

# ASSIGNMENT

## Coding and Code Testing

### GROUP-5:-

RITISH BANSAL - 190101076

SURYANSH SINGH - 190101089

ANANT SHANKHDHAR - 190101011

MAYANK CHANDAK - 190101052

### **Overview of the Document**

This document contains the overview of how we have implemented the project and an overview of the code testing we performed.

We have used Python with the Django framework for implementing our project, Placement Helper. As a part of this website, there are several features implemented including a facility for the seniors to upload their preparation phase experience and their company working experience, seeing the summarised statistics consolidated company wise, branch wise, year wise, and role wise, searching through this data based on different criteria, such as skills required, profiles and companies etc and a personalised feed showing the top viewed profiles.

### **Scope of the System and Target Audience**

The Placement Helper is intended to be used by the students during preparation for placements and internship drives. The Placement Helper System is for the students of IIT-Guwahati to have easy access to important information and resources directly from their seniors who went through the process which will make their intern and placement preparation easier and more convenient.

### **Purpose of the Document**

The purpose of this report is to document the Black Box and the White Box testing of our website, Placement Portal Platform. Unit testing has been performed only after the corresponding module was coded and successfully reviewed. As a summary, this document is a means to equip the reader with the bugs, errors and the shortcomings of the app.

### **Mapping between Design Document and Coding**

The modules/submodules of the DFD and the corresponding code in the project implemented have been indicated. Taking into consideration that all features need not have to be implemented, we have also added the potential functions signatures which we would add later.

|     | <b>Module/ Submodule of the DFD</b> | <b>Function in the Code</b>                 |
|-----|-------------------------------------|---|
| 1.1 | Create Account                      | SignupView(request)                         |
| 1.2 | Login                               | auth_views.LoginView.as_view(template_name) |
| 1.3 | Forgot Password                     | ForgotPassword(request, pk)                 |
| 2.1 | Create Experience                   | CreateExperience(request)                   |
| 2.2 | Edit Experience                     | EditExperience(request, pk)                 |
| 3.1 | Search Name                         | SearchExperience(request)                   |
| 3.2 | Search Company                      |   |
| 3.3 | Search Job Profile                  | SearchExperience(request)                   |
| 4.1 | View Year Summary                   | YearSummary(request)                        |
| 4.2 | View Branch Summary                 | BranchSummary(request)                      |
| 4.3 | View Job Profile Summary            | JobProfileSummary(request)                  |
| 4.4 | View Company Summary                | CompanySummary(request)                     |
| 5.1 | Like/Dislike                        | Like/Dislike(request, pk)                   |
| 5.2 | Bookmark Experience                 | BookmarkExperience(request,pk)              |
| 5.3 | View Bookmarks                      | ViewBookmarks (request)                     |
| 6.1 | Ask Question                        | CreateQuestion(request)                     |
| 6.2 | Give Answer                         | CreateAnswer(request)                       |
| 6.3 | Search Q/A                          | SearchQA(request)                           |

### **Functions Chosen for the Representative Cases**

Since the functions are usually short, one single function was not completely representing the behaviour of the core functionality of our project. So, we added multiple functions from the chosen representative modules.

We have chosen four representative cases and added the functions corresponding to them:

1. Adding Recruitment Experience
2. Bookmarking a particular Experience
3. Searching Experience based on keyword
4. Creating Account

## Adding Recruitment Experience

### Justification for using this representative case

The core idea of our project is to get placement/internship experiences and display them to the aspiring candidates. So, adding experiences is a core functionality of our project and one of the most primary representative use cases. Thus, we have included this representative use case and the related functions' code and the corresponding testing.

This case is modularised into three functions, dealing with the fetching of experience from the data entered on the website, and adding it to the database, and providing an acknowledgement to the user.

These are:

*CreateExperience(request)* -> This is the function handling the rendering of the page for creating experience, fetching the details from the user, and also adding them to the database in the required format. This function also invokes the *CreateRound* function for handling round wise experience addition to the database.

*CreateRound(request, pk, n)* -> This function is invoked by *CreateExperience* to handle the round wise interview experience. This is a recursive function and calls itself for all the rounds and calls the *CreateEffort* function after all the rounds are completed.

*CreateEffort(request, pk)* -> This function is called by *CreateRound* after all the roundwise experience addition is done and the remainder of the details are to be fetched and added to the database.

```

@login_required
# Main function invoked to handle Experience Creation for the website
1 def CreateExperience(request):
2     if request.method=='POST':    # When the Experience Creation form is filled
3         form=ExperienceForm(request.POST)
4         if form.is_valid():        # If the form is filled as expected
5             experience=form.save(commit=False)
6             experience.user=request.user
7             experience.save()      # Save the Experience to the database
8             return redirect('experience:create_round', pk=experience.id,
n=experience.recruitment_process)
9         else:    # If the form is not filled as expected
10            print(form.errors)
11            return render(request, 'experience/experience_create.html', {'form':form})
                # Redirect to beginning of the page
12 else:    # On clicking the Create Experience Page
13     form=ExperienceForm()
14     return render(request, 'experience/experience_create.html', {'form':form})

```

```

@login_required
# Function invoked to handle Roundwise Experience Addition to the website
1 def CreateRound(request, pk, n):
2     if request.method=='POST':    # When the Experience Addition is filled
3         form=RoundForm(request.POST)
4         if form.is_valid():        # If the form is filled as expected
5             round=form.save(commit=False)
6             exp=get_object_or_404(Experience, pk=pk)
7             round.experience=exp
8             round.save()    # Save the form data to the database
9             if n == 1 :    # If all the rounds have been filled
10                return redirect('experience:create_effort', pk=pk)
11             else :        # If addition for some experiences remains
12                return redirect('experience:create_round', pk=pk, n=n-1)

