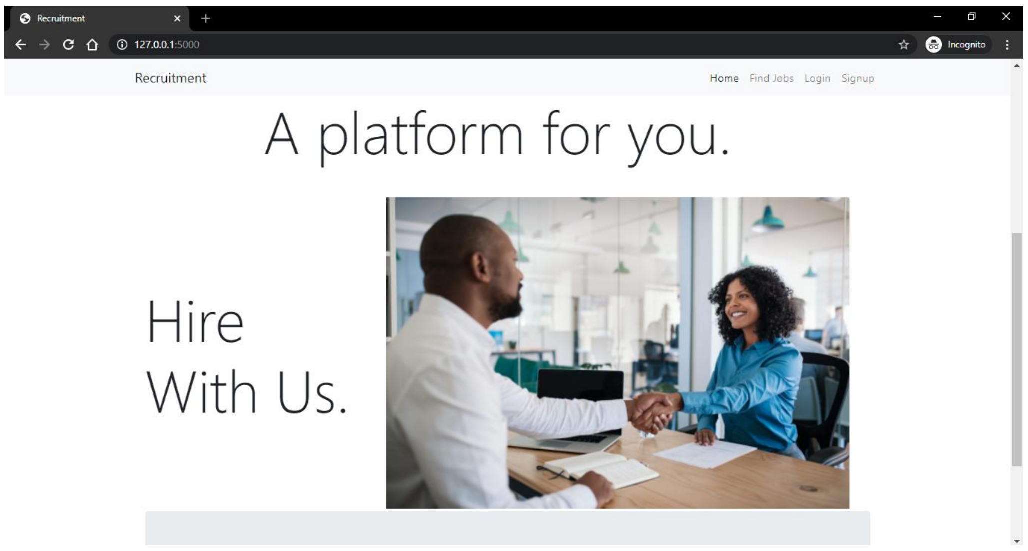
```python
from flask import Flask, render_template, request, redirect, url_for, flash, session
# To establish connection between SQLite db and backend
from flask_sqlalchemy import SQLAlchemy
from datetime import datetime, timedelta, date
import json
from sqlalchemy.orm import sessionmaker

DEVELOPMENT_ENV  = True

# For Flask to cofigure the templates and static files.
app = Flask(__name__)

# Instanciating the db using SQLAlchemy
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///site.db'

# Creating a secret key to ignore external requests
app.config['SECRET_KEY'] = 'testkey'

# Initialising the db as 'db'
db = SQLAlchemy(app)

# Creating database classes
class User(db.Model):
   username = db.Column(db.String(100), primary_key=True)
   name = db.Column(db.String(100), nullable=False)
   password = db.Column(db.String(100), nullable=False)
   user_type = db.Column(db.String(100), nullable=False)

class Company(db.Model):
    company_id = db.Column(db.String(), primary_key=True)
    company_name = db.Column(db.String(100), nullable=False)
    company_email = db.Column(db.String(100), nullable=False)
    company_vacancies = db.Column(db.Integer())
    company_type =  db.Column(db.String(100), nullable=False)
```

FLOW OF CONTROL WITH EXPLANATION OF THE PURPOSE OF EACH INTERFACE PAGE/FORM IN DETAIL:

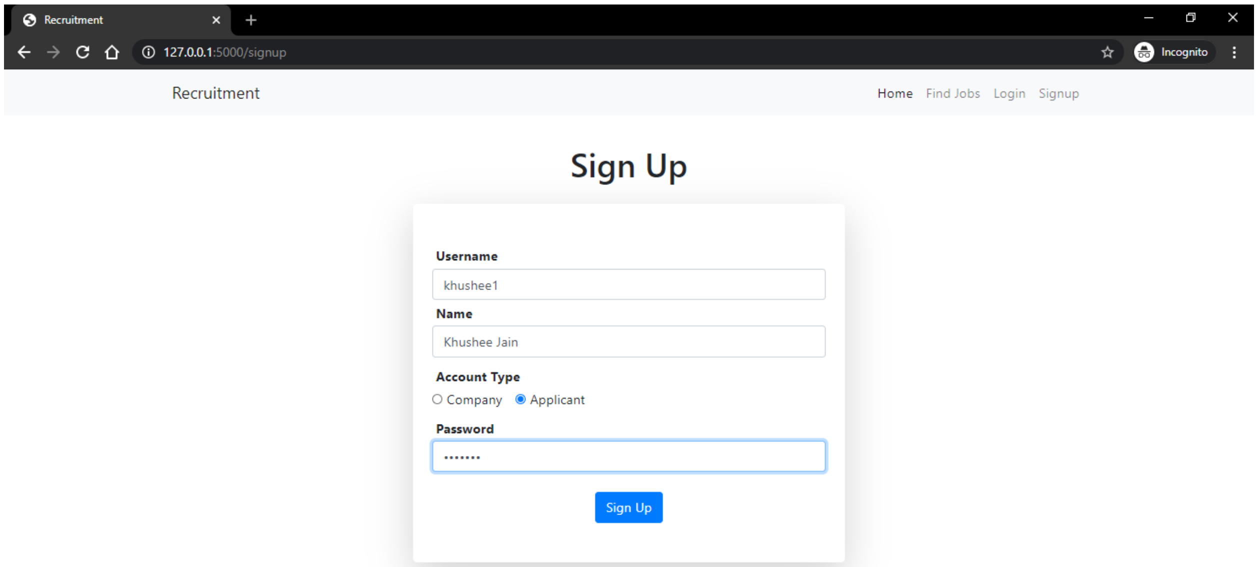

Home.html : Application
findjob.html: where the job seeker can find a job post without creating an account
signup.html : Sign up where the user creates a account
What is the user type
Sign up as a Job Seeker
Sign up as a Company
login.html: Log in
What is the user type?
account.html: Job Seeker page where the job seeker can view the account detail and other functionality.
account.html Company page where the company can view account detail and other functionality
findjob.html Find Job browse the job posts put by companies
account.html Check Application status and account details
Send Application Write your qualification and send application on a job post
newpost.html Create a new job Post with requirements for the candidate
account.html Check the status of job post and account details
WAITING LIST wait for interview
Interview
Interview Status
HIRED
OFFER LETTER the offer letter of the applicant who is hired
REJECT

SCREENSHOTS OF WORKING PROJECT:

1. Home Page.



2. **Sign up Page** with Applicant as user type.



Recruitment

Home Find Jobs Login Signup

Sign Up

Username

khushee1

Name

Khushee Jain

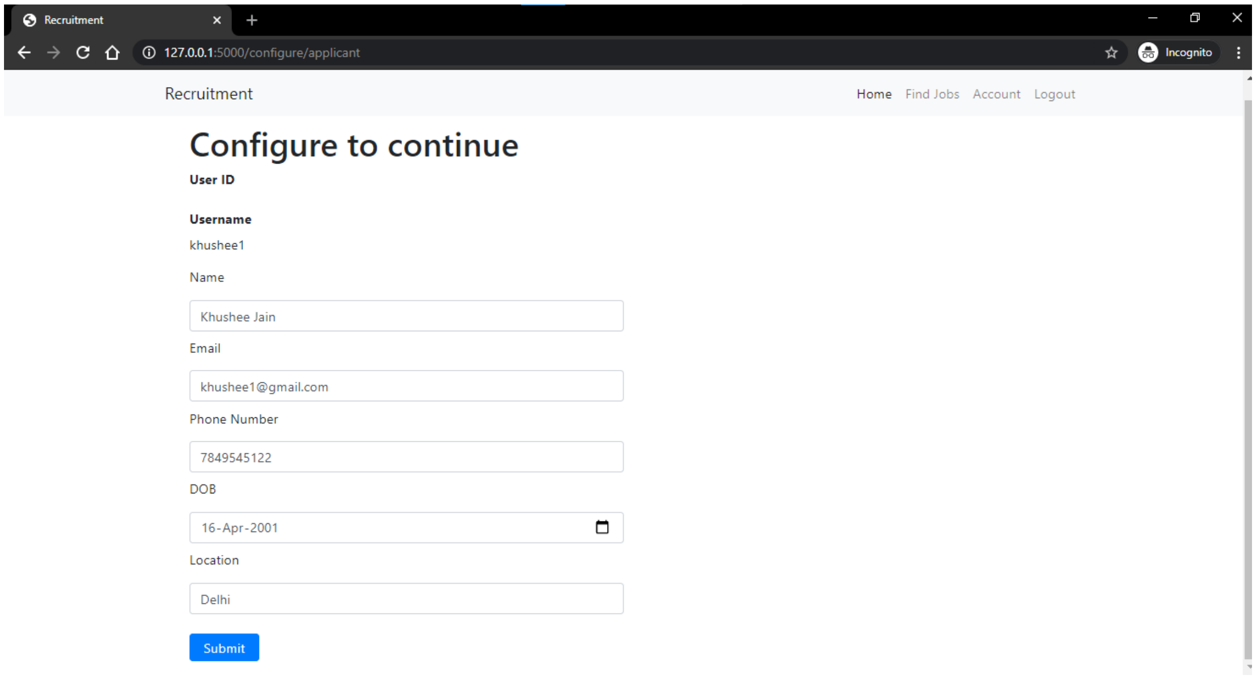
Account Type

Company [✓ Applicant]

