

Job Analytics Testing Documentation

short line

Daniel O’Sullivan  
Student Number: 14307881

Supervisor: Suzanna Little

Table of Contents

* Introduction………………………………………………………………………………………………2
  + Problems my project presents……………………………………………………...2
  + Frameworks and libraries used……………………………………………………..2
* Unit Tests…………………………………………………………………………………………………..3
  + Models…………………………………………………………………………………………….3
  + Forms……………………………………………………………………………………………….7
* Integration Tests……………………………………………………………………………………….8
  + Views in Django……………………………………………………………………………..8
  + Tests………………………………………………………………………………………………..8
* Coverage Report……………………………………………………………………………………..12
* End-to-End Tests……………………………………………………………………………………..13
  + Test Cases……………………………………………………………………………………..13
* Machine Learning algorithm testing……………………………………………………...14
* User Testing…………………………………………………………………………………………….18
  + User Interface Questionnaire……………………………………………………….18
  + Employer and applicant user testing………………………………………….24

Introduction

In this document, I will discuss the numerous tests I carried out in the development of my application, Job Analytics.

Problems my project presents

There are a number of potentials problems in making a python job posting application. First and foremost the application should be reliable, meaning that it should it carry out the tasks that it was implemented to do when requested. Not only that but it should carry out these tasks flawlessly without error (to some degree). For my project, there are a number of key constraints that need to be tested. Some examples are:

* The CV parser of my application should only take one format of curriculum vitae.
* Requirements must be specified in a specific format.
* Applicants shouldn’t be able to apply for a job twice.
* When summarizing applicant details, the data should be accurate.
* Views should be restricted to users who aren’t logged in or don’t have permission to see them
* User should only be able to delete jobs they have created, along with the applicants for those jobs.

Frameworks and libraries used

**Unittest**

Unittest (also known as PyUnit) is a unit testing framework written in python. It is based off the Java testing framework, JUnit. Unittest provides features such as test automation, sharing of setup and shutdown code for tests and independence of tests from the reporting framework.

**Sikuli**

Sikuli is an automation framework written in Jython (a mix between Python and Java). Sikuli automates anything you see on the screen. It uses image recognition to identify and control GUI components. Very useful for end-to-end tests.

**Coverage.py**

Coverage.py is a tool for measuring code coverage of Python programs. It monitors your program, noting which parts of the code have been executed, then analyzes the source to identify code that could have been executed but was not. Good for measuring how much of your code unit tests cover. It’s good for measuring integration test coverage.

Unit Tests

For unit testing, I used the python unit testing framework unittest. I only tested any code or component that i have implemented myself. I won’t be testing any django aspects of my project e.g. testing whether a django import works or not, as this would already extensively tested by Django themselves.

Models

**JobPosting Model**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Name** | **Description** | **Criticality** | **Inputs** | **Expected Result** |
| 1 | test\_job\_title | Tests that the field job\_title in JobPosting is given proper label | Medium | JobPosting object’s field label for job\_title | Field label should match |
| 2 | test\_employer | Tests that the field employer is given proper label | Medium | JobPosting object’s field label for employer | Field label should match |
| 3 | test\_address\_line\_1 | Tests that the field address\_line\_1 is given proper label | Low | JobPosting object’s field label for address\_line\_1 | Field label should match |
| 4 | test\_address\_line\_2 | Tests that the field address\_line\_2 is given proper label | Low | JobPosting object’s field label for address\_line\_2 | Field label should match |
| 5 | test\_city | Tests that the field city is given proper label | Medium | JobPosting object’s field label for city | Field label should match |
| 6 | test\_county | Tests that the field county is given proper label | Medium | JobPosting object’s field label for county | Field label should match |
| 7 | test\_employer\_logo | Tests that the field employer\_logo is given proper label | Low | JobPosting object’s field label for employer\_logo | Field label should match |
| 8 | test\_job\_description | Tests that the field job\_description is given proper label | Medium | JobPosting object’s field label for job\_description | Field label should match |
| 9 | test\_jobposting\_str | Test that the jobposting str() returns the expected string | Low | JobPosting object | String should match |
| 10 | test\_jobposting\_valid\_creation | Tests that a JobPosting object is created with valid field inputs | High | Valid inputs for creating a JobPosting object | JobPosting object should be created |
| 11 | Test\_jobposting\_invalid\_creation | Tests that you cannot create a JobPosting object with invalid field inputs | High | Invalid inputs for creating a JobPosting object | JobPosting object shouldn’t be created |

**Requirement Model**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Name** | **Description** | **Criticality** | **Inputs** | **Expected Result** |
| 12 | test\_requirement\_detail | Test that the field requirement\_detail is given the proper label | Medium | Requirement object’s field label for requirement\_detail | Field label should match |
| 13 | test\_requirement\_type | Test that the field requirement\_type is given the proper label | Medium | Requirement object’s field label for requirement\_type | Field label should match |
| 14 | test\_requirement\_keyword | Test that the field requirement\_keyword is given the proper label | Medium | Requirement object’s field label for requirement\_keyword | Field label should match |
| 15 | test\_requirement\_str | Test that the requirement str() returns the expected string | Low | Requirement Object | String should match |