```

```

13     else:      # If the form is not filled as expected
14         print(form.errors)
15         return render(request, 'experience/round_create.html', {'form':form})
16     else:      # When CreateFunction invokes this function
17         form=RoundForm()
18         exp=get_object_or_404(Experience, pk=pk)
19         numberOfRounds=exp.recruitment_process-n+1    # Calculating the number of rounds
20         return render(request, 'experience/round_create.html', {'form':form,
    'numberOfRounds':numberOfRounds})

```

```

@login_required
# Function invoked to add Resume and Effort Time to the website
1 def CreateEffort(request, pk):
2     if request.method=='POST':      # If the form is filled as expected
3         form=EffortForm(request.POST)
4         if form.is_valid():    # If the form is filled as expected
5             effort=form.save(commit=False)
6             exp=get_object_or_404(Experience, pk=pk)
7             effort.experience=exp
8             effort.save()    # Save the data to the database
9             return redirect('home')
10        else: # If the form is not filled as expected
11            print(form.errors)
12    else: # When the function is called by CreateRound
13        form=EffortForm()
14        return render(request, 'experience/effort_create.html', {'form':form})

```

## Bookmarking a particular Experience

### Justification for using this representative case

We have added a feature to bookmark a particular experience so that the candidates can save certain experiences for easy reference. So, bookmarking a particular experience is a very significant and non-trivial use case in our project.

There is only one function involved in bookmarking experiences. It is:

*BookmarkExperience(request,pk)* -> It is responsible to fetch the profile bookmarked by the user and save the updates to the database.

```
@login_required
# Function invoked to Bookmark an Experience
1 def BookmarkExperience(request, pk):
2     if request.method=='POST': # When the Bookmark button is clicked
3         form=BookmarkForm(request.POST)
4         if form.is_valid(): # When the button is clicked
5             bookmark=form.save(commit=False)
6             exp=get_object_or_404(Experience, pk=pk)
7             bookmark.experience=exp
8             bookmark.user=request.user
9             bookmark.save() # Save the Bookmarked Profile to the database
10            return redirect('home')
11        else: # Invalid request
12            print(form.errors)
13    else: # When the Bookmarks page is called
14        form=BookmarkForm()
15        return render(request, 'experience/effort_create.html', {'form':form})
```

## Searching Experience based on Keyword

### Justification for using this representative case

Just like adding experiences, viewing experiences is also a core functionality of our project and one of the most primary representative use cases. For viewing experiences, we need to have some criteria to filter the necessary experiences and this is being done by this functionality. Thus, we have included this representative use case and the related functions' code and the corresponding testing. Right now, two categories of Keywords are being supported: Company and Candidate's name.

There are two functions dealing with the fetching of criteria, and searching the required results, and rendering them to the user.

These are:

*SearchExperience(request)* -> This is the main function handling the search experience functionality. It also calls a helping function, *getExperiences* to fetch the experiences satisfying the criteria from the database.

*getExperiences(request, pattern)* -> This function is invoked by *SearchExperience* to fetch all the required experiences from the database based on criteria passed.



```

@login_required
# Function invoked to Search Experience based on Name or Company
1 def SearchExperience(request):
2     if request.method=='POST':    # When the Search Form is filled
3         form=SearchForm(request.POST.dict())
4         if form.is_valid():    # The form is filled as expected
5             data=form.cleaned_data
6             pattern=data['pattern']
7             context={}    # To store all the results
8             context['experiences']=getExperiences(request, pattern)
9             context['form']=BookmarkForm()
10            context['searchform']=SearchForm()
11            context['searchvalue']=pattern
12            return render(request, 'experience/home.html', context) # Return the search results to
the user
13        else:    # The form is not filled as expected
14            print(form.errors)
15            return redirect('home')    # Return to the homepage
16    else:    # When the Search Field is rendered
17        form=SearchForm
18        return redirect('home')

```

```

# Function invoked to fetch Experiences from the database based on Name or Company
1 def getExperiences(request,pattern):
2     companywise_experiences=Experience.objects.all().filter(company__icontains=pattern)
3     namewise_experiences=[]    # Container to store experiences based on pattern
4     users_list=User.objects.all().filter(username__icontains=pattern)
5     for user in users_list:    # Iterate over all the filtered experiences
6         for exp in user.experiences.all():
7             namewise_experiences.append(exp)
8
9     experiences=list(set(companywise_experiences) | set(namewise_experiences))
10
11    flag=[]    # To keep track of bookmarked experiences to render accordingly
12    for exp in experiences:

```

```
11     aux=Bookmark.objects.all().filter(experience=exp).filter(user=request.user)
12     if len(aux) == 0:    # If not bookmarked
13         flag.append(0)
14     else :    # If bookmarked
15         flag.append(1)
16     return zip(experiences, flag)
```

## Creating Account

### Justification for using this representative case

For accessing the website, a user has to first create his/her account. Only then is the data on the website accessible. Thus, because creating an account is a crucial requirement for accessing the data, we have chosen this as a representative case, and added the related code.

There are two functions dealing with the creation of an account on the website.