Password

•••••••

3. Configure the applicant details.



Recruitment

Home Find Jobs Account Logout

Configure to continue

User ID

Username

khushee1

Name

Khushee Jain

Email

khushee1@gmail.com

Phone Number

7849545122

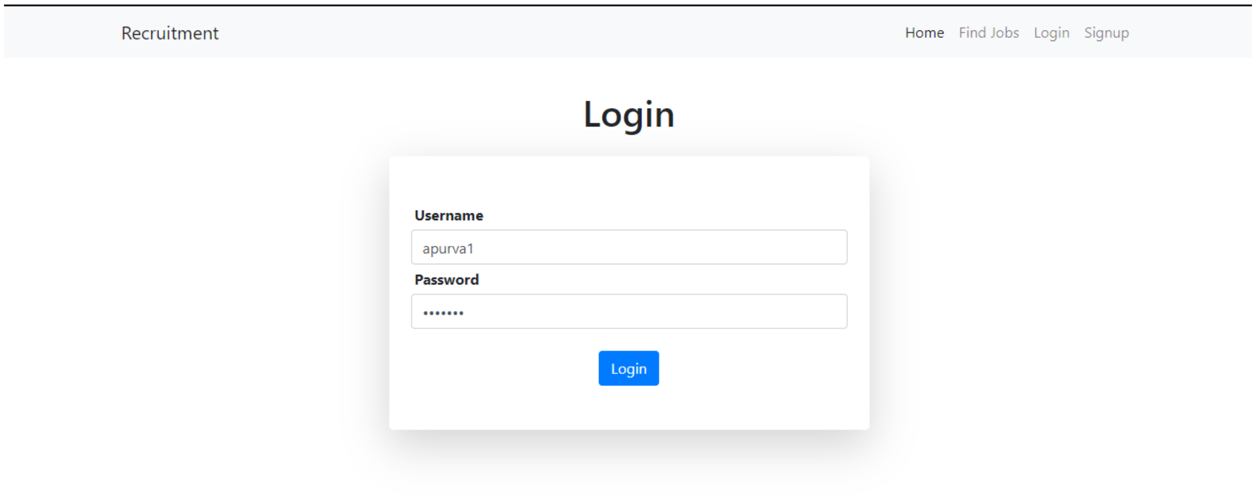
DOB

16-Apr-2001

Location

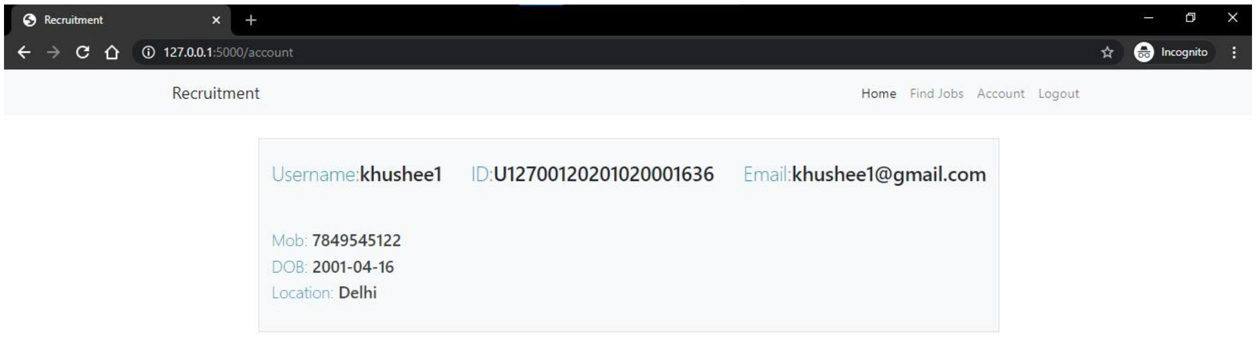
Delhi

4. Login in as either an applicant or a company.



Username
apurva1
Password
•••••••

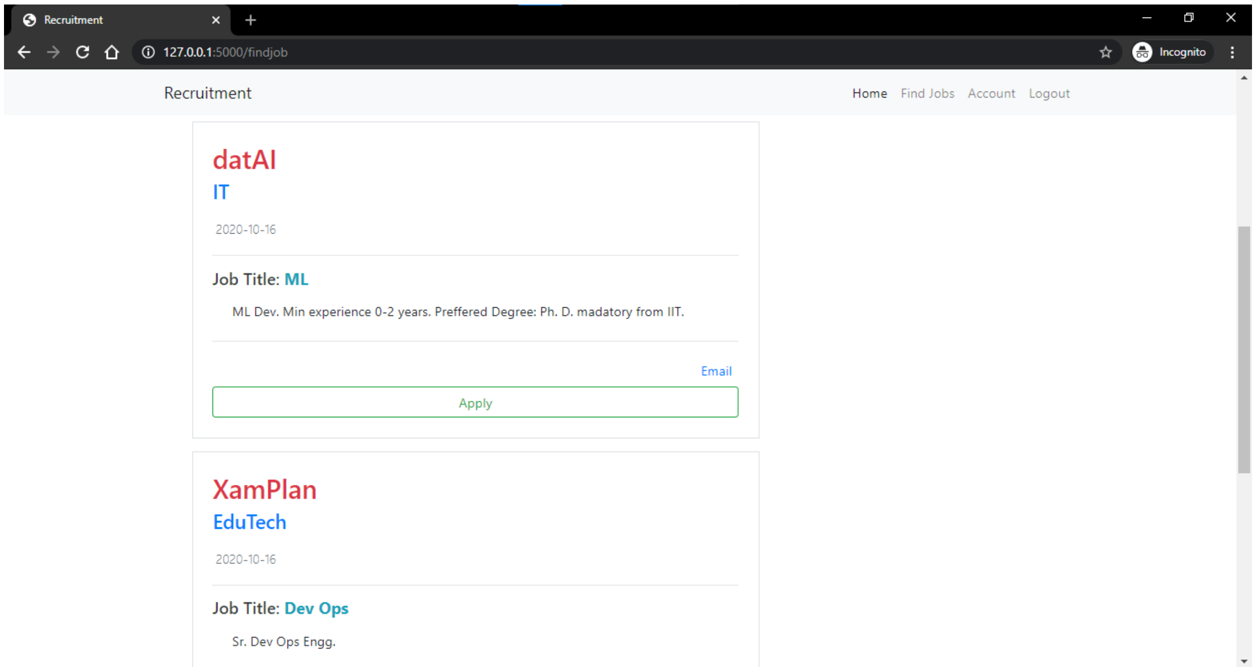
5. Account details of the Applicant.