**UserApplication Model**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Name** | **Description** | **Criticality** | **Inputs** | **Expected Result** |
| 16 | test\_first\_name | Test that the field first\_name is given the proper label | Medium | UserApplication object’s field label for first\_name | Field label should match |
| 17 | test\_last\_name | Test that the field last\_name is given the proper label | Medium | UserApplication object’s field label for last\_name | Field label should match |
| 18 | test\_email | Test that the field email is given the proper label | Medium | UserApplication object’s field label for email | Field label should match |
| 19 | test\_phone\_number | Test that the field phone\_number is given the proper label | Medium | UserApplication object’s field label for phone\_number | Field label should match |
| 20 | test\_education | Test that the field education is given the proper label | Medium | UserApplication object’s field label for education | Field label should match |
| 21 | test\_work\_experience | Test that the field work\_experience is given the proper label | Medium | UserApplication object’s field label for work\_experience | Field label should match |
| 22 | test\_skills | Test that the field skills is given the proper label | Medium | UserApplication object’s field label for skills | Field label should match |
| 23 | test\_satisfaction\_level | Test that the field satisfaction\_level is given the proper label | Medium | UserApplication object’s field label for satisfaction\_level | Field label should match |
| 24 | test\_number\_of\_projects | Test that the field number\_of\_projects is given the proper label | Medium | UserApplication object’s field label for number\_of\_projects | Field label should match |
| 25 | test\_average\_salary | Test that the field average\_salary is given the proper label | Medium | UserApplication object’s field label for average\_salary | Field label should match |
| 26 | test\_work\_accident | Test that the field work\_accident is given the proper label | Medium | UserApplication object’s field label for work\_accident | Field label should match |
| 27 | test\_promotion\_in\_last\_5\_years | Test that the field promotion\_in\_last\_5\_years is given the proper label | Medium | UserApplication object’s field label for promotion\_in\_last\_5\_years | Field label should match |
| 28 | test\_performance | Test that the field performance is given the proper label | Medium | UserApplication object’s field label for performance | Field label should match |
| 29 | test\_match\_ratio | Test that the field match\_ratio is given the proper label | Medium | UserApplication object’s field label for match\_ratio | Field label should match |
| 30 | test\_userapplication\_str | Test that the userapplication str() returns the expected string | Low | UserApplication Object | String should match |

**MatchedRequirement Model**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Name** | **Description** | **Criticality** | **Inputs** | **Expected Result** |
| 31 | test\_match | Test that the field match is given the proper label | Medium | MatchedRequirement object’s field label for match | Field label should match |

**CVUpload Model**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Name** | **Description** | **Criticality** | **Inputs** | **Expected Result** |
| 32 | test\_file\_field | Test that the field file is given the proper label | Medium | CVUpload object’s field label for first\_name | Field label should match |
| 33 | test\_education\_section | Test that the field education\_section is given the proper label | Medium | CVUpload object’s field label for education\_section | Field label should match |
| 34 | test\_work\_experience\_section | Test that the field work\_experience\_section is given the proper label | Medium | CVUpload object’s field label for work\_experience\_section | Field label should match |
| 35 | test\_cvupload\_str | Test that the CVUpload str() returns the expected string | Low | CVUpload Object | String should match |

Forms

**ApplicantForm**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Name** | **Description** | **Criticality** | **Inputs** | **Expected Result** |
| 36 | test\_form\_with\_valid\_number\_without\_plus | Test that the form can be filled out with a phone number variation that doesn’t begin with a ‘+’ | High | Form data with valid phone number format that doesn’t begin with a ‘+’ | Form inputs should be valid |
| 37 | test\_form\_with\_valid\_number\_with\_plus | Test that the form can be filled out with a phone number variation that begins with a ‘+’ | High | Form data with valid phone number format that begins with a ‘+’ | Form inputs should be valid |
| 38 | test\_form\_with\_invalid\_number | Test that the form can’t be filled out with a invalid phone number | High | Form data with an invalid phone number format | Form inputs should be invalid |

