Project #2 - Jobster Report

Team Member: Yixuan Tang (yt1369@nyu.edu), Isha Chaturvedi (ic1018@nyu.edu)

1. Project Description

In this project, both frontend and backend of the 'Jobster' website is designed and developed using the relational database that was designed in the first project. The database schema is updated and improved to improve the functionality of the website. Jobster is a place where students can look for jobs as well as apply, send messages to their friends, follow a company, send friend requests, search for people on the website etc.

2. Introduction

This is a web development project for to develop and design the frontend and backend of a job-hunting website called 'Jobster'. The the focus of the website is to provide an online portal for graduating students to look for jobs, the frontend and backend are designed and developed to facilitate the objective. The website is designed to serve the following roles:

- The site lets the students to sign up and add their resume and other relevant information to the website. Students can update their information including updating password as well (except loginname and student name).
- Students can also edit or set their private settings as public or friendly public (described in detail in section 4.b.3).
- The site is also a medium for the companies to connect with the students or future employees, and it lets the companies to sign up as well. Moreover, companies can also update their profile.
- The companies can also post their information and job announcements for the vacant positions.
- A student can follow the companies of his/her interest.
- Apart from following companies, students can also apply for jobs (primary role of the website), and can connect with other students by being friends with them as well.
- Once a student get friends with someone, he/she can send and receive messages to and from that friend respectively
- A company can have multiple followers, and a student can follow multiple companies. Similarly, a student can have multiple friends.
- A student can search for other students and jobs positions, and a company can look for students matching a particular job requirement (or any keyword).

3. Workflow

Here is the whole procedure of how the website was designed and developed (further explanations of functionality and mechanisms are section 4 and 5):

The project is divided into two main parts:

Backend:

1. create_db.sql: The database can be created using this sql file, which loads into all the relevant schema tables and dummy data.
2. functions.py: All the queries in the form of functions, are run through this python 2.7 script.

Frontend:

1. main.py: This is the main flask (micro web framework using Python) file to manage all the requests on the backend. (This is the
2. auth.py: This is the authentication file for login and signout, which is using Flask-Login framework.
3. forms.py: This is a python file for creating all the relevant forms in the website. It is using Flask-Form framework.

Extended Functionalities:

1. There is real-time messaging (1 second lag) incorporated in the website in a smooth imessage like user-interface.
2. The password are hashed (salt + hashing), to ramp up the security of the student and company profiles on the website.
3. The session storage function is incorporated in the website, which saves the user's session even if they have closed the tab. Further users can sign out of the website.
4. The website is made user-friendly using base template HTML-UP Alpha, which has pre-included CSS files.

The website can be run in the following way:

1. All the files are in a Jobster folder. All the required sources such as flask, flask_login, wtforms and flask_wtf needs to be installed before running the website.
2. The database needs to be loaded (using create_db.sql) using MySQL database management system (MySQL workbench).
3. The function.py connects the python script to the database, and thus care should be taken that the mysql server is properly connected to the python script. For this,

mysql.connector needs to be installed. Further care should be taken that python mysql connector takes in correct username, password, hostname, port number and database name.

4. Along with this hashlib library needs to be installed to enable the hashing password feature in the website.
5. One all the installations are done, and database is also loaded successfully, main.py needs to be called on the terminal (python main.py). After this, the address: http://localhost:5000/ should be typed in the web browser to launch the website.

4. Backend

MySQL relational database management system is used to create the backend of the website, and a python script (function.py) is used to query the database.

4.a Revised Database Schema

- The database is developed using MySQL (MySQL workbench).
- All the data about students, companies, job announcements (positions), connections (friendships and followings), notifications(eg. message, friend request, job notification) and job applications are stored in a relational database.
- The database is first designed using Entity-Relationship (ER) model, and later it is translated to a Relational model.
- The database went through several iterations of testing (the database is populated with sample data to carry out the testing) and re-design phases, to make sure it is equipped to execute the required sql queries and operations.
- Further, the database schema is updated while developing the website, to improve the functionality of the website.

4.a.1 Entity-Relationship (ER) model

The Message table is updated to incorporate the receive relationship as well. The application and notification are changed to relations (relations between student and position). Further, privacy settings column is included in the student table. Since Request and Message are weak entities, the solid dashed lines of rdate and mdate are changed to dotted lines. Further, now the schema as the feature of notification and message being read or not. Figure 1 below shows the updated working schema, that is used to develop the Jobster website.

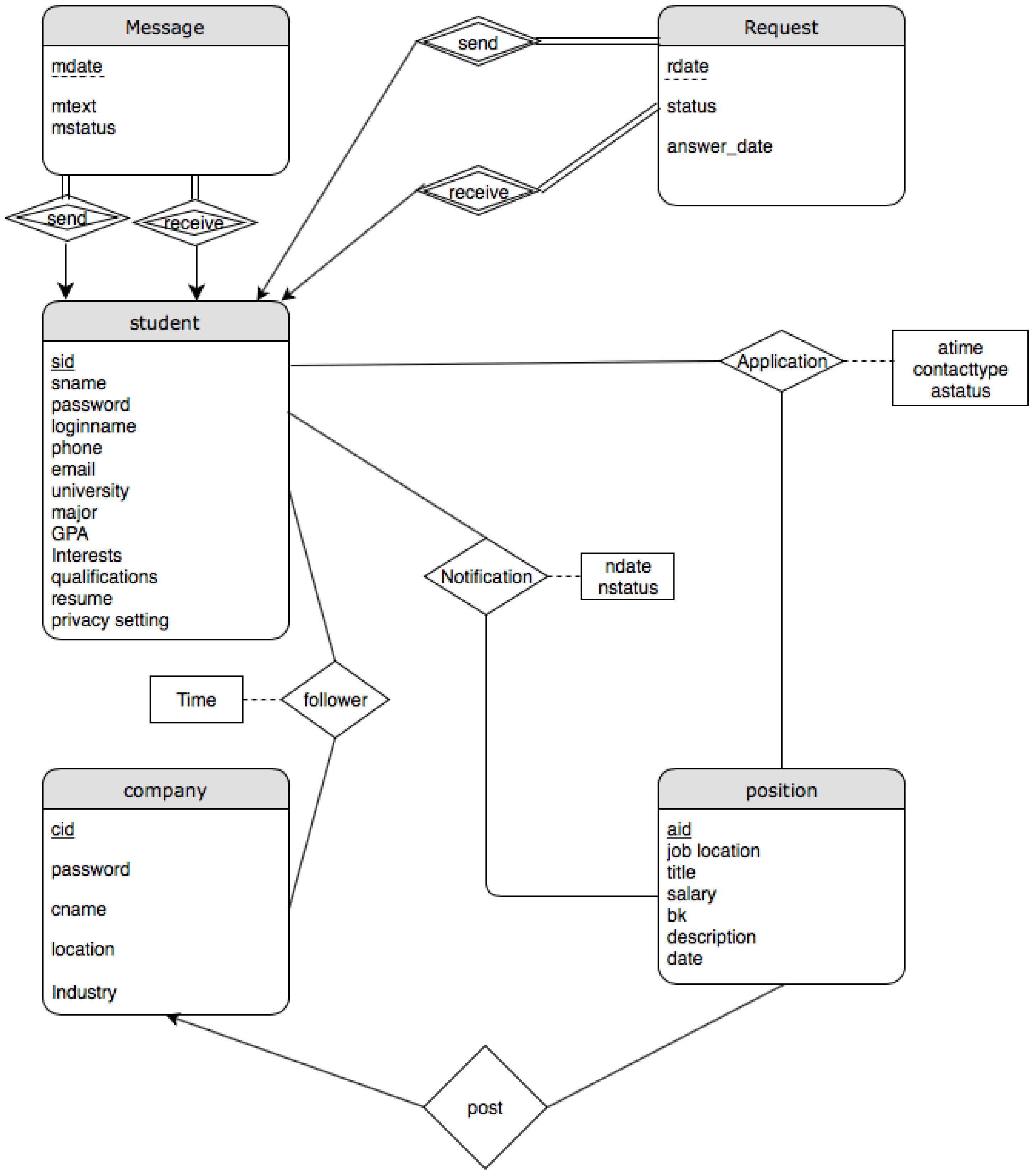


Figure 1: Revised working ER diagram for the Jobster website.

