**SC2006 Software Engineering**

****

**Lab#5 Deliverables**

Lab Group A29

Group Name: Mala

Lab Supervisor: Dr Liu Yang

| Group Members | |
| --- | --- |
| Name | Matriculation No. |
| Pavanraj Selvaraju | U2122616H |
| Ng Yao Ming | U2122168F |
| Low Yong Keng | U2122645L |
| Teo Kai Jie Keith | U2121074G |
| Oak Soe Khant | U2121901K |

**Table of Contents**

[**1. Introduction and Overall Descriptions 5**](#_heading=h.2pk9g52740o1)

[1.2 Intended Audience 5](#_heading=h.1fob9te)

[**2. Documentation of Functional and Non-Functional Requirements 6**](#_heading=h.4yqzpkc7dfk0)

[2.1 Functional Requirements 6](#_heading=h.jnmag6txxq8w)

[2.2 Non-Functional Requirements 11](#_heading=h.4d34og8)

[**3. Data Dictionary 12**](#_heading=h.p36nw3i0w2c9)

[**4. Use Cases Diagram 14**](#_heading=h.ccf3btac9pzn)

[4.1 Use Case Diagram 14](#_heading=h.c7lbx58wu5ar)

[**5. Use Cases Descriptions 15**](#_heading=h.3ghu779u2myp)

[5.1.1 Navigate to Tracker Page 15](#_heading=h.ht1hpwbkepbu)

[5.1.2 Navigate to Searcher Page 16](#_heading=h.18d44l6do6o5)

[5.1.3 Navigate to Booster Page 17](#_heading=h.hj1t5dxa3a5v)

[5.1.4 Navigate to Forum Page 18](#_heading=h.wsfravfipxwj)

[5.2.1 Create Account 19](#_heading=h.jo8jsccnf2a6)

[5.2.2 Login 21](#_heading=h.vl40c2o3byjj)

[5.2.3 Logout 23](#_heading=h.kbsqi7sg0wwh)

[5.3.1 Update Job Search Fields 24](#_heading=h.o1rphbl993er)

[5.3.2 Search Jobs Available 25](#_heading=h.98kvxq3zert8)

[5.3.3 Browse Jobs 26](#_heading=h.nogxfly1s7pm)

[5.3.4 View Wishlist Entry 27](#_heading=h.hh1q362jni6g)

[5.3.5 Delete Wishlist Entry 28](#_heading=h.oq8wlblivt4x)

[5.4.1 View Tracker Entry 29](#_heading=h.y6ee8vr7h2ye)

[5.4.2 Create Tracker Entry 30](#_heading=h.i9jruis84iq5)

[5.4.3 Delete Tracker Entry 31](#_heading=h.d236b8b7nghq)

[5.4.4 Update Tracker Entry 32](#_heading=h.k40fs5p98pdd)

[5.5.1 View Forum 33](#_heading=h.y6v588sde2in)

[5.5.2 Create Forum Post 34](#_heading=h.wsypy65100t5)

[5.5.3 Create Forum Comment 35](#_heading=h.pqtxetxz0i3u)

[5.6.1 Set Resume 36](#_heading=h.yde2bxu6nhue)

[5.6.2 Boost Resume 37](#_heading=h.uzbxts1dljph)

[5.7.1 Access User Database API Call 38](#_heading=h.fpjsctywsmci)

[5.7.2 Google Maps API Call 39](#_heading=h.dp4e7v67rf2m)

[5.7.3 OpenAI API Call 40](#_heading=h.98gpbby3rhge)

[5.7.4 myCareersFuture Jobs API Call 41](#_heading=h.egfxeh4mzus5)

[5.7.5 Access Forum Database API Call 42](#_heading=h.rzjo0z6bsjps)

[**6. Sequence Diagrams 43**](#_heading=h.rws6mqdnvsyh)

[6.1.1 Navigate To Tracker Page 43](#_heading=h.ahfv7admctcx)

[6.1.2 Navigate To Searcher Page 43](#_heading=h.agao8qctwyhg)

[6.1.3 Navigate To Booster Page 44](#_heading=h.g9yahegu1aun)

[6.1.4 Navigate To Forum Page 44](#_heading=h.7af1be1670cx)

[6.2.1 Create Account 45](#_heading=h.utgu101lbrg3)

[6.2.2 Login 46](#_heading=h.wb8o9iohbxxk)

[6.2.3 Logout 47](#_heading=h.nj737o56jrc3)

[6.3.1 Update Job Search Fields 47](#_heading=h.boume5f04yw)

[6.3.2 Search Jobs Available 48](#_heading=h.8wjlza866nrq)

[6.3.3 Browse Jobs 49](#_heading=h.k9gc266jhyrl)

[6.3.4 View Wishlist Entry 50](#_heading=h.u5ubzu7khv3b)

[6.3.5 Delete Wishlist Entry 50](#_heading=h.h305d7e2kiii)

[6.4.1 View Tracker Entry 51](#_heading=h.pkg88c7bageq)

[6.4.2 Create Tracker Entry 51](#_heading=h.j4w7pkoslp7s)

[6.4.3 Delete Tracker Entry 52](#_heading=h.iugbovwqa2bo)

[6.4.4 Update Tracker Entry 52](#_heading=h.a9043m58j48k)

[6.5.1 View Forum 53](#_heading=h.39dm2vvh20rc)

[6.5.2 Create Forum Post 54](#_heading=h.6hkuorz88oxk)

[6.5.3 Create Forum Comment 55](#_heading=h.bu4qo6oybocy)

[6.6.1 Set Resume 56](#_heading=h.mk0on8v9uku1)

[6.6.2 Boost Resume 56](#_heading=h.unm5g9rykmi0)

[**7. Entity Class Diagram 57**](#_heading=h.c47q3ilexud1)

[**8. Boundary and Control Class Diagram 58**](#_heading=h.uwn97ulwzqc9)

[**9. Dialog Map 59**](#_heading=h.aqfu1gq66auf)

[**10. UI Mockups (Initial and Final) 60**](#_heading=h.px26fsd9trcd)

[10.1. Landing Page 60](#_heading=h.tp1nfmrls5ss)

[10.2. Signup Page 61](#_heading=h.9szf8o2cfeac)

[10.3. Login Page 62](#_heading=h.mkrzw7brespp)

[10.4. Searcher Set Up Page 63](#_heading=h.om8yex7ojyvz)

[10.5. Searcher Collapsed Card 64](#_heading=h.sycr5gvknkin)

[10.6. Searcher Expanded Card 65](#_heading=h.87liz6o08de1)

[10.7. Tracker Page (Collapsed Entry) 66](#_heading=h.95ebhxhq220a)

[10.8. Tracker Page (Expanded Entry) 67](#_heading=h.caz6mi92hnmo)

[10.9. Add Tracker Entry Page 68](#_heading=h.swep1mc73r02)

[Initial Mockup 68](#_heading=h.dxbubivmadsp)

[10.10. Forum Page 69](#_heading=h.wu0833gv9hcp)

[Initial Mockup 69](#_heading=h.7swefshrb8jf)

[10.11. Individual Forum Thread Page 70](#_heading=h.tx714s9vjlmk)

[10.12. Update Resume Details Page 70](#_heading=h.yksql3eqt4yx)

[10.13. Resume Booster Page (Collapsed View) 71](#_heading=h.3ouaowju8c8p)

[10.14. Resume Booster Page (Expanded View) 71](#_heading=h.oh085nkwu48x)

[**11. System Architecture 72**](#_heading=h.zd5t73tvljse)

[**12. Unit Testing 73**](#_heading=h.6tjgh037hzpc)

[12.1 Black Box Testing 73](#_heading=h.psiofcfpyqnx)

[12.1.1 Black Box Testing (Sign Up) 74](#_heading=h.999av2z1bhkw)

[12.1.2 Black Box Testing (Searcher Set Up Page) 76](#_heading=h.mtb8bjm8e3jg)

[12.2 White Box Testing 79](#_heading=h.l7f5jlea50cv)

[12.2.1 Sign Up White Box Testing 79](#_heading=h.ql4uh8ojo1zd)

[11.2.2 getJobs White Box Testing 81](#_heading=h.84bb00xfu6vg)

# Introduction and Overall Descriptions

1.1 Purpose

Searching for and tracking job applications is tedious, and one needs to be meticulous in order to not get overwhelmed. Our group, Mala, aims to create a job platform that helps job seekers find job listings and keep track of job applications, all in one place. The Job Searcher shows a list of jobs that best match the user in a "dating app" manner, allowing the user to quickly and effectively browse and select those that interest them. The Job Tracker assists them with organising all their job applications, streamlining the process of finding a job. The Resume Booster aids the user in creating a targeted resume which is sure to please their potential employers and land them a job. The Forum also helps users connect with like minded job searchers to have meaningful discussions and conversations, as well as sharing tips with each other.