*SignupView(request)* -> Handles rendering the account creation page, parsing details from the page to the database, and returning an active session for the user

*StoreUser(user\_form,profile\_form)* -> Stores the user passed to the function to the database and returns the rendered page to *SignupView*.

```
# Handles the Account Creation for the Website
```

```
1 def SignupView(request):
2     if request.method=="POST":    # When the account creation form is filled
3         user_form=UserForm(data=request.POST)
4         profile_form=UserProfileForm(data=request.POST)
5         if(user_form.is_valid() and profile_form.is_valid()):    # The form is filled as expected
6             return StoreUser(user_form,profile_form)
7         else:
8             print(user_form.errors, profile_form.errors)    # If the form was not filled as
expected
9         else:
10            user_form=UserForm()    # Load the Create Account page
11            profile_form=UserProfileForm()
12            return render(request, 'account/signup.html',
{'user_form':user_form,'profile_form':profile_form})    # Load the account creation page
```

*# Stores the user (passed argument) to the database*

```
1 def StoreUser(user_form,profile_form):
2     user=user_form.save(commit=False)
3     user.set_password(user.password)
4     user.save()    # Adds the user credentials to database
5     username=user_form.cleaned_data['username']
6     password=user_form.cleaned_data['password']
7     profile=profile_form.save(commit=False)
8     profile.user=user
9     profile.save()    # Adds the user details to database
10    user=authenticate(username=username , password=password)
11    login(request, user)    # Create the session for the user
12    messages.success(request, "You were successfully signed-in")
13    return redirect('home')
```

## **Black Box Testing**

The Black Box testing overview and results obtained for the above chosen representative cases have been included below. For each function for each representative case, we have included the equivalence classes based on possible input cases (including errors). We have also taken into consideration the boundary values which are possible (Boundary value analysis) and took care to include those test cases as well. Based on these equivalence classes and boundary value analysis, we have included all the necessary representative test cases, also, we have included the expected output and the actual output obtained. In our testing, we did not find any output which did not match our expected outputs in the final project implemented. We have included several screenshots from the implemented website as well, some indicating the input passed for the particular test case and others showing the actual output obtained.

## **Test Suite**

### **CreateExperience**

#### **Equivalence class testing**

Set 1 - Set of user details having expected Company and Number of Rounds.

Set 2 - Set of user details having empty field in Company or Number of Rounds.

Set 3 - Set of user details entering less than 1 value in the Number of Rounds.

#### **Boundary Testing**

For the number of rounds, the expected value is greater than 0.

Therefore we have a boundary at 0

Boundary TestCase 1 - User entered expected company name with 0 number of rounds.

Boundary TestCase 2 - User entered expected company name with 1 number of rounds

## Set 1

**Input** - User details having expected Company and Number of Rounds.

**Expected Output** - Moving to next form for providing details about Rounds.

**Actual Output** - Moving to the next form for entering details about Rounds.

Add Experience

Company\*

Amazon

Experience type\*

Placement

Number of Rounds\*

2

Next

Round : 1

Type\*

Coding

Description

Next

## Set 2

**Input** - User details having empty field in Company or Number of Rounds.

**Expected Output** - Prompt Message to User to enter all required fields.

**Actual Output** - Prompt Message to User to enter all required fields.

Add Experience

Company\*

Experience type\*

Placement

Number of Rounds\*

2

Next

Add Experience

Company\*

Experience type\*

Placement

Number of Rounds\*

Please fill in this field.

2

Next

### Set 3

**Input** - User details entering less than 1 value in the Number of Rounds.

**Expected Output** - Prompt Message to User that Number of Rounds must be greater than 0.

**Actual Output** - Prompt Message to User that Number of Rounds must be greater than 0.

Placement Portal Create Search  anant Logout

Add Experience

Company\*

Amazon

Experience type\*

Placement

Number of Rounds\*

-2

Next

Placement Portal Create Search  anant Logout

Add Experience

Company\*

Amazon

Experience type\*

Placement

Number of Rounds\*

-2

Number of Rounds must be greater than 0.

Next



## Boundary Test Case 1

**Input** - User details with expected Company and 0 Number of Rounds.

**Expected Output** - Prompt Message to User that Number of Rounds must be greater than 0.

**Actual Output** - Prompt Message to User that Number of Rounds must be greater than 0.

Add Experience

Company\*

Amazon

Experience type\*

Placement

Number of Rounds\*

0

Next

Add Experience

Company\*

Amazon

Experience type\*

Placement

Number of Rounds\*

0

Number of Rounds must be greater than 0.

Next

## Boundary Test Case 2

**Input** - User details with expected Company and 1 Number of Rounds.

**Expected Output** - Moving to next form for providing details about Rounds.

**Actual Output** - Moving to the next form for entering details about Rounds.

Placement Portal Create Search  anant Logout

Add Experience

Company\*

Amazon

Experience type\*

Placement

Number of Rounds\*

1

Next

Placement Portal Create Search  anant Logout

Round : 1

Type\*

Coding

Description

Next

## CreateRound

### Equivalence class testing

Set 1 - Set of Users just entering the type of round.

Set 2 - Set of Users entering type of round and description of round.

## Boundary Value Analysis

If the user has entered only 1 round then he will be directed to the next form else he will be asked details about next rounds.

Boundary TestCase 1 - Only 1 round is entered by user

Boundary TestCase 2 - 2 rounds are entered by user


## Set 1

**Input** - User details with only giving type of round


**Expected Output** - Saving Round details and moving to next round detail(if any) or moving to next form.

**Actual Output** - Saving Round details and moving to next round detail(if any) or moving to next form.


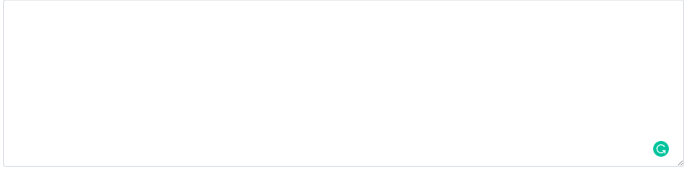
Placement Portal Create Search  anant Logout