4.a.2 Relational model and Database tables

The relational model (Figure 2) is updated according to the updated version of the ER diag above (Figure 1). Figure 2 shows the database tables, the column names and their types (with the constraints on the types), the default values of the columns and information about the NOT NULL constraint. Along with the constraints shown in the Figure 2, here are further more constraints (the constraints and defaults values are similar to that of in Project 1):

1. Every student user should have unique: student id, login name, email address and phone number, that is there cannot be two students with same login name and contact information (email address and phone number).
2. Every company should have unique: company id and company name.
3. Every position id (aid) should be unique.

These constraints makes sure that a particular student or a company don't register into the database repeatedly, as well don't apply the same job application again. The constraints are handled through the keyword "UNIQUE".

4. Key constraints (Primary and Foreign Key):

Primary Keys:
Student: sid
Company: cid
Position: aid
Follower: cid and sid (together)
Request: rdate, sid, Friendid (together)
Message: mdate, sid1, sid2 (together)
Notification: aid and sid (together)
Application: aid, sid (together)

Foreign Keys:
position.cid REFERENCES Company.cid;
Follower.cid REFERENCES Company.cid,
Follower.sid REFERENCES Student.sid;
Request.sid REFERENCES Student.sid,
Request.Friendid REFERENCES Student.sid;
Message.sid1 REFERENCES Student.sid,
Message.sid2 REFERENCES Student.sid;
Notification.aid REFERENCES position.aid,
Notification.sid REFERENCES Student.sid;
Application.aid REFERENCES position.aid,
Application.sid REFERENCES Student.sid;

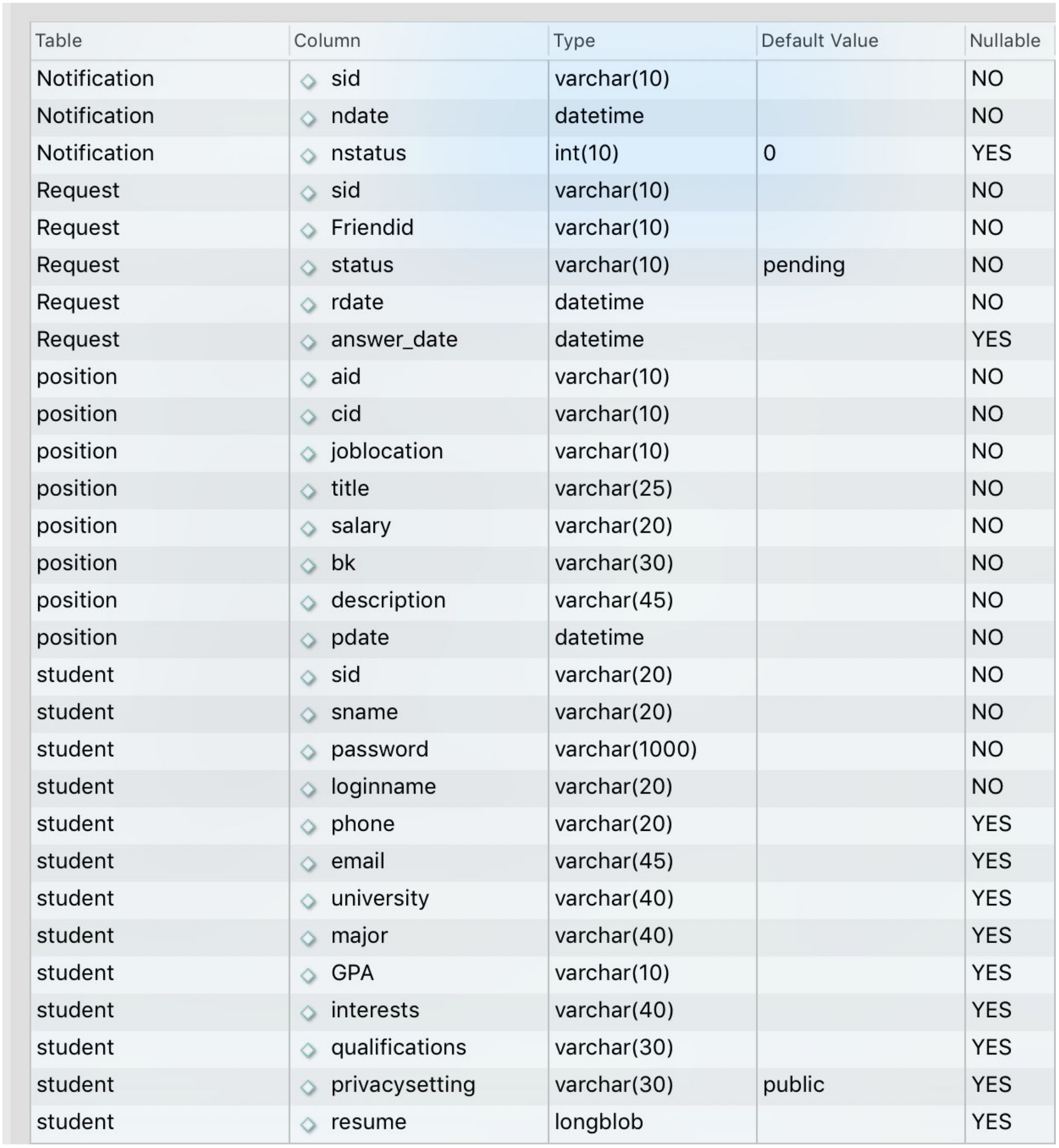


Table	Column	Type	Default Value	Nullable
Notification	sid	varchar(10)		NO
Notification	ndate	datetime		NO
Notification	nstatus	int(10)	0	YES
Request	sid	varchar(10)		NO
Request	Friendid	varchar(10)		NO
Request	status	varchar(10)	pending	NO
Request	rdate	datetime		NO
Request	answer_date	datetime		YES
position	aid	varchar(10)		NO
position	cid	varchar(10)		NO
position	joblocation	varchar(10)		NO
position	title	varchar(25)		NO
position	salary	varchar(20)		NO
position	bk	varchar(30)		NO
position	description	varchar(45)		NO
position	pdate	datetime		NO
student	sid	varchar(20)		NO
student	sname	varchar(20)		NO
student	password	varchar(1000)		NO
student	loginname	varchar(20)		NO
student	phone	varchar(20)		YES
student	email	varchar(45)		YES
student	university	varchar(40)		YES
student	major	varchar(40)		YES
student	GPA	varchar(10)		YES
student	interests	varchar(40)		YES
student	qualifications	varchar(30)		YES
student	privacysetting	varchar(30)	public	YES
student	resume	longblob		YES