## 1.2 Intended Audience

Our intended audience for the application is anyone who is seeking an all-in-one platform to organise and source for jobs so as to increase their likelihood of finding one that they are pleased with and at the same time reduce the trouble they would have to go through in the process.

# Documentation of Functional and Non-Functional Requirements

## 2.1 Functional Requirements

1. Homepage

1.1. Users must be able to access the login and sign up page from the home page.

2. Account

2.1. Users must be able to sign up for a new account at the sign up page.

2.1.1. System must allow users to create a new account with an existing Google account.

2.1.1.1. System should be able to use the user’s existing display picture when they create a new account successfully through the google account path.

2.1.1.2. System should be able to use the user’s existing display name when they create a new account successfully through the google account path.

2.1.2. System must create a new account for the user once email and password requirements are fulfilled.

2.1.2.1. System must prompt the user to re-enter the password if it didn’t adhere to the requirements: at least 8 characters, password must be a combination of letters, numbers and a special character, 2 passwords do not match.

2.1.2.2. System must be able to hide the password from view.

2.1.2.3. System must prompt the user to re-enter the email address if it is already linked to an existing account.

2.1.2.4. System must set a default display picture for the user when they create a new account successfully through the email and password path.

2.2. Users must be able to login to the account with their email and password or their Google account at the login page.

2.2.1. System must validate the user's account when users enter their email and password.

2.2.2. System must be able to hide the password from view.

2.2.3. System must redirect users to the searcher page after their account is validated.

3. Navigation Bar

3.1. Users must be able to access the job tracker page, job searcher page, resume booster page and forum page from the Navigation Bar.

3.2. Users must be able to log out from the Navigation Bar and System must redirect users to the home page when they log out.

3.3 User profile picture must be shown.

4. Job Searcher

4.1. System must redirect users to the searcher setup page to set up their information the first time they try to search for jobs and hold the information.

4.1.1. Users must be able to update searcher setup information.

4.1.1.1. Searcher setup information includes School, Major, Citizenship, Location, JobType and Minimum Salary.

4.1.1.1.1. Minimum Salary should be within 0 to 20000

4.1.1.1.2. All the fields are required

4.2. System must show jobs suitable to users based on the information held from the searcher setup page and the input given by the user in the search bar.

4.3. Users must be able to browse through jobs at the job search page by swiping left or right on the collapsed searcher card.

4.3.1. Job information must be displayed.

4.3.1.1. Job company name must be displayed.

4.3.1.2. Job company logo must be displayed.

4.3.1.3. Job roles must be displayed.

4.3.1.4. Job salary range must be displayed.

4.3.1.5. Job location must be displayed.

4.3.1.6. Job required skills must be displayed.

4.3.2. Job entry must be added to the wishlist in the tracker page when users swipe right.

4.3.3. Job entry must be discarded when users swipe left.

4.4 Users must be able to expand the collapsed searcher card to view the expanded searcher card.

4.4.1. Job information must be displayed.

4.4.1.1. Job company name must be displayed.

4.4.1.2. Job company logo must be displayed.

4.4.1.3. Job roles must be displayed.

4.4.1.4. Job salary range must be displayed.

4.4.1.5. Job type must be displayed.

4.4.1.6. Job location must be displayed.

4.4.1.7. Job address must be displayed.

4.4.1.7.1. If the address is not provided, nil must be displayed instead.

4.4.1.8. Job live Google Maps location must be displayed.

4.4.1.8.1. Users must be able to interact with the Google Maps view.

4.4.1.8.2. If the address does not exist, “Address not found” must be displayed instead.

4.4.1.9. Job date posted must be displayed.

4.4.1.10. Job due date must be displayed.

4.4.1.11. Job required skills must be displayed.

4.4.1.12. Job description must be displayed.

4.4.2. Users must be able to access the job website.

5. Job Tracker

5.1. Job tracker page must display a list of tracked job entries and a list of wish-listed job entries from the job search page.

5.1.1. Job entry information must be displayed.

5.1.1.1. Job company names must be displayed for both lists.

5.1.1.2. Job roles must be displayed for both lists.

5.1.1.3. Job location must be displayed for both lists.

5.1.1.4. Job status must be displayed for the track list.

5.1.2. Users must be able to toggle between the views: just the wishlist, just the track list and a view showing both.

5.2. Users must be able to search for job entries by their job company name for both the wishlist and the track list.

5.3. Users must be able to move a job entry from the wishlist to the track list.

5.4. Users must be able to create new job entries in the track list.

5.5. Users must be able to update their job entries in the track list.

5.5.1. System must be able to create a new job entry in the track list once all the requirements are fulfilled.

5.5.1.1. Job company name, job role and job location are required fields.

5.5.1.2. Skills cannot comprise only whitespaces.

5.6. Users must be able to delete their job entries in both the track list and the wishlist.

5.7. Users must be able to toggle the status of the job entry in the track list.

5.7.1. The statuses are cycled in this order (Preparing, Applied, Replied, OA Due, OA Done, Interview, Offered, Rejected, Ghosted).

5.8. Users must be able to expand the view of their job entries in both the track list and wish list.

5.8.1. Job entry information must be displayed.

5.8.1.1. Job company names must be displayed for both lists.

5.8.1.2. Job roles must be displayed for both lists.

5.8.1.3. Job location must be displayed for both lists.

5.8.1.4. Job status must be displayed for the track list.

5.8.1.5. Job descriptions must be displayed for both lists.

5.8.1.6. Job required skills must be displayed for both lists.

5.8.1.7. Job pay range must be displayed for both lists.

5.8.1.7.1. Job pay range must be displayed in the format: “minSalary - maxSalary SGD”.

5.8.1.8. Job addresses must be displayed for both lists.

5.8.1.8.1. If the address is not provided, nil must be displayed instead.

5.8.1.9. Job live Google Maps locations must be displayed.

5.8.1.9.1. Users must be able to interact with the Google Maps view.

5.8.1.9.2. If the address does not exist, “Address not found” must be displayed instead.

5.8.2. Users must be able to access the job website.

6. Booster

6.1. Resume booster page must display a list of tracked job entries from the tracklist in the job tracker page.

6.2. Users must be able to search for a tracked job entry by their job company name.

6.3. Users must be able to update their resume information.

6.3.1. Resume information includes “Project Experiences”, “Work Experiences”, “Co-Curricular Activities”, “Achievements” and “Skills”.

6.4. Users must be able to boost their resume based on the job entry information of each tracked job entry from the tracklist in the job tracker pager and their resume information.

6.4.1. System must be able to boost multiple resumes at the same time.

6.4.2. Users must be able to view their boosted resume only when a job entry has a corresponding boosted resume data.

6.4.2.1. The boosted resume data must include a boosted cover letter, project experiences, work experiences, co-curricular activities, achievements and skills.

7. Forum

7.1. Users must be able to create forum posts at the forum page.

7.1.1. System must create forum posts for users when requirements are fulfilled.

7.1.1.1. Forum posts title cannot consist of only whitespaces.

7.1.1.2. Forum posts content must be within 5 to 5000 characters.

7.1.2. Forum post information must be displayed.

7.1.2.1. Forum posts title must be displayed.

7.1.2.2. Forum posts content must be displayed.

7.1.2.3. Forum posts author name must be displayed.

7.1.2.4. Forum posts author profile logo must be displayed.

7.1.2.5. Forum posts comment number must be displayed in the general forum page.

7.1.2.6. Forum posts date published must be displayed.

7.1.2.7. Forum posts must be sorted such that posts with higher comment number must be on top and if posts have the same number of comments, the newest posted one must be shown on top.