Integration Tests

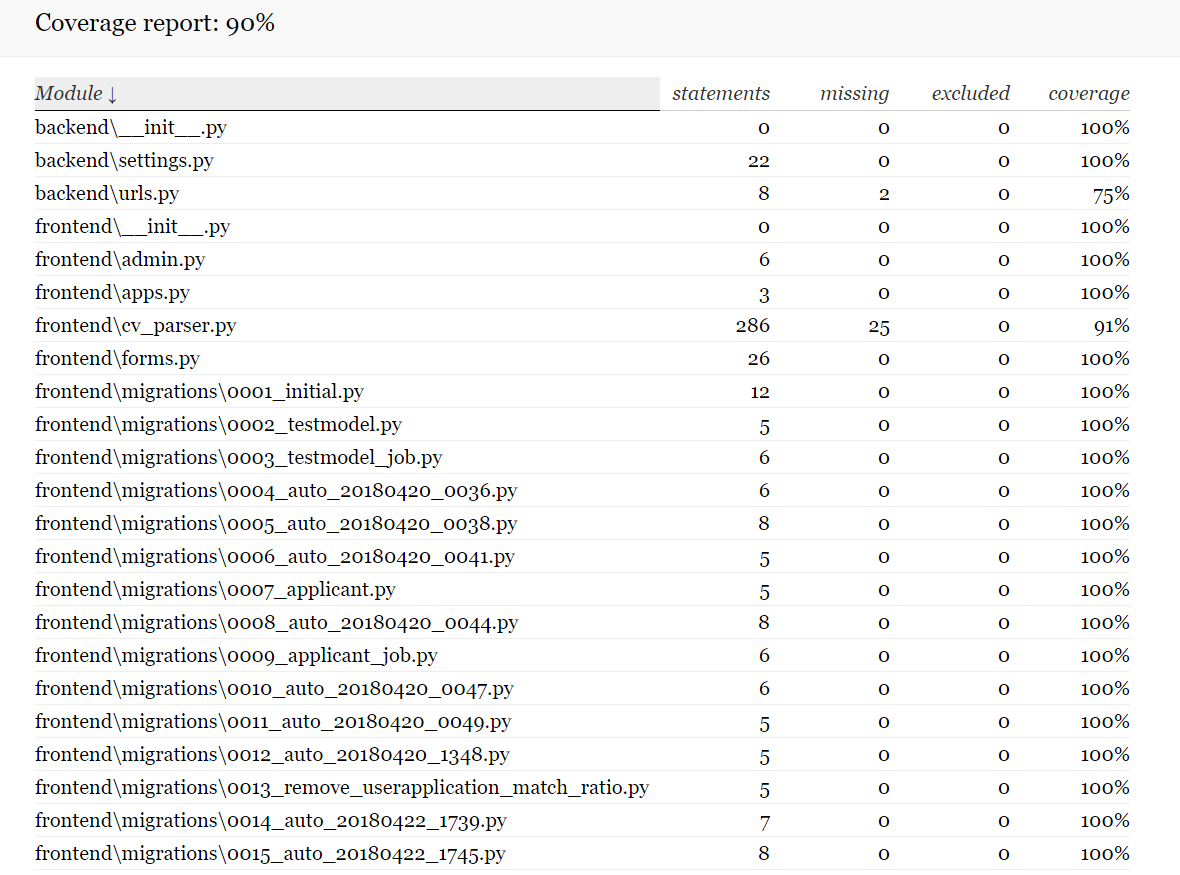
Views in Django

The views in my django application are responsible for calling all component functions. Models, forms and any external library or framework is called inside a view function. Therefore, carrying out unit tests on these views will test all the components and their corresponding functions, integrated together. Therefore, they can be regarded as integration tests.

Tests

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Test ID** | **Test Name** | **Description** | **Criticality** | **Inputs** | **Expected Result** |
| 39 | test\_index\_view | Tests the index view for a successful response | High | Request from a mocked client | Should return HTTP response code 200 |
| 40 | test\_register\_view | Tests the register view for a successful response | High | Request from a mocked client | Should return HTTP response code 200 |
| 41 | test\_login\_page\_view | Tests the login page view for a successful response | High | Request from a mocked client | Should return HTTP response code 200 |
| 42 | test\_logout\_view | Tests the logout view for a successful response | High | Request from a mocked client | Should return HTTP response code 302 |
| 43 | test\_job\_board\_view | Tests the job board view for a successful response | High | Request from a mocked client | Should return HTTP response code 302 |
| 44 | test\_job\_board\_view\_query | Tests the job board view with a search query for a successful response | Medium | Request from a mocked client, along with two queries for job searches | Should return HTTP response code 200 for both queries |
| 45 | test\_add\_job | Tests the add job view for a successful response when a user is logged on and redirect response when a user isn’t | High | Request from a mocked client and user login | Should return HTTP response code 200 when user logged in and a 302 response code when not logged in |
| 46 | test\_add\_job\_with\_post | Tests the add job view with form data to post a job | High | Request from a mocked client, user login and input data for add job form in view | Should return HTTP response code 302 and form data should be valid |
| 47 | test\_cv\_format | Tests the cv format view for a successful response | High | Request from a mocked client | Should return HTTP response code 200 |
| 48 | test\_data\_report\_with\_redirect | Tests that the data report view is restricted to users who aren’t logged in | Medium | Request from a mocked client | Should return HTTP response code 302 |
| 49 | test\_data\_report\_with\_access | Tests that the data report view is accessible for users who are logged in | Medium | Request from a mocked client | Should return HTTP response code 200 |
| 50 | test\_previous\_posts | Tests the previous post view can be accessed by a logged on user | Medium | Request from a mocked client and user login | Should return HTTP response code 200 |
| 51 | test\_previous\_posts\_with\_redirect | Tests the previous post view is restricted to users who aren’t logged on | Medium | Request from a mocked client and user login | Should return HTTP response code 302 |
| 52 | test\_add\_job\_with\_previous\_posts | Tests the previous post view can be accessed by a logged on user | Medium | Request from a mocked client and user login | Should return HTTP response code 302 |
| 53 | test\_user\_dashboard\_redirect\_if\_not\_logged\_in | Tests user dashboard can’t be accessed to users who aren’t logged on | High | Request from a mocked client | Should return HTTP response code 302 |
| 54 | test\_login\_with\_valid\_data | Tests that an existing user can log in | High | Request for a mocked client and user login data | User status should be active |
| 55 | test\_login\_with\_invalid\_data | Tests that a non existing user can’t login | High | Request from a mocked client and user login data | User status should not be active |
| 56 | test\_registration | Tests that a user can register | High | Request from a mocked client and registration data | Should return HTTP response code 302 and registration data valid |
| 57 | test\_registration\_with\_invalid\_data | Tests that a user can’t register with invalid data | High | Request from a mocked client and registration data | Should return error |
| 58 | test\_job\_info | Tests job info view | High | Request from a mocked client | Should return HTTP response code 200 |
| 59 | test\_job\_info\_with\_logged\_in\_user | Tests job info with logged on user | High | Request from a mocked client and user login | Should return HTTP response code 200 |
| 60 | test\_cv\_upload\_view | Tests cv upload view | High | Request from a mocked client | Should return HTTP response code 200 |
| 61 | test\_cv\_upload\_with\_valid\_data | Test uploading a valid cv through the cv upload view | High | Request from a mocked client and properly formatted cv | Should return HTTP response code 302 and CV form data should be valid |
| 62 | test\_cv\_upload\_with\_invalid\_data1 | Test uploading an invalid cv through the cv upload view | High | Request from a mocked client and improperly formatted cv | Should return HTTP response code 200 and CV form data should be invalid |
| 63 | test\_cv\_upload\_with\_invalid\_data2 | Test uploading an invalid cv through the cv upload view | High | Request from a mocked client and improperly formatted cv | Should return HTTP response code 200 and CV form data should be invalid |
| 64 | test\_applicant\_form\_view | Tests applicant form view | High | Request from a mocked client | Should return HTTP response code 200 |
| 65 | test\_applicant\_form\_view\_with\_valid\_data | Tests applicant view and posting valid data to form | High | Request from a mocked client and valid applicant form data | Should return HTTP response code 302 and applicant form data should be valid |
| 66 | test\_applicant\_form\_view\_with\_invalid\_data | Tests applicant view and posting invalid data to form | High | Request from a mocked client and invalid applicant form data | Should return HTTP response code 302 and applicant form data should be invalid |
| 67 | test\_applicant\_form\_view\_with\_existing\_applicant | Tests applicant view and posting data of a user who has already applied for the job in question | Medium | Request from a mocked client and existing applicant form data | Should return HTTP response code 302 and applicant form data should be invalid |
| 68 | test\_application\_form\_view | Tests application view and valid application data | High | Request from a mocked client and valid application form data | Should return HTTP response code 302 and application form data should be valid |
| 69 | test\_applicant\_list\_view\_with\_redirect | Tests that applicant list view is restricted to users who aren’t logged in | High | Request from a mocked client | Should return HTTP response code 302 |
| 70 | test\_applicant\_list\_view\_with\_login | Tests that applicant list view is not restricted to users who are logged in | High | Request from a mocked client and user login | Should return HTTP response code 200 and user status should be active |
| 71 | test\_update\_job\_view | Tests update job view with user logged in | High | Request from a mocked client and user login | Should return HTTP response code 200 and user status should be active |
| 72 | test\_update\_job\_form | Tests the update job form while an applicant is logged on | High | Request from a mocked client, user login and valid updata job form data | Should return HTTP response code 200, user status should be active and form data should be valid |
| 73 | test\_applicant\_info\_view | Tests that applicant info view is not restricted to users who are logged in | High | Request from a mocked client and user login | Should return HTTP response code 200 and user status should be active |
| 74 | test\_delete\_job\_view | Tests that delete job view is not restricted to users who are logged in | High | Request from a mocked client and user login | Should return HTTP response code 200 and user status should be active |