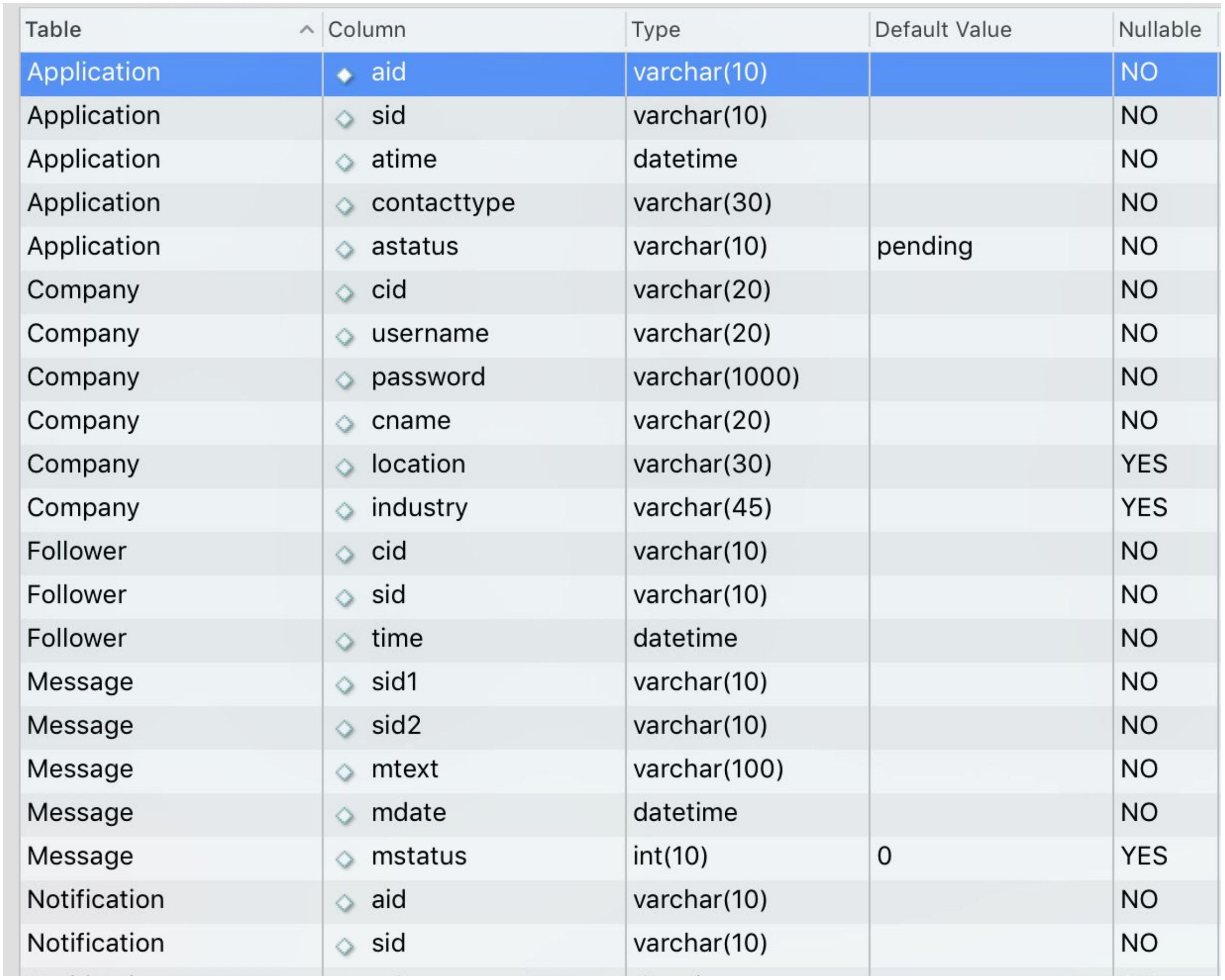


Table	Column	Type	Default Value	Nullable
Application	◆ aid	varchar(10)		NO
Application	◇ sid	varchar(10)		NO
Application	◇ atime	datetime		NO
Application	◇ contacttype	varchar(30)		NO
Application	◇ astatus	varchar(10)	pending	NO
Company	◇ cid	varchar(20)		NO
Company	◇ username	varchar(20)		NO
Company	◇ password	varchar(1000)		NO
Company	◇ cname	varchar(20)		NO
Company	◇ location	varchar(30)		YES
Company	◇ industry	varchar(45)		YES
Follower	◇ cid	varchar(10)		NO
Follower	◇ sid	varchar(10)		NO
Follower	◇ time	datetime		NO
Message	◇ sid1	varchar(10)		NO
Message	◇ sid2	varchar(10)		NO
Message	◇ mtext	varchar(100)		NO
Message	◇ mdate	datetime		NO
Message	◇ mstatus	int(10)	0	YES
Notification	◇ aid	varchar(10)		NO
Notification	◇ sid	varchar(10)		NO

Figure 2: Database Tables along with the column names and types and their respective NULL constraints and default values.

Here, size of the columns' type are relaxed to accommodate longer inputs in the field. For example, the size of the password column for both Student and Company tables is increased to 1000, as the password gets hashed once they are entered in the field, and the resulting hashed password is stored on the database (which is longer, due to the added salt in the hash).

4.b.1 SQL Queries

All the SQL Queries as encoded into Python functions in functions.py. The return type of the function queries is a dictionary, which is set at the top of the file.

4.b.2 Security

The website has good level of security as it prevents SQL injection and implements hashing of the passwords. To prevent SQL injection and other vulnerabilities, the queries are coded as : cur.execute("select * from request r join student s where r.Friendid = %s and s.sid = r.sid;", (sid,)). This form of syntax sanitizes the input and and prevents sql injection in Python file.

The passwords are hashed using hashlib.sha512 and salt (salt = 'askjdfhkjashr43iuwq78efbqwuig 7wr83gewguifgiwu3 27eg3e') is added to increase the password security. The function is implemented in the following way:

```
def hash_password(password):
   hashed_password = hashlib.sha512(password.encode('utf-8') +
salt.encode('utf-8')).hexdigest()
   return hashed_password
```

4.b.3 Jobster student profile privacy

A student user have the option to specify his/her privacy settings. That is, a student can make his profile ‘public’ or ‘friendly public’. In public setting, the student’s profile is public to other users, that is both students and companies. In friendly public setting, the student profile is visible only to the friends of the student. This way a student can decide whether or not he/she wants to restrict his/her profile to only his/her friends.

5. Frontend

The frontend is implemented to achieve the website functionalities mentioned in Section 2.

Following tools and methods are used to develop the frontend:

1. Flask, which is a micro web framework (written in Python) and based on Jinja2 template engine, is used for serving up the web-server. It is handling all the server side requests, and thus is managing the requests to the backend database. The calls to the database are made through the sql queries in functions.py as mentioned above.
2. Jinja2, which is a modern and designer-friendly templating language for Python (modelled after Django’s templates), is used for rendering templates and used as programming language for generating HTML.

3. wtforms, which is a flexible forms validation and rendering library for Python, is used as a python interface for managing forms (job posting form, login form, search form, student and company form). Here the default version of wtforms: HTML5 is used.
4. Alpha HTML UP, is used as a base template for the website. The template has pre-included CSS in it.
5. Flask-Login, which is user session management for Flask, is used to handle logging in, logging out, and remembering users' sessions over extended periods of time. That is, Flask-Login is used to handle the persistence of user authentication, so that it maintains the sessions storage, even after the tab is closed down.
6. AJAX calls, using Jquery, are made whenever there is a request being made on a page load. It is typical page request method, but it is done through javascript, and the return type is a JSON file. AJAX calls are used used for following a company, sending/accepting friend requests, and message.
7. A message interface is implemented in two ways: There is a refresh button on the page, which one can use to get the message, but this is not compulsory. Since it takes a while and effort to continuously refresh a page, a regular request method using AJAX calls is implemented. This is done using a polling fashion, so every second, requests are being sent to the server, which runs the query to get new messages.

6. Further Discussion and Conclusion

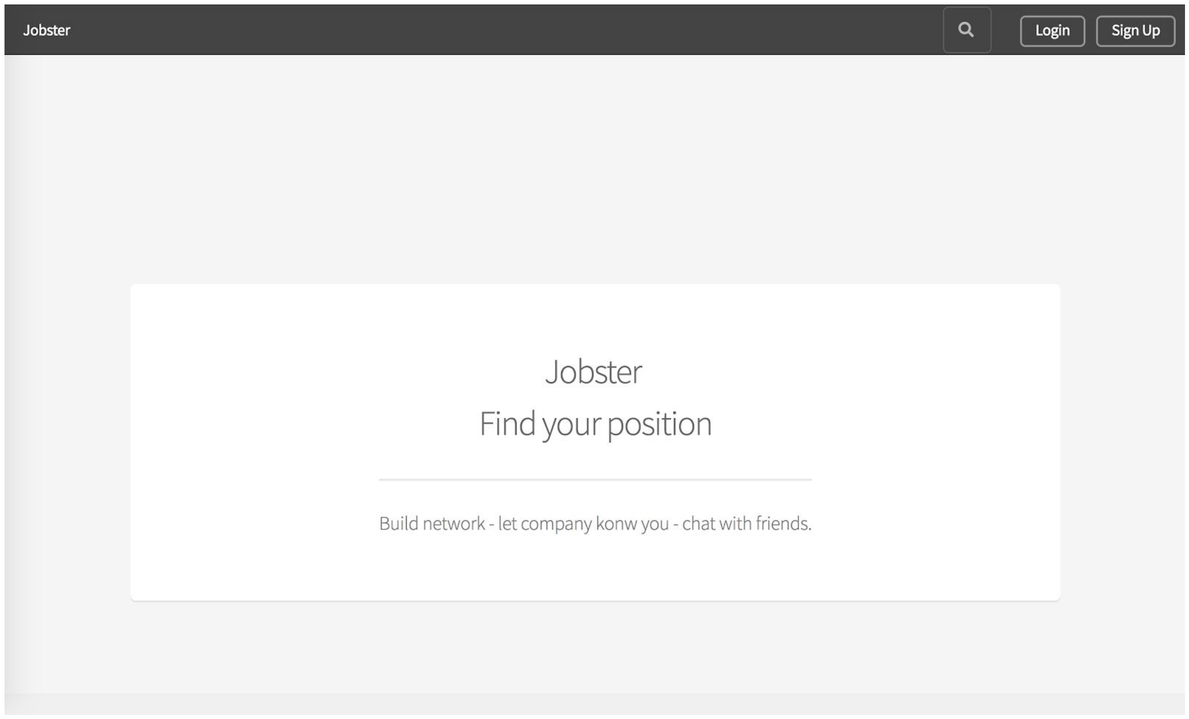
In this project, a job hunting and networking website for university students called 'Jobster' was designed and developed using a MySQL relational database system, Flask python microweb framework, Jinja templates for rendering, wtforms for managing forms, and HTML5UP - Alpha for basic template form (included CSS). Further Ajax calls (implemented using Jquery) are used to manage page requests on a page load. The website has good quality of security, as it prevents SQL injection and implements hashing + salt of the passwords. The database was designed keeping in mind that it is space efficient, non-redundant and functional. However, further improvements in the frontend is needed to improve the functionality and UI of the website.