Username:khushee1 ID:U12700120201020001636 Email:khushee1@gmail.com

Mob: 7849545122
DOB: 2001-04-16
Location: Delhi

6. Find Jobs and Send Applications to the recruiting company.



Recruitment

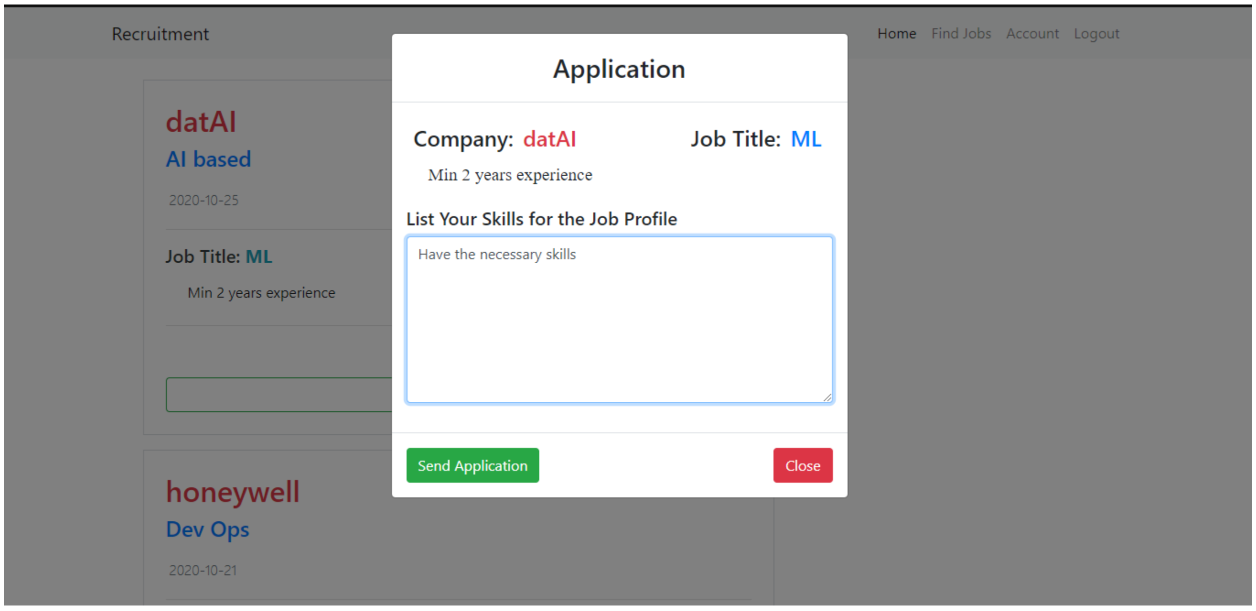
127.0.0.1:5000/findjob

Recruitment · Home · Find Jobs · Account · Logout

datAI
IT
2020-10-16
Job Title: ML
ML Dev. Min experience 0-2 years. Preffered Degree: Ph. D. madatory from IIT.
Email
Apply

XamPlan
EduTech
2020-10-16
Job Title: Dev Ops
Sr. Dev Ops Engg.

7. List your skills and create an application.



Recruitment · Home · Find Jobs · Account · Logout

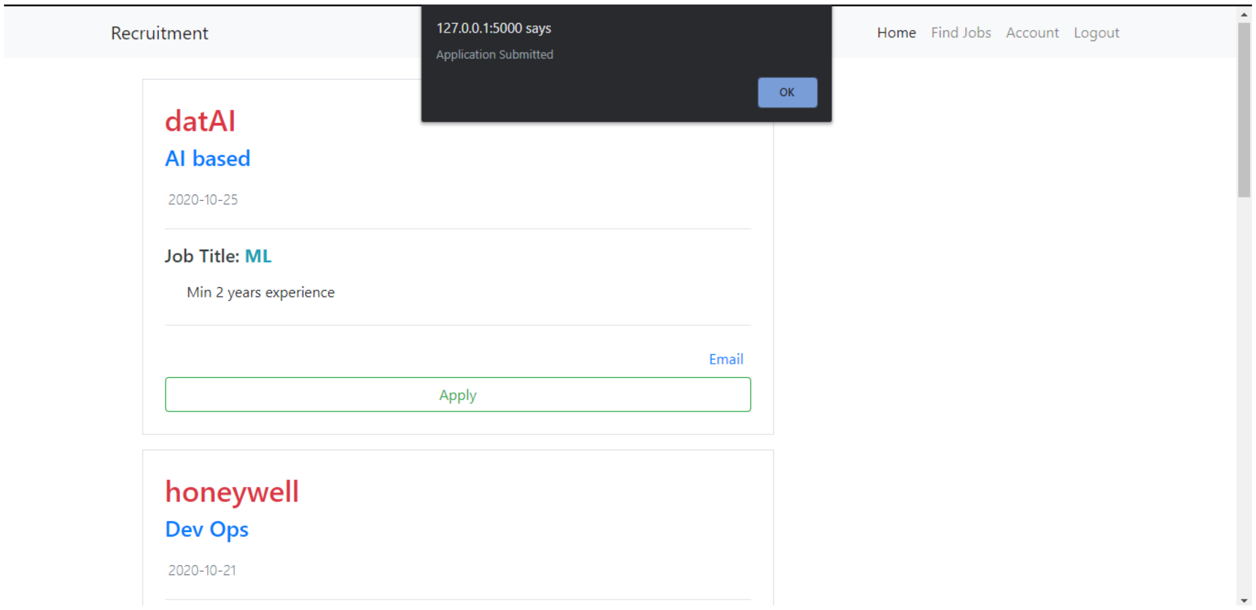
Application

Company: datAI · Job Title: ML
Min 2 years experience
List Your Skills for the Job Profile
Have the necessary skills
Send Application · Close