7.2. Users must be able to comment on forum posts.

7.2.1. System must create comments for users when requirements are fulfilled.

7.2.1.1. Forum comments content must be within 5 to 5000 characters.

7.2.2. Forum post comment information must be displayed

7.2.2.1. Forum post comments content must be displayed.

7.2.2.2. Forum post comments author name must be displayed.

7.2.2.3. Forum post comments author profile logo must be displayed.

7.2.2.4. Forum post comments date published must be displayed.

7.2.2.5. Forum post comments must be sorted such that comments which are newly posted are shown on top.

7.3. Users must be able to search for forum posts by their title.

## 2.2 Non-Functional Requirements

1. Usability
   1. The system should include prompts that help users understand how to use the app.
   2. Users can access the Searcher, Tracker, Booster and Forum page from any page.
   3. Users remain logged in even if they close the application, until the browser is closed.
2. Reliability

2.1. After a system reboot, the full system functionality must be restored within 5

minutes.

1. Performance
   1. The matching job entries shall load in within 30 seconds.
   2. Resume booster shall boost the resume within 2 minutes.
   3. The sign up and sign in page shall load in within 2 seconds.
   4. Each function module shall load in within 2 seconds.
   5. System must add the new job entry to the tracker listing within 5 seconds.
   6. System must delete job entries from the tracker listing within 5 seconds.
   7. System must update job entries in the tracker listing within 5 seconds.
   8. Details of the job shall load within 2 seconds.
   9. The application shall exit within 2 seconds.
   10. All alerts shall load in within 1 second.
2. Supportability

4.1. The database must be replaceable with any commercial product supporting

standard SQL queries.

1. Safety Requirements
   1. The team shall maintain the system regularly.
   2. The team shall test and debug the application beforehand.
2. Security Requirements
   1. Users must be logged in to access the application pages.
   2. Users must be redirected to the landing page if they are not logged in.
   3. The system shall handle exceptions and display them as alerts.
   4. The application must not leak user information.
      1. User information must not be seen or used by others.
      2. User information must be deleted if the user account is deleted.

# Data Dictionary

| Term | Definition |
| --- | --- |
| User | A person who has a registered account and is using the services provided by the application. |
| Account | The profile of the user in our system which contains details (Username, Email, photoURL, searcherBoolean, searcherData, resumeData, uuid) associated with them. |
| Username | A name that is associated with the user. |
| Password | A string of characters (letters, numbers, special symbols) that allow the user to access their account. |
| Job Searcher | The page where users can browse through different jobs recommended to them based on the information gathered by the system from the searcher set up page and what the user input in the search bar. |
| Job Searcher Set Up Information | A set of data about the user such as school, major, skills, minimum salary, jobType and citizenship, used so that the recommendations from the job searcher page can be streamlined to suit their needs. |
| Match | A match is the case where a job is compatible with a user based on the information gathered by the system from the searcher set up page and what the user input in the search bar. |
| Job | An instance of a job from the myCareersFuture API with details like the dueDate, jobTitle, skills[], location, company name, company logo URL, jobType, description, datePosted, salaryRange and jobURL. Users can add jobs from the job searcher page to the wish-list in the job tracker page. |
| Job Tracker | A page where users can find their wishlist and track list. |
| Track list | A page where users can find a list of all the jobs they applied to and their current statuses. |
| Wishlist | A list containing the jobs added by the users from the job searcher page which they are interested in. |
| Job Tracker Entry | An instance of a job entry of the user with details like the dueDate, jobTitle, skills[], location, company name, company logo URL, jobType, description, datePosted, salaryRange, jobURL and status. |
| Job Booster | A page where users can see their tracked jobs and create a boosted resume for them. |
| Resume Boost | Generate a resume for the users based on the resumeData the user inputted and the job entry information with edits by ChatGPT using the OpenAI API. |
| Forum | A community where users can interact with one another and share their thoughts regarding job applications. |
| Post | A post contains a title and content. Users may make comments on posts. |
| Comment | A comment is a reply to a post. |

# 

# Use Cases Diagram

## 4.1 Use Case Diagram

## 

## 

# Use Cases Descriptions

## 5.1.1 Navigate to Tracker Page

| Use Case ID: | **NAV1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Navigation to Tracker Page** | | |
| Created By: | **Low Yong Keng** | Last Updated By: | **Oak Soe Khant** |
| Date Created: | **12.02.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **With the use of the navigation bar and router, users can navigate to the Tracker Page.** |
| Preconditions: | 1. **User must be logged in.** |
| Postconditions: | 1. **User will be navigated to the tracker page** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User clicks on “Tracker” button in NavigationBarUI** 2. **RedirectToTracker() called** 3. **User directed to Tracker page** |
| Alternative Flows: |  |
| Exceptions: | **User can not see the navigation bar if they are not logged in.** |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.1.2 Navigate to Searcher Page

| Use Case ID: | **NAV2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Navigate to Searcher Page** | | |
| Created By: | **Low Yong Keng** | Last Updated By: | **Oak Soe Khant** |
| Date Created: | **12.02.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **With the use of the navigation bar and router, users can navigate to the Searcher Page.** |
| Preconditions: | 1. **User must be logged in.** |
| Postconditions: | 1. **User will be navigated to the searcher page** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User clicks on “Searcher” button in NavigationBarUI** 2. **redirectToSearcher() called** 3. **User directed to Searcher page** |
| Alternative Flows: |  |
| Exceptions: | **User can not see the navigation bar if they are not logged in.** |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.1.3 Navigate to Booster Page

| Use Case ID: | **NAV3** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Navigate to Booster Page** | | |
| Created By: | **Low Yong Keng** | Last Updated By: | **Oak Soe Khant** |
| Date Created: | **12.02.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **With the use of the navigation bar and router, users can navigate to the Booster Page.** |
| Preconditions: | 1. **User must be logged in.** |
| Postconditions: | 1. **User will be navigated to the booster page** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **Clicks “Booster” button in NavigationBarUI** 2. **redirectToBooster() called** 3. **User redirected to Booster page** |
| Alternative Flows: |  |
| Exceptions: | **User can not see the navigation bar if they are not logged in.** |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.1.4 Navigate to Forum Page

| Use Case ID: | **NAV4** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Navigate to Forum Page** | | |
| Created By: | **Low Yong Keng** | Last Updated By: | **Oak Soe Khant** |
| Date Created: | **12.02.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **With the use of the navigation bar and router, users can navigate to the Forum Page.** |
| Preconditions: | 1. **User must be logged in.** |
| Postconditions: | 1. **User will be navigated to the forum page** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User clicks "Forum" button in NavigationBarUI** 2. **redirectToForum()** 3. **User redirected to Forum page** |
| Alternative Flows: |  |
| Exceptions: | **User can not see the navigation bar if they are not logged in.** |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.2.1 Create Account

| Use Case ID: | **ACC1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Create Account** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Oak Soe Khant** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **New User** |
| --- | --- |
| Description: | **Create Account allows new users to create an account in our User Database. To create an account, users must input Username, Email, Password.** |
| Preconditions: | **(For email account creation)**   1. **Passwords must meet the requirements:**     1. **At least 1 special character**    2. **At least 1 capital character**    3. **At least 8 characters.** 2. **Email must be unique.**   **(For Google OAuth Login)**   1. **Must have a Google account.** |
| Postconditions: | **(For email account creation)**   1. **User data stored in the user database using default information.** 2. **System redirects the user to the Searcher Page.**   **(For Google OAuth Login)**   1. **User data stored in the user database using Google account information** 2. **System redirects the user to the Searcher Page** |
| Priority: | **High** |
| Frequency of Use: | **Low** |
| Flow of Events: | **(Email)**   1. **User inputs required information: Email, Username, Password and Confirm Password and user clicks "Sign Up"** 2. **User inputs sent to system.** 3. **isValid(signupDetails) is true** 4. **User data is stored into User Database** 5. **User account has been created** 6. **System sends redirect to Searcher Page** 7. **User redirected to Searcher Page**   **(Google OAuth)**   1. **User clicks on "Login with Google" button and selects account** 2. **User data sent to loginCtr** 3. **loginGmail(gmailDetails)** 4. **If loginGmail(gmailDetails) is successful** 5. **System sends redirect to Searcher Page** 6. **User redirected to Searcher Page** |
| Alternative Flows: | **(Email)**  **isValid(signupDetails) is false**   1. **Displays relevant error message.** 2. **User submits new details**   **If loginGmail(gmailDetails) is unsuccessful**   1. **Display ask for another Gmail account** 2. **User sees Gmail login page** 3. **User resubmits login with Gmail** |
| Exceptions: | **Users can have same username** |
| Includes: | **Access User Database API Call (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.2.2 Login