7. References/Tools Used

MySQL Workbench
Python 2.7
Entity-Relationship Model
Alpha HTML5 UP: https://html5up.net/alpha

Flask: http://flask.pocoo.org/
Flask-Login: https://flask-login.readthedocs.io/en/latest/
WTF forms: https://wtforms.readthedocs.io/en/stable/
Jinja Template: http://jinja.pocoo.org/docs/2.10/

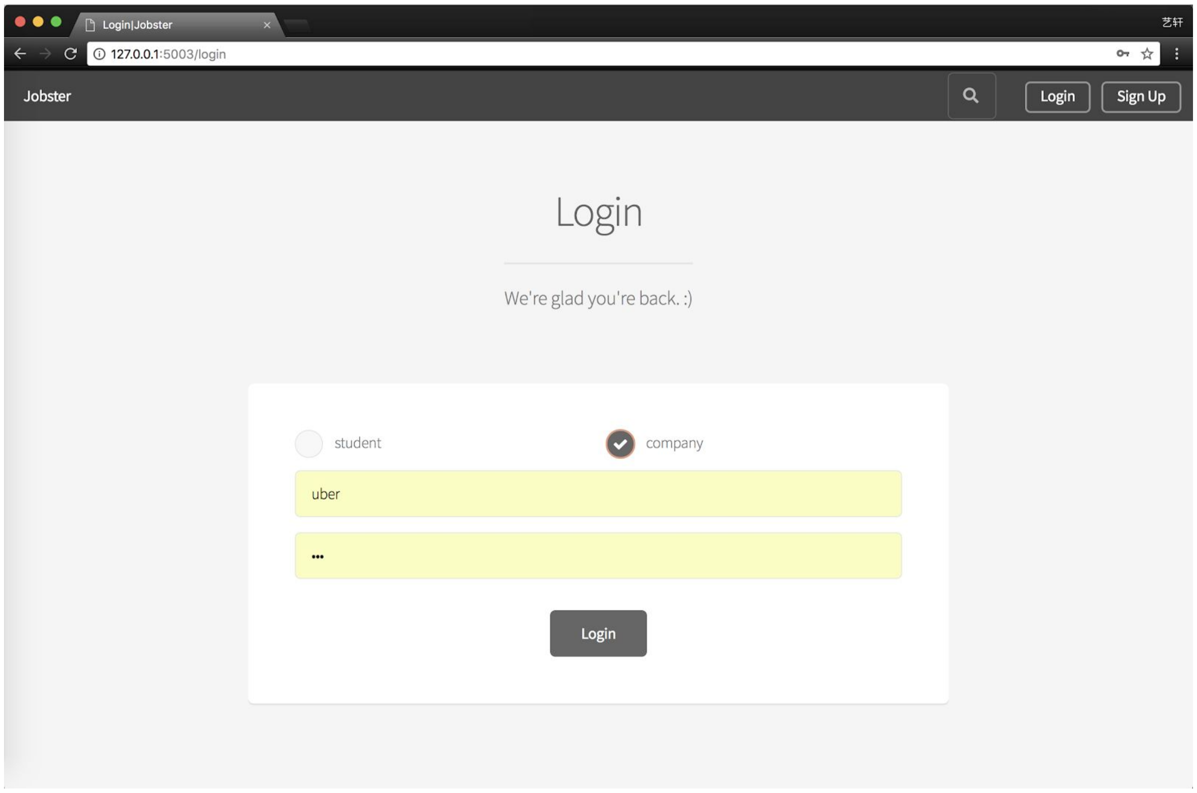
Appendix: Website Interface:



Jobster
Find your position

Build network - let company konw you - chat with friends.

Home Page



Login

We're glad you're back. :)

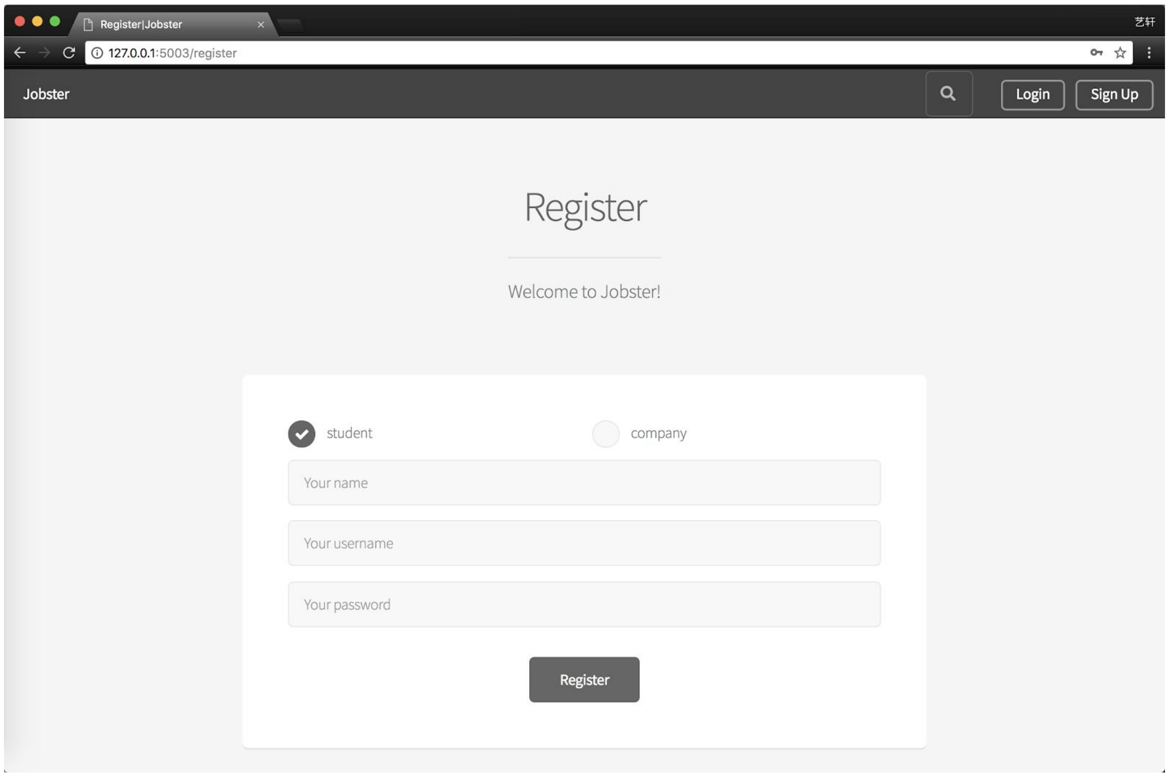
student [✓ company]

uber

•••

Login

Login page



Register

Welcome to Jobster!