Round : 1 

Type\*  


Coding 

Description  



Next

Placement Portal Create Search  anant Logout

Add Efforts made by you 

Resume  


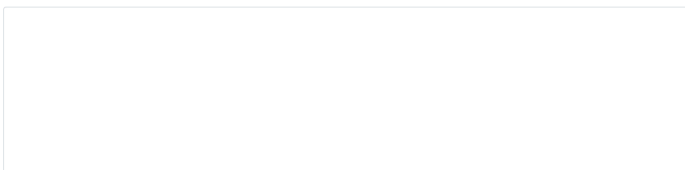
Choose file

No file chosen


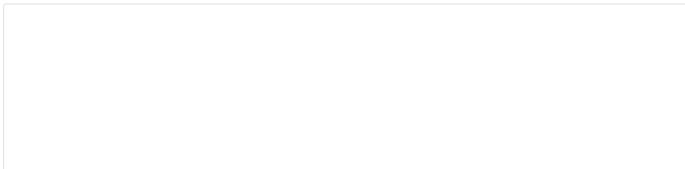
Efforts put in :  

hh hrs / day

Resources  



Tips  



Next

## Set 2

**Input** - User details with expected type of round and its description.

**Expected Output** - Saving Round details and moving to next round detail(if any) or moving to next form.

**Actual Output** - Saving Round details and moving to next round detail(if any) or moving to next form.

Round : 1

Type\*  
Coding

Description  
1 was told to solve 2 coding problems  
The first problem was a leetcode hard dp problem  
second was a simple puzzle.

Next

Add Efforts made by you

Resume  
Choose file No file chosen

Efforts put in :  
hh hrs / day

Resources

Tips

Next

## Boundary Test Case 1

**Input** - User has only 1 round and given its details in this

**Expected Output** - Redirecting user to Add Efforts form.

**Actual Output** - Redirecting user to Add Efforts form.

Placement Portal   Create         [anant](#)   [Logout](#)

Round : 1

Type\*  

Coding

Description

Next

Placement Portal   Create         [anant](#)   [Logout](#)

Add Efforts made by you

Resume  

Choose file

 No file chosen

Efforts put in :  

hh hrs / day

Resources

Tips

Next

## Boundary Test Case 2

**Input** - User has 2 rounds and given its details in this first round.

**Expected Output** - Redirecting user to next Round form.

**Actual Output** - Redirecting user to next Round form.

Round : 1

Type\*  
Coding

Description  
4 simple coding questions

Next

Round : 2

Type\*  
Coding

Description

Next

## CreateEffort

### Equivalence Class Testing

Set 1 - Resume and details are entered by the User.

Set 2 - Only details are entered by the User.

Set 3 - Only the resume is entered by User.

Set 4 - None of Resume and details are entered by the User.

## Set 1

**Input** - User has entered the resume and details

**Expected Output** - Redirected to Homepage and experience has been recorded.

**Actual Output** - Redirected to Homepage and experience has been recorded.

Placement Portal>Create

Q

RohanLogout

Add Efforts made by you

Resume

Choose file1645905130030.pdf

Efforts put in :  
3hrs a day 2 months

Resources

[GFG](#)  
[Codeforces](#)

Tips

Having a good knowledge of graph algorithms is necessary  
You should have good projects

Submit

|  |   |  |             |
|--|---|--|-------------|
| Placement Portal>Create                                  | Search  | Q  | RohanLogout |
| Intern<br>IIT Guwahati<br>Number of rounds : 1           | Scholarship<br>flipkart<br>Number of rounds : 3       | Intern<br>Adobe<br>Number of rounds : 1              |             |
| Ritish<br>Intern<br>Quadeye<br>Number of rounds : 4      | Mayank<br>Placement<br>Google<br>Number of rounds : 3 | Rahul<br>Placement<br>Amazon<br>Number of rounds : 2 |             |
| Rohan<br>Placement<br>Salesforce<br>Number of rounds : 1 | Rohan<br>Intern<br>JPMC<br>Number of rounds : 1       | Rohan<br>Intern<br>Microsoft<br>Number of rounds : 1 |             |
| Rohan<br>Placement<br>Amazon<br>Number of rounds : 2     |   |  |             |

## Set 2

**Input** - User has entered the details only.

**Expected Output** - Redirected to Homepage and experience has been recorded.

**Actual Output** - Redirected to Homepage and experience has been recorded.

Placement Portal

Create

Rohan

Logout

Add Efforts made by you

Resume

Choose file | No file chosen

Efforts put in :  
5hrs a day 2 months

Resources

GFG  
CLRS book  
Intro to competitive programming

Tips

Submit

Placement Portal

Create

Search

Rohan

Logout

|   |  |   |
|---|--|---|
| <div>adesh</div> <div>Intern<br/>IIT Guwahati<br/>Number of rounds : 1</div>  | <div>suryansh</div> <div>Scholarship<br/>flipkart<br/>Number of rounds : 3</div> | <div>Anant</div> <div>Intern<br/>Adobe<br/>Number of rounds : 1</div>     |
| <div>Ritish</div> <div>Intern<br/>Quadeye<br/>Number of rounds : 4</div>      | <div>Mayank</div> <div>Placement<br/>Google<br/>Number of rounds : 3</div>       | <div>Rahul</div> <div>Placement<br/>Amazon<br/>Number of rounds : 2</div> |
| <div>Rohan</div> <div>Placement<br/>Salesforce<br/>Number of rounds : 1</div> |  |   |