| Use Case ID: | **ACC2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Login** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Oak Soe Khant** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Login allows users to enter the app to use features catered for registered users of the app. To login to the system, the user must input an existing account details which consists of email and password.**  **Users can also login using Google OAuth with their existing Google account.** |
| Preconditions: | **(Email)**   1. **Must have an existing account in the user database.**   **(Google OAuth)**   1. **Must have a Google account.** |
| Postconditions: | **(Email)**   1. **System directs the user to the Searcher Page.**   **(Google OAuth)**  **(If user has not register an account)**   1. **System will create and store the user account into the user database and direct the user to the Searcher Page.**   **(If user has created an account using Google OAuth)**   1. **System directs the user to the Searcher Page.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | **(Email)**   1. **User enters Email and Password and clicks on "Log In" button** 2. **User data sent to LoginCtr** 3. **LoginCtr request data from User Database** 4. **Sends all User data to LoginCtr** 5. **Direct User to Searcher Page**   **(Google OAuth)**   1. **User clicks on "Login with Google" button** 2. **User selects account to login from pop up UI** 3. **User data sent to LoginCtr** 4. **Retrieve user details from User Database** 5. **Sends all User data to LoginCtr** 6. **Direct User to Searcher Page** |
| Alternative Flows: | **(Email)**  **isEmpty == True**   1. **System displays an error message.** 2. **System prompts user to input email and password.**   **accountMatch() == False**   1. **Error message to prompt user to enter valid email and password**   **(Google OAuth)**  **New User (accountExists() == False)**   1. **New User data is sent to User Database** 2. **Successful account created** 3. **Direct User to Searcher Page** |
| Exceptions: | **Users using Google Login will not be able to login using the default method.** |
| Includes: | **Access User Database API Call (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 

## 5.2.3 Logout

| Use Case ID: | **ACC3** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Logout** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Logout allows users to log out from the app. To logout, the user must be logged in.** |
| Preconditions: | 1. **User must be logged in** 2. **Users can be in any of the Searcher, Tracker, Booster, Forum pages.** |
| Postconditions: | 1. **User is sent to landing page** |
| Priority: | **High** |
| Frequency of Use: | **Low** |
| Flow of Events: | 1. **User clicks on “Log Out” button.** 2. **User is redirected to landing page** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.3.1 Update Job Search Fields

| Use Case ID: | **SEA1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Update Job Search Fields** | | |
| Created By: | **Keith** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Allows users to update their survey results used to search for jobs.** |
| Preconditions: | 1. **Users must be logged in.** |
| Postconditions: | 1. **Job search fields updated.** 2. **Users are allowed to start searching for jobs.** |
| Priority: | **High** |
| Frequency of Use: | **Medium** |
| Flow of Events: | 1. **User goes to SetupJobSearcherPage** 2. **Users select and update fields (School, Major, Location, Type of Job, Citizenship, Minimum Salary, Skills).** 3. **User clicks on “Submit” button**   **if isValidSearchFields == True**   1. **User data sent to UserDatabase** 2. **User Database updates successfully** 3. **Redirects user to Searcher Page** |
| Alternative Flows: | **if isValidSearchFields == False**   1. **Error prompt user to enter correct data fields** 2. **User sees error message** |
| Exceptions: |  |
| Includes: | **Access User Database API Call (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.3.2 Search Jobs Available

| Use Case ID: | **SEA2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Search Jobs Available** | | |
| Created By: | **Keith Teo** | Last Updated By: | **Yong Keng** |
| Date Created: | **12.02.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **For users to see all the jobs available to the user based on the Job Search Fields filled by the user.** |
| Preconditions: | 1. **Users must have filled up the Job Search Fields in order for the System to show all jobs that match the user.** |
| Postconditions: | 1. **Lists all jobs available to the user based on the Job Search Fields.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User clicks on the search icon** 2. **Request for search jobs** 3. **ifExistsSearcherSetup(userID) exists** 4. **RetreiveSearcherProfile(userID) called** 5. **Returns searcher profile** 6. **getJob(searcherProfile) called** 7. **Returns list of jobs** 8. **Displays job browsing cards** 9. **User sees first job card** |
| Alternative Flows: | **searchSetup is false**   1. **Redirect to searcher setup page** 2. **User is redirected to Searcher Setup Page** |
| Exceptions: |  |
| Includes: | **myCareersfuture Jobs API Call (API4)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 

## 5.3.3 Browse Jobs

| Use Case ID: | **SEA3** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Browse Jobs** | | |
| Created By: | **Keith Teo** | Last Updated By: | **Yong Keng** |
| Date Created: | **12.02.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **For users to see all the jobs available to the user based on the Job Search Fields filled by the user.** |
| Preconditions: | 1. **User has clicked on the search button** 2. **List of jobs already loaded into the web application** 3. **UI has finished loading and displays a job for the user** 4. **User is able to drag the card to the left or to the right** |
| Postconditions: | 1. **Lists all jobs available to the user based on the Job Search Fields.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User sent here after clicking on search button and jobs loaded** 2. **Requests list of jobs** 3. **Displays first job card**   **While browsing is true**   1. **If user swipes right, Job added to wishlist** 2. **Shows next card** 3. **User sees next job** |
| Alternative Flows: | **if userSwipe is left**   1. **Job not added to wishlist** 2. **Shows next card** 3. **User sees next card** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.3.4 View Wishlist Entry

| Use Case ID: | **TRA1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **View Wishlist Entry** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Users can view all the jobs that the user has selected from the Searcher Page by swiping right.** |
| Preconditions: | 1. **Users must be logged in.** 2. **User wish list will only show if User has swiped at least 1 job in Searcher Page.** |
| Postconditions: | 1. **System displays all user entries.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User goes to the Tracker Page.** 2. **TrackerPage requests for Wishlist Entry data** 3. **getWishlistEntry(userID)** 4. **Return data** 5. **Displays user's entries** 6. **User sees Job Entries** |
| Alternative Flows: | **if isExpanded is true**   1. **User clicks on the blue down arrow on the right side of the job listing.** 2. **User can view more information such as Job Description, Skills, Pay, Job Address, the Job Url and the Google Maps of the actual location based on the Job Address.** 3. **User sees information on Job Entries.** |
| Exceptions: |  |
| Includes: | **Access Forum Database API Call (API5)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.3.5 Delete Wishlist Entry

| Use Case ID: | **TRA3** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Delete Wishlist Entry** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Keith** |
| Date Created: | **27.01.2023** | Date Last Updated: | **03.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Delete Wishlist Entry allows users to remove an existing job that the user created previously.** |
| Preconditions: | 1. **Entry must exist before allowing users to delete** |
| Postconditions: | 1. **Entry is deleted from the user database.** |
| Priority: | **High** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on the red “x”** 2. **Delete request sent** 3. **deleteEntry(entryID) called** 4. **Successful delete from User Database** 5. **Displays without deleted entry** 6. **User sees entry disappear** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: | **Access User Database API Call (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.4.1 View Tracker Entry

| Use Case ID: | **TRA4** | | |
| --- | --- | --- | --- |
| Use Case Name: | **View Tracker Entry** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Keith** |
| Date Created: | **27.01.2023** | Date Last Updated: | **03.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Users can view all the jobs that the user has selected from their Wishlist.** |
| Preconditions: | 1. **Users must be logged in.** |
| Postconditions: | 1. **User Tracker Entry is displayed** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User goes to the Tracker Page.** 2. **TrackerPage requests for Tracker Entry data** 3. **getTrackerEntries() called** 4. **Return Tracker Entry data from Database** 5. **Displayed after Job Company Name, Job Role, the Location, the Job Url and the Status of the job.** 6. **User sees Job Entries** |
| Alternative Flows: | **isExpanded == true**   1. **User clicks on "View More" button** 2. **TrackerPage requests for moreTracker Entry data** 3. **User can view more information such as Job Description, Skills, Pay, Job Address, the Job Url and the Google Maps of the actual location based on the Job Address.** 4. **User sees more information on Job Entries** |
| Exceptions: |  |
| Includes: | **Access User Database API Call (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.4.2 Create Tracker Entry

| Use Case ID: | **TRA5** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Create Tracker Entry** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **27.01.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Users can create Tracker Entry from their Wishlist Entry. Users can also manually add in their wanted jobs from another website to keep track of their job applications.** |
| Preconditions: | 1. **Users must be logged in.** |
| Postconditions: | 1. **A new entry is created.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User goes to the Tracker Page.** 2. **Users click on the green tick in the Wishlist job entries to add jobs they wish to apply to into their Tracker Entry.** 3. **Wishlist job entry is removed and then added into Tracker Entry in the User Database.** |
| Alternative Flows: | **User manually adds in a new job entry**   1. **User clicks on the “+” button beside Applications.** 2. **A Job Entry UI appears with 10 fields that requires the user to fill up. (Company, Job Title, Country, Address, Minimum Pay, Maximum Pay, Currency, Description, Url, Skills)** 3. **If User leaves either the fields Company, Job Title or Country empty, or the user enters white spaces into those 3 fields, a prompt will appear under the fields that the User did not meet the requirements.** 4. **Once the User finishes filling up the fields, User can click on the “Save Details” button and a new Job Entry will be created under Applications.** |
| Exceptions: | **Users can leave the fields Minimum Pay, Maximum Pay, Currency, Description, Url and Skills empty.** |
| Includes: | **Access User Database API Call (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 