[✓ student] company

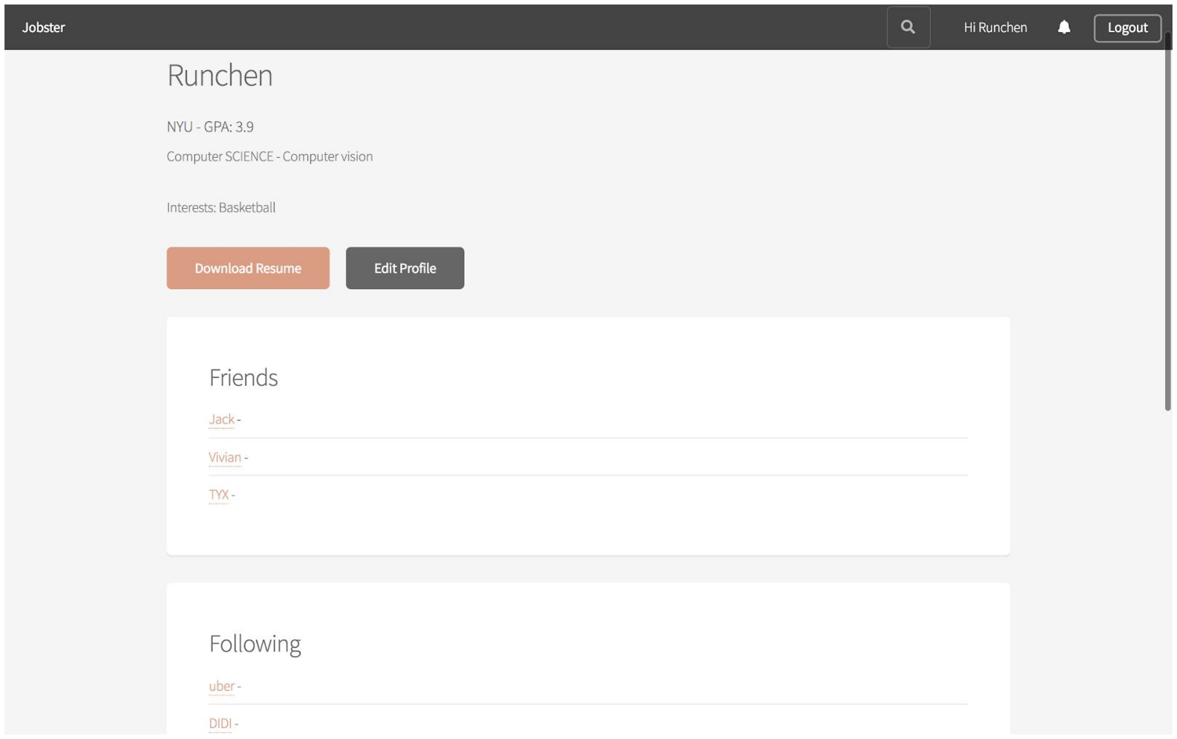
Your name

Your username

Your password

Register

Register Page



Jobster

Hi Runchen

Logout

Runchen

NYU - GPA: 3.9

Computer SCIENCE - Computer vision

Interests: Basketball

Download Resume

Edit Profile

Friends

Jack -

Vivian -

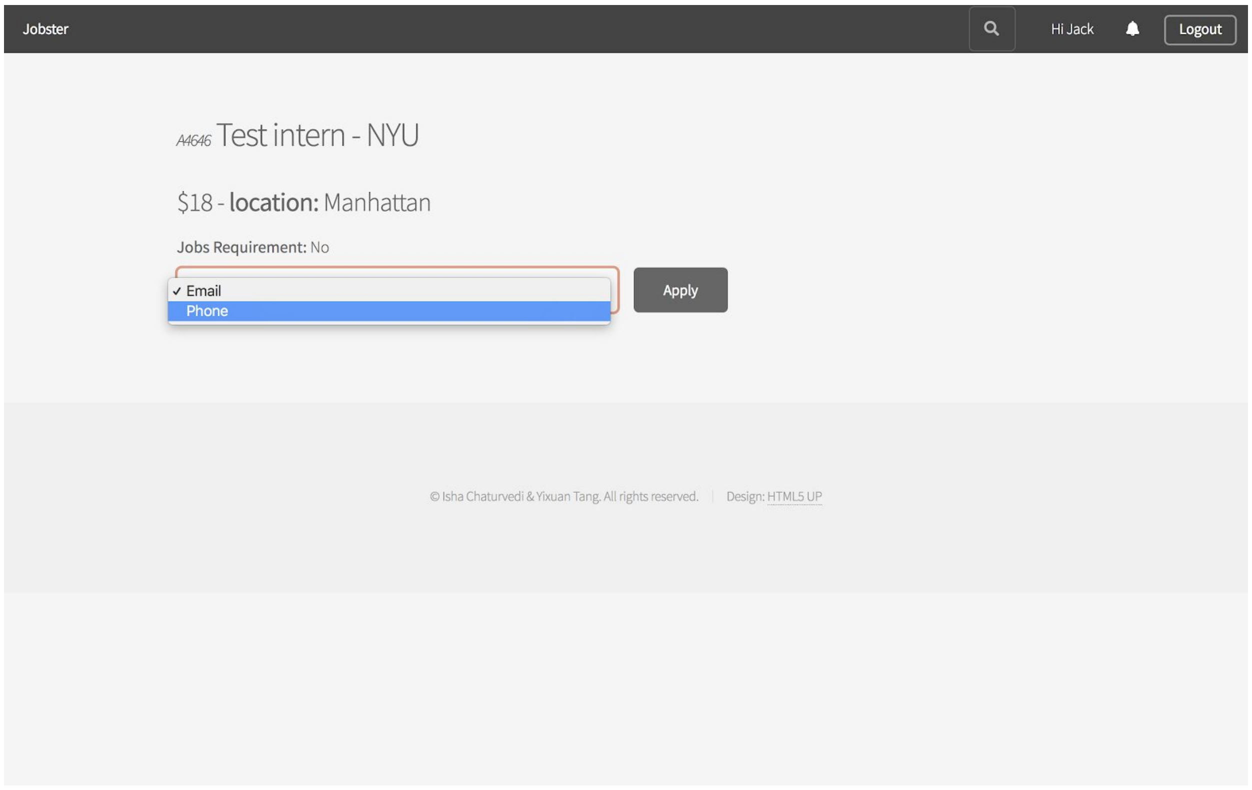
TYX -

Following

uber -

DIDI -

Student Home Page



Jobster

Hi Jack

Logout

A4646 Test intern - NYU

$18 - **location:** Manhattan