### Set 3

**Input** - User has entered only resume

**Expected Output** - Redirected to Homepage and experience has been recorded.

**Actual Output** - Redirected to Homepage and experience has been recorded.

Placement Portal

Create

Q

Rohan

Logout

Add Efforts made by you

Resume

Choose file

1645905130030.pdf

Efforts put in :

hh hrs / day

Resources

Tips

Submit

Placement Portal

Create

Search

Q

Rohan

Logout

adesh

Intern

IIT Guwahati

Number of rounds : 1

suryansh

Scholarship

flipkart

Number of rounds : 3

Anant

Intern

Adobe

Number of rounds : 1

Ritish

Intern

Quadeye

Number of rounds : 4

Mayank

Placement

Google

Number of rounds : 3

Rahul

Placement

Amazon

Number of rounds : 2

Rohan

Placement

Salesforce

Number of rounds : 1

Rohan

Intern

JPMC

Number of rounds : 1

## Set 4

**Input** - User has entered nothing

**Expected Output** - Redirected to Homepage and experience has been recorded.

**Actual Output** - Redirected to Homepage and experience has been recorded.

Add Efforts made by you

Resume

Choose file

No file chosen

Efforts put in :

hh hrs / day

Resources

Tips

Submit

|   |  |   |
|---|--|---|
| <div>adesh</div> <div>Intern<br/>IIT Guwahati<br/>Number of rounds : 1</div> <div></div>  | <div>suryansh</div> <div>Scholarship<br/>flipkart<br/>Number of rounds : 3</div> <div></div> | <div>Anant</div> <div>Intern<br/>Adobe<br/>Number of rounds : 1</div> <div></div>     |
| <div>Ritish</div> <div>Intern<br/>Quadeye<br/>Number of rounds : 4</div> <div></div>      | <div>Mayank</div> <div>Placement<br/>Google<br/>Number of rounds : 3</div> <div></div>       | <div>Rahul</div> <div>Placement<br/>Amazon<br/>Number of rounds : 2</div> <div></div> |
| <div>Rohan</div> <div>Placement<br/>Salesforce<br/>Number of rounds : 1</div> <div></div> | <div>Rohan</div> <div>Intern<br/>JP MC<br/>Number of rounds : 1</div> <div></div>            | <div>Rohan</div> <div>Intern<br/>Microsoft<br/>Number of rounds : 1</div> <div></div> |

## BookmarkExperience

### Equivalence Class testing

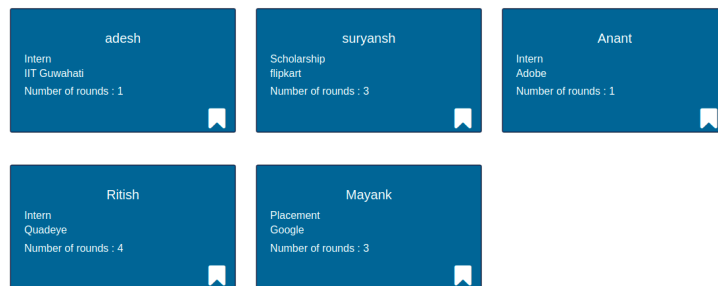
Set 1 - User clicks on bookmark button to bookmark a review

#### Set 1

**Input** - User clicks on bookmark button to bookmark a review

**Expected Output** - Successful in bookmarking experience.

**Actual Output** - Successful in bookmarking experience.



# SearchExperience

## Equivalence Class Testing

Set 1 - User entering some text and click search

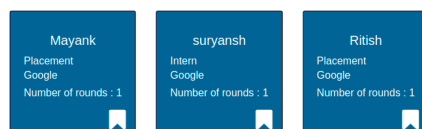
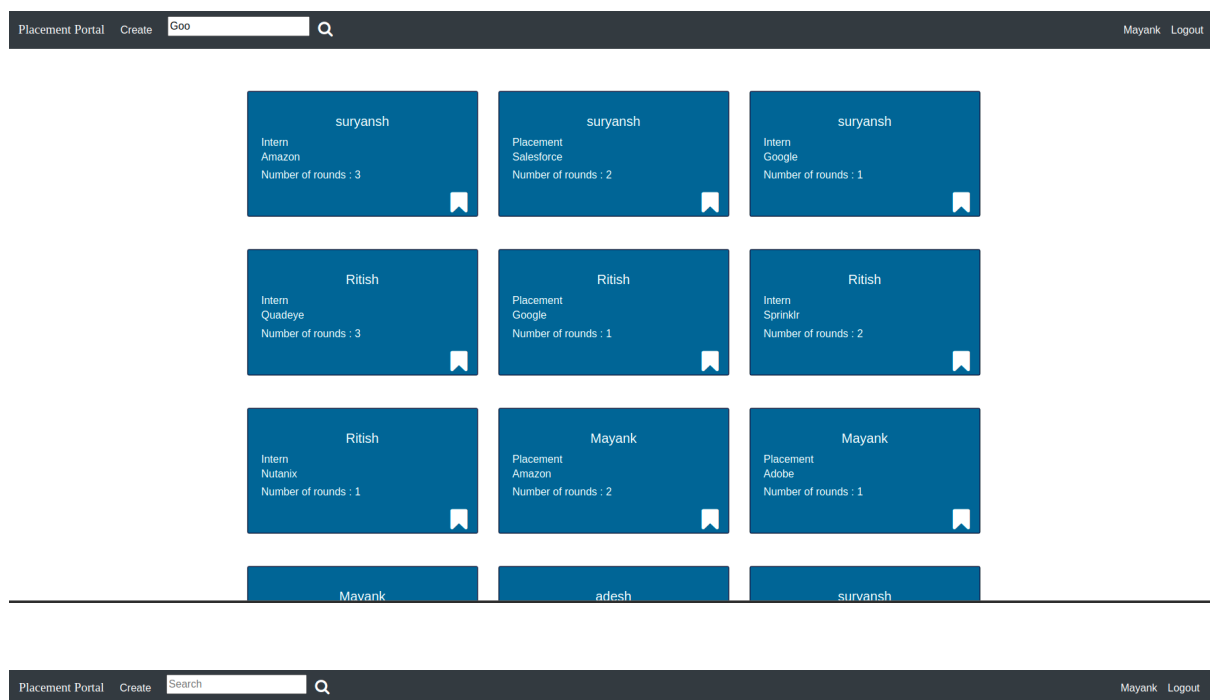
Set 2 - User entering nothing and just click search