Coverage Report



In total, my unit and integration tests cover 90% of my application code. The coverage of the application’s individual main components are:

* **CV Parser:** 91%
* **Forms.py:** 100%
* **Models.py:** 100%
* **Random Forest Classifier:** 80%
* **Views.py:** 74%

End-to-End Tests

I used the Sikuli automation framework to conduct end-to-end tests on my application. The goal of these tests were to test the flow of tasks in my application. I wanted to construct test cases that would execute as much user tasks as possible and carry out these tasks in different orders and variations to test all possible flow of user actions. Anything that was created during the execution of these scripts is deleted once the test is finished. These scripts were ran several times a day and after there was any major change regarding the application’s functionality or a UI change.

Test Cases

Test Case 1: User creation and job posting

**Steps (flow of user tasks):**

1. Register a new user in the application
2. Log on with this newly created user
3. Create a job posting
4. Delete the job
5. Delete User

**Functionality tested:**

* User Registration
* User login
* Job creation
* Job Deletion
* User Deletion

Test Case 2: Apply for a job then view applicants

**Steps (flow of user tasks):**

1. Search for job using the search bar
2. Apply for a job
3. Login as job poster
4. View applicant list
5. View applicant summary
6. Delete applicant
7. Logout

**Functionality tested:**

* Job search queries
* Applying for a job (which tests cv parser, machine learning and form validation)
* User login
* Statistics shown in applicant list
* Applicant summaries
* Applicant deletion
* User logout

Test case 3: Apply for job with existing applicant and view applicant

**Steps (flow of user tasks):**

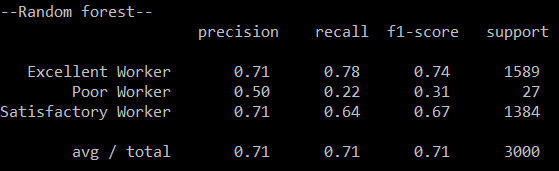
1. Search for job using the search bar
2. Apply for job with CV of applicant who has already applied for this job
3. Login as job poster
4. View existing applicant
5. Logout

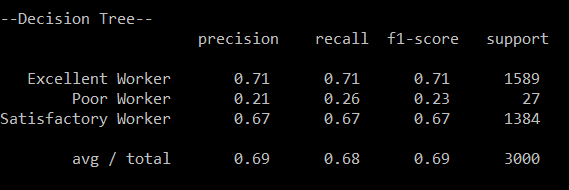
**Functionality Tested**

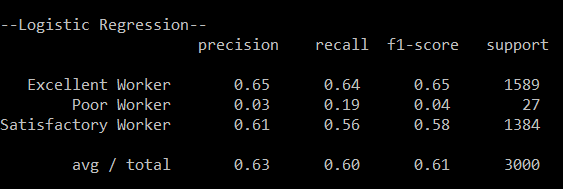
* Job search queries
* Applying for a job with rejection due to applicant attempting to apply twice
* User login
* Statistics shown in applicant list
* Applicant summaries
* User logout

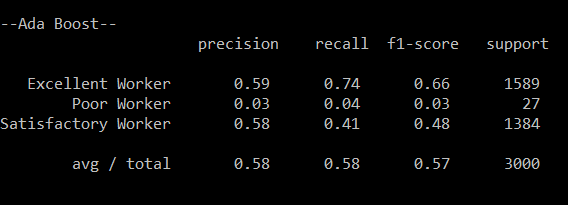
Machine Learning algorithm testing

When designing my machine learning algorithm, i had to decide on what classifier to use. I experimented with a number of models and used Scikit-learn’s classification report functionality to test the models precision and accuracy. I also had to tweak parameters, such as the criterion for attribute splits, and see how this affected the performance of the classifier.