Jobs Requirement: No

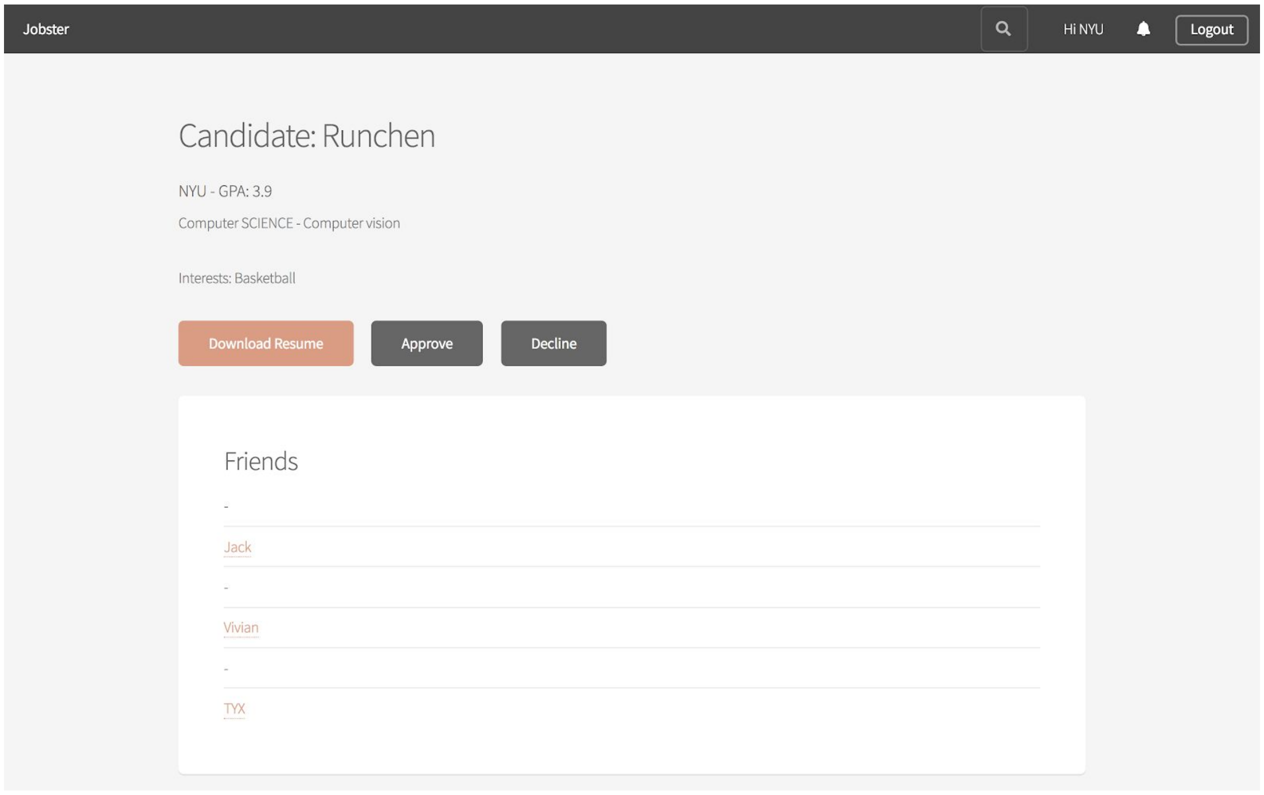
✓ Email

Phone

Apply

© Isha Chaturvedi & Yixuan Tang. All rights reserved. | Design: HTML5 UP

Job Application page


Jobster

Hi NYU

Logout

Candidate: Runchen

NYU - GPA: 3.9

Computer SCIENCE - Computer vision

Interests: Basketball

Download Resume

Approve

Decline

Friends

-

Jack

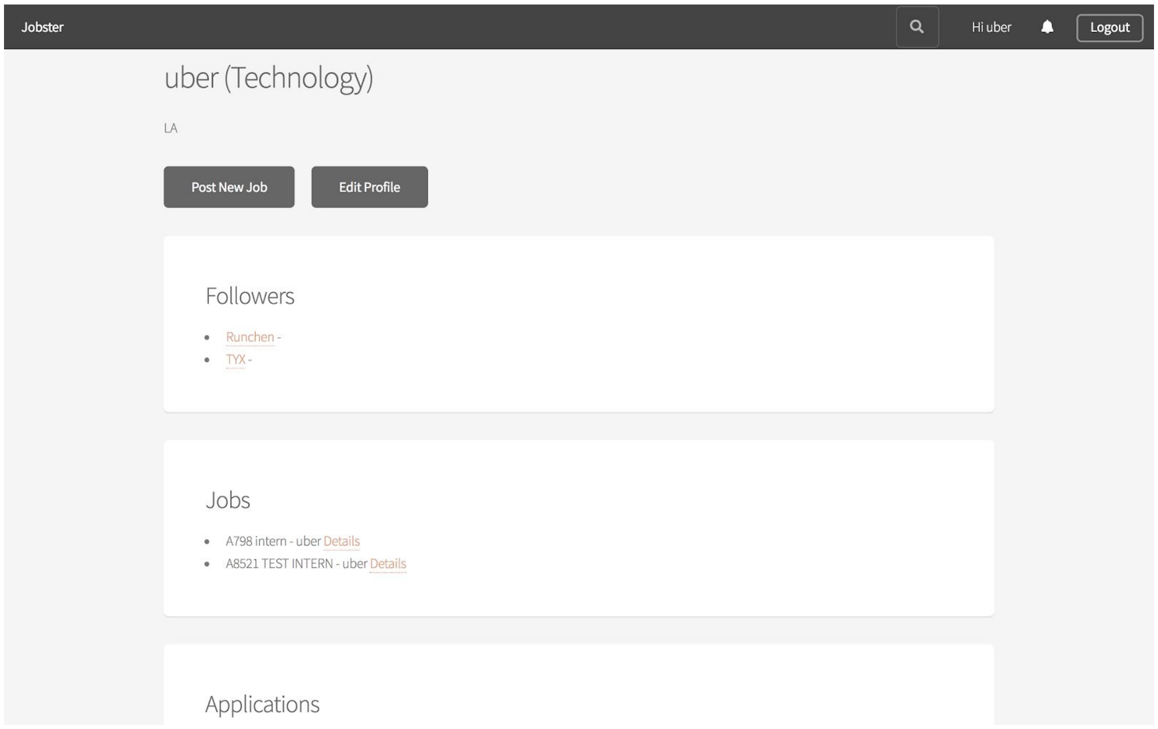
-

Vivian

-

TYX

Candidate / decision Page


Jobster

Hi uber

Logout

uber (Technology)

