COS333 Final Project: AACCNJ Job Board

Tanushree Banerjee*, Chris Pan*, Haoyuan He*, Yutong Shen*
* denotes equal contribution

Product Evaluation

Table of Contents

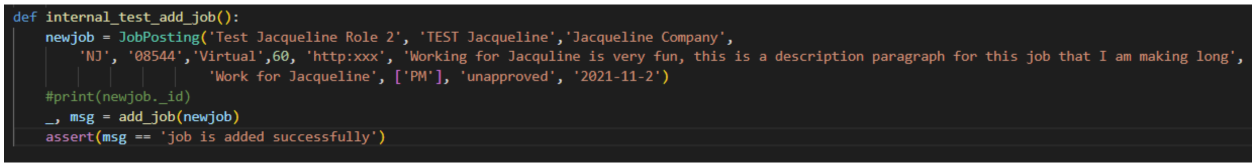
Introduction

The purpose of this document is to present a thorough evaluation of our final product for the class: our three-tier web application, *AACCNJ Opportunities for Success!* In the first part, we cover thorough documentation of the testing we conducted to evaluate the extent to which our product behaves as we intend it to behave. Next, in the second part, we cover a thorough account of evaluation by users in order to understand how well our product meets the needs of its intended users. Lastly, in part 3, we include a thorough evaluation of experts based on the heuristic evaluation and cognitive walkthrough techniques. We include a complete user task list used for evaluating our application in Appendix A. All notes that were taken during the formal evaluation of real users are in Appendix B.

Part 1: Testing

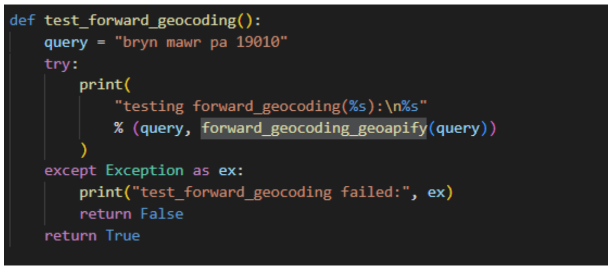
Internal testing

On the database side, we wrote functions for querying/managing data in four tables in the database, including the job_posting, resume_book, company_accounts, admin_accounts tables. Each function was tested by unit test before connecting to the backend jobboardapp.py. Example of unit test:



```python
def internal_test_add_job():
    newjob = JobPosting('Test Jacqueline Role 2', 'TEST Jacqueline','Jacqueline Company',
        'NJ', '08544','Virtual',60, 'http:xxx', 'Working for Jacquline is very fun, this is a description paragraph for this job that I am making long',
                    'Work for Jacqueline', ['PM'], 'unapproved', '2021-11-2')
    #print(newjob._id)
    _, msg = add_job(newjob)
    assert(msg == 'job is added successfully')
```

For the functions used for filter by location functionality based on calling the external API Geoapify, we also wrote unit tests to ensure that our functions are working properly before connecting to the backend jobboardapp.py. Example of unit test:



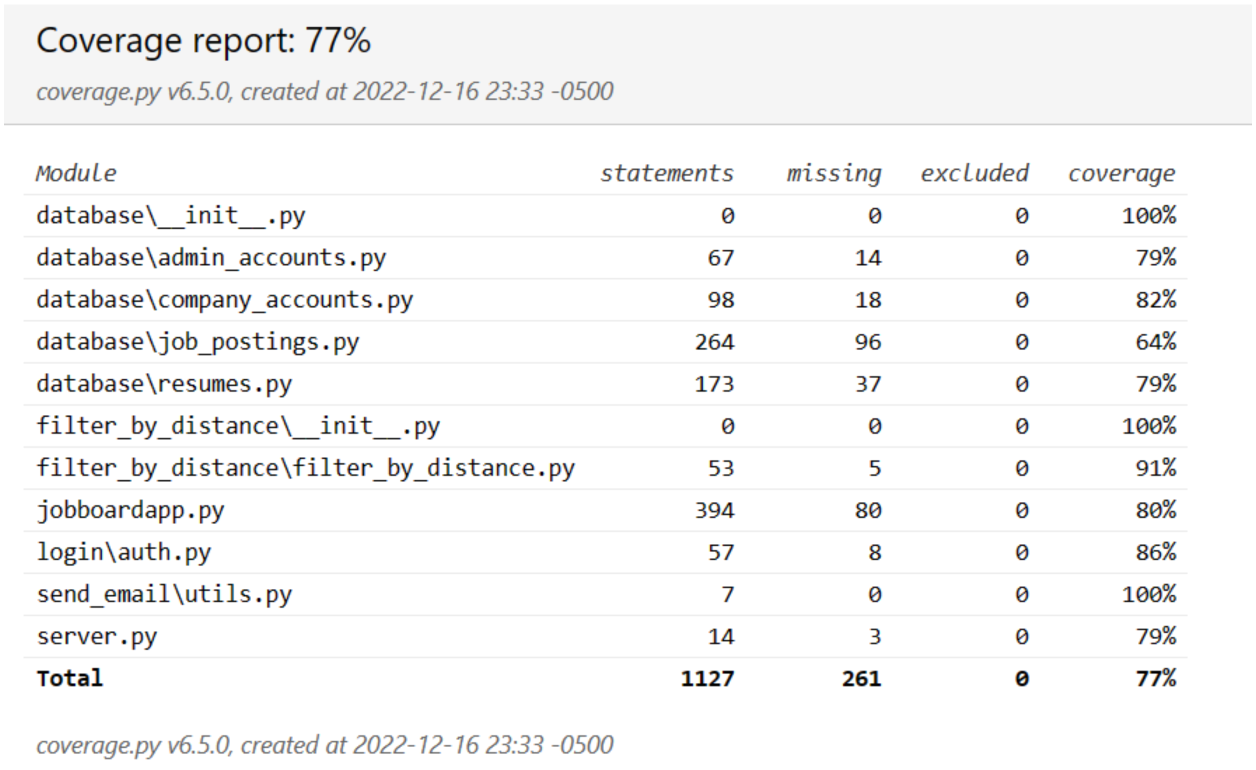
```python
def test_forward_geocoding():
    query = "bryn mawr pa 19010"
    try:
        print(
            "testing forward_geocoding(%s):\n%s"
            % (query, forward_geocoding_geoapify(query))
        )
    except Exception as ex:
        print("test_forward_geocoding failed:", ex)
        return False
    return True
```

On the controller side for jobbaord.app, we did a client unit test to initiate a test client that call get method to get the url in order to ensure the redirection or urls were corrected as expected and the general users could only see the page of job postings, filter by location, add resume, and job postings by company.

White-box external testing

Statement testing

We generated a coverage report for our project.