datAI
AI based
2020-10-25
Job Title: ML
Min 2 years experience

honeywell
Dev Ops
2020-10-21

8. Prompt message after sending the application.



127.0.0.1:5000 says

Application Submitted

datAI

AI based

2020-10-25

Job Title: ML

Min 2 years experience

Email

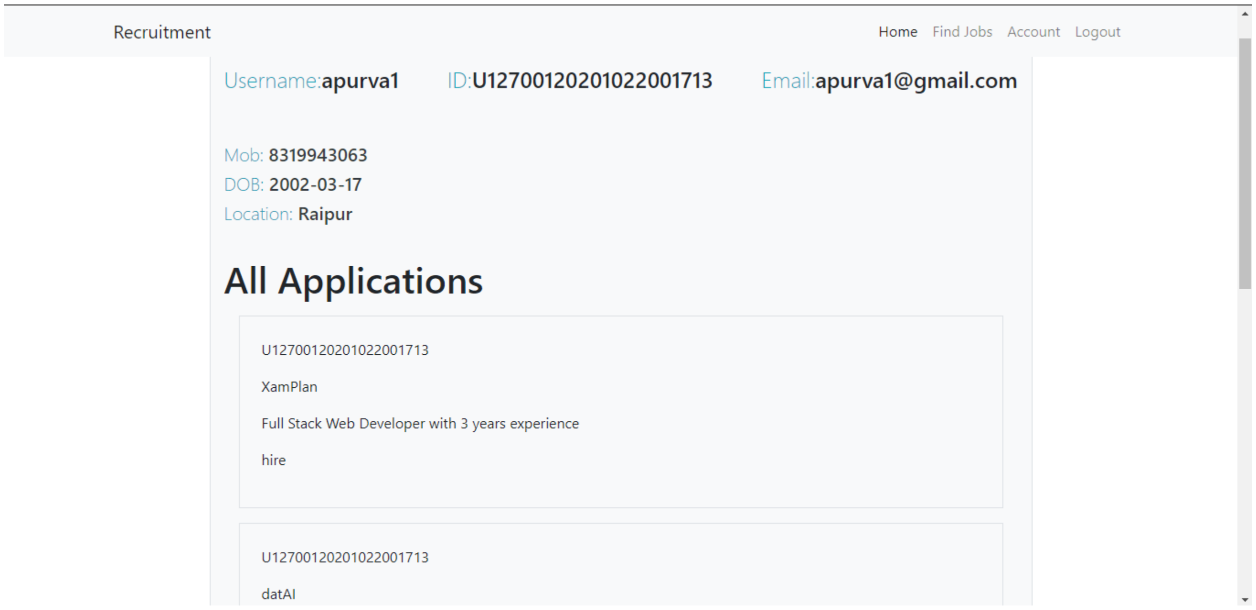
Apply

honeywell

Dev Ops

2020-10-21

9. Check the status of all application (here: hired)



Username:apurva1 ID:U12700120201022001713 Email:apurva1@gmail.com

Mob: 8319943063
DOB: 2002-03-17
Location: Raipur

All Applications

U12700120201022001713

XamPlan

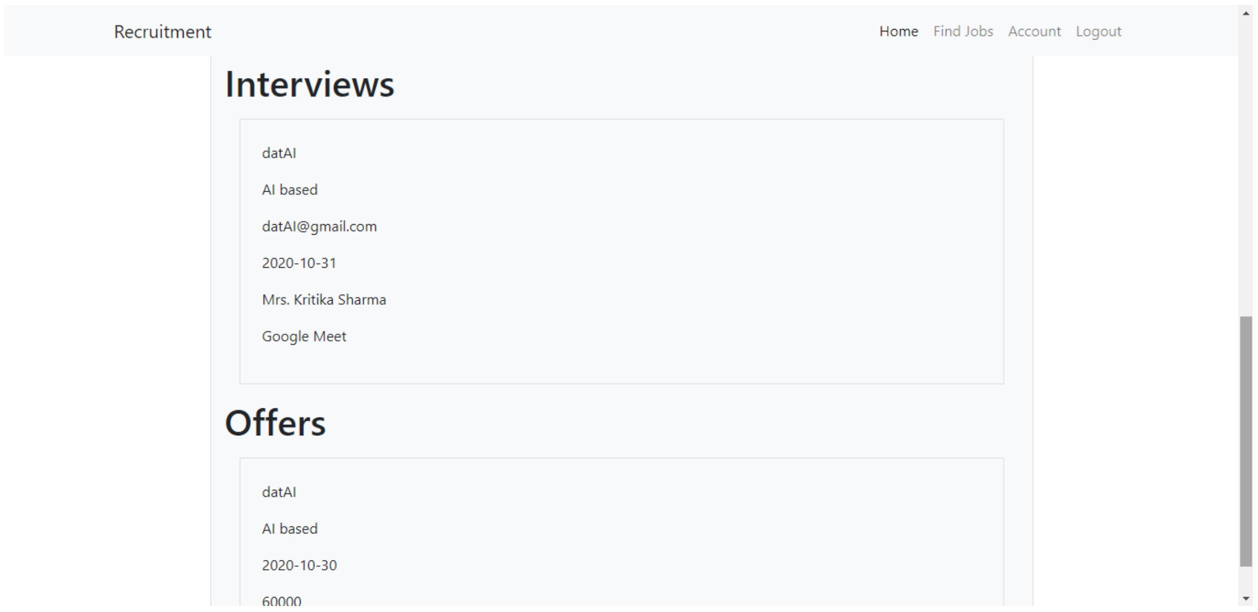
Full Stack Web Developer with 3 years experience

hire

U12700120201022001713

datAI

10. Check all your scheduled interviews and offers from the company


Recruitment

Home Find Jobs Account Logout

Interviews

datAI

AI based

datAI@gmail.com

2020-10-31

Mrs. Kritika Sharma

Google Meet

Offers

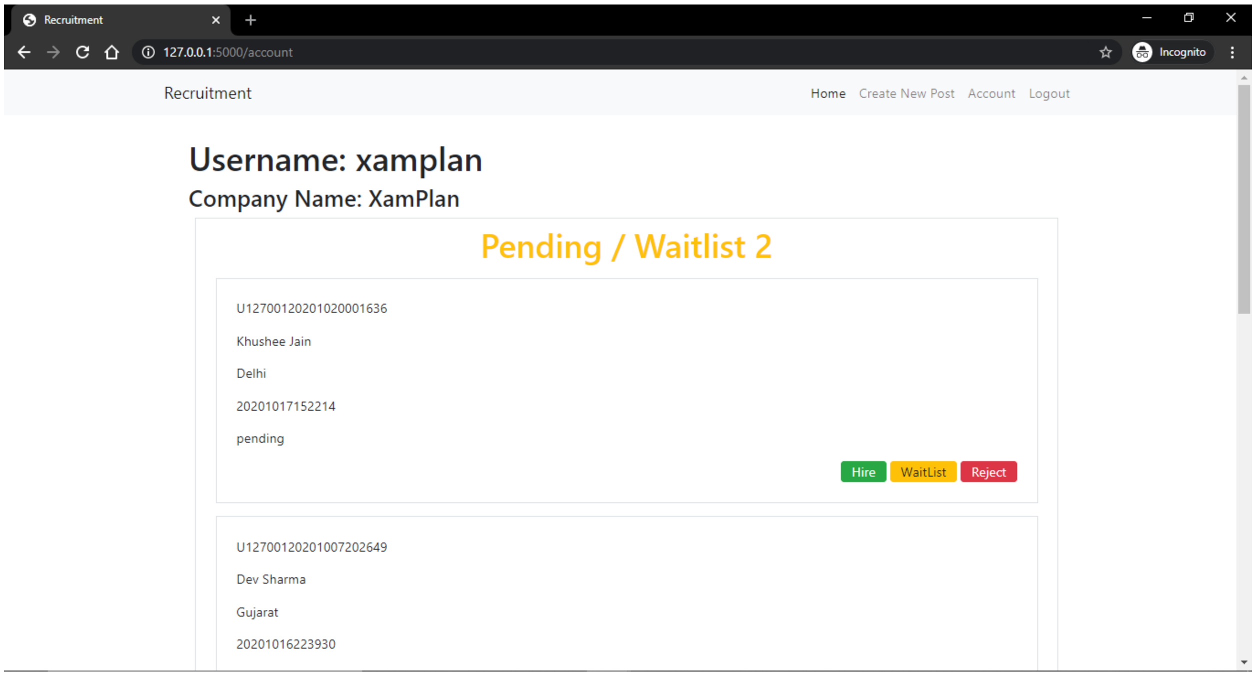
datAI

AI based

2020-10-30

60000

11. Add the applicant to the waiting list/ reject or hire him/her. This interface is for **Company.**


Recruitment

127.0.0.1:5000/account

Recruitment

Home Create New Post Account Logout

Username: xamplan

Company Name: XamPlan

Pending / Waitlist 2

U12700120201020001636

Khushee Jain

Delhi

20201017152214

pending

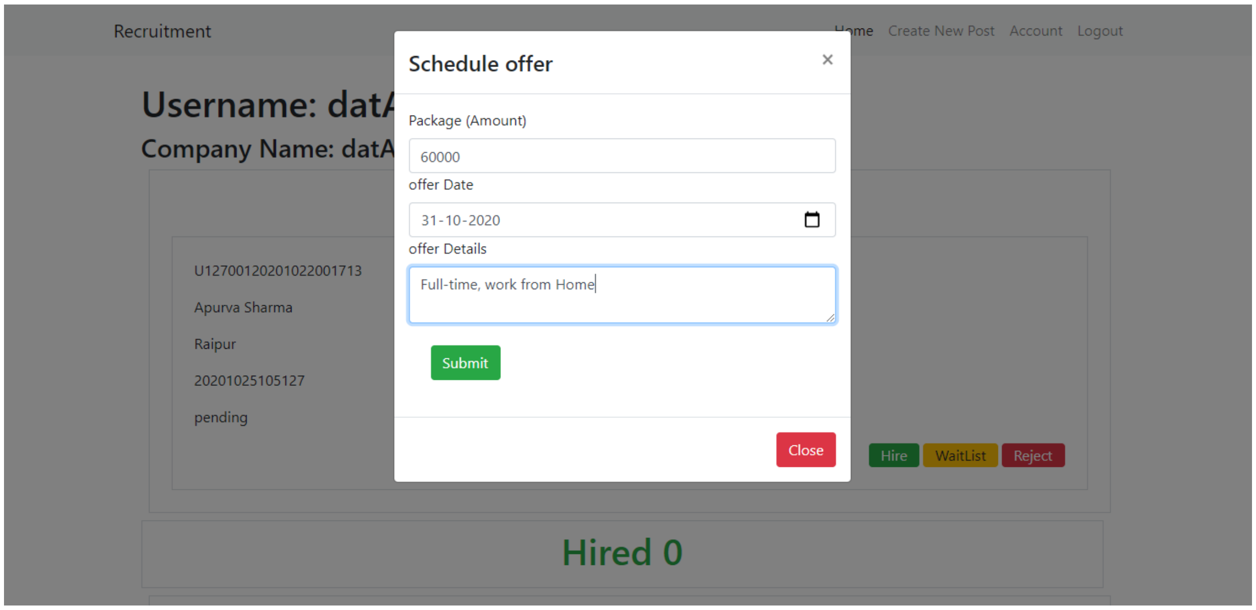
U12700120201007202649

Dev Sharma

Gujarat

20201016223930

12. Add the offer letter when you hire an applicant


Recruitment

Home Create New Post Account Logout

Username: datA

Company Name: datA

U12700120201022001713

Apurva Sharma

Raipur

20201025105127

pending

Schedule offer

Package (Amount)