LA

Post New Job

Edit Profile

Followers

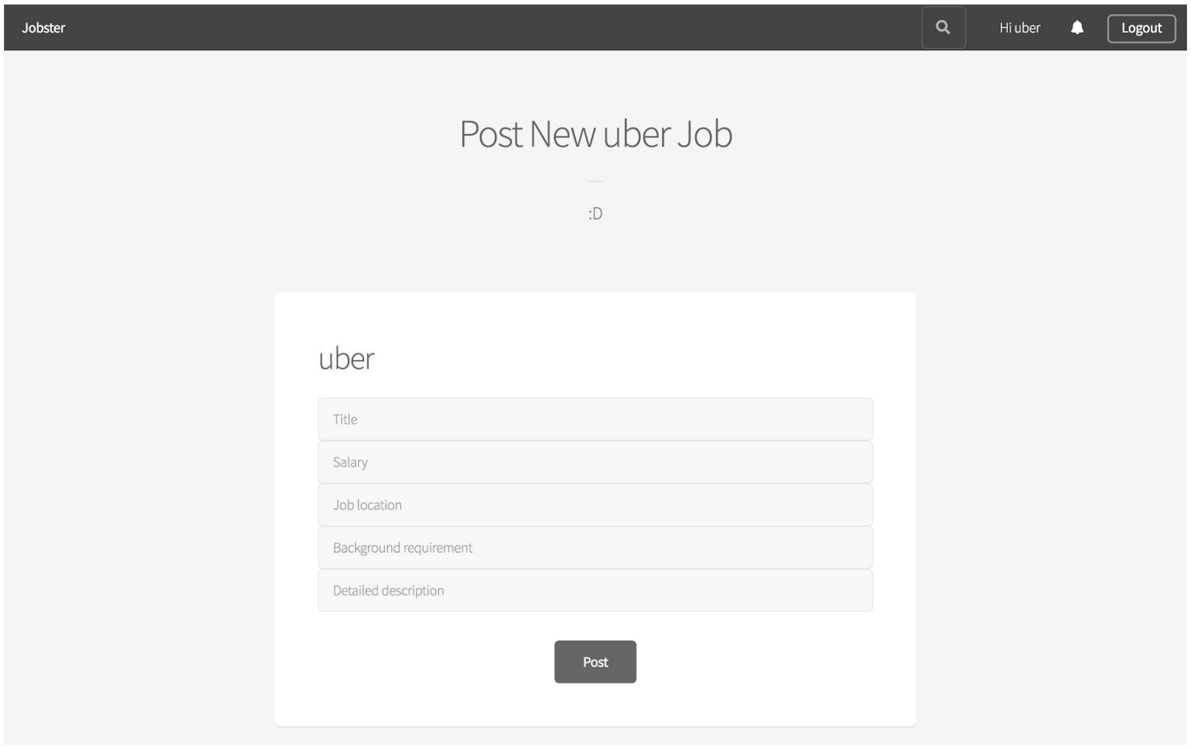
- Runchen -
- TYX -

Jobs

- A798 intern - uber Details
- A8521 TEST INTERN - uber Details

Applications

Company Home Page


Jobster

Hi uber

Logout

Post New uber Job

:D

uber

Title

Salary

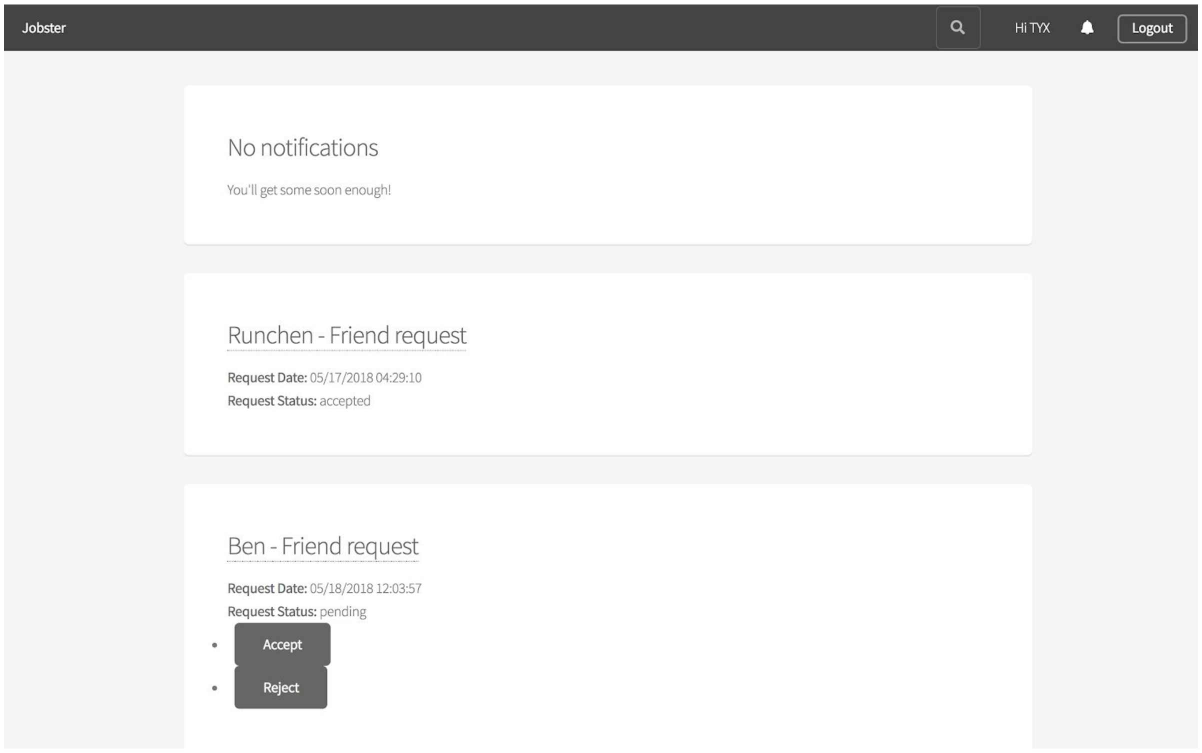
Job location

Background requirement

Detailed description

Post

Job Posting Page


Jobster

Hi TYX

Logout

No notifications

You'll get some soon enough!

Runchen - Friend request

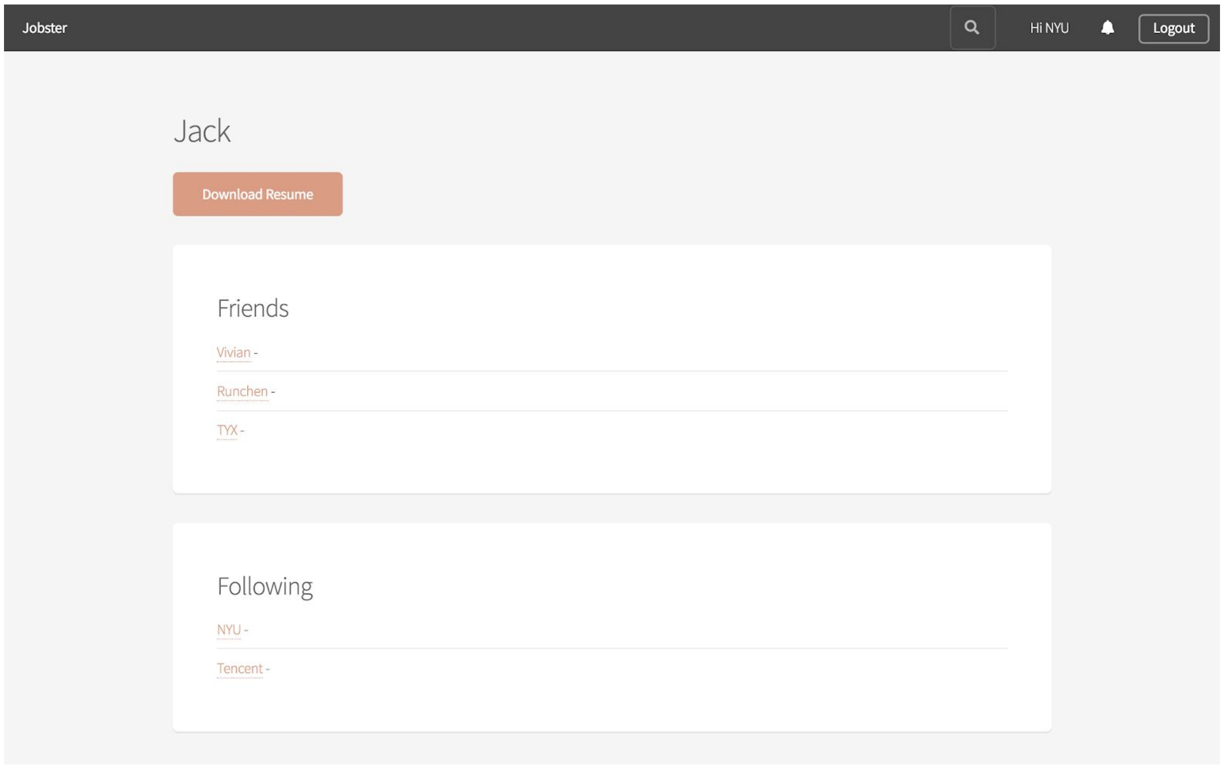
Request Date: 05/17/2018 04:29:10
Request Status: accepted

Ben - Friend request

Request Date: 05/18/2018 12:03:57
Request Status: pending

- Accept
- Reject

Notification / Friend Request Decision Page


Jobster

Hi NYU

Logout

Jack

Download Resume

Friends

Vivian -

Runchen -

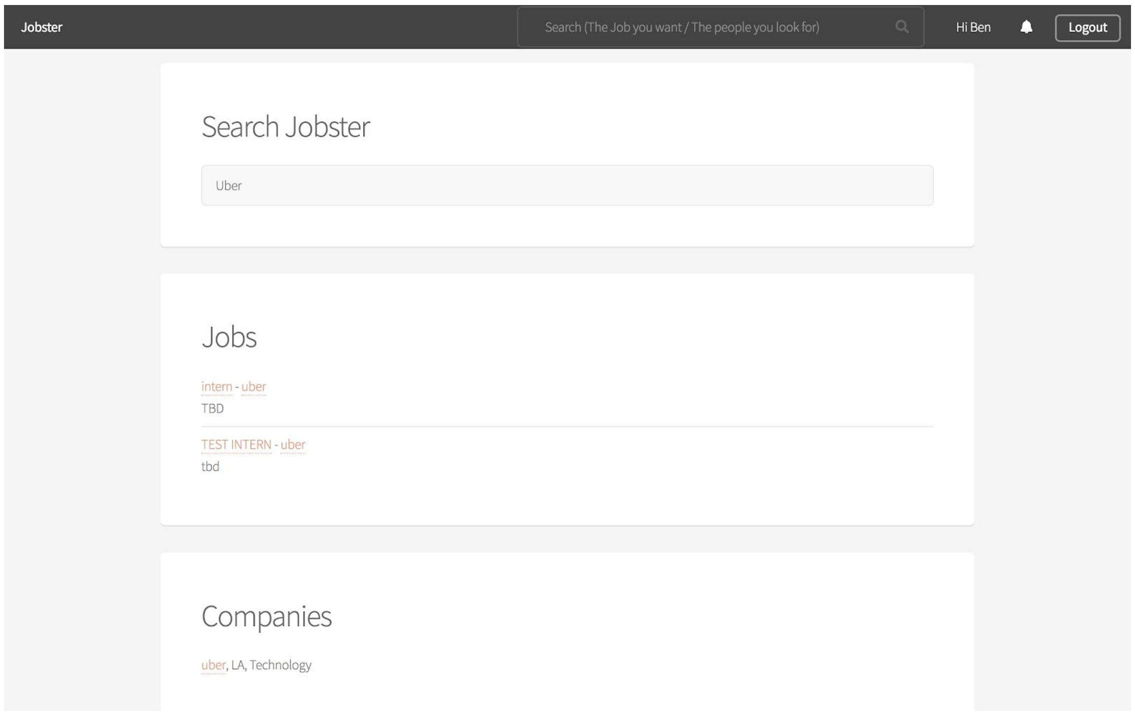
TYX -

Following

NYU -

Tencent -

Privacy Student Home Page View


Jobster

Search (The Job you want / The people you look for)