## 5.4.3 Delete Tracker Entry

| Use Case ID: | **TRA6** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Delete Tracker Entry** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **27.01.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Updates user’s previously created entries if the user did not input the correct details.** |
| Preconditions: |  |
| Postconditions: |  |
| Priority: | **High** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on the red "x" beside the job they wish to delete** 2. **Delete request sent** 3. **trackerEntryDelete() called** 4. **Delete successful from User Database** 5. **Displays remaining entries in Tracker Page** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: | **Access User Database API Call (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 

## 5.4.4 Update Tracker Entry

| Use Case ID: | **TRA7** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Update Tracker Entry** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **27.01.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Updates user’s previously created entries if the user did not input the correct details.** |
| Preconditions: | 1. **Must have at least 1 entry created by the user.** |
| Postconditions: | 1. **Selected entry is updated.** |
| Priority: | **Moderate** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on the blue down arrow beside the job entry they wish to update.** 2. **User then clicks on the “Edit Details” button.** 3. **The Job Entry UI pops up with all the already available details filled.** 4. **Users then change the values of the fields they wish to update.** 5. **User clicks on the “Save Details” button to save all the changes and update the job entry in the User Database.** |
| Alternative Flows: | **Company, Job Title or Country field is/are empty/contains white spaces**   1. **Users cannot save details and a prompt under the field, asking User to fill up.**   **Changing of Status of the job**   1. **Users can change the Status of the jobs by clicking on the Status bubble between the Job Location and the blue down arrow key.** 2. **Users can rotate within these 9 Job Status. (Preparing, Applied, Replied, OA Due, OA Done, Interview, Offered, Rejected, Ghosted)** 3. **Everytime the User clicks on it, the Job Entry will be updated in the User Database.** |
| Exceptions: | **Users can leave the fields Minimum Pay, Maximum Pay, Currency, Description, Url and Skills empty.** |
| Includes: | **Access User Database API Call (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 

## 5.5.1 View Forum

| Use Case ID: | **FOR2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **View Forum** | | |
| Created By: | **Keith** | Last Updated By: | **Pavan** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Allows users to view posts on the forum.** |
| Preconditions: | 1. **User must be logged in** |
| Postconditions: | 1. **User views selected forum posts.** |
| Priority: | **Moderate** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on “Forum” button** 2. **Request for list of posts** 3. **getListOfPosts() called** 4. **Forum Database returns list of posts** 5. **List of posts displayed** 6. **User sees posts** 7. **User clicks on a posts** 8. **Request for replies to posts** 9. **getReplies() called** 10. **Forum Database returns replies** 11. **Display replies** 12. **User sees full posts** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: | **Access Forum Database API Call (API5)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.5.2 Create Forum Post

| Use Case ID: | **FOR1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Create Forum Post** | | |
| Created By: | **Keith** | Last Updated By: | **Pavan** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Allows users to make a post on the forum.** |
| Preconditions: | 1. **Users must be logged in to make a post.** |
| Postconditions: | 1. **A new post is created and a corresponding entry inserted into the Forum Database.** |
| Priority: | **Moderate** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on “+ Post”** 2. **Page displays create post card** 3. **User enters post title, content and presses “Post”** 4. **Post details submitted** 5. **addNewPost(post)** 6. **New post added** 7. **New post displayed** 8. **User sees new post** |
| Alternative Flows: | **isEmpty(postTitle) is true**   1. **Display Message “Please fill in this field.”** 2. **New post submitted**   **isEmpty(postContent) is true**   1. **Display Message “Please fill in this field.”** 2. **New post submitted** |
| Exceptions: |  |
| Includes: | **Access Forum Database API Call (API5)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.5.3 Create Forum Comment

| Use Case ID: | **FOR1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Create Forum Comment** | | |
| Created By: | **Keith** | Last Updated By: | **Pavan** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Allows users to make a comment on the forum post.** |
| Preconditions: | 1. **Users must be logged in to make a comment.** |
| Postconditions: | 1. **A new comment is created in the replies to post and corresponding entry inserted into the Forum Database.** |
| Priority: | **Moderate** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on “+ Comment”** 2. **Page displays create comment card** 3. **User enters post title, content and presses "Comment"** 4. **Comment details submitted** 5. **isEmpty(comment) ran to check if comment empty** 6. **New post submitted** 7. **addNewComment(comment) called** 8. **New comment added** 9. **New comment displayed** 10. **User sees new comment** |
| Alternative Flows: | **isEmpty(comment) is true**   1. **Display Message “Please fill in this field.”** 2. **New comment submitted** |
| Exceptions: |  |
| Includes: | **Access Forum Database API Call (API5)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.6.1 Set Resume

| Use Case ID: | **BOO1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Set Resume** | | |
| Created By: | **Low Yong Keng** | Last Updated By: | **Pavan** |
| Date Created: | **12.02.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **User uploads resume for the System to add in more details suited for the job selected by the user.** |
| Preconditions: | 1. **Users must have a resume written.** |
| Postconditions: | 1. **Resume is uploaded to System** |
| Priority: | **High** |
| Frequency of Use: | **Medium** |
| Flow of Events: | 1. **User clicks on “Resume Upload”** 2. **Page displays upload resume card** 3. **User enters resume details** 4. **Resume details submitted** 5. **createResume(resume) called** 6. **Returns confirmation resume created** 7. **Displays confirmation resume uploaded** 8. **User sees new resume** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: | **Access User Database API Call (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 

## 

## 5.6.2 Boost Resume

| Use Case ID: | **BOO2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Boost Resume** | | |
| Created By: | **Low Yong Keng** | Last Updated By: | **Pavan** |
| Date Created: | **12.02.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Resume uploaded by the user will be edited by the System based on the related job description that the user chose.** |
| Preconditions: | 1. **User is logged in.** 2. **At least one resume must have been uploaded.** 3. **User pressed the “Boost” button.** |
| Postconditions: | 1. **System sends an updated resume to the user.** |
| Priority: | **Moderate** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on the “Boost”** 2. **Resume sent to application** 3. **boostResume(resume) called from OpenAI API** 4. **Returns boosted resume** 5. **updateResume(resume) called from User Database** 6. **Update successful** 7. **User sees updated resume** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: | **Access User Database API Call (API1), OpenAI API Call (API3)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.7.1 Access User Database API Call

| Use Case ID: | **API1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Access User Database API Call** | | |
| Created By: | **Keith** | Last Updated By: | **Ng Yao Ming** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **User Database** |
| --- | --- |
| Description: | **API that allows access to user information for login, account creation and updating information.** |
| Preconditions: | 1. **Account data must exist in database** |
| Postconditions: | 1. **Frontend receives account information** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **Frontend sends JSON request to user database for account information** 2. **User database replies with relevant information** |
| Alternative Flows: | **Database fetch unsuccessful**   1. **Sends error to Frontend** 2. **Frontend will render and display default information** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 5.7.2 Google Maps API Call

| Use Case ID: | **API2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Google Maps API Call** | | |
| Created By: | **Keith** | Last Updated By: | **Ng Yao Ming** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **Google Maps API** |
| --- | --- |
| Description: | **API that returns the location of an address on a map.** |
| Preconditions: | 1. **Location must be provided** |
| Postconditions: | 1. **Frontend receives a map with location indicated** |
| Priority: | **Low** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **Frontend sends JSON request to Google Maps API for address information** 2. **Google Maps replies with relevant information** |
| Alternative Flows: | 1. **If location does not exist or address is missing, frontend will display “Address not found”** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: | **Google API key remains valid.** |
| Notes and Issues: |  |

