Master Test Plan

1. Test Plan Identifier: OPIGS_TEST_ASSGN6

2. Authors:

Pranav Mehrotra, Saransh Sharma, Abhijeet Singh

3. References:

a. SRS submitted as a part of Assignment
b. Project Description
c. https://jmpovedar.files.wordpress.com/2014/03/ieee-829.pdf

4. Introduction:

This is the Master Test Plan for the Online Placement Information Gathering System (OPIGS) submitted. This plan will address testing of all items and elements that are related to the OPIGS, both directly and indirectly. The project will have unit testing, the details for each test are addressed in the appropriate section.

5. Test Items:

a. Admin Class:
   - Access, Modify, Delete
   - UploadJSON()
   - ApproveComp()

b. Student Class:
   - getCVs()
   - getProfiles()
   - getCompApplied()
   - changeStatus()
   - EditDetails()
   - requestFeedback()
   - applyCompany()

c. Alumni Class:
   - viewStudentDetails()
   - giveFeedback()

d. Company Class:
   - shortlistStudent()
   - viewStudentDetails()
   - editDescription()

6. Features to be tested:

a. Login Interface
b. Signup Interface
c. Notification panel
d. Feedback panel
e. Upload CV

7. Features not to be tested:

a. All HTML, CSS and Bootstrap will NOT be tested
b. All client-side scripts written in JavaScript would NOT be tested
c. The server-side code written using Django would NOT be tested
d. The database deployed via sqlite3 would NOT be tested

8. Pass/Fail Criteria:

We will be providing outputs for all unit tests, which need to be matched for a unit test to be considered as “Passed”, any other result would be treated as “Failed”.

9. Test Plans and Scenarios:

1. Admin Portal class:

1. username → String
2. password → String (hashed internally while storing)

The institute will already be provided with a username and password which can be used to log into the superuser account.

Plan:

1. To Log into the portal using the given password and username.
2. Change password(if required) and reattempt to log in.
3. View individual accounts of students, alumni and company; modify or delete the entities.
4. Upload JSON file containing details of registered students.
5. Change status of company to active or inactive using checkboxes (active makes a company eligible for hiring students)
6. Change the status of the students, who have got an offer from a company and at the end of the placement season, mark all the remaining students as Inactive, so that they cannot participate next year in the placement process.

2. Student Class:

a. Sign up:
   Input:(all fields are compulsory to be filled, empty field would reprompt user to fill that field.
   first_name = CharField(max_length=100)
   last_name = CharField(max_length=100)
   email = EmailField(unique=True)
   contact_number = CharField(max_length=12)
   password = character field (max_length = 128)
   confirm password = character field (max_length = 128)
   department = CharField(max_length=60)
   roll_number =CharField(max_length=9,unique=True)
   SDprofile = BooleanField(default=True)
   DAprofile = BooleanField(default=False)
   CV_SD = FileField (upload_to='accounts/resumes')
   CV_DA = FileField (upload_to='accounts/resumes')

Scenarios:

1. Successful Sign up:
    a. Output: Student dashboard page is displayed. Student constructor will be called and the object will be updated in the database.
2. Sign-up failure:
    a. Password mismatch: user would be prompted to retype a new password (both password fields don’t match error message is displayed).
    b. Email id pre-exists:User with this Email already exists error message will be displayed and user will be asked to refill the form.
    c. Roll number pre-exists: UNIQUE constraint failed: accounts_student.roll_number error message.
    d. Roll number not registered in uploaded json file: No such student exists message is displayed.

b. Log in:

1. Successful log-in:
    a. Input: Correct username (registered email id will be used as username) and password is entered
    b. Output: Student dashboard page is displayed
2. Log-in failure:
    a. Input: wrong username or wrong password
    b. Output: Invalid username or password error message is displayed

c. EditDetails:

1. Input:
    first_name = CharField(max_length=100) (Readonly)
    last_name = CharField(max_length=100) (Readonly)
    email = EmailField(unique=True) (Readonly)
    contact_number = CharField(max_length=12) (Updatable)
    department = CharField(max_length=60) (Readonly)
    roll_number =CharField(max_length=9,unique=True) (Readonly)
    SDprofile = BooleanField(default=True) (Updatable)
    DAprofile = BooleanField(default=False) (Updatable)
    CV_SD = FileField (upload_to='accounts/resumes') (Updatable)
    CV_DA = FileField (upload_to='accounts/resumes') (Updatable)