Based off these results, I decided to go with a random forest classifier. One major issue I noticed was the low number of employees labelled a “poor worker”. However, I couldn’t tweak class labels to improve this and did not want to compromise prediction accuracy. I then proceeded to carry out test cases for predictions. These tests involved changing different attributes inputs together and recording the result. I wanted to test the effect of individual parameters, so I gave each parameter a standard value and then enhanced each parameter for each test case. The standard values for each attribute are:

* **Satisfaction Level:** 6
* **Average Monthly Hours:** 156
* **Time spent at company:** 2
* **Work Accident:** No
* **Promotion in the last 5 years:** No
* **Left:** Yes
* **Salary:** Medium
* **Number of projects:** 4

I did a train and test split of 80% (training data) to 20% (testing data).

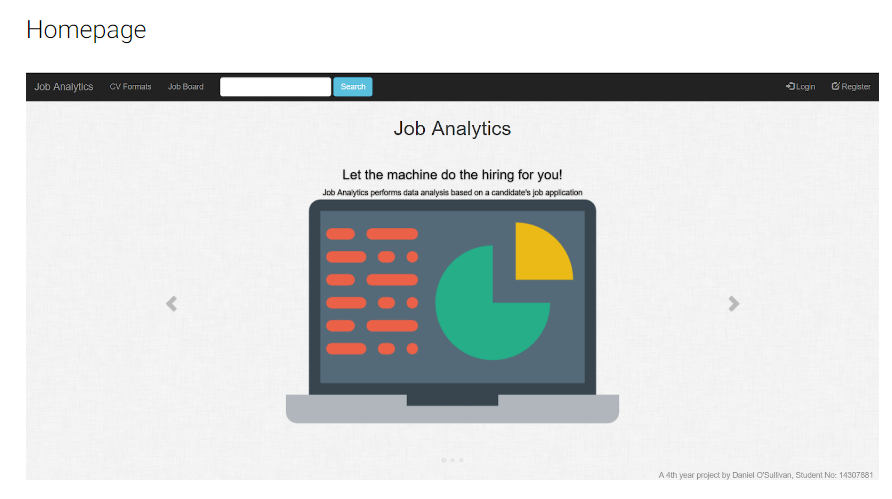
**Tests**

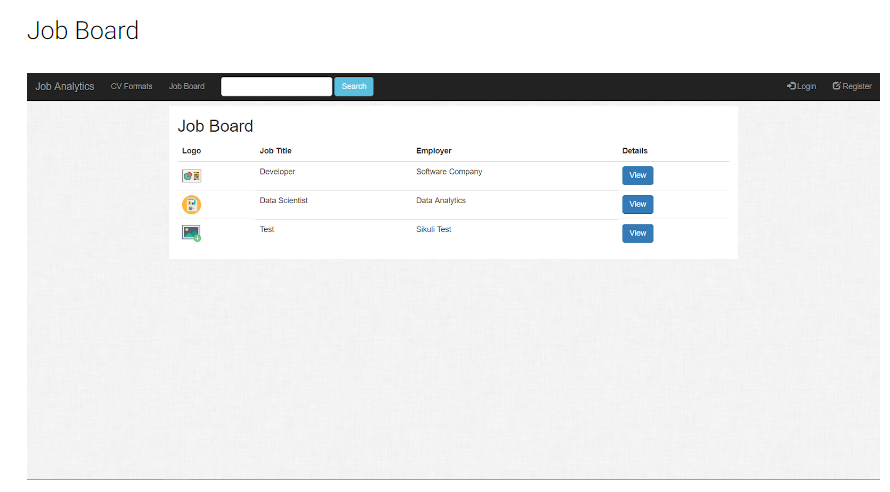
|  |  |  |
| --- | --- | --- |
| **Test ID** | **Inputs** | **Result** |
| 75 | Applicant with standard attributes but with high satisfaction level | Satisfactory Worker |
| 76 | Applicant with standard attributes but with low satisfaction level | Satisfactory Worker |
| 77 | Applicant with standard attributes but with high average monthly hours | Excellent Worker |
| 78 | Applicant with standard attributes but with low average monthly hours | Satisfactory Worker |
| 79 | Applicant with standard attributes but with high time spent at the company | Satisfactory Worker |
| 80 | Applicant with standard attributes but with low time spent at the company | Satisfactory Worker |
| 81 | Applicant with standard attributes and no promotion in the last 5 years | Satisfactory Worker |
| 82 | Applicant with standard attributes and a promotion in the last 5 years | Satisfactory Worker |
| 83 | Applicant with standard attributes and had no work accident | Satisfactory Worker |
| 84 | Applicant with standard attributes and had a work accident | Satisfactory Worker |
| 85 | Applicant with standard attributes and a low salary | Satisfactory Worker |
| 86 | Applicant with standard attributes and a medium salary | Excellent Worker |
| 87 | Applicant with standard attributes and high salary | Satisfactory Worker |
| 88 | Applicant with standard attributes and a high number of projects worked on | Excellent Worker |
| 89 | Applicant with standard attributes and a low number of projects worked on | Excellent Worker |