## 5.7.3 OpenAI API Call

| Use Case ID: | **API3** | | |
| --- | --- | --- | --- |
| Use Case Name: | **OpenAI API Call** | | |
| Created By: | **Keith** | Last Updated By: | **Ng Yao Ming** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **OpenAI API** |
| --- | --- |
| Description: | **OpenAI API takes in general text prompts and returns answers based on its NLP engine. This is used to suggest edits to the resume of the user.** |
| Preconditions: | 1. **Resume must have been set up first.** |
| Postconditions: | 1. **Returns an edited version of the resume with suggestions.** |
| Priority: | **Low** |
| Frequency of Use: | **Low** |
| Flow of Events: | 1. **Frontend sends API requests to OpenAI containing user resumes and prompts for improvements.** 2. **API responds with suggestions.** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: | **Open AI API key remains valid.** |
| Notes and Issues: |  |

## 5.7.4 myCareersFuture Jobs API Call

| Use Case ID: | **API4** | | |
| --- | --- | --- | --- |
| Use Case Name: | **myCareersFuture Jobs API Call** | | |
| Created By: | **Keith** | Last Updated By: | **Ng Yao Ming** |
| Date Created: | **27.01.2023** | Date Last Updated: | **02.04.2023** |

| Actor: | **myCareersFuture Jobs API** |
| --- | --- |
| Description: | **Fetches a list of jobs from API based on user job search query and job searcher profile.** |
| Preconditions: | 1. **Job searcher profile needs to be completed.** |
| Postconditions: | 1. **Returns list of jobs related to user query and/or survey.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **View Jobs Available use case sends API requests to myCareersFuture Jobs API.** 2. **API responds with a list of results** |
| Alternative Flows: | **API fetch unsuccessful**   1. **Sends error to Frontend** 2. **Frontend will render and display default information** |
| Exceptions: | 1. **No job available for search query and profile.** |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

# 

## 

## 5.7.5 Access Forum Database API Call

| Use Case ID: | **API5** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Access Forum Database API Call** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Ng Yao Ming** |
| Date Created: | **12.02.2023** | Date Last Updated: | **02.04.2023** |

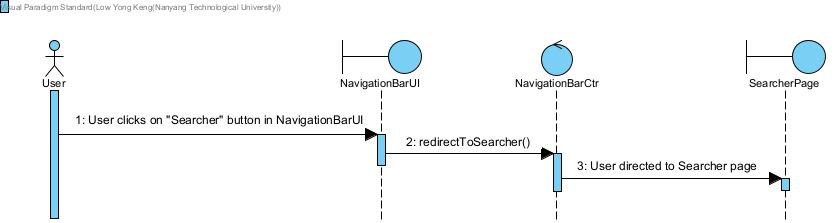
| Actor: | **Forum Database** |
| --- | --- |
| Description: | **Create and read operations for the post and comments in the forum database.** |
| Preconditions: | 1. **User must be logged in** |
| Postconditions: | 1. **Create/read operation is completed.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | **Create operation:**   1. **Application sends create operation with relevant data.** 2. **A new post or comment entry in the database is created.** 3. **Forum database responds with a message confirming successful operation.** |
| Alternative Flows: | **Read operation:**   1. **Application sends read operation for a certain post or comment entry.** 2. **Forum database returns the post entry.** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