2. Output:Update the Data in the database of that particular student and redirect the student to the Student Dashboard.

d. getCVs (query to sqlite3 database):

1. Input: Student username
2. Output: uploaded CVs of student

e. getProfiles (query to sqlite3 database):

1. Input: Student username
2. Output: list of profiles chosen by the student

f. getCompApplied (query to sqlite3 database):

1. Input: Student username
2. Output: list of companies applied by student in the form of string

g. requestFeedback:

1. Input: Request to an alumnus to give feedback to the Student Profile and CVs.
2. Output: Append the request to the Alumni Dashboard, for the alumnus to give feedback
   .

h. applyCompany:

1. Input: student request by clicking apply button
2. Output: Student roll number is appended to the list of students field in companies database.

3. Company Class:

a. Sign-up:Input:(all fields are compulsory to be filled, empty field would reprompt user to fill that field.

Form 1:

email = EmailField (unique=True)
contact_number= CharField (max_length=12)
company_name=CharField (max_length=70)
address =CharField (max_length=100)
profile = CharField(max_length=5)
password = CharField (max_length = 128)
confirm password = CharField(max_length = 128)
verify_doc = file

Form 2:

overview= TextField
work_environ = TextField
job_desc = TextField
other_details = TextField

Scenarios:

1. Successful Sign-up:
    a. Output: After the successful submission of both the Forms, the Company object is constructed and saved in the database. Then the object is made *inactive*, in order to prevent the company from taking part in the placement process, until the approval from the Institute Admin.
2. Sign-up failure:
    a. Password mismatch: user would be prompted to retype a new password (both password fields don't match error message is displayed).
    b. Email id pre-exists:User with this Email already exists error message will be displayed and user will be asked to refill the form.

b. Log in:

1. Successful log-in:
    a. Input: Correct username (registered email id will be used as username) and password is entered after verification from institute.
    b. Output: Company dashboard page is displayed
2. Log-in failure:
    a. Input: wrong username or wrong password
    b. Output: Invalid username or password error message is displayed

c. shortlistStudent:

1. Input: The list of students applied would be given as input by extracting the string and splicing it appropriately.
2. Output: the company will receive an option to either approve or reject. Once approved a student's status should change to place. Further, communication regarding other technicalities is expected to be delivered via mail by the HR of the company..

d. viewStudentDetails:

1. Input: Student list from company database would be extracted using queries to sqlite3.
2. Output: The string returned by the query would be splice to retrieve individual usernames of students. From this username, CV and other important details would be displayed to the company.

e. editDescription:

1. Input: Text typed in text box
2. Output: the pre-written description would be overwritten by the new text on clicking the button.

4. **Alumni Class:**

a. Sign-up: Input:(all fields are compulsory to be filled, empty field would reprompt user to fill that field.)(all fields are compulsory to be filled, an empty field would reprompt user to fill that field.

Input:

first_name = CharField(max_length=100)

last_name = CharField(max_length=100)

email = EmailField(unique=True)

contact_number = CharField(max_length=12)

password = character field (max_length = 128)

confirm password = character field (max_length = 128)

department = CharField(max_length=60)

year_of_graduation =IntegerField(max_length=4,default="2020")

roll_number =CharField(max_length=9,unique=True)

Scenarios:

1.Successful Sign up:

a. Output: Alumni dashboard page is displayed. Alumni constructor will be called and the object will be updated in the database.

2. Sign-up failure:

a. Password Mismatch: user would be prompted to retype a new password (both password fields don't match error message is displayed).

b. Email id pre-exists:User with this Email already exists error message will be displayed and the user will be asked to refill the form.

c. Roll number pre-exists: UNIQUE constraint failed: accounts_alumni.roll_number error message.

d. Roll number not registered in uploaded json file: No such student exists message is displayed.

b. Log-in:

1. Successful log-in:

a. Input: Correct username (registered email id will be used as username) and password is entered after verification from institute.

b. Output: Alumni dashboard page is being displayed where there is an option to edit profile, give feedback to students and chat with the students.