Coverage report: 77%

coverage.py v6.5.0, created at 2022-12-16 23:33 -0500

Module	statements	missing	excluded	coverage
database__init__.py	0	0	0	100%
database\admin_accounts.py	67	14	0	79%
database\company_accounts.py	98	18	0	82%
database\job_postings.py	264	96	0	64%
database\resumes.py	173	37	0	79%
filter_by_distance__init__.py	0	0	0	100%
filter_by_distance\filter_by_distance.py	53	5	0	91%
jobboardapp.py	394	80	0	80%
login\auth.py	57	8	0	86%
send_email\utils.py	7	0	0	100%
server.py	14	3	0	79%
Total	**1127**	**261**	**0**	**77%**

coverage.py v6.5.0, created at 2022-12-16 23:33 -0500

Boundary Testing

On the database side, each querying/managing data function for job_posting, resume_book, company_accounts, admin_accounts tables is tested by boundary testing. The functions that were tested include:
job_postings.py
get_jobpostings
search_job_by_id,search_job_by_location,search_job_by_company_name,
search_job_by_zipcode, add_job, count_jobs, delete_jobs, approve_job
resume_book_database.py - get_resume
resumes.py
search_resume_by_id, add_resume, approve_resume, delete_resume
admin_accounts.py + company_accounts.py
get_admin_from_db,get_company_from_db,add_to_db,get_admin_accounts,
delete_admin_accounts, get_company_accounts, approve_company_accounts,
delete_company_accounts, get_company_name_logo

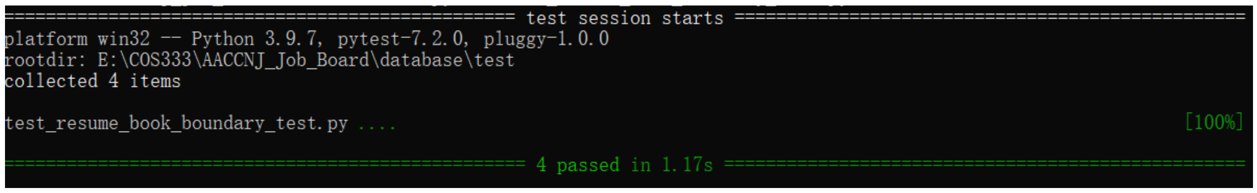
We placed all the boundary testing files under the directory of /database/test. We used a python package called Pytest to support running the test files. The main corner cases we used for boundary testing in each function on database side are:

1. **Empty input:**

The case of empty input was tested when searching in the database with an empty value, and adding an empty object into the database. Our product would still work properly in both cases, with a successful status flag. Furthermore, the case of adding an empty object is avoided from the frontend code since we ensure that each input element that users need to fill out for the job posting/resume information will never be empty. If a user tends to submit the form with empty input, our product will pop up a notification message of “this field could not be empty!” and thus reject submission.

2. **Long input**
   The case of long input was tested when adding objects with fields that are long strings into the database. Our product could work properly with an object that has a string field of 10000 characters, or an object with at most five fields with 5000 characters. If the length of string input exceeds 10000 or the object has more than five fields with 5000 characters, the product tends to get “connection failed” error when connecting to elephantSQL.
3. **Url input**
   The case of having url text in input was tested when adding objects with fields that have url text. Our product could work properly with objects that have url text.
4. **Escape word input**
   The case of input containing escape words were tested for each search function, and our product works properly when users enter queries containing escape words such as “%”,”_”. We used exact match at our search function when querying from the database, so if there is no data inside the table that has the exact format of the input, the search function will return an empty result. Our get functions will return every single row of data to be visualized in the frontend, so the get functions will never have the issue of searching with escape words.
5. **Wrong type input**
   The job posting object has an integer field of “work per hour”, and we tested the case of adding an object with “work per hour” that has a string type. Our product could work properly to return a failure status flag when the integer field has a string type.
   The job posting and resume_book also has a field of “skills” that has a type of list, and we tested the case of adding an object with “skills” that has a string type. Our product could work properly to add the object into the table and regard the string type as a single-item for skills.

Result examples from Pytest:



```
============================================= test session starts =============================================
platform win32 -- Python 3.9.7, pytest-7.2.0, pluggy-1.0.0
rootdir: E:\COS333\AACCNJ_Job_Board\database\test
collected 4 items

test_resume_book_boundary_test.py ....                                                                   [100%]

============================================== 4 passed in 1.17s ==============================================
```

For the functions used for filter by location functionality based on calling the external API Geoapify, we tested on the product page by typing corner case input since these functions are closely integrated and with the frontend code. The main corner cases we used are:

1. **Empty input**
   We tested the case when the input location query or the radius is empty. Our product could work properly with empty input and display the query result for nearby jobs with a default location of “Princeton, NJ” and the radius of 25 km.
2. **Random input**
   We tested the case when the input location query is a random string or a string containing special characters including ‘%’,’_’,’^’. The random input will lead to zero address results returned by the Geoapify API, and the autocompletion will make empty suggestions since the input doesn’t make actual sense. Our product could work properly to handle the empty result obtained by Geoapify API with random input and display an alert box of ‘please enter a valid address!’.
3. **Long input**
   We tested the case when the input location query is a long string with more than 1000 characters. Our product could work properly to handle the long input and display an alert box of ‘please enter a valid address!’. Our product effectively set a length limit to the location query
4. **Wrong type input**

   We tested the case when the input radius is a wrong type that couldn’t be converted to integer type. Our product could work properly to handle the wrong type input and display an alert box of ‘please enter a valid address!’.

Black-box external testing

Use case testing

1. We conducted a use case testing with our major client, Carmen Gates, from the AACCNJ. We sent her the link with the product, and she tried to use the product on her own computer. She was instructed by us to test thoroughly on every page and functionality of the jobboard, with the role of an applicant, an admin, and an employer. During the testing, the product behaved properly without unexpected issues.
2. We conducted a user case testing with three interviewees Diya, Garner, Ameya, each of the interviewees takes the role as the applicant, admin, and the employer. Each interviewee was instructed by us to test thoroughly on the page visible to the corresponding role. During the testing, the product behaved properly without unexpected issues.
3. We conducted a user case testing within the group as programmers. Each of us tested thoroughly on every page for the role of an applicant, an admin, and an employer. One of the teammates, Jacqueline, noticed several cases that might cause inconvenience to the users.

Case as admin:

1. Manage Company Accounts

When the admin tries to approve two rows of company account information with exactly the same information, he couldn't approve both of them. If one row is approved, the other row couldn't be approved. When the admin tries to delete one row, the page will eventually delete both rows.
2. Employer submits long text in input
When employers submit job posting information with extremely long text, the layout of the "Manage Job Posting" page for admin will be affected so that it is harder for admin to delete the job posting.

Stress testing

After figuring out that the unpaid PostGreSQL is unstable for the basic connection, we utilized a connection pool to support querying a large number of calls. The minimum connection is set to 1 and maximum to 10.

We timed each bankend function call with a 100-loop and 300-loop. The result time is shown in tables below

Job Postings

Methods	100 calls(s)	300 calls(s)
search_by_company_name	2.27	7.34
search_job_by_zipcode	2.25	7.25
search_job_by_id	2.39	7.42
search_job_by_location	2.27	7.46
add_job	2.46	7.58
count_jobs	2.52	7.07
approve_job	2.51	7.15
get_job	2.38	7.72

Resume Book

Methods	100 calls(s)	300 calls(s)
search_resume_by_id	2.50	7.45
approve_resume	2.50	7.33
delete_resume	2.45	7.32

get_resume	2.44	7.12
add_resume	2.43	6.86

Login

Methods	100 calls(s)	300 calls(s)
get_admin_from_db	2.64	7.67
get_company_from_db	2.62	7.74
get_adminaccounts	2.51	7.40
get_companyaccounts	2.61	7.47
get_company_name_logo	2.53	7.50
Company.add_to_db	2.37	7.34
Admin.add_to_db	2.40	7.46

We also did stress testing on functions calling Geoapify API for 100 times.

Filter Location

Methods	100 calls(s)
forward_geocoding_geoapify	49.87
get_zipcode	35.28
zipcode_near_users	39.56

Test automation

We use python package Pytest to as the test automation to support running the testing files for boundary and stress testing.