# Sequence Diagrams

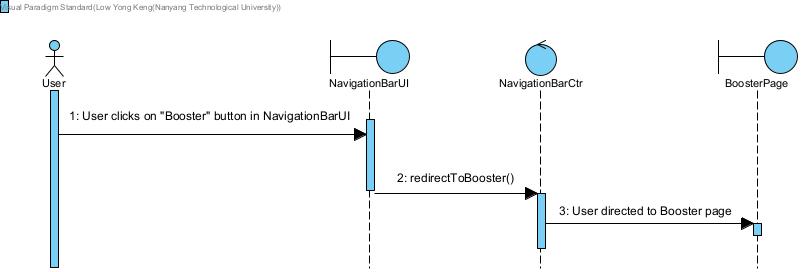
## 6.1.1 Navigate To Tracker Page

## 

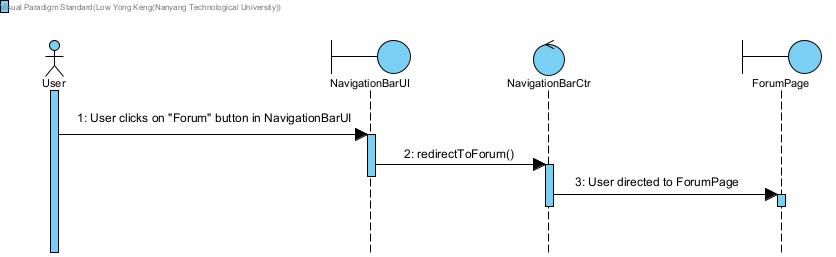
## 6.1.2 Navigate To Searcher Page



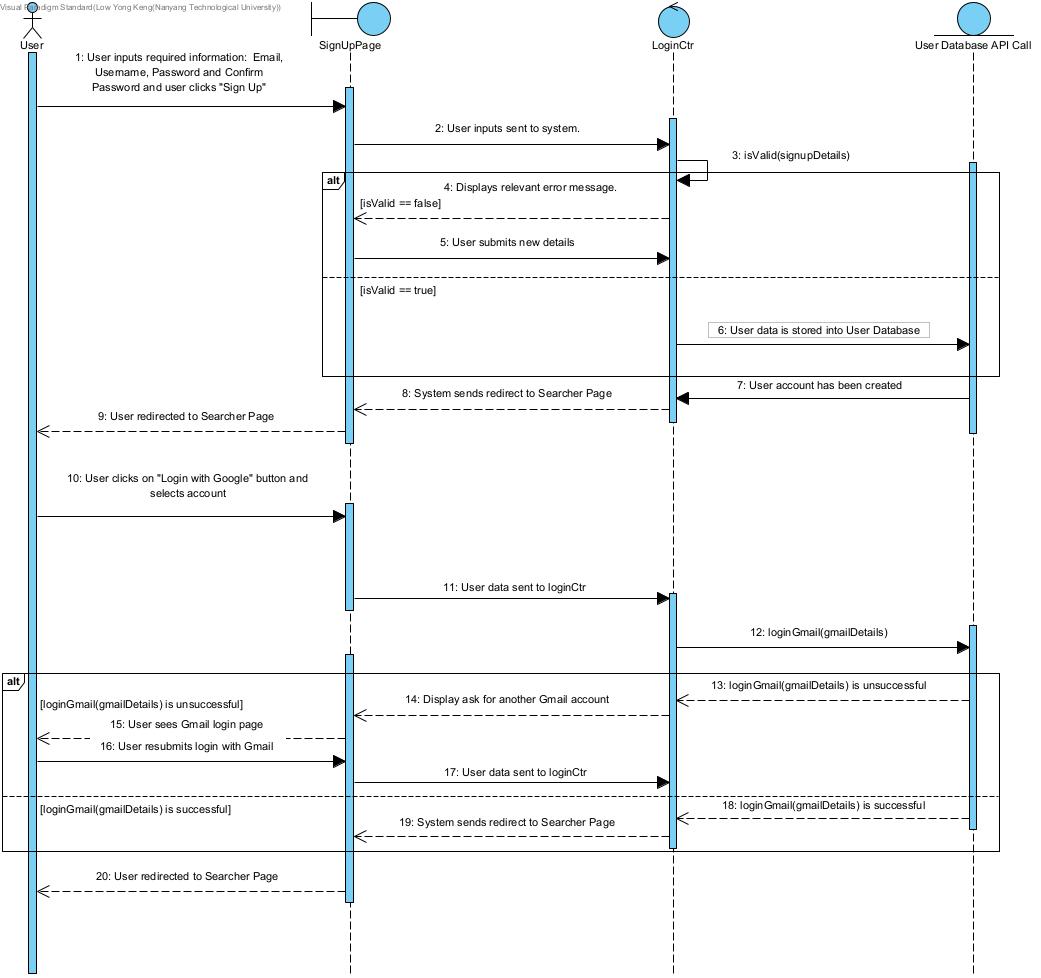
## 6.1.3 Navigate To Booster Page



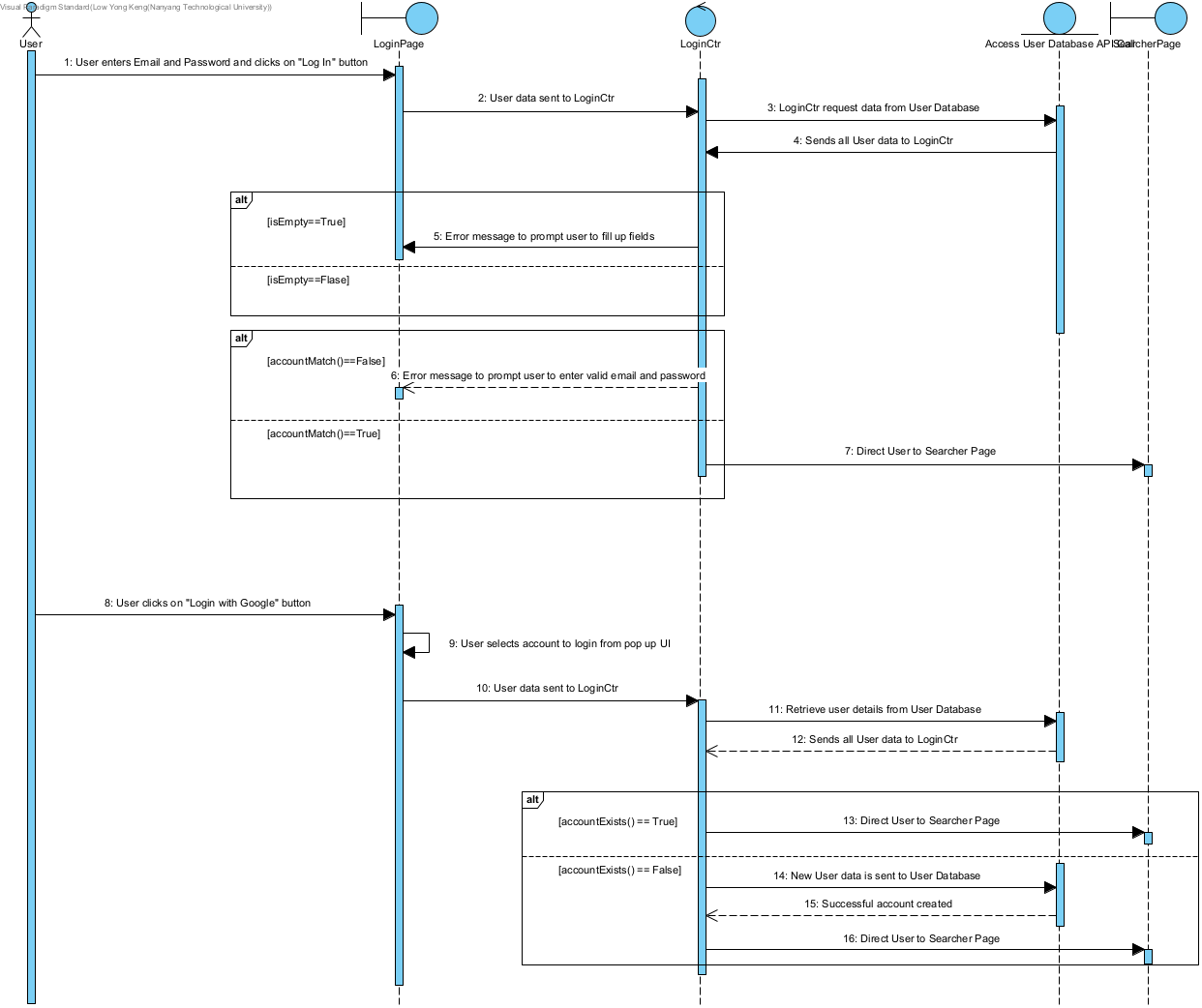
## 6.1.4 Navigate To Forum Page



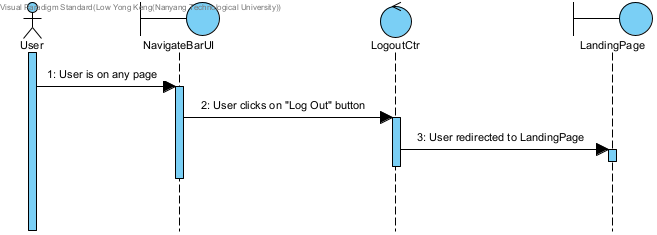
## 6.2.1 Create Account



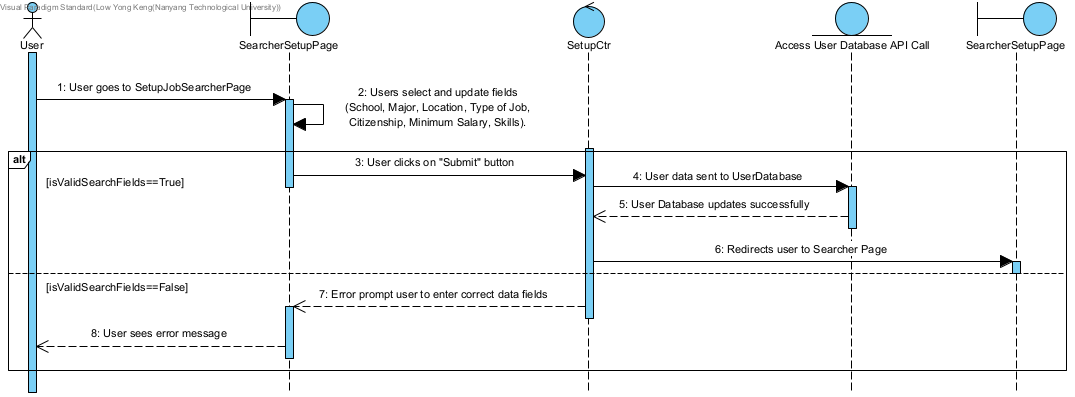