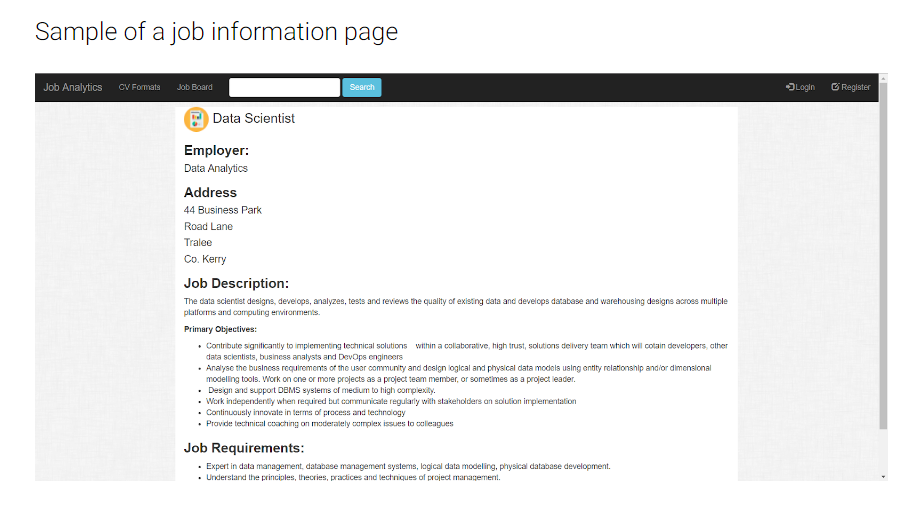
User Testing

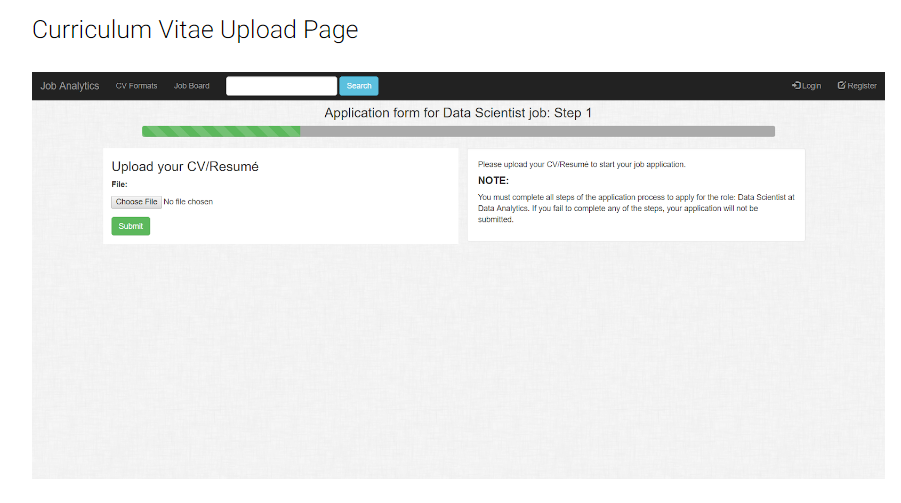
User Interface Questionnaire

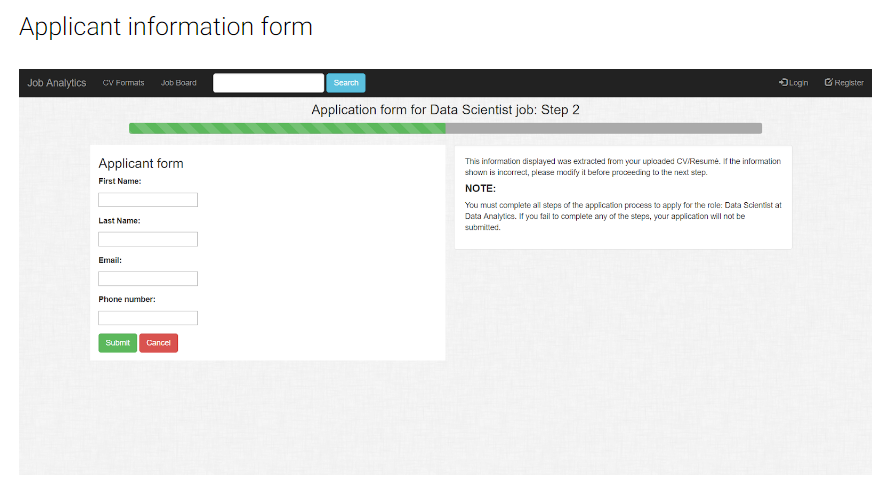
I made a questionnaires regarding the design of the application’s user interface. I posted it on to Facebook and asked for people’s opinions on the design and how it looked. In the questionnaire, users were asked to observe the following images and then answer questions about the user interface.