Robustness of the aspects of our product

Robust aspects

1. Security

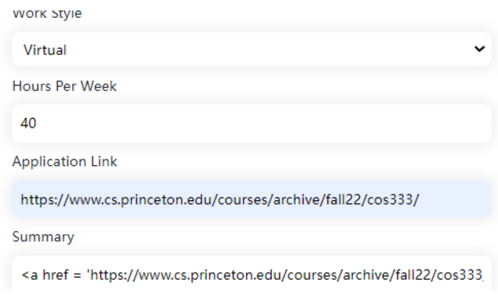
SQL Injection Attack

Our product is robust against the SQL injection attack. All functions used on the database side are based on PostGreSQL prepared statements. If a malicious user inputs SQL commands or strings with single quotes, the product will work properly without getting unexpected errors. In addition, the delete functionality was designed for only admins and company accounts. Users could only delete by clicking on the button but not entering text.

XSS Attack

Our product is robust against the XSS attack. The user-provided strings are sanitized with Jinja2 to ensure that the user input won't attack the web. If a malicious user inputs html elements in the input, the product will display in the text form of the input.

Examples of XSS attack testing:



Work Style

Virtual

Hours Per Week

40

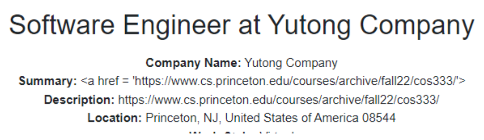
Application Link

https://www.cs.princeton.edu/courses/archive/fall22/cos333/

Summary

<a href = 'https://www.cs.princeton.edu/courses/archive/fall22/cos333

Case: user input a link element for job summary in the input form when submitting a new job posting. The job details page will not display the summary as a link.



Software Engineer at Yutong Company

Company Name: Yutong Company
Summary: <a href = 'https://www.cs.princeton.edu/courses/archive/fall22/cos333/'>
Description: https://www.cs.princeton.edu/courses/archive/fall22/cos333/
Location: Princeton, NJ, United States of America 08544

Login system

Our product utilized the google login for user authentication without storing the password into database tables. Employers/Admin must register with a valid email and password before accessing the employer/admin page. The admins are employees from AACCNJ who are assigned the right to get admin accounts. Employers will be able to submit/manage job postings only after registration and getting approval from admins.

CSRF attack

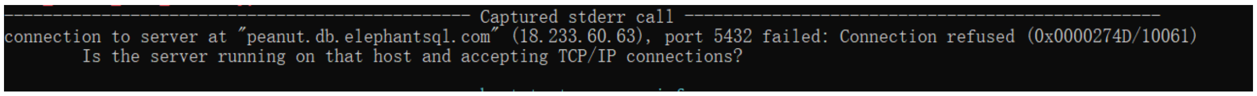
Our product is robust against the CSRF attack with the use of Flask-WTF. The product ensures that POST request it receives was sent from a page that the jobboardapp creates

RECAPTCHA check

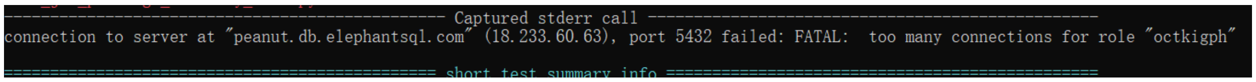
Our product utilizes the Recaptcha check to ensure that only real users could submit their resumes to the jobboard. The Recaptcha check will distinguish between real humans and robots and if a robot tends to submit a resume, the submission will be rejected by failing the Recaptcha check.

Fragile aspects

Unpaid Elephant SQL service



```
----------------------------------------------- Captured stderr call ---------------------------------------------------
connection to server at "peanut.db.elephantsql.com" (18.233.60.63), port 5432 failed: Connection refused (0x0000274D/10061)
        Is the server running on that host and accepting TCP/IP connections?
```



```
----------------------------------------------- Captured stderr call ------------------------------------------------
connection to server at "peanut.db.elephantsql.com" (18.233.60.63), port 5432 failed: FATAL:  too many connections for role "octkigph"

============================================ short test summary info ============================================
```

We used an unpaid version of the Elephant SQL to store the data in three tables based on the three main types of users. During the testing, when executing function calls for more than 100 times, the Elephant SQL platform tended to be unstable and throw the “failed to connect” error. In addition, the Elephant SQL as an online database has the risk of unpredictable server down. The product would be relatively fragile if the request flow gets high, or during the night time when the probability of server down issue increases.

Unpaid external API

We used an incorporated external API called Geoapify for the filter by the address functionality of our product, which is highly dependent on the performance of the API provider’s end. When filtering job addresses within a large radius, such as 1000km, the API will behave slower. Stability would also be a major concern of our product when facing unexpected issues when calling the API.

Known bugs

1. A random bug occurred on the web when deleting/approve a company account with an empty email field. The bug is caused by the database end inserting bad data into the table and is avoided since the frontend ensures companies will never enter empty email during registration. Error message below:
   ```
   in sendmail
   raise SMTPRecipientsRefused(senderrs)
   smtplib.SMTPRecipientsRefused: {'': (555, b'5.5.2 Syntax error. x10-20020a05620a258a00b006fefa5f7fc9sm12744805qko.134 - gsmtp')}
   ```
2. A bug occurred on the web when filtering jobs nearby with a non-integer radius during the development of filter by location functionality using Geoapify API. The bug caused internal server errors that crash the page. The bug was currently fixed with statements to ensure that the radius input from form is integer.
3. A bug occurred on the database side when calling the search functions with an empty id. The bug was caused when the functions return nothing after checking the empty id. The jobboardapp was not able to get the status flag from the search functions. The bug was fixed after adding the return item from the search functions.
4. Database connection failure occurred due to the unstableness of ElephantSQL.
   ```
   connection to server at "peanut.db.elephantsql.com" failed: Connection refused Is the server running on that host and accepting TCP/IP connections?
   ```
5. Database connection failure occurred due to the unstableness of ElephantSQL.(too many connections)
   ```
   connection to server at "peanut.db.elephantsql.com" (18.233.60.63), port 5432 failed: FATAL:  too many connections for role "octkigph"
   ```
6. CSRF token expired error occurred when registering a company account. The bug was caused due to the reason when the page request is stale for security reasons. The bug was solved by reopening the page and connecting to the port.
7. A bug occurred on the Job Openings by Company page that when clicking on the company logo, the layout of the images was moved and changed.
8. A bug occurred when a company registered the account with the company logo, the table in the “Manage Company Accounts” displayed columns in mis-alignment. The bug was caused by forgetting to update the company object class and related PostGreSQL queries.
9. A bug of “No module database found” occurred when running functions in resumes.py. The bug was caused when executing the python file in the “database” directory.
10. A syntax error occurred in the PostGreSQL query statement of the method search_resume_by_id(resume_id) that there is an additional comma. The bug was fixed by correcting the query

Part 2: Evaluation by users

Procedure

User eval (interviews)

1. Interviewed 4 real users in-depth about their experience: Ms Gates, Diya Dalia ‘24, Garner Thompson ‘24, and Ameya Vaidya ‘24.
2. Ms. Gates tested each of the three interfaces, while Diya, Garner and Ameya tested out the job seeker, employer, and admin interfaces respectively.
3. Each user was provided with a short description of our application, along with a specific task list for them to try out on the application.
4. The task list given to each interviewee is in *Appendix A*
5. The exact packet of information sent to each interviewee is included in *Appendix C.*

The exact procedure for each interview is detailed below.

```
For each user:
        (If necessary) give the user a short introduction to the application.
        Ask the user to read the short introduction.
        Confirm that the user understands the short introduction.
        Give the user a written task sequence.
        For each task:
                Ask the user to read the task.
                Confirm that the user understands the task.
                Ask the user to use your application to perform the task.
                Ask (force!) the user to talk aloud while performing the task.
                Take copious notes.
```

User eval (questionnaires/surveys)

1. A survey was sent out to all friends to try out the job seeker interface of our app.
2. Survey link is provided in the Appendix.