60000

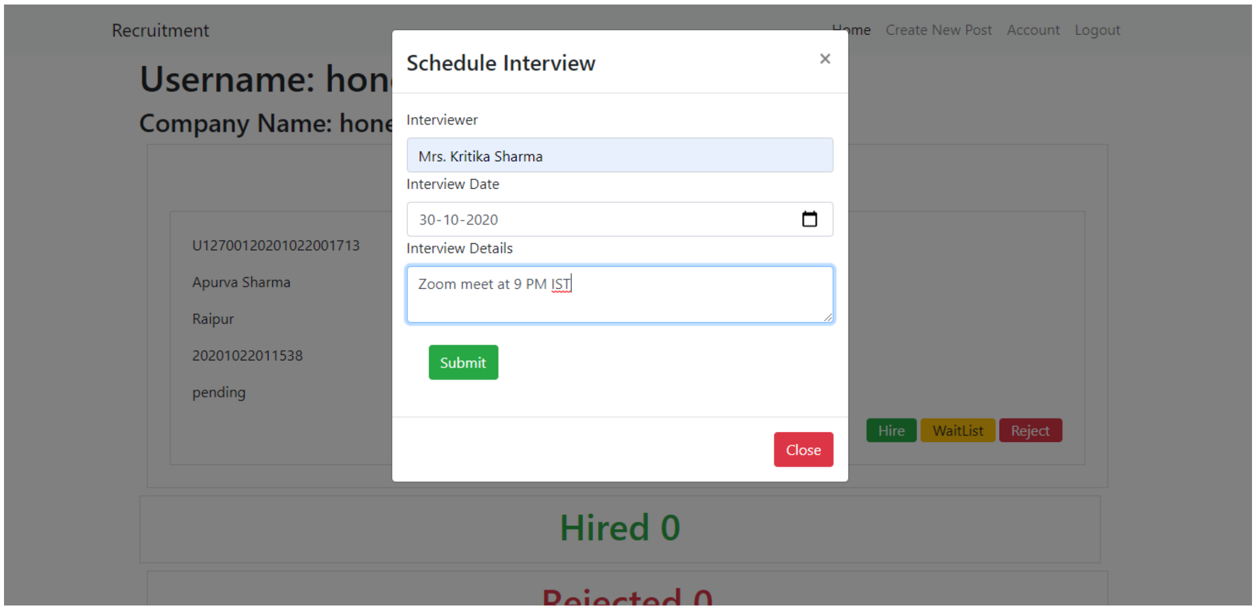
offer Date

31-10-2020

offer Details

Full-time, work from Home

13. Add the interview if you want to add someone to add someone to the waiting list.


Recruitment

Home Create New Post Account Logout

Username: hon

Company Name: hone

U12700120201022001713

Apurva Sharma

Raipur

20201022011538

pending

Schedule Interview

Interviewer

Mrs. Kritika Sharma

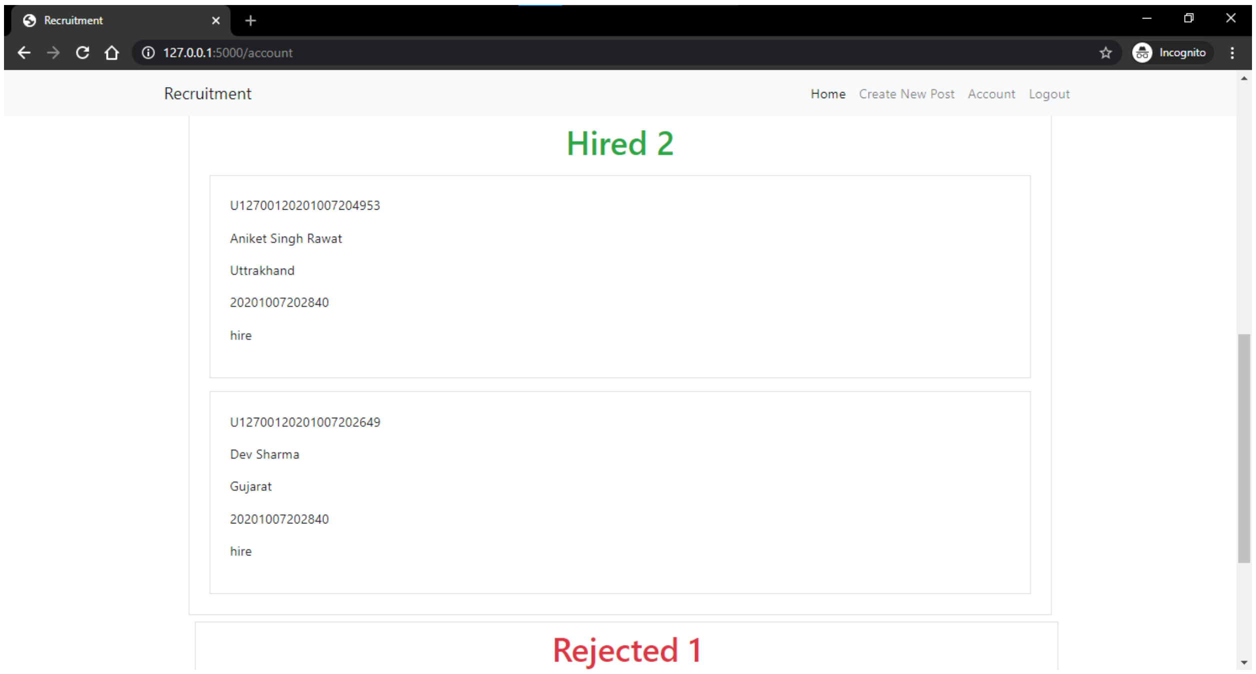
Interview Date

30-10-2020

Interview Details

Zoom meet at 9 PM IST

14. Check which applicant is hired/rejected or have been added to the waiting list. This interface is for **Company.**


Recruitment
127.0.0.1:5000/account
Incognito
Recruitment Home Create New Post Account Logout