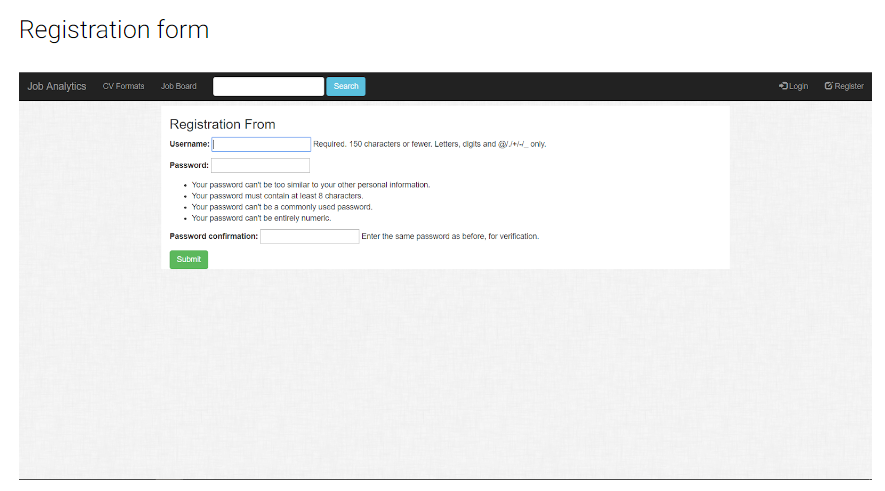


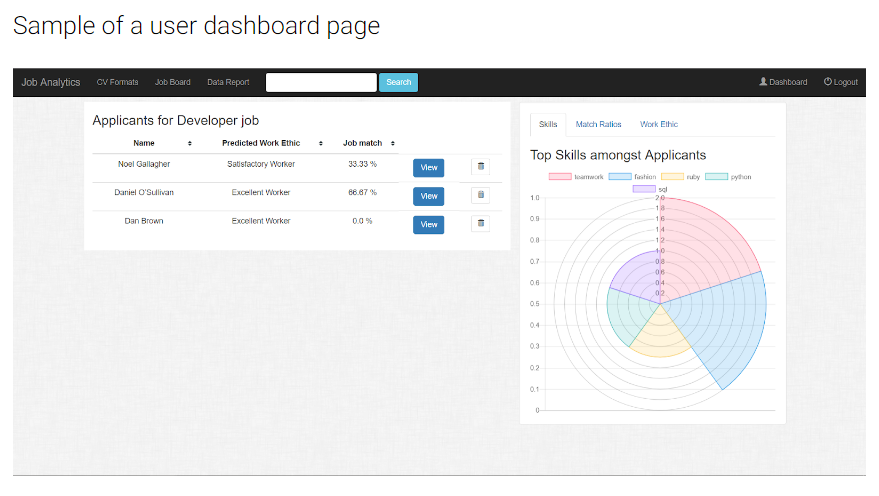




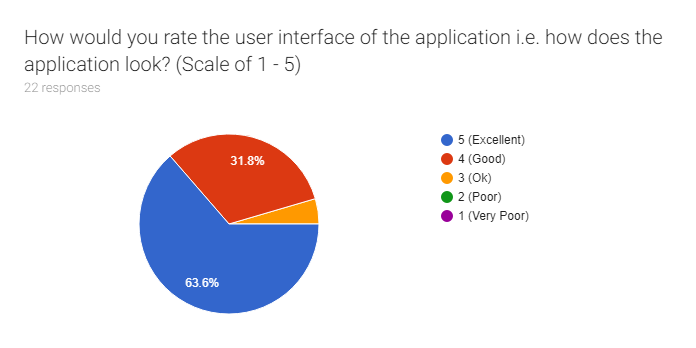


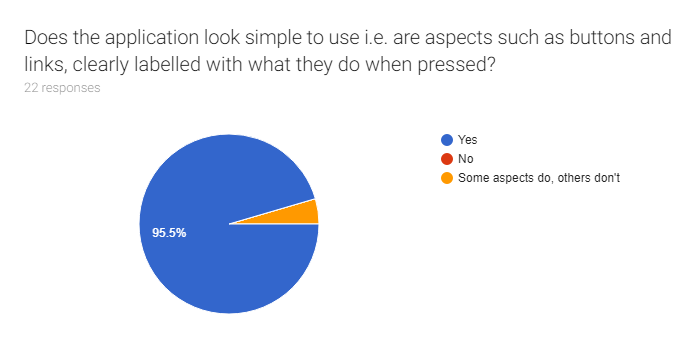


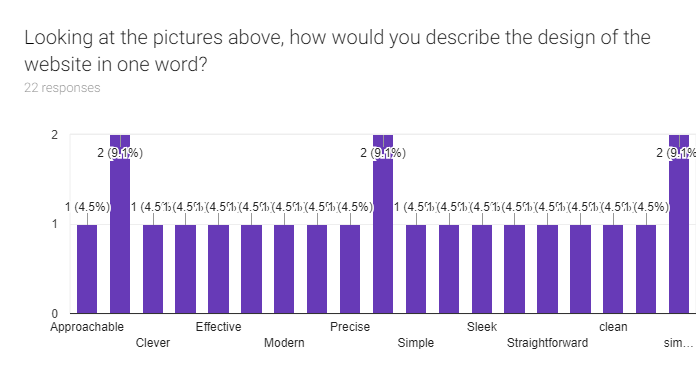




These are the results I got back.







**Top 3 words used were:**

* Clean
* Professional
* SImple

Are there any improvements you would suggest?

|  |
| --- |
| Utilise more screen space. With bigger buttons and text. I’m quite elderly and have poor sight. |
| More colour |
| Looks really good! Very simplistic but effective design! Could you include a my profile page if users crest accounts? It could be synced with LinkedIn. |
| None |
| Maybe try fill some pages ? |
| If I understand the final image correctly, maybe add an option so applicants can't see the scores of other applicants if they would prefer not to. Maybe devise a page that lists all of the individual's pending applicantions and scores too. |
| Brighter navbar |
| More graphical job info, cv upload etc |
| Synergise |
| i think it looks perfect to me nothing really i would change |
| Some pages are very sparse, and others text heavy. Distribute data more evenly through out pages |
| None, smooth clean and easy to use |
| Great App! |

The overall feedback was that the application looks really well designed and has a professional look to it. The main concern people had was that some content on some pages looked too distributed, while some in contrast looked too condensed.

Employer and applicant user testing

I conducted anonymous user testing. I got 4 users who cover different aspects of my user demographic:

**2 people who work in recruitment roles:**

* One person works for a software development company and is familiar with using technology including applications similar to Job Analytics. We’ll call this person employer1.
* One person works for a van courier company and would not be very familiar with using recruitment technology. We’ll call this person employer2.