## 6.2.2 Login



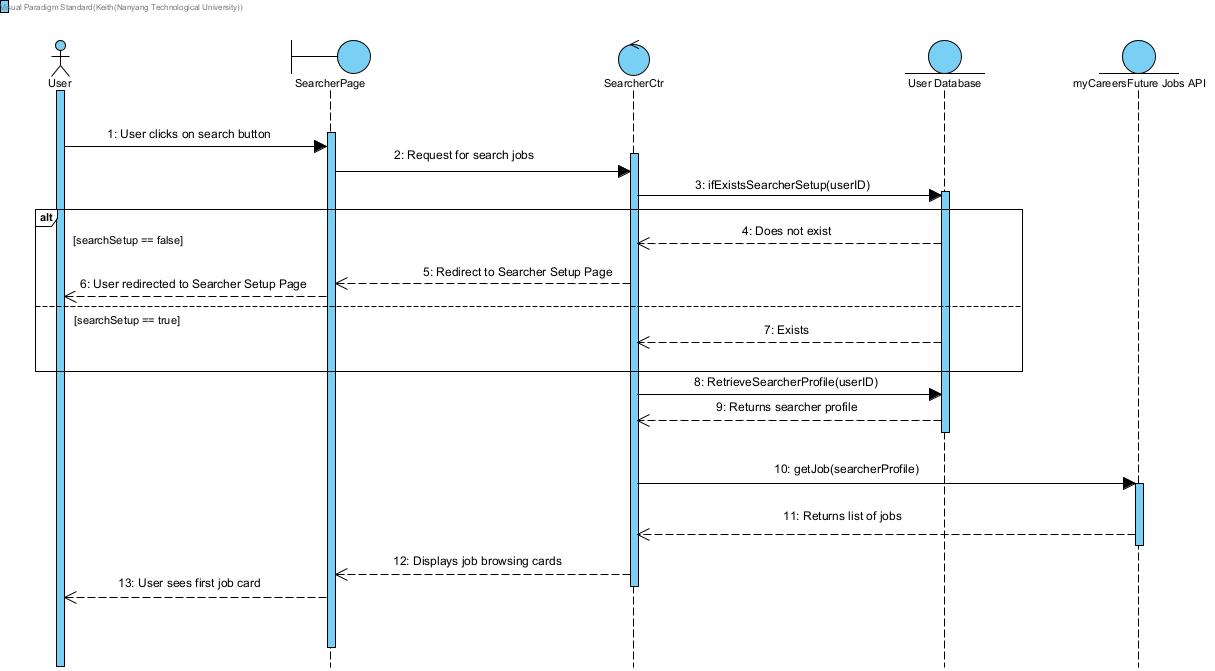
## 6.2.3 Logout



## 6.3.1 Update Job Search Fields

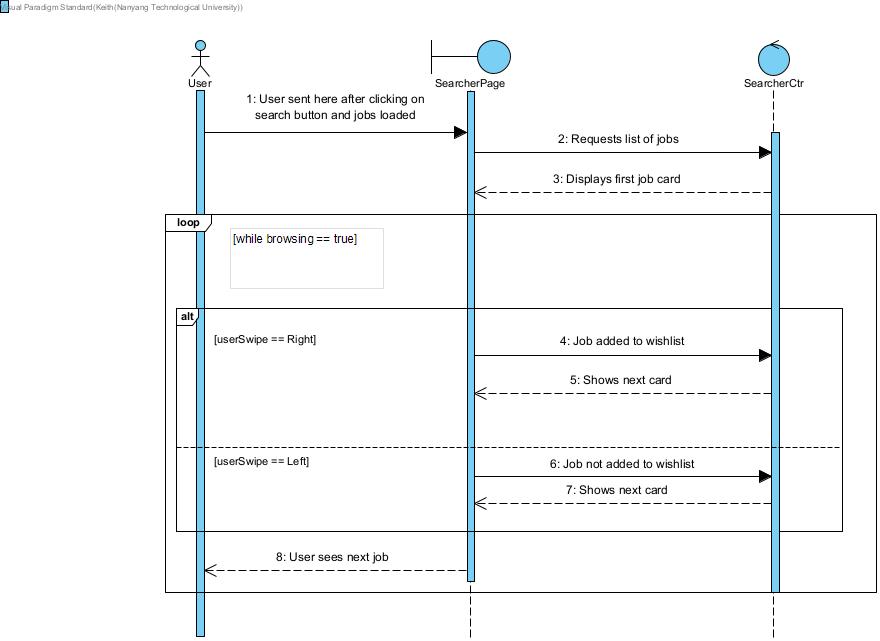


## 6.3.2 Search Jobs Available

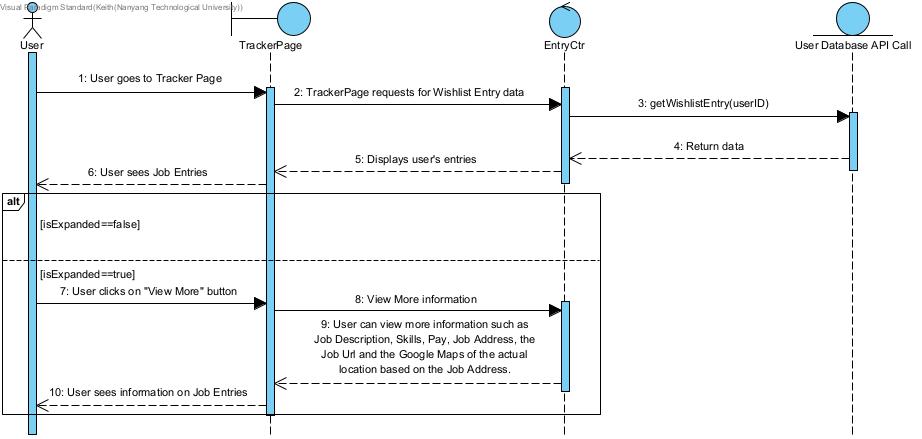


## 

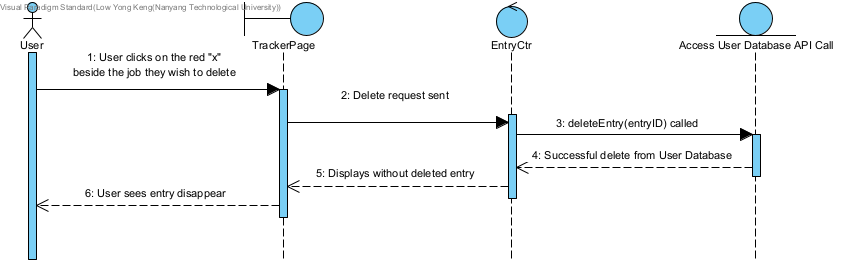
## 6.3.3 Browse Jobs



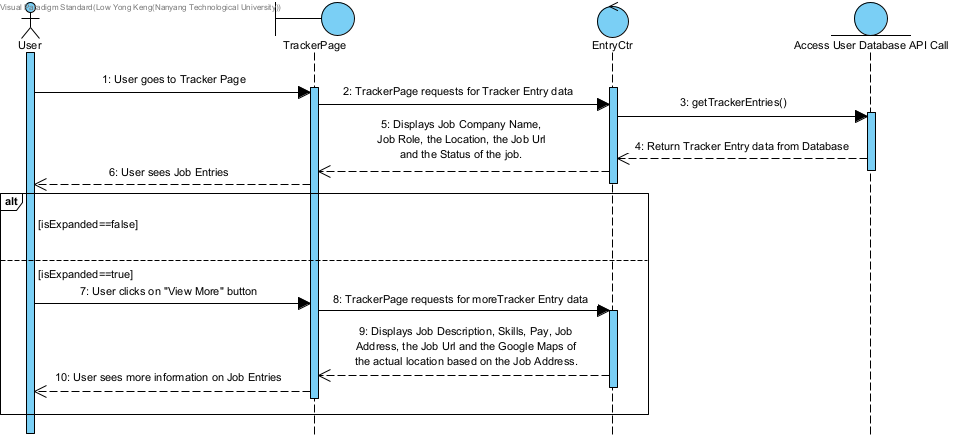
## 6.3.4 View Wishlist Entry



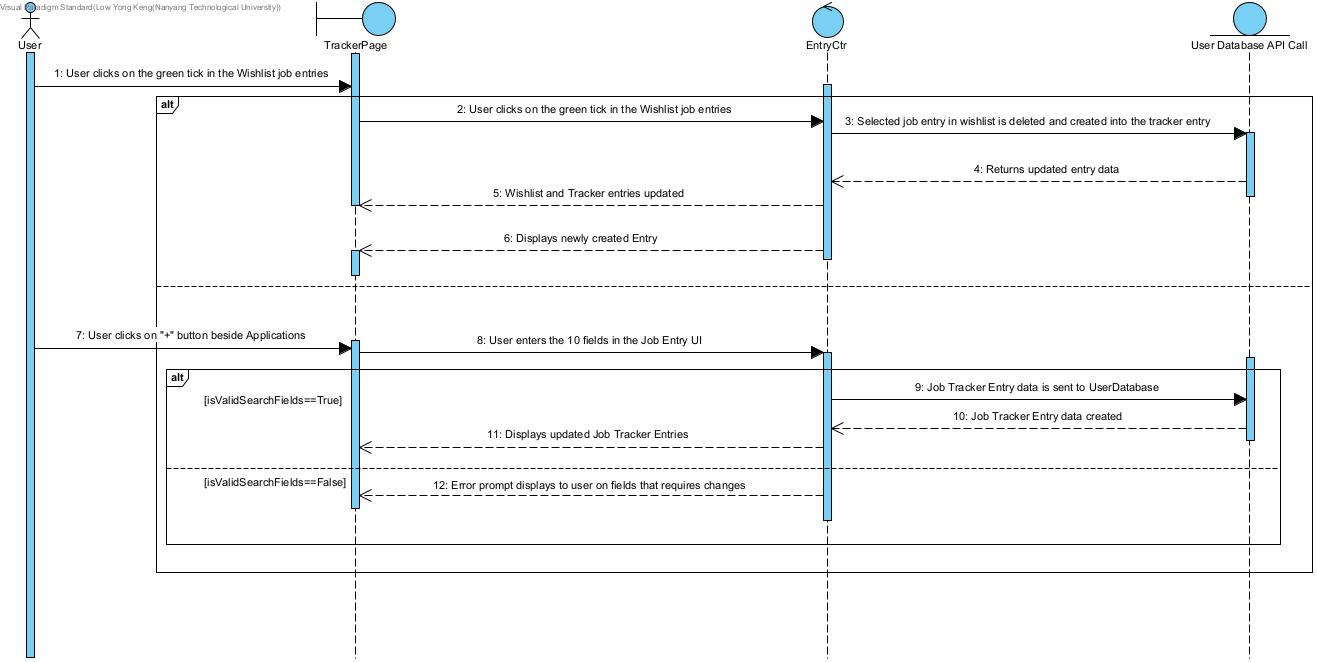
## 6.3.5 Delete Wishlist Entry