Written user evaluation from intended users - representatives from the AACCNJ

1. We asked our contacts from the AACCNJ to provide us with a written evaluation of our system. Their response is attached in the Appendix.

Summary of results

Summary of in-depth interviews with real users

A summary of the findings from the in-depth interviews with real users is provided below, organized by each interface. In-depth notes from each interview are in *Appendix B.*

Job seeker interface

1. Job openings tab: Some of the job title names are not very descriptive or confusing. Moreover the location filter on this tab matches by exact string match, so only NJ is picked up, and new jersey is not picked up. Since we have a different tab to search by location proximity, this filter should be deleted since the difference between the two is ambiguous. Any keyword filter is somewhat redundant and should be removed - its function isn't very obvious, or maybe include a description in placeholder text for the field
2. Add resume tab: Most people store their resumes as PDFs, and not everyone knows how to store their resumes as a link. Moreover, the link to each resume should open up on a new tab. Moreover, no verification is done on the form fields, and it isn't obvious which fields are required. Should add asterisks for fields that are required. After a ReCaptcha error, an option to go back to the form and redo the ReCaptcha verification should exist
3. Browse by company logo tab: When a company hasn't uploaded a logo, nothing should be displayed - currently, a blank box is displayed. When there are no jobs available for a particular company, instead of "no data available" a message that says "no jobs available at this company at the time, please check back again later" should be displayed.
4. View jobs nearby: should ask for radius in miles instead of kilometers, since miles is the default unit of distance in the United States. The autocomplete suggestions are somewhat laggy and the drop down with suggestions show up in an awkward place. The autocomplete feature exists isn't very obvious.
5. Job details page: the application link is sometimes hard to see, since one needs to scroll down. Also, a button to go back to the main job openings page should exist (maybe an arrow button?). Should also have the option to contact the employer directly for more information. Perhaps could be a new feature - where job seeker can send a message to the employer directly through the app and the employer receives an email with the message and email of the job seeker to be able to respond to the job seeker directly).

Employer interface

1. Account approval: after the account is approved, it isn't completely obvious what to do in order to sign in as an employer and access the employer interface - perhaps including instructions in the approval email would be useful. In addition, a link to the employer interface login should be included in that email as well.
2. Posting a job opening: Required fields weren't completely obvious. Should add asterisks to indicate required fields. Role name and title are redundant and one of them should be removed

3. Job postings: link to the job details page is not included (unlike the other interfaces)
4. Deleting job - should ask for confirmation before deleting a job
5. Browse resumes: shouldn't show the array, perhaps display the skills in bubbles or boxes separately.
6. Logging out: should ask for confirmation before logging out

Admin Interface

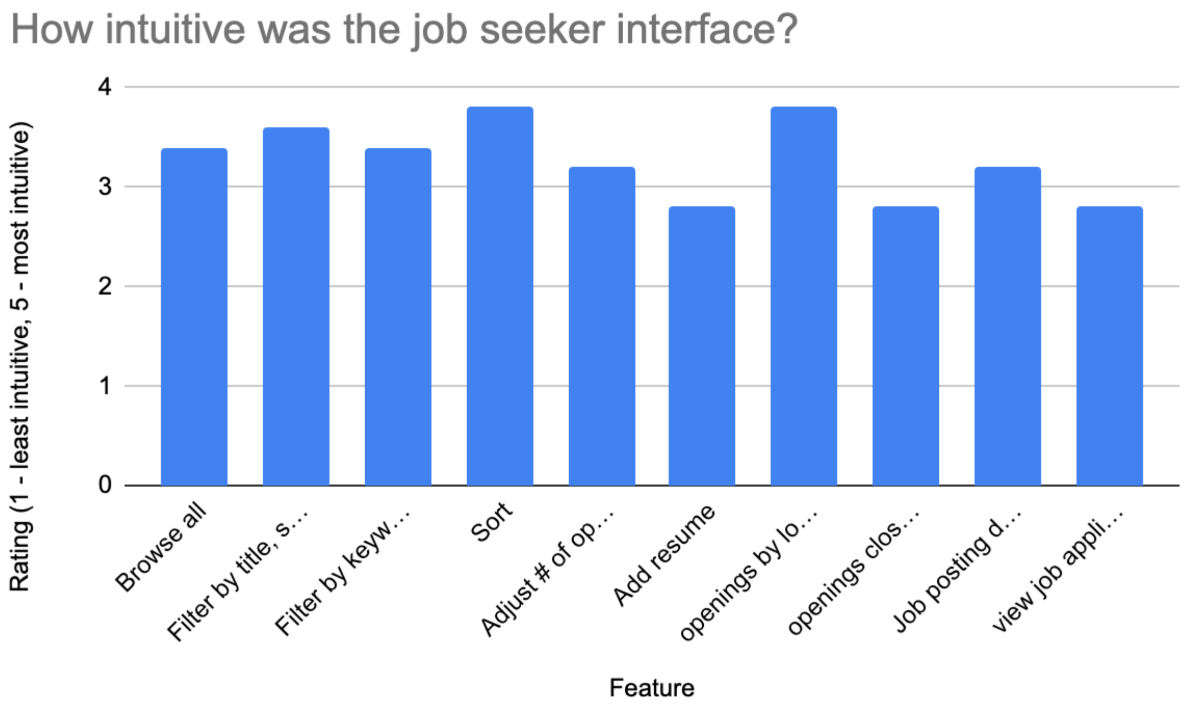
1. Approve/delete jobs and resumes: should ask for confirmation before deleting or approving. Also, it is not immediately obvious whether the approve/delete action went through - perhaps should show a confirmation message in a dialogue box or new page. Once a job is approved, instead of removing the approve button, it should be greyed out instead (indicating that the action is disabled), or an unapprove button should be shown.
2. Adding new admins: currently, it is too easy too do - perhaps have a designated super admin - only this super admin can add or delete new admins, and is able to transfer super adminship to any one other admin before leaving AACCNJ. Alternatively, new admins should request to be approved like employers rather than be manually added by existing admin using the form
3. Once logged in, not obvious which interface youre logged into - should display that on the page header somewhere
4. Sorting jobs: by default, should sort jobs such that unapproved jobs show up on the top by default - right now it is sorted alphabetically by email which is unintuitive. There shouldn't be any sorting enabled on the delete button column

General

1. The green text on the navigation bar should take the user to the "home page" for the respective interface
2. On each page, generally, there should be a button to go back to the 'home page" or previous page for the respective tab
3. Should make the website responsive - doesn't resize well
4. Buttons should be rounded like the login button and should change color when the mouse is hovered over the button
5. A confirmation message after logging out or any other irreversible action such as approve or deleting jobs should be displayed
6. The current tab we are on should be highlighted on the navigation bar

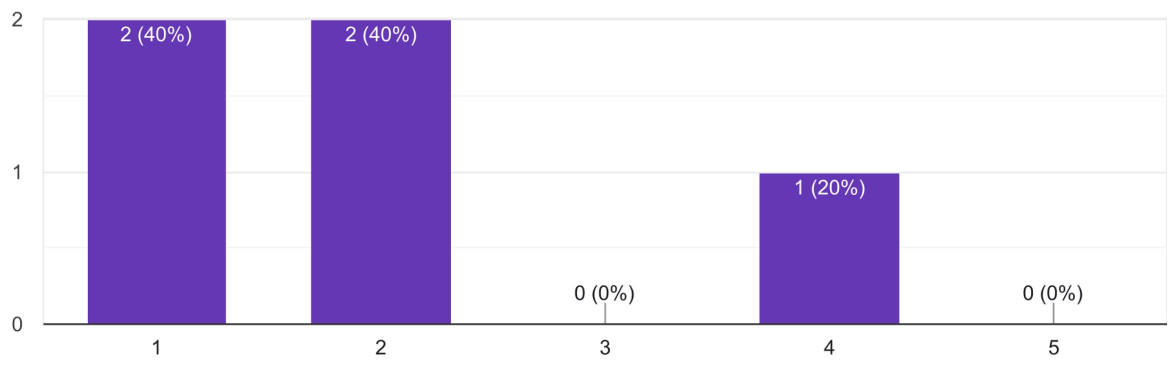
Summary of general survey for job seeker interface:

Generally, the scores submitted by users in the survey suggest that the interface was moderately intuitive to use. More detailed data collected from the survey is included in the Appendix.