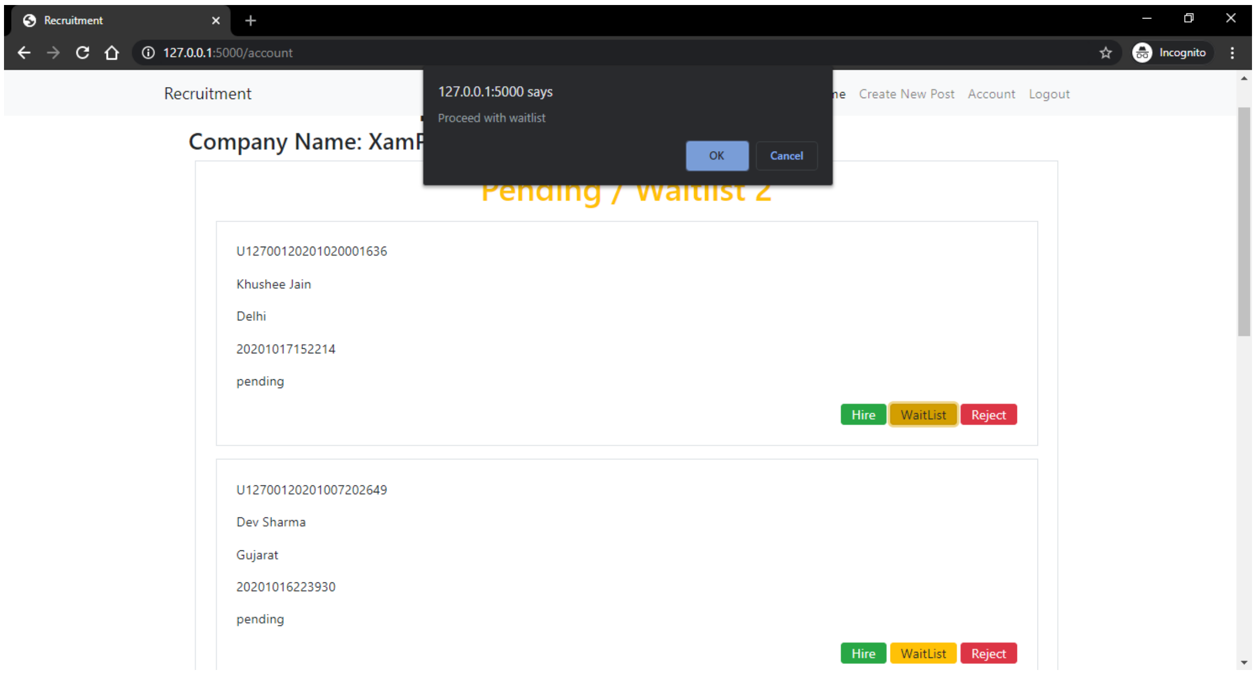
Hired 2

U12700120201007204953
Aniket Singh Rawat
Uttrakhand
20201007202840
hire

U12700120201007202649
Dev Sharma
Gujarat
20201007202840
hire

Rejected 1

15. Hire Khushee Jain, prompt message for confirmation.


Recruitment
127.0.0.1:5000/account
Incognito
Recruitment Create New Post Account Logout

127.0.0.1:5000 says
Proceed with waitlist
OK Cancel

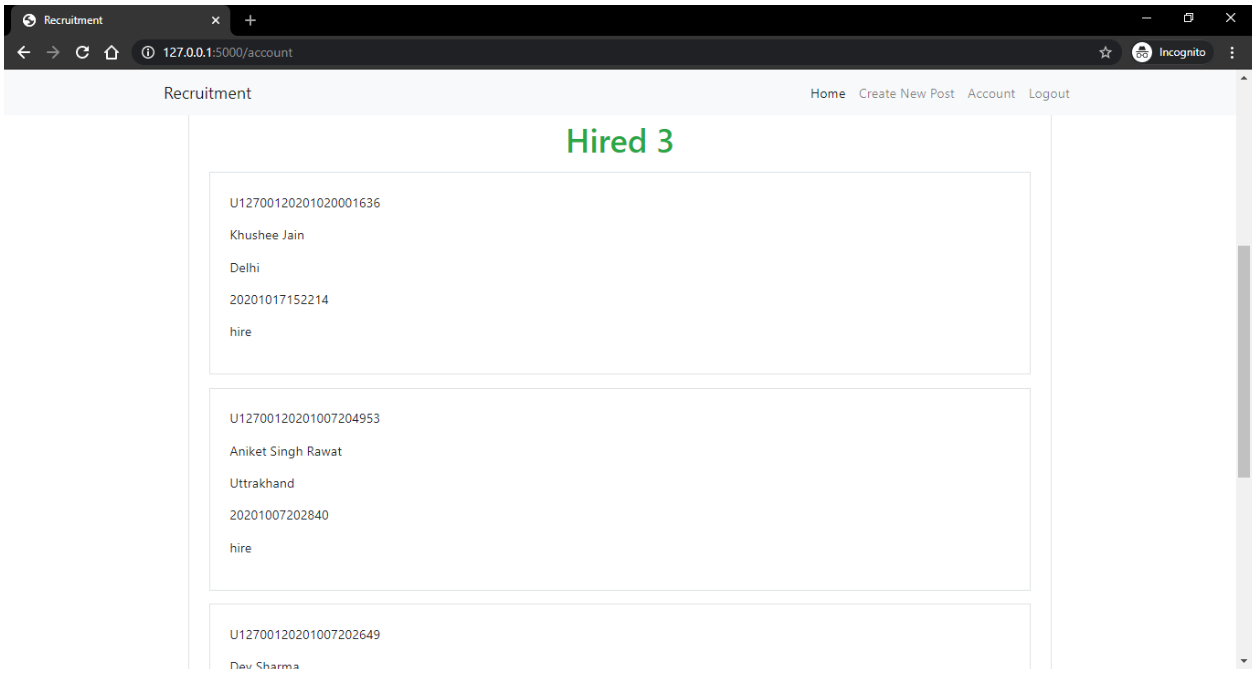
Company Name: XamF

Pending / Waitlist 2

U12700120201020001636
Khushee Jain
Delhi
20201017152214
pending
Hire WaitList Reject

U12700120201007202649
Dev Sharma
Gujarat
20201016223930
pending
Hire WaitList Reject