### Set 1

**Input** - User entering some Name or Company and click search

**Expected Output** - Homepage showing specific experiences containing the text entered.

**Actual Output** - Homepage showing specific experiences containing the text entered.



## Set 2

**Input** - User entering nothing just click search

**Expected Output** - Homepage showing all the experiences.

**Actual Output** - Homepage showing all the experiences.

|   |   |  |
|---|---|--|
| adesh<br>Intern<br>IIT Guwahati<br>Number of rounds : 1 | suryansh<br>Scholarship<br>flipkart<br>Number of rounds : 3 | Anant<br>Intern<br>Adobe<br>Number of rounds : 1     |
| suryansh<br>Intern<br>Amazon<br>Number of rounds : 3    | suryansh<br>Placement<br>Salesforce<br>Number of rounds : 2 | suryansh<br>Intern<br>Google<br>Number of rounds : 1 |
| Ritish<br>Intern<br>Quadeye<br>Number of rounds : 3     | Ritish<br>Placement<br>Google<br>Number of rounds : 1       | Ritish<br>Intern<br>Sprinklr<br>Number of rounds : 2 |
| Ritish  | Mayank  | Mayank   |

|   |   |  |
|---|---|--|
| adesh<br>Intern<br>IIT Guwahati<br>Number of rounds : 1 | suryansh<br>Scholarship<br>flipkart<br>Number of rounds : 3 | Anant<br>Intern<br>Adobe<br>Number of rounds : 1     |
| suryansh<br>Intern<br>Amazon<br>Number of rounds : 3    | suryansh<br>Placement<br>Salesforce<br>Number of rounds : 2 | suryansh<br>Intern<br>Google<br>Number of rounds : 1 |
| Ritish<br>Intern<br>Quadeye<br>Number of rounds : 3     | Ritish<br>Placement<br>Google<br>Number of rounds : 1       | Ritish<br>Intern<br>Sprinklr<br>Number of rounds : 2 |
| Ritish  | Mayank  | Mayank   |

## **SignUp**

### **Equivalence Class testing**

Set 1 - Set of user details having expected username, and password, and email ID.

Set 2- Set of user details having empty field in either username, and/or password, and/or email.

Set 3 - Set of user details having duplicate username.

Set 4 - Set of user details having email ID with unexpected domain (other than @iitg.ac.in).

Set 5- Set of user details having empty LinkedIn and/or Github user ID.

Set 6 - Set of user details with expected LinkedIn and Github user ID.

## Set 1

**Input** - User details with expected username, and password, and email ID.

**Expected Output** - Successful Sign In

**Actual Output** - Successful Sign In and redirect to Home Page.

### Sign Up

Username\*

Anant

Password\*

\*\*\*\*\*

Email\*

anant.shankhdhar@iitg.ac.in

Branch\*

CSE

Graduation year\*

2022

Programme\*

B. Tech.

Linkedin User ID

Github User ID

Sign Up

Already have an account? [Login](#)

You were successfully signed-in

|  |  |  |
|--|--|--|
| <div>adesh</div> <div>Intern<br/>IIT Guwahati<br/>Number of rounds : 1</div> | <div>suryansh</div> <div>Scholarship<br/>flipkart<br/>Number of rounds : 3</div> | <div>Mayank</div> <div>Intern<br/>Adobe<br/>Number of rounds : 1</div> |
|--|--|--|

## Set 2

**Input** - User details with empty password, and email ID.

**Expected Output** - Prompt Message to fill all required fields.

**Actual Output** - Prompt Message to fill the password.

### Sign Up

Username\*

Ritish

Password\*

Password

Email\*

E-mail ID

Branch\*

CSE

Graduation year\*

2022

Programme\*

B. Tech.

Linkedin User ID

Github User ID

Sign Up

Already have an account? [Login](#)

### Sign Up

Username\*

Ritish

Password\*

Password

Email\*

E-mail ID

Branch\*

CSE

Graduation year\*

2022

Programme\*

B. Tech.

Linkedin User ID

Github User ID

Sign Up

Already have an account? [Login](#)

Please fill in this field.



## Set 3

**Input** - User details with duplicate username(Anant)

**Expected Output** - Error Message that username is already taken.

**Actual Output** - Error Message that username is already taken.

Sign Up

Username\*

Anant

Password\*

.....

Email\*

anantshankhdhar0808@iitg.ac.in

Branch\*

CSE

Graduation year\*

2022

Programme\*

B. Tech.

Linkedin User ID

linkedin.com/anant

Github User ID

Sign Up

Already have an account? [Login](#)

✕

# Sign Up

Username\*

Anant

ⓘ

A user with that username already exists.