How intuitive was this interface to use on the whole?

5 responses



Overall Summary:

The application's basic functionality works well. However, some aspects of the user interface and front-end design could be improved in order to make the interface even more intuitive and easy to use for intended users. Adding confirmation messages once actions like approve, delete, etc are successfully conducted, as well as buttons allowing users to go back to the home page or previous page could be useful.

Part 3: Evaluation by experts

Heuristic evaluation

The heuristic evaluation technique, as developed by Jakob Nielsen, involves evaluating the whole system generally using a set of 10 heuristics (Nielsen, 1994). Our evaluation using each of these 10 heuristics is given below.

Heuristic #1: Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within a reasonable time.

Employer

The employer is kept in the loop about the status of their account and their job postings through email notifications and displayed messages in the web app as appropriate.

1. Employer is notified when their email is not registered to be able to post job postings and access the employer interface and is prompted to fill out a form for approval accordingly.
2. Email notifications are sent to the employer when their account is approved and when their job postings are deleted by the administrator.
3. Employer job posting dashboard clearly shows whether each job posting has been approved or not by the administrator

Administrator

1. Admin is informed about the status of all job postings and resumes submitted, i.e. whether they have been approved or not through the fields in the tables in the relevant tabs on the admin interface
2. However, they are not informed each time a new employer account is created or a new job posting or resume is submitted and is expected to check the platform themselves regularly

Job seeker

1. Job-seeker is not emailed when their resume is approved to be visible to employers on the website.
2. However, the job seeker is informed through appropriate messages on the platform when they fail to complete the ReCaptcha authentication or fill in a required field in any form (such as for submitting resumes)

Summary: users are informed of what's going on in the system for critical functions. However, there is some scope for improvement in keeping users better in the loop about the status of their requests on the system (through appropriate messages in the app as well as through email notifications, or even text notifications).

Heuristic #2: Match between the system and the real world

The system should speak the user's language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. Follow real-world conventions, making information appear in a natural and logical order.

The labels of the table that are displayed in every job posting table are based on the suggestions of Ms. Gates, our AACCNJ contact, who is one of the primary intended users of the system since she would be the primary system administrator of this system once the application is handed over to the AACCNJ. She also closely works with member organizations and is involved with this initiative herself, so understands what language best aligns with the words, phrases and concepts familiar to the user.

We also try to emulate the web apps we have seen in real life and the sequence or order in which things usually work on these systems. For example, the sequence of messages that appear on the screen when an employer is trying to log into the employer interface is based on our experience logging into similar systems that require special access for approval, such as an attempt to log in to Facebook with an account that doesn’t exist yet, so the Facebook page, in this case, displays the appropriate messages and displays a signup form for their website. Since this sequence of screens is logical and appears in many other web apps, users are more used to this sequence of events and need less help trying to navigate the app and its different screens.

Heuristic #3: User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. Support undo and redo

Our app doesn't let employers, job seekers, or admin to undo tasks like approving/deleting a job. Thus, there isn't much support for undoing or redoing actions on the app.

The user is also not prompted with any warning before logging out of the employer or admin screens.

Admin is not asked for confirmation before deleting jobs, resumes, or other admin accounts.

Thus, our application does not perform very well against this heuristic.

Heuristic #4: Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. Follow platform conventions.

Our system follows the principles of consistency. We made sure that the same worrds, situations, or actions mean the same thing. For example, the “Delete” button in every row would mean the same thing: delete this row from the table.

Heuristic #5: Error prevention

Even better than good error messages is a careful design which prevents a problem from occurring in the first place. Either eliminate error-prone conditions or check for them and present users with a confirmation option before they commit to the action.

On the database side, our app utilizes the with statement for connecting to the ElephantSQL database to enable the error catching of database related errors. Functions called by the backend are all wrapped into try-catch statements that enable our app to notify users with pop up alert boxes and page redirection when an error occurs. The design of input when user completes and submits a form ensures that no empty field exists in the object sent to the backend and inserted into the database, which prevents database query errors and displaying errors on the frontend.

Heuristic #6: Recognition rather than recall

Minimize the user's memory load by making objects, actions, and options visible. The user should not have to remember information from one part of the dialogue to another. Instructions for use of the system should be visible or easily retrievable whenever appropriate.

Our app’s design of a navbar instructs users on the functionality of each page of the jobboard. When a user clicks on a tab, the page displayed is easy to recognize and use since the main content of the page corresponds to the title of tab in the navbar. When a user wants to search a job inside the table, he could simply type queries into the search bar and get results. When a user wants to add/delete a job posting or company accounts as an admin, he could simply click on the button with the marks of “add” and “delete”

Heuristic #7: Flexibility and efficiency of use

Accelerators—unseen by the novice user—may often speed up the interaction for the expert user such that the system can cater to both inexperienced and experienced users. Allow users to tailor frequent actions.

On the database side, our app utilizes a connection pool that supports larger flow of database queries with higher speed. Our app is deployed to render so users could open the jobboard with a url link. Google login system in our app will support auto-login after a successful registration and login.

Heuristic #8: Aesthetic and minimalist design

Dialogues should not contain information that is irrelevant or rarely needed. Every extra unit of information in a dialogue competes with the relevant units of information and diminishes their relative visibility.

Our system does not have any extra texts or instructions that are hard to read or redundant. We try to communicate the instructions through images, buttons, and icons as much as possible.

Heuristic #9: Help users recognize, diagnose, and recover from errors

Error messages should be expressed in plain language (no codes), precisely indicate the problem, and constructively suggest a solution.

We have distinct error messages for every type of error we could identify. We're making sure the user will not see errors like "Internal server error" and not know what to do. For example, if the data format in the form the user filled out is not corrent, the system will ask the user to reexamine the data format.

Heuristic #10: Help and documentation

Even though it is better if the system can be used without documentation, it may be necessary to provide help and documentation. Any such information should be easy to search, focused on the user's task, list concrete steps to be carried out, and not be too large.

Our Team has composed the user guide to provide comcrete steps of instructions for each of the user case. We have provide good help and instructions in our app, including the descriptions for each field in the job posting form as placeholders so that users can understand the required inputs and their formats.

Cognitive walkthrough

The cognitive walkthrough method was developed by Cathleen Wharton and Jakob Nielsen (Rogers et al. 2011) in order to evaluate a part of the system in detail. The process involves repeatedly asking the following set of questions.

1. **Will the correct action be sufficiently evident to the user?**

For most functionality, the correct action is sufficiently evident to the user. There are some cases, such as search by any keyword, which is not easily evident how to perform by the user, however.

2. **Will the user know what to do to achieve the task?**

Each tab on the website is descriptively labeled with the main task it helps the user achieve, with links on the navigation bar of the web app. (Add job postings, add resume, etc). This makes it clear to the user that they need to navigate to that tab by clicking on the appropriate tab name on the navigation bar.