Hi Ben

Logout

Search Jobster

Uber

Jobs

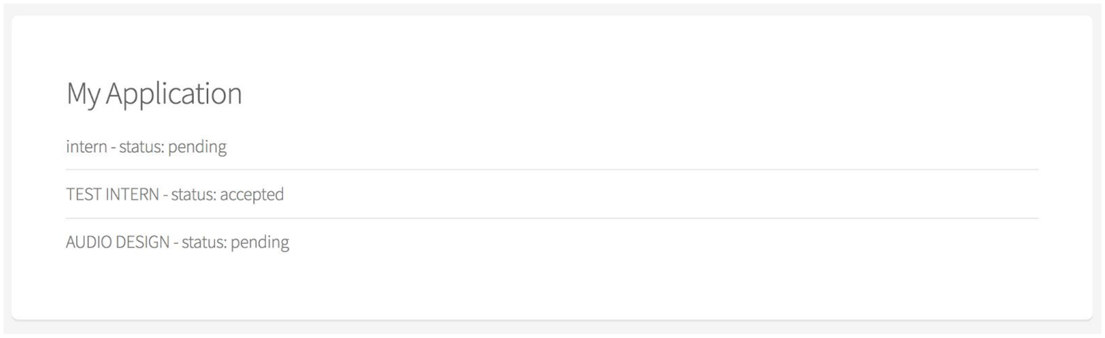
intern - uber
TBD

TEST INTERN - uber
tbd

Companies

uber, LA, Technology

Search Result Page



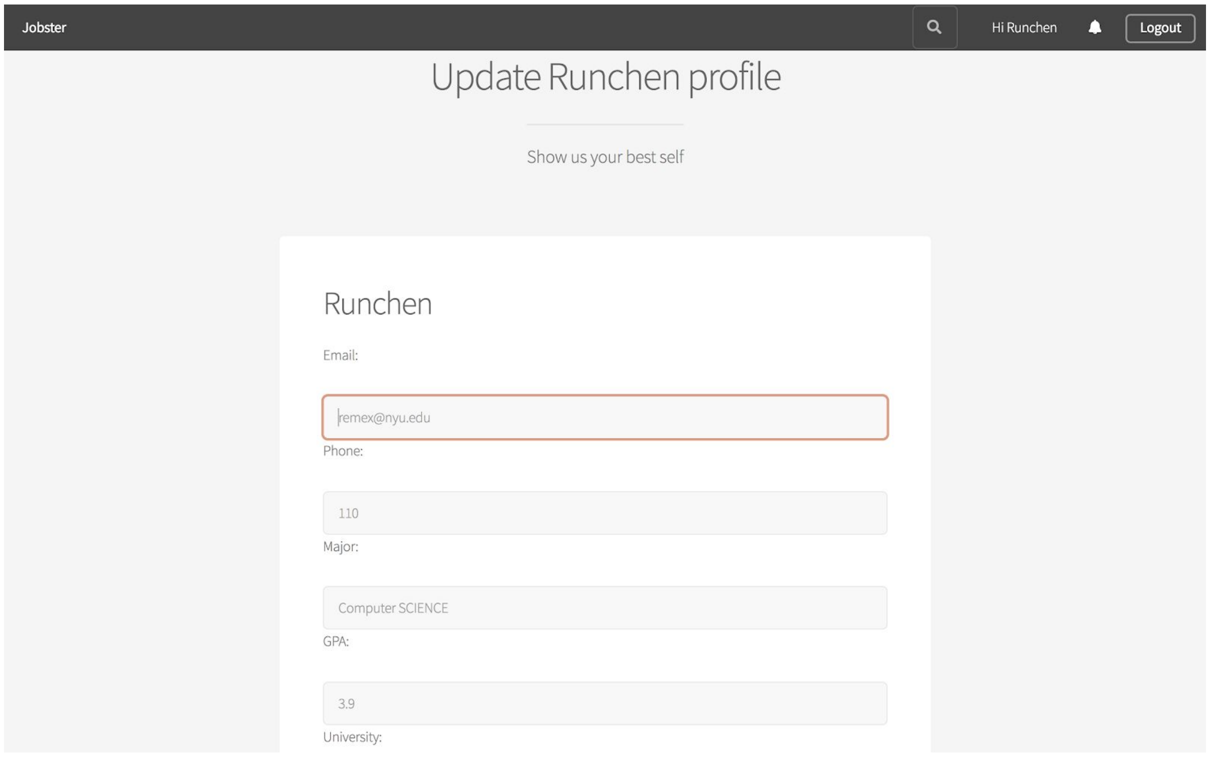
My Application

intern - status: pending

TEST INTERN - status: accepted

AUDIO DESIGN - status: pending

Track Application Status



Update Runchen profile

Show us your best self

Runchen

Email:

remex@nyu.edu

Phone:

110

Major:

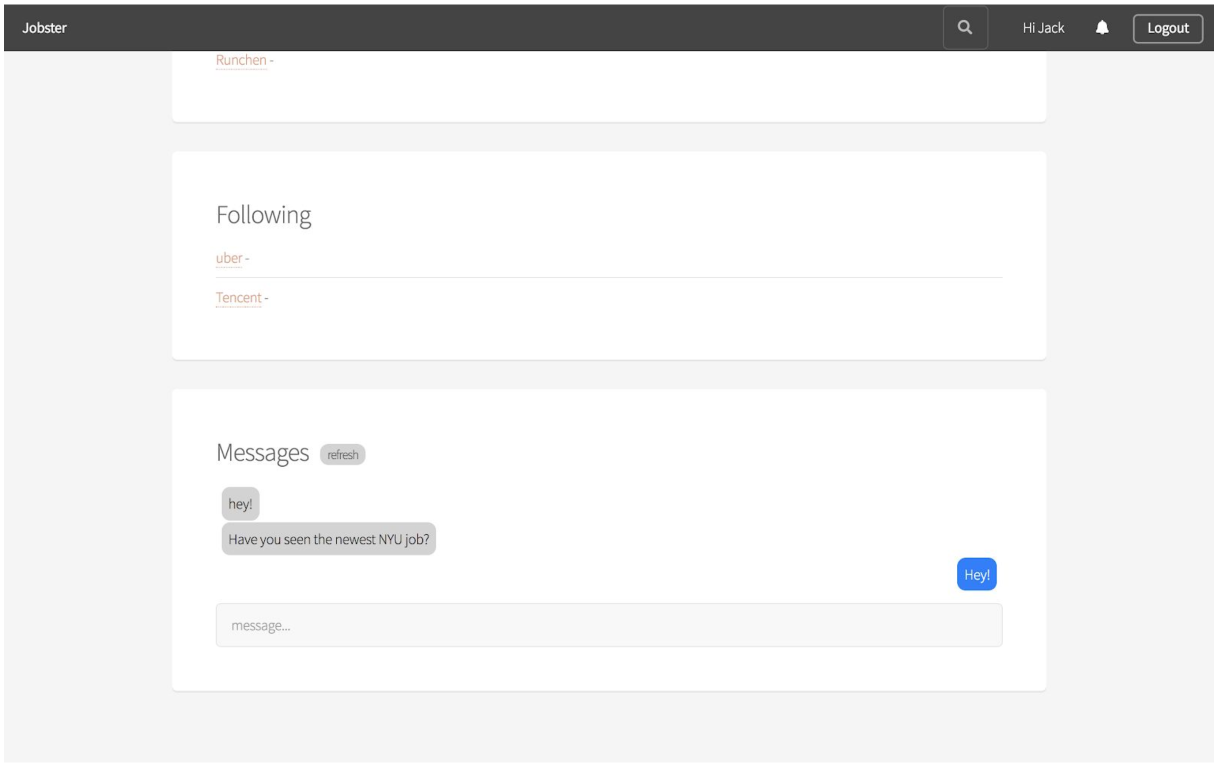
Computer SCIENCE

GPA:

3.9

University:

Student Profile Update



Runchen -

Following

uber -

Tencent -

Messages

hey!

Have you seen the newest NYU job?

Hey!

Message