Password\*

Password

Email\*

anantshankhdhar0808@iitg.ac.in

Branch\*

CSE

▼

Graduation year\*

2022

▼

Programme\*

B. Tech.

▼

Linkedin User ID

linkedin.com/anant

Github User ID

Sign Up

Already have an account? [Login](#)

## Set 4

**Input** - User details Email ID as [ritish@gmail.com](mailto:ritish@gmail.com)(other than @iitg.ac.in)

**Expected Output** - Error message that Email ID must be of @iitg.ac.in domain.

**Actual Output** - Error message that Email ID must be of @iitg.ac.in domain.

### Sign Up

Username\*

Ritish

Password\*

\*\*\*\*\*

Email\*

ritish@gmail.com

Branch\*

CSE

Graduation year\*

2022

Programme\*

B. Tech.

Linkedin User ID

Github User ID

ritish01

Sign Up

Already have an account? [Login](#)

### Sign Up

Username\*

Ritish

Password\*

Password

Email\*

ritish@gmail.com

Branch\*

CSE

Graduation year\*

2022

Programme\*

B. Tech.

Linkedin User ID

Github User ID

ritish01

Sign Up

Already have an account? [Login](#)

You must use E-mail ID provided by IIT-Guwahati.

## Set 5

**Input** - User details with empty Linkedin and github ID.

**Expected Output** - Successful Sign In

**Actual Output** - Successful Sign In and redirect to Home Page.

### Sign Up

Username\*

Anant

Password\*

\*\*\*\*\*

Email\*

anant.shankhdhar@iitg.ac.in

Branch\*

CSE

Graduation year\*

2022

Programme\*

B. Tech.

Linkedin User ID

Github User ID

Sign Up

Already have an account? [Login](#)

You were successfully signed-in

|   |   |   |
|---|---|---|
| adesh<br>Intern<br>IIT Guwahati<br>Number of rounds : 1 | suryansh<br>Scholarship<br>flipkart<br>Number of rounds : 3 | Mayank<br>Intern<br>Adobe<br>Number of rounds : 1 |
|---|---|---|

## Set 6

**Input** - User details with non-empty Linkedin and github ID.

**Expected Output** - Successful Sign In

**Actual Output** - Successful Sign In and redirect to Home Page.

### Sign Up

Username\*

Ritish

Password\*

\*\*\*\*\*

Email\*

ritishb@iitg.ac.in

Branch\*

CSE

Graduation year\*

2022

Programme\*

B. Tech.

Linkedin User ID

linkedin.com/ritish

Github User ID

ritish01

Sign Up

Already have an account? [Login](#)

You were successfully signed-in

adesh  
Intern  
IIT Guwahati  
Number of rounds : 1

suryansh  
Scholarship  
flipkart  
Number of rounds : 3

Mayank  
Intern  
Adobe  
Number of rounds : 1

## White Box Testing

We have included the Control Flow Graphs for each function for each representative case we have mentioned above.

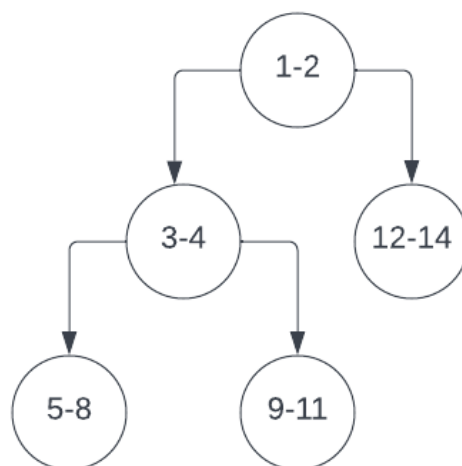
We have compressed all the nodes which were in a straight line and did not have scope for any branching. Because it was clear that such nodes will definitely be traversed, such compression should also give us all the required test cases.



As required in the assignment we have used path coverage to ensure all the required test cases are included. We have listed the paths we found in the report as well. Just like the black box testing, we have listed the inputs, and outputs we found. Also, for all the test cases we found, our expected output matched the actual output we found, so we did not add expected output separately.

### 1) Add Experience

#### **CreateExperience (request)**



### **Path 1**

Path - (1-2) -> (12-14)

Test Case - Click create to get Add experience form.

Output - Display form to add experience details.

### **Path 2**

Path - (1-2) -> (3-4) -> (9-11)

Test Case - Some essential details are not provided.

Output - Display Error to add all details.

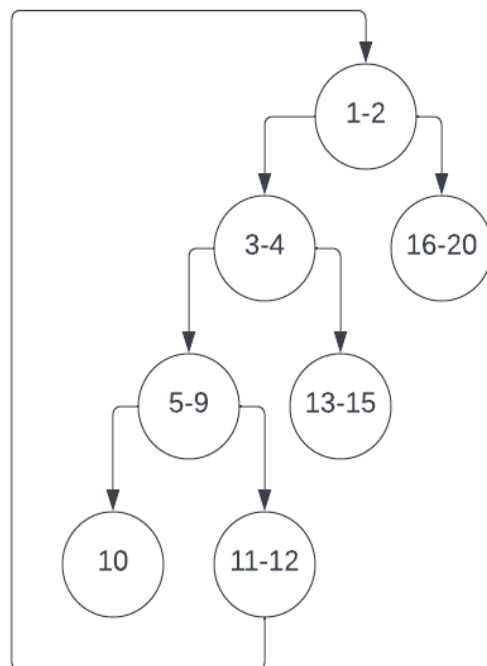
### **Path 3**

Path - (1-2) -> (3-4) -> (5-8)

Test Case - All the details were provided.

Output - Move to the next form to provide round wise details.

### **Create Round (request, pk)**



### **Path 1**

Path - (1-2) -> (16-20)

Test Case - When CreateRound function is called by CreateExperience function.

Output - Display form to add round wise experience details.

### **Path 2**

Path - (1-2) -> (3-4) -> (13-15)

Test Case - User enters detail of the last round.

Output - Effort Form is displayed.