16. Now there are 3 hired people for the particular job post.


Recruitment

127.0.0.1:5000/account

Recruitment

Hired 3

U12700120201020001636

Khushee Jain

Delhi

20201017152214

hire

U12700120201007204953

Aniket Singh Rawat

Uttrakhand

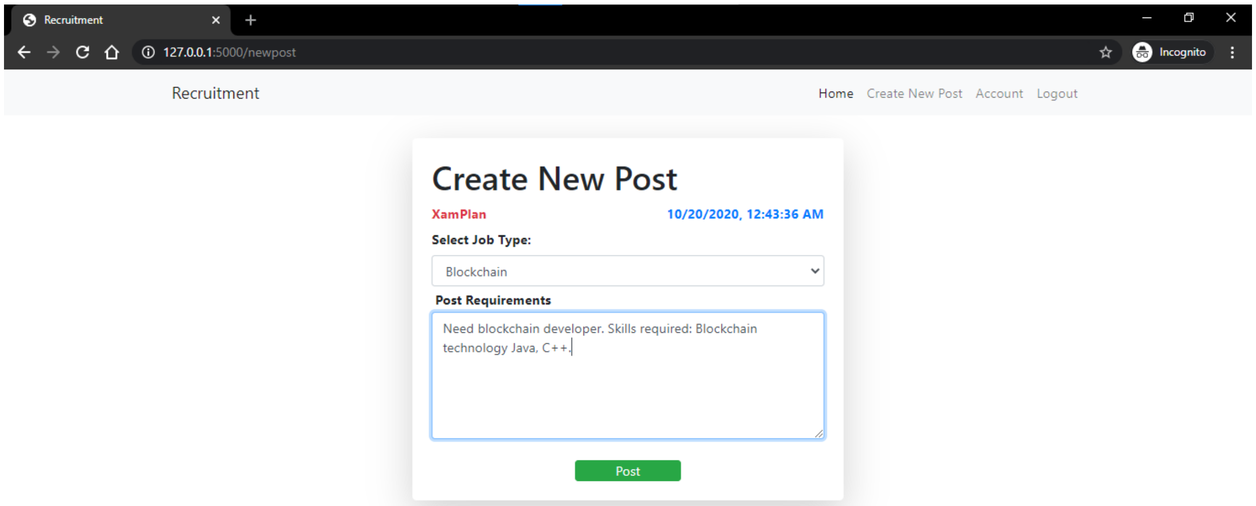
20201007202840

hire

U12700120201007202649

Dev Sharma

17. Create a new job post. This interface is for **Company.**


Recruitment

127.0.0.1:5000/newpost

Recruitment

Create New Post

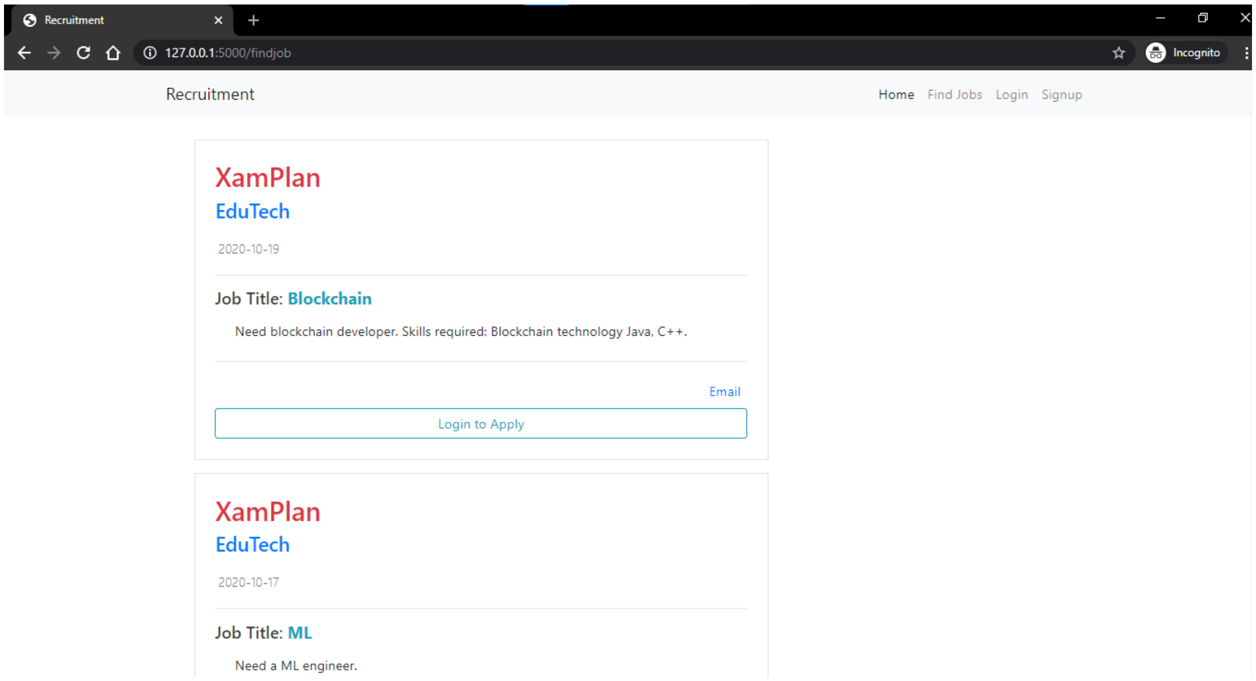
XamPlan 10/20/2020, 12:43:36 AM

Select Job Type: Blockchain

Post Requirements

Need blockchain developer. Skills required: Blockchain technology Java, C++.

18. Find Jobs without creating an account. And Login for sending an application.



XamPlan
EduTech
2020-10-19

Job Title: Blockchain

Need blockchain developer. Skills required: Blockchain technology Java, C++.

Email

Login to Apply

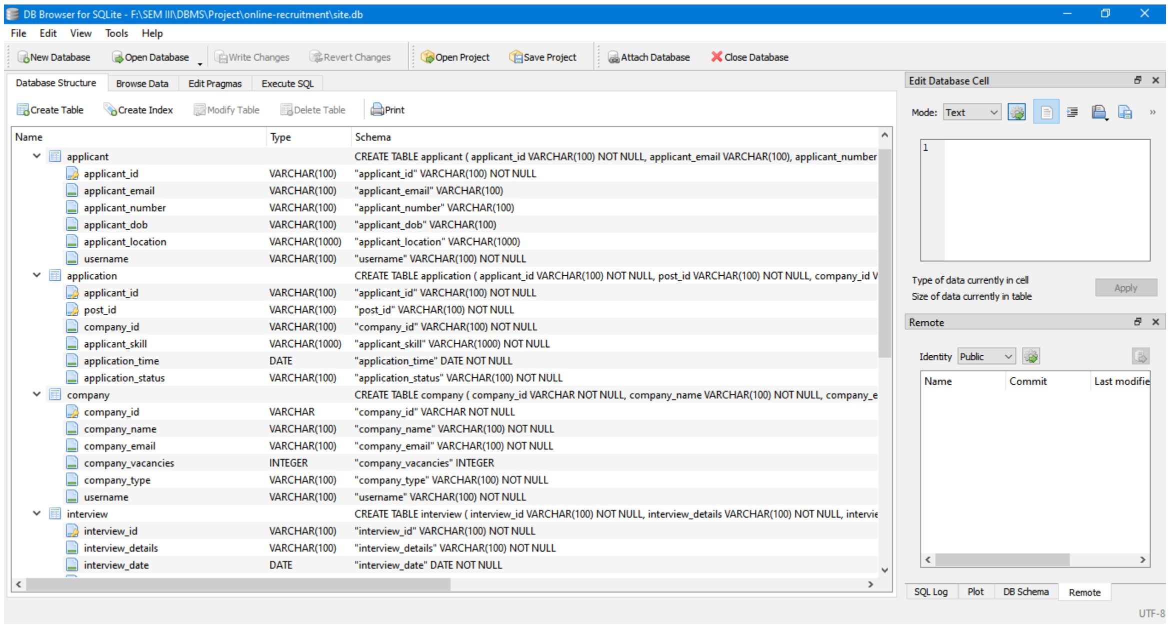
XamPlan
EduTech
2020-10-17

Job Title: ML

Need a ML engineer.

19. Glimpse of the database(SQLite) :

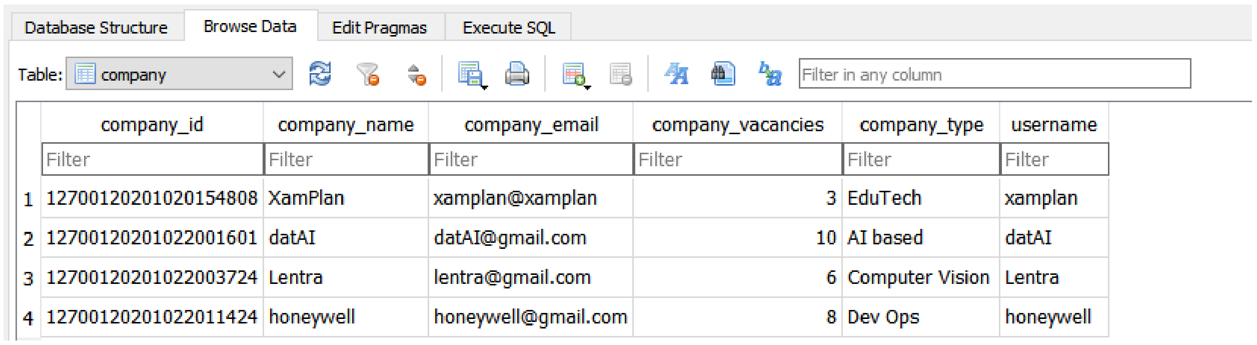
All the tables:



Name	Type	Schema
applicant		CREATE TABLE applicant (applicant_id VARCHAR(100) NOT NULL, applicant_email VARCHAR(100), applicant_number
applicant_id	VARCHAR(100)	"applicant_id" VARCHAR(100) NOT NULL
applicant_email	VARCHAR(100)	"applicant_email" VARCHAR(100)
applicant_number	VARCHAR(100)	"applicant_number" VARCHAR(100)
applicant_dob	VARCHAR(100)	"applicant_dob" VARCHAR(100)
applicant_location	VARCHAR(1000)	"applicant_location" VARCHAR(1000)
username	VARCHAR(100)	"username" VARCHAR(100) NOT NULL
application		CREATE TABLE application (applicant_id VARCHAR(100) NOT NULL, post_id VARCHAR(100) NOT NULL, company_id V
applicant_id	VARCHAR(100)	"applicant_id" VARCHAR(100) NOT NULL
post_id	VARCHAR(100)	"post_id" VARCHAR(100) NOT NULL
company_id	VARCHAR(100)	"company_id" VARCHAR(100) NOT NULL
applicant_skill	VARCHAR(1000)	"applicant_skill" VARCHAR(1000) NOT NULL
application_time	DATE	"application_time" DATE NOT NULL
application_status	VARCHAR(100)	"application_status" VARCHAR(100) NOT NULL
company		CREATE TABLE company (company_id VARCHAR NOT NULL, company_name VARCHAR(100) NOT NULL, company_e
company_id	VARCHAR	"company_id" VARCHAR NOT NULL
company_name	VARCHAR(100)	"company_name" VARCHAR(100) NOT NULL
company_email	VARCHAR(100)	"company_email" VARCHAR(100) NOT NULL
company_vacancies	INTEGER	"company_vacancies" INTEGER
company_type	VARCHAR(100)	"company_type" VARCHAR(100) NOT NULL
username	VARCHAR(100)	"username" VARCHAR(100) NOT NULL
interview		CREATE TABLE interview (interview_id VARCHAR(100) NOT NULL, interview_details VARCHAR(100) NOT NULL, intervie
interview_id	VARCHAR(100)	"interview_id" VARCHAR(100) NOT NULL
interview_details	VARCHAR(100)	"interview_details" VARCHAR(100) NOT NULL
interview_date	DATE	"interview_date" DATE NOT NULL