3. **Will the user notice that the correct action is available?**

All three types of user (job applicants, employers, admins) will be able to notice that the correct action is available. Each of our interactive elements in the frontend are conspicuous and we provide instructions on what each button or input field will do.

4. **Can users see the button or menu item that they should use for the next action?**

All of the menu items and buttons in our system are conspicuous and self-explanatory. They are also distinguishable from their colors. For example, the delete buttons are in red, and the approve buttons are in green.

5. **Will the user associate and interpret the response from the action correctly?**

Most of our clickable buttons have hover-on effects. They will respond to the user when the cursur interacts with them. The buttons will also respond to the user after clicking them, mainly by showing a success or failure message.

6. **Will users know from the feedback that they have made the correct or incorrect choice of action?**

We made our system as interactive and responsive as possible. If the user made a correct choice of action, such as successful submitted a form, the system would show a massage saying their form has been submitted successfully. If the user made an incorrect choice of action, such as trying to submit a form without filling out all the required fields, the system would pop up a message under the unfilled field and instruct the user to fill it.

Acknowledgments

Special thanks to Ms. Gates from the AACCNJ for her support throughout the project and constant feedback on how we can make our system better and more user-friendly, as well as suggestions for functionality and structure of the interface that would help the application better meet the users' needs.

I would also like to thank Diya Dalia ‘24, Ameya Vaidya ‘24, and Garner Thompson ‘24 for taking the time to let me interview them in depth about their experience using our application, as well as their invaluable insights and feedback on how to make the interface more intuitive for users as well as how to make it better align with the needs of users.

Appendices

Appendix A: User task list

Job seeker

1. Browse job openings
    a. all
    b. by title, company, location, summary, any criteria
    c. View more than 10 job openings on a single page
    d. Sort openings by title, company, location, summary, etc.
2. Add their resume link to the resume book
3. View job openings by company
4. View jobs close to their location (adjusting location and radius of search)
5. View details of a job posting

AACCNJ member (i.e. employer)

1. Logging in as employer using Google login
2. Requesting their company account to be approved for being able to post job openings on the web app
3. Posting a job opening once their account is approved by admin
4. Browse existing job openings posted that are live on the web app
5. View status of posted job openings (Approved/unapproved)
6. Delete old job postings
7. Search job postings by keyword
8. Browse resumes in resume book by keyword
9. View a resume of a prospective employee
10. Log out of employer account

AACCNJ administrator

1. Logging in as admin using google login
2. Approve/delete job postings
3. Sort job postings by status, title, location, etc
4. Approve/delete company accounts that have applied for approval
5. approve/delete resumes submitted
6. Add a new admin account
7. Delete an existing admin account
8. Logging out of the Admin interface

Appendix B: Complete notes during the formal evaluation of real "users"

Interview 1: Ms Gates <> Tanushree

Role 1: Administrator of the AACCNJ Opportunities for success website

1. Logging in as admin using google login
    - Easy and intuitive, just like the normal Google login so is familiar to use

2. Approve/delete job postings
    - Should ask for confirmation before deleting and approving job posting

3. Sort job postings by status, title, location, etc
    - Intuitive, exactly what she had asked us for at the beginning of the project

4. Approve/delete company accounts that have applied for approval
    - Again, we should ask for confirmation before approving or deleting any company account

5. approve/delete resumes submitted
    - Again, should ask for confirmation before approving or deleting any company account

6. Add a new admin account
    - Too easy to add a new admin currently
    - Solution might be to designate one super admin, who has the ability to transfer super-adminship to anyone other admin before leaving AACCNJ for good
    - Only super admin would be able to add or delete admins

7. Delete an existing admin account
    - Same as above - too easy to do, maybe implementing the super admin idea would help solve this

8. Logging out of the Admin interface
    - Should ask for confirmation before doing this as well, otherwise too easy to do and the user could have clicked the button in error

Role 2: AACCNJ chamber member hoping to advertise a new job posting on the AACCNJ opportunities for success web app

1. Logging in as employer using Google login
    a. Again, familiar and easy to use

2. Requesting their company account to be approved for being able to post job openings on the web app
    a. Easy to use, exactly what she envisioned at the beginning of the project
    b. Simplistic interface and easy to use, “I like that”

3. Posting a job opening once their account is approved by admin
    a. Role name and title field are redundant, unclear what the difference is between the two fields

4. Browse existing job openings posted that are live on the web app
    a. Easy to use, great!

5. View the status of posted job openings (Approved/unapproved)
    a. Also intuitive and easy to use!

6. Delete old job postings
    a. Should ask for confirmation before deleting

7. Search job postings by keyword
    a. Also easy to use, exactly what she envisioned earlier

8. Browse resumes in the resume book by keyword
    a. Also intuitive and easy to use

9. View a resume of a prospective employee
    a. Also easy to use, though do people usually store resumes on google drive/know how to store it as a link? Most people just keep it as a pdf
    b. But this link method is a good way to avoid storage costs I suppose
    c. When you view a resume, then we should be able to go directly back to the browse resume page, not the job openings page! The current system is unintuitive since it takes you back to job openings rather than resume book page
    d. Either this or it should take you back to the previous page, or can open the resume in a new tab each time rather than the same tab/window

10. Log out of employer account
    a. Again, should ask for confirmation before doing so, may hit the button in error

Role 3: A potential job seeker looking for job opportunities near where they live, i.e. Trenton NJ

1. Browse job openings
    - all
    - by title, company, location, summary, any criteria
    - View more than 10 job openings on a single page
    - Sort openings by title, company, location, summary, etc.

2. Add their resume link to the resume book
    - Most people store resumes as pdf… don't have resumes stored on google drive as a link

3. View job openings by company
    - When the employer doesn't upload an image, shouldn't see an empty box, instead, we should see nothing at all

4. View jobs close to their location (adjusting location and radius of search)
    - Change kilometers to miles
    - Weird internal server error when not entering space and not selecting autocomplete suggestions from drop-down menu

5. View details of a job posting
    - Couldn't immediately see the application link…

General comments:
- Green text should take you to the home page

Interview 2: Diya <> Tanushree

Role: A job seeker

Scenario: looking for a job close to her home in East Windsor, NJ

Task list, with corresponding comments of the user (notes taken during the interview):

1. Browse all job openings
    - One can clearly see details like name, location, etc.
    - Easy to do, intuitive interface
    - Some of the job titles names are not very descriptive/confusing, e.g. Jacqueline's company

2. Browse job openings by Title, Company, Location, Summary, any keyword
    - Also easy and intuitive

- Location search only matches the string, and “New Jersey” isn't picked up, but NJ is - with filter by location tab is redundant, so you should just remove the option to be able to do that
- Any keyword search is redundant, not obvious that's an option or what that is supposed to even do - better to remove that

3. Adjust the number of job openings viewable on a single page
   - Easy and intuitive

4. Sort openings by Title, Company, Location, and Summary
   - Also easy and intuitive

5. Add your resume to the AACCNJ resume book
   - No verification of form fields…
   - Required fields should have asterisks
   - Should have a button to go back to the homepage (job openings)
   - Option to go back to the previous page after the ReCaptcha error should exist

6. View job openings by a specific company logo
   - When no jobs are available under the company, should show “no openings at this time, check again later” rather than “no data available”

7. View jobs close to your location by adjusting your current location and radius of the search
   - Change to miles and not km
   - Autocomplete suggestions show up in an awkward place, not obvious auto-complete exists and there's a bit of a lag for autocomplete suggestions to show up
   - Internal server error weirdly sometimes - and also when non-integer radius entered, should instead round to nearest integer on our end

8. View details of a job posting
   - The option to go back to the main job openings page should exist (as an explicit back button)

9. View the external application for a job through the job details page
   - External application link sometimes takes the user to a page they don’t expect to see (but that's the link the employer submitted for the job posting…)
   - Option to contact the employer for more information?
   - Need to scroll down to see the link to apply - not always obviously visible

General comments

- Should make the website responsive… doesn't resize well
- Using bootstrap CSS would have been better
- Should have a link to go back to job openings/homepage on every page - make the link obvious
- Remove the footer - comes in the way

Interview 3: Garner <> Tanushree

Role: Owner of PJ's Hotdogs, a local African-American owned business and member of the AACCNJ

Scenario: company based in New Brunswick, NJ. Wants to advertise a new job opening for a cashier at their New Brunswick location, as well as browse for potential candidates that could be a good fit for the role on the AACCNJ resume book database

1. Logging in as an employer using Google login
    - Login is the regular google login so is familiar and easy to use

2. Requesting their company account to be approved for being able to post job openings on the web app
    - After the account is approved, it is not completely obvious how to log in as an employer
    - Maybe a link in the approval email notification should take the user to log in to the portal directly, or at least include a screenshot indicating how to access the employer interface

3. Posting a job opening once their account is approved by admin
    - Wasnt completely clear which fields were required (for e.g. logos aren't required, and some companies may not have a logo)
    - Include an asterisk for required fields
    - Difference between title, role name and one sentence summary isn't clear
    - Remove title field - redundant

4. Browse existing job openings posted that are live on the web app
    - Link to job details page not included in the employer postings page

5. View status of posted job openings (Approved/unapproved)
    - Intuitive and easy to understand and interpret

6. Delete old job postings
    - Also intuitive
    - Should ask for confirmation before deleting job

7. Search job postings by keyword
    - Intuitive and easy to use

8. Browse resumes in the resume book by keyword
    - Also intuitive and easy to use
    - Shouldn't display the skills in an array with square brackets - instead just the contents separated by commas is neater

9. View a resume of a prospective employee
    - Easy and intuitive interface and process

10. Log out of employer account
    - Intuitive, but should ask for confirmation before actually logging out

General comments:

- Remove the footer
- Was able to submit the form for approval to be able to post as an employer twice - shouldn't be allowed to do so

Interview 4: Ameya <> Tanushree

Role: Administrator of the AACCNJ website

Scenario: wants to approve/delete jobs and add a new admin

1. Logging in as admin using google login
    - Good
    - Thought not very obvious that you're in the admin interface and logged in successfully - should say that somewhere on the page

2. Approve/delete job postings
    - Not immediately sure if the approve or delete went through - maybe should show a new confirmation page or confirmation message that the approve or delete went through?
    - When hovering over a button, the color should change
    - Either have an unapprove button once a job is approved, or grey out the approve button once a job is already approved, indicating that action is disabled

3. Sort job postings by status, title, location, etc
    - Simple interface
    - Thought why is the approve button column allowing you to sort? That doesn't make sense - not obvious what that sort does
    - Delete button sort? Weird and not required

    - Job postings should be default sorted to have unapproved jobs on the top - right now the default is alphabetical on the email which is unintuitive and unhelpful

4. Approve/delete company accounts that have applied for approval
    - Company account approval forms submitted twice shows up as an additional entry whose approve button doesn't work since that account is approved already - should not be able to submit the form twice ideally

5. approve/delete resumes submitted
    - Resume links should open up on a new tab
    - Get rid of brackets in skills
    - Maybe show each skill in a separate little box or bubble

6. Add a new admin account
    - Too easy to add - should ask for confirmation?
    - Perhaps change the model such that a person needs to request access to the admin interface, then the current admin accepts or rejects their request (similar to the employer interface)

7. Delete an existing admin account
    - Easy to do - almost too easy
    - Should ask for confirmation before deleting

8. Logging out of the Admin interface
    - Easy and intuitive
    - Maybe also should show a confirmation message

General comments:

1. Remove the footer - it shows up in a weird location
2. Clicking on the green text on the navigation bar should take you back to the “homepage” of the web app
3. Should show what tab you are on by highlighting the appropriate tab name on the navigation bar
4. The job openings table doesn't resize to page size - need to awkwardly scroll instead
5. add/delete buttons should look like the logout button - change color when hovering over/click, rounded edges

Appendix C: User evaluation interview packet

See the next page for the information we gave each user before they tested our system during the in-depth user evaluation interviews.

AACCNJ Opportunities for Success!

User evaluation interview packet

Introduction to our system

AACCNJ Opportunities for Success is a centralized job board that allows members of the African-American Chamber of Commerce of New Jersey (AACCNJ) and other approved employers to publicize job openings at their business on the AACCNJ website, as well as allow job seekers to search for the most relevant jobs for them on the board. It also allows employers to search a database of job seekers based on keywords and view their resumes submitted to the resume book. Lastly, website administrators are able to seamlessly approve and delete job postings and resumes that are live on the website and viewable to other users.

Link to our web app: https://aaccnj-job-board-ton2.onrender.com/

Please join the zoom link below and share your screen.

Zoom link: https://princeton.zoom.us/my/tanushree.banerjee

Please read the brief introduction to our system above and confirm that you understand it.

Task sequence

For each task listed below, please read each task and confirm that you understand it before attempting to perform the task.

Please try to talk aloud while performing each task, or I will force you to do so.

Job seeker

1. Browse all job openings
2. Browse job openings by
    a. Title
    b. Company
    c. Location
    d. Summary
    e. Any keyword
3. Adjust the number of job openings viewable on a single page
4. Sort openings by

    a. Title
    b. Company
    c. Location
    d. Summary

5. Add your resume to the AACCNJ resume book
6. View job openings by a specific company logo
7. View jobs close to your location by adjusting your current location and radius of search
8. View details of a job posting
9. View the external application for a job through the job details page

Link to survey: https://forms.gle/7AfC2jiWf51oWuA5A

Employer

1. Log in as an employer to view the employer interface
2. Submit a request for your company account to be approved for being able to post job openings on the web app
3. Post a new job opening once your account is approved by the admin
4. Browse existing job openings posted by you that are live on the web app
5. View the status of your posted job openings (Approved/unapproved)
6. Delete old job postings
7. Search job postings by keyword
8. Browse resumes in the resume book by keyword
9. View a resume of a prospective employee
10. Log out of employer account

Link to survey: https://forms.gle/7AfC2jiWf51oWuA5A

Administrator

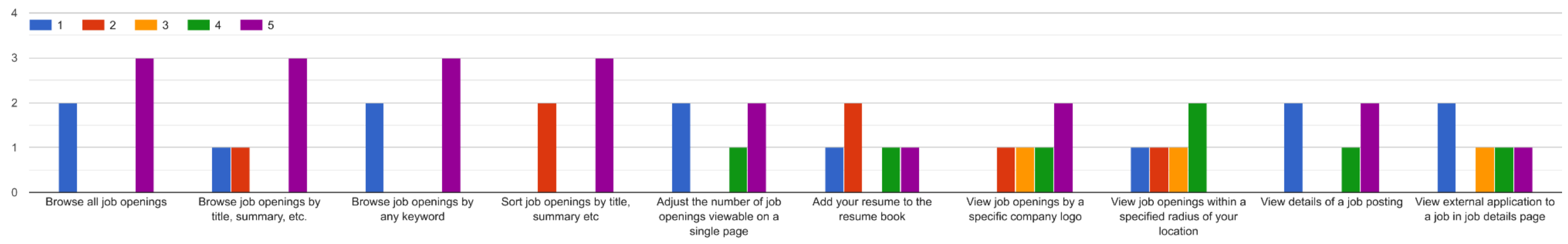
1. Log in as an admin to view the admin interface
2. Approve/delete job postings
3. Sort job postings by
    a. Status
    b. Title
    c. Location, etc
4. Approve/delete company accounts that have applied for approval
5. Approve/delete resumes submitted
6. Add a new admin account
7. Delete an existing admin account