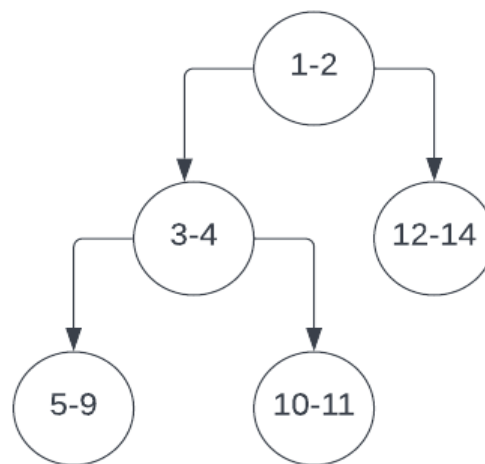
### **Path 3**

Path - (1-2) -> (3-4) -> (5-9) ->(11-12) -> (1-2) -> (3-4) ->(5-9) -> (10)

Test Case - Previously User has entered 2 rounds and this cycle is followed to add details of two rounds.

Output - Next Round Form is displayed.

### **Create Effort (request, pk)**



### **Path 1**

Path - (1-2) -> (12-14)

Test Case - When CreateEffort function is called by CreateRound function.

Output - Display Homepage..

### **Path 2**

Path - (1-2) -> (3-4) -> (10-11)

Test Case - Resume is provided in some other format than .pdf

Output - Error is returned to the user.

### **Path 3**

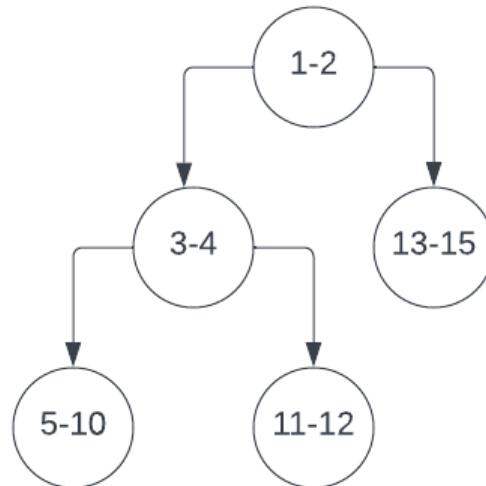
Path - (1-2) -> (3-4) -> (5-9)

Test Case - All the details were provided.

Output - Submit the form and save the experience.

## 2) Bookmarks

### **BookmarkExperience (request, pk)**



#### **Path 1**

Path - (1-2) -> (13-15)

Test Case - Bookmark icon is rendered on every experience.

Output - Displays bookmark icon on each experience depending upon whether the experience has been bookmarked or not..

#### **Path 2**

Path - (1-2) -> (3-4) -> (11-12)

Test Case - User is not authenticated and tries to bookmark an experience.

Output - Error is returned to the user and the user is sent to the login page.

#### **Path 3**

Path - (1-2) -> (3-4) -> (5-10)

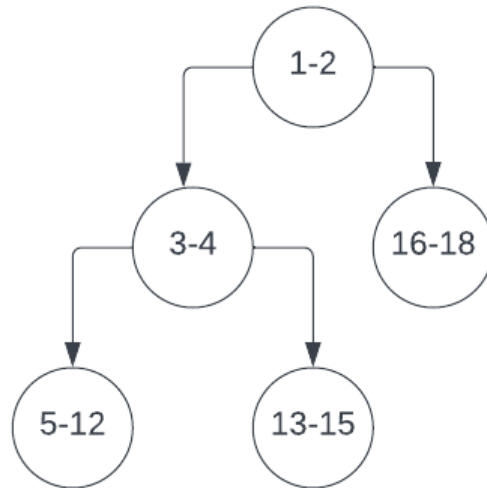
Test Case - Bookmarking an experience by clicking on the bookmark icon.

Output - Bookmark is saved and colour is changed



### 3) Search

#### **SearchExperience(request)**



#### **Path 1**

Path - (1-2) -> (16-18)

Test Case - Search icon and bar is rendered on every experience.

Output - Search icon and bar is displayed on homepage.

#### **Path 2**

Path - (1-2) -> (3-4) -> (13-15)

Test Case - User is not authenticated and tries to search for an experience.

Output - Error is returned to the user and the user is sent to the login page.

#### **Path 3**

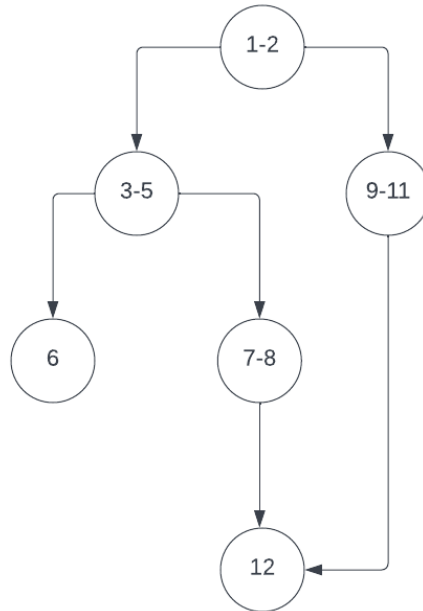
Path - (1-2) -> (3-4) -> (5-12)

Test Case - User is authenticated and tries to search for some name or recruiter.

Output - Specific experiences are displayed to the user according to his requirements..

## 4) SignUp

### SignupView(request)



#### Path 1

Path - (1-2) -> (9-11) -> (12)

Test Case - When the SignUp function is called by the user.

Output - Sign Up form is displayed.

#### Path 2

Path - (1-2) -> (3-5) -> (6)

Test Case - User enters all the details correctly and clicks sign up.

Output - User is signed in and redirected to Home page.

#### Path 3

Path - (1-2) -> (3-5) -> (7-8) -> (12)

Test Case - User forgets to enter some essential detail and tries to sign up.

Output - All the errors are shown and user is expected to sign up again.

---