SOME Database Tables :- Data Dictionary s

20. The company database:

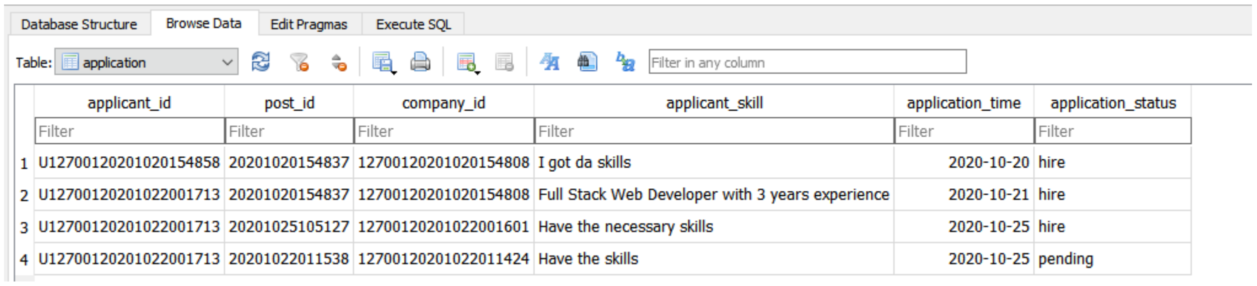


Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: company

	company_id	company_name	company_email	company_vacancies	company_type	username
1	12700120201020154808	XamPlan	xamplan@xamplan	3	EduTech	xamplan
2	12700120201022001601	datAI	datAI@gmail.com	10	AI based	datAI
3	12700120201022003724	Lentra	lentra@gmail.com	6	Computer Vision	Lentra
4	12700120201022011424	honeywell	honeywell@gmail.com	8	Dev Ops	honeywell

21. The application database:

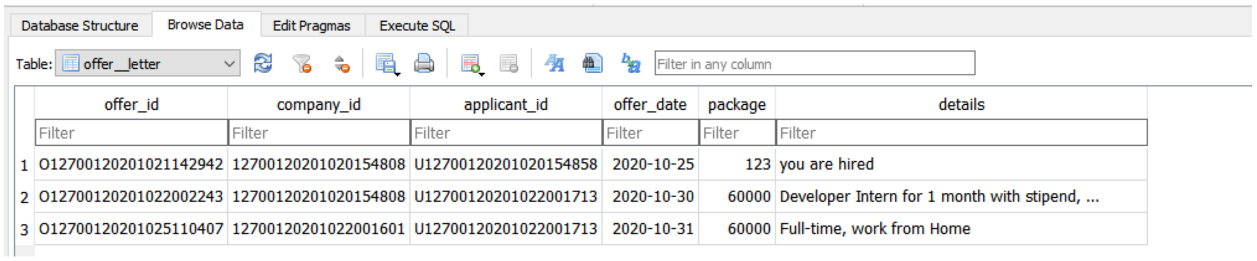


Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: application

	applicant_id	post_id	company_id	applicant_skill	application_time	application_status
1	U12700120201020154858	20201020154837	12700120201020154808	I got da skills	2020-10-20	hire
2	U12700120201022001713	20201020154837	12700120201020154808	Full Stack Web Developer with 3 years experience	2020-10-21	hire
3	U12700120201022001713	20201025105127	12700120201022001601	Have the necessary skills	2020-10-25	hire
4	U12700120201022001713	20201022011538	12700120201022011424	Have the skills	2020-10-25	pending

22. Offer_letter database

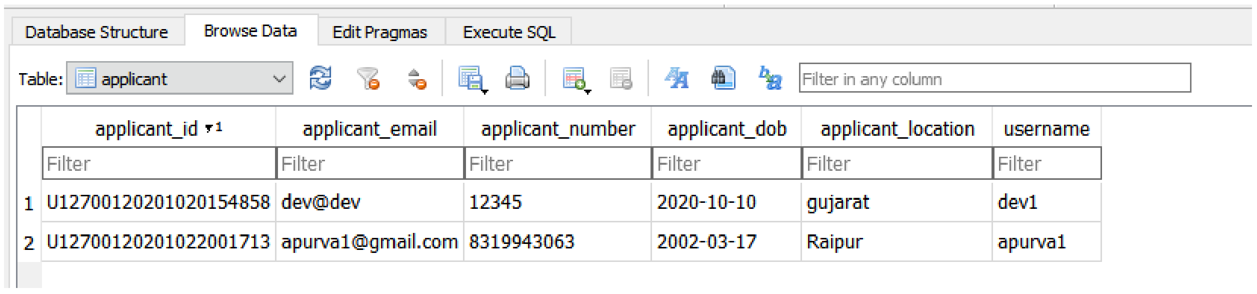


Database Structure | Browse Data | Edit Pragmas | Execute SQL

Table: offer__letter

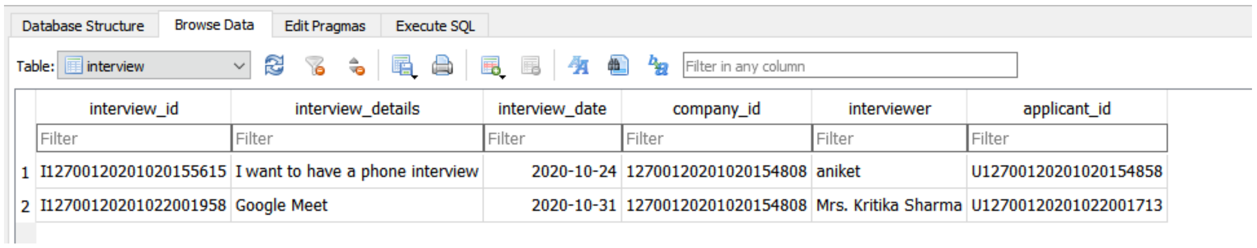
	offer_id	company_id	applicant_id	offer_date	package	details
1	O12700120201021142942	12700120201020154808	U12700120201020154858	2020-10-25	123	you are hired
2	O12700120201022002243	12700120201020154808	U12700120201022001713	2020-10-30	60000	Developer Intern for 1 month with stipend, ...
3	O12700120201025110407	12700120201022001601	U12700120201022001713	2020-10-31	60000	Full-time, work from Home

23. The applicant database



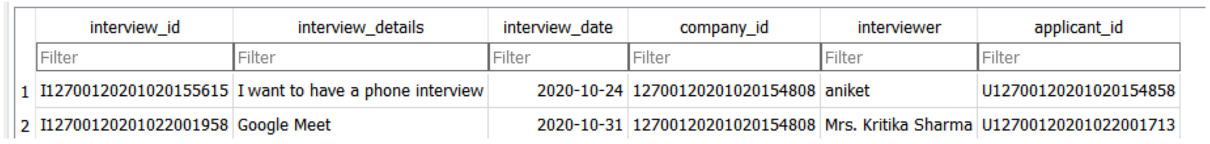
	applicant_id	applicant_email	applicant_number	applicant_dob	applicant_location	username
	Filter	Filter	Filter	Filter	Filter	Filter
1	U127001202010201548 58	dev@dev	12345	2020-10-10	gujarat	dev1
2	U127001202010220017 13	apurva1@gmail.com	8319943063	2002-03-17	Raipur	apurva1

24. The interview database



	interview_id	interview_details	interview_date	company_id	interviewer	applicant_id
	Filter	Filter	Filter	Filter	Filter	Filter
1	I127001202010201556 15	I want to have a phone interview	2020-10-24	127001202010201548 08	aniket	U127001202010201548 58
2	I127001202010220019 58	Google Meet	2020-10-31	127001202010201548 08	Mrs. Kritika Sharma	U127001202010220017 13

25. The post database



	interview_id	interview_details	interview_date	company_id	interviewer	applicant_id
	Filter	Filter	Filter	Filter	Filter	Filter
1	I127001202010201556 15	I want to have a phone interview	2020-10-24	127001202010201548 08	aniket	U127001202010201548 58
2	I127001202010220019 58	Google Meet	2020-10-31	127001202010201548 08	Mrs. Kritika Sharma	U127001202010220017 13