2. Log-in failure:
   a. Input: wrong username or wrong password
   b. Output: Invalid username or password error message is displayed

c. viewStudentDetails
   1. Input: List of students who have applied for a feedback from that particular alumni would be visible to the alumni in the student details section from the alumni database and would be extracted using queries to sqlite3.
   2. Output: The string returned by the query would be splice to retrieve individual username of students. From this username, CV and other important details would be displayed to the company.

d. giveFeedback:
   1. Input: The alumni enters necessary feedback after seeing their CV and profile as deemed by him for those students who have requested feedback from him/her in the form of a string.
   2. Output: The feedback would be stored as a string in the database which would then be displayed to the student when he would login to his profile.

5. Chat Features:

1. Chat feature on the portal is only visible for the student and the Alumni.
2. First step would be that a student would request feedback from alumni.
3. Then the alumni would have a request being displayed on his side to give Feedback.
4. If the alumni gives feedback for a particular student then this would enable chat Functionality between the student and the alumni.
5. Students can send a message to one of the alumni who has given feedback on his resume.
6. Student sends a message which is received by the Alumni in his chat box on the Portal.
7. Alumni can then give relevant reply back to the student which would be received by the student in his chat box.

6. Notification Panel:

1. The institute admin would monitor all the job opportunities offered by all the companies.

2. Whenever a new job opportunity is floated by a company or some relevant information regarding the placement process is obtained, the institute would take note of it.
3. Then the Institute admin would put up notifications for all such relevant information.
4. These notifications can be viewed by the student by logging in to the portal in the notification panel.

Interface Testing

1. **Sign Up/Login Interface:**
    a. The displayed form would be easily understandable/self explanatory and should handle multiple fail scenarios. The form will prompt the user if some required field is left out or some information entered is of wrong format than the required format by the database.
    b. If during Signup, a student entered an Invalid Roll Number or an already existing Roll Number, then the form will prompt the student for the same and will not accept the Signup Request.
    c. In Login Interface, if the user entered the wrong username and password combination, then the user will get an error message for the same.

2. **Edit Profile Interface:**

    a. The Edit Profile form should only allow the user to change the editable fields(Contact Number, Profiles(for students and companies) and the other non-editable fields(Email , First Name , Last Name, Roll Number(for Students and Alumni) and Company Name(for Company)) should be read only. They will only be displayed for user's satisfaction and to make the software more user friendly.

3. **Apply for Company Interface:**
    a. This Interface will show the list of all Companies that are available for applications in the recruitment process.
    b. The student should be able to apply for a Company by clicking on the Apply button beside the Company name. The Companies that the student has already applied to should not be available to apply again.
    c. Once the student has confirmed their application for the Company, then the Student id should be appended to the Company's list of Students that have applied.

4. **Notifications Interface:**
    a. All the notifications posted by Institute Admin must be visible to all the Students.
    b. All the newly posted notifications would come on reloading the Student Dashboard.
    c. The Notifications should come in Sorted order according to their posting time, i.e., the recent posts must come before the older notifications.

GUI Testing

- **Check Basic GUI elements**
  For the following Tkinter GUI elements, we describe the basic properties that must be tested for appropriate/error-free behavior wherever they appear in our GUI
  1. **Buttons**
  Check if all buttons are clickable and active
  2. **RadioButtons**
  Check if exactly one is selected
  3. **CheckBoxes**
  Check if atleast one is selected
  4. **TextBoxes**
  Check for text entry is not empty
  5. **DropDown Menus**
  Check if exactly one option is selected
  6. **ListBoxes/ComboBoxes**
  Check if atleast one option is selected

- **Check Common GUI features**
  1. **'Back' Buttons**
  Check if provided in every page to Go Back to the previous page
  2. **Submit/OK Buttons**
  Check if all 'Required' text entries are filled before execution
  3. **'LogIn' and 'LogOut' Buttons**
  Check if these buttons safely execute login and logout for Institute Admin, Students , Company Admin and Alumni