Link to survey: https://forms.gle/7AfC2jiWf51oWuA5A

Appendix D: User evaluation general survey

Link to form: https://docs.google.com/forms/d/e/1FAIpQLScw25XqL10ivy9x1JRDMqOI_t6nx3HjV0cNS9HjSE98x_LH7Q/viewform

For each of the following features that you tried, how easy and intuitive were they? (on a scale of 1 to 5, with 1 being very unintuitive and 5 being very intuitive)



	For each of the following features that you tried, how easy and intuitive were they? (on a scale of 1 to 5, with 1 being very unintuitive and 5 being very intuitive)

How intuitive was this interface to use on the whole?	[Browse all job openings]	[Browse job openings by title, summary , etc.]	[Browse job openings by any keyword]	[Sort job openings by title, summary etc]	[Adjust the number of job openings viewable on a single page]	[Add your resume to the resume book]	[View job openings by a specific company logo]	[View job openings within a specified radius of your location]	[View details of a job posting]	[View external application to a job in job details page]	Improvement suggestions
1	5	5	5	5	5	4	5	4	5	5	“Filter locations" and "Search by location" seem redundant to me (not sure if they are the same or not); "add resume" could be separated maybe
2	1	2	1	2	1	1	3	2	1	1	
4	5	5	5	5	4	5	4	3	4	4	for locations, indicating xxx miles away from user's current location or incorporating a map view would be more intuitive
2	5	5	5	5	5	2	5	4	5	3	I might be dumb but I'm not sure how to make a link for a resume. Other websites usually allow for uploading files. The specified radius of location filtering is in km which might be non intuitive to many Americans. The button for "Click here to apply!" often gets covered (and unable to be used) by

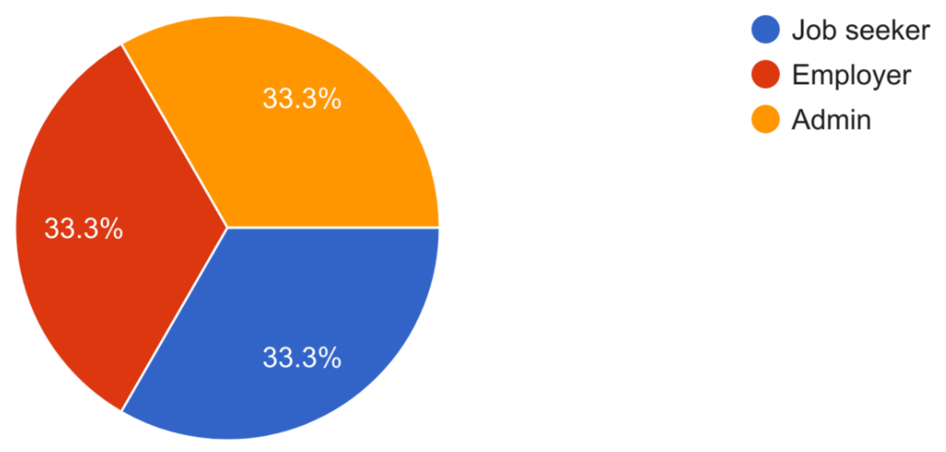
											the credit text ("Created by... etc"). I think that should move with the page so it stays below everything. Also this survey is slightly non-intuitive because the first question has 5 being not intuitive but the rest of the questions are opposite lol. Very cool website though, I like the responsiveness of the search boxes. I will consider applying for senior manager for Chris.
1	1	1	1	2	1	2	2	1	1	1	

Appendix E: User evaluation interview survey

Link to form: https://docs.google.com/forms/d/e/1FAIpQLScAmGiW0f6khbI9QbRxN8I0CCTQAJSWa6Rdl6qsHtMS4AW21g/viewform

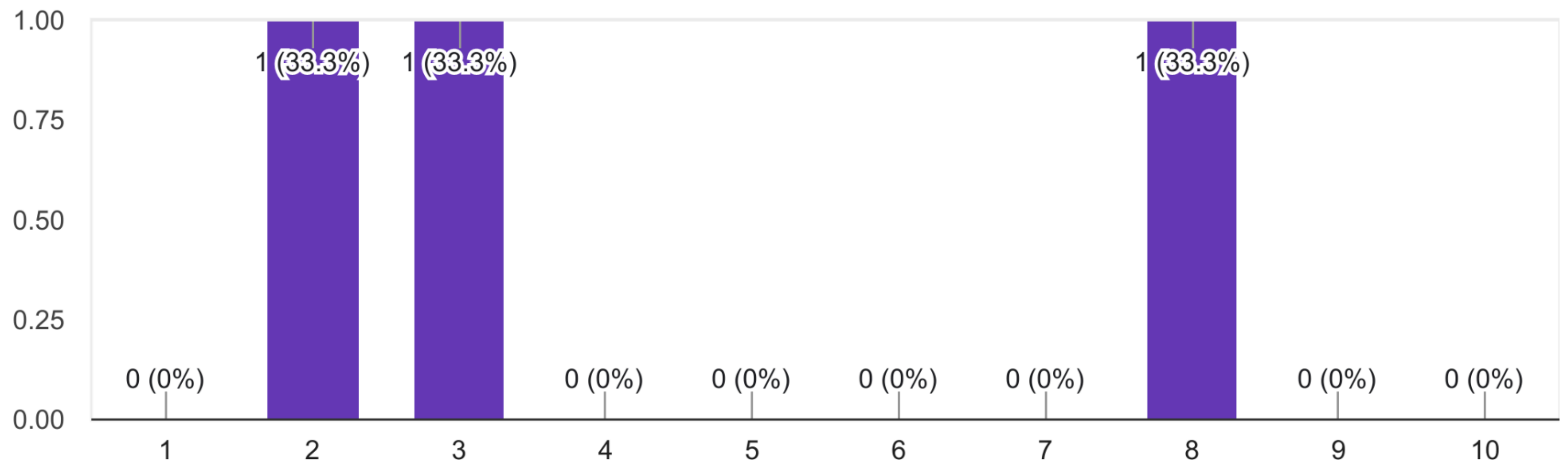
Which interface did you test out for us just now?

3 responses



How intuitive was this interface to use on the whole?

3 responses



Timestamp	Email Address	Which interface did you test out for us just now?	How intuitive was this interface to use on the whole?	How well does your product meet the needs of its users?	For any features you feel that were not easy/intuitive to use, please let us know if you have any ideas for how they could be made more intuitive to use that you didn't already mention during the interview, or if you have any other

					thoughts/ideas/comments.
12/14/2022 19:34:27	ddalia@princeton.edu	Job seeker	2	8	Some of the links were hard to click
12/14/2022 20:29:44	mgt2@princeton.edu	Employer	2	8	
12/16/2022 2:49:31	avaidya@princeton.edu	Admin	3	8	

Appendix F: User evaluation interview survey

Hello Yutong,

The newly created job opportunities site is a job well done. We'll need to add the safety catches on the logout and delete buttons ("Are you sure you want to logout").

Congratulations on delivering just what we ordered.

Ferlanda's evaluation/comments are below.

Thank you.

Hi Carmen,

This looks really good. The search functions cover all search logic iterations. The response rate is fast. The design is sleek and user-friendly. The prototype satisfies all aspects of the scope of work we submitted. Congratulations to the students and you on a job well done!!

(We will need to update the logo with the most recent U.S. Chamber of Commerce accreditation insignia.)

Again, kudos!

Regards,
Ferlanda