**2 recent college graduates:**

* One person who has recently graduated from a computer science course. We’ll call this user applicant1.
* One person who has recently graduated from a political based college course. We’ll call this person applicant2.

The thought process behind getting these types of users was that I have 2 users (one employer and one applicant) who would be quite familiar with using technologies such as a web application. I would also have 2 users (one employer and one applicant) who wouldn’t be familiar with using such technologies. I assigned each set of users a set of tasks and asked them to answer questions after they have completed them

Employer Tasks

**Creating a user**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Employer1** | **Employer2** | **Comments** |
| How easy did you find the task on a scale of 1 - 5 (with 5 being really easy one being very difficult). | 5 | 5 | Both users stated that they found that the task was relatively simple to do |
| How clear were the instructions on the application in regards to this task, on a scale of 1 -5 (with 5 being really easy one being very difficult). | 5 | 5 | Both users stated that instructions were clear |

**Logging on with their newly created user**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Employer1** | **Employer2** | **Comments** |
| How easy did you find the task on a scale of 1 - 5 (with 5 being really easy one being very difficult). | 5 | 4 | Both users stated that they found that the task was relatively simple to do, but employer2 was confused as to why you had to login after you registered. He/She assumed they would be logged on automatically after registration. |
| How clear were the instructions on the application in regards to this task, on a scale of 1 -5 (with 5 being really easy one being very difficult). | 5 | 5 | Both users stated that instructions for logging on were clear. |

**Posting a job**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Employer1** | **Employer2** | **Comments** |
| How easy did you find the task on a scale of 1 - 5 (with 5 being really easy one being very difficult). | 3 | 2 | Both users stated that they found that the task more difficult than it should have been. Employer1 expressed their view that the requirements should not be constrained to a certain format. |
| How clear were the instructions on the application in regards to this task, on a scale of 1 -5 (with 5 being really easy one being very difficult). | 3 | 3 | Both users stated that instructions were a little vague and somewhat difficult to follow. |

**Viewing applicant statistics and summaries**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Employer1** | **Employer2** | **Comments** |
| How easy did you find the task on a scale of 1 - 5 (with 5 being really easy one being very difficult). | 4 | 4 | Both users stated that they found that the task simple but both expressed where improvements could be made e.g. better labelling of graphs. |
| How clear were the instructions on the application in regards to this task, on a scale of 1 -5 (with 5 being really easy one being very difficult). | 5 | 4 | Employer1 stated the instructions were simple to follow while employer2 said the only problem they had was when observing an applicant’s CV. They didn’t see the tab menu on top for ”Formatted CV” and “Raw CV”. Said that they should be made more clear to see. |

**Deleting a job post**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Employer1** | **Employer2** | **Comments** |
| How easy did you find the task on a scale of 1 - 5 (with 5 being really easy one being very difficult). | 5 | 5 | Both users stated that they found that the task was relatively simple to do |
| How clear were the instructions on the application in regards to this task, on a scale of 1 -5 (with 5 being really easy one being very difficult). | 5 | 5 | Both users stated that instructions were clear |

**Deleting an applicant**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Employer1** | **Employer2** | **Comments** |
| How easy did you find the task on a scale of 1 - 5 (with 5 being really easy one being very difficult). | 5 | 5 | Both users stated that they found that the task was relatively simple to do |
| How clear were the instructions on the application in regards to this task, on a scale of 1 -5 (with 5 being really easy one being very difficult). | 5 | 5 | Both users stated that instructions were clear |

**Updating a job post**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Employer1** | **Employer2** | **Comments** |
| How easy did you find the task on a scale of 1 - 5 (with 5 being really easy one being very difficult). | 3 | 3 | Both users stated that their previous concerns with the job posting functionality. Employer2 gave a slightly better score than previously as they were more familiar with how to do it after doing it the first time. |
| How clear were the instructions on the application in regards to this task, on a scale of 1 -5 (with 5 being really easy one being very difficult). | 3 | 3 | Both users stated their previous comments in regards to the job posting instructions. |

Applicant Tasks

**Search for a job using search bar**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Applicant1** | **Applicant2** | **Comments** |
| How easy did you find the task on a scale of 1 - 5 (with 5 being really easy one being very difficult). | 4 | 5 | Both users stated that they carried out the tasks with ease. Applicant1 stated their view that you should be able to filter for jobs by more than just the job title. |
| How clear were the instructions on the application in regards to this task, on a scale of 1 -5 (with 5 being really easy one being very difficult). | 3 | 5 | Applicant1 stated that there were no instructions given on the application but said the task was self explanatory. Stated that someone might not know how to do it without proper instructions. |

**Applying for a job**

|  |  |  |  |
| --- | --- | --- | --- |
| **Question** | **Applicant1** | **Applicant2** | **Comments** |
| How easy did you find the task on a scale of 1 - 5 (with 5 being really easy one being very difficult). | 4 | 3 | Applicant1 stated the task was easy but said that they felt the last step of the application process was unnecessary. Applicant 2 was confused as to why their CV has to be in a specific format. |
| How clear were the instructions on the application in regards to this task, on a scale of 1 -5 (with 5 being really easy one being very difficult). | 5 | 3 | Despite Applicant1’s comments, they stated the instructions were very clear. Applicant2 stated the instructions were clear but didn’t give reasoning to why they were there. |

The overall feedback was that the majority of tasks were simple to execute. The big concern was the job posting functionality for employers, while applicants were concerned with (not as concerned as employers were about job posting) some of the aspects of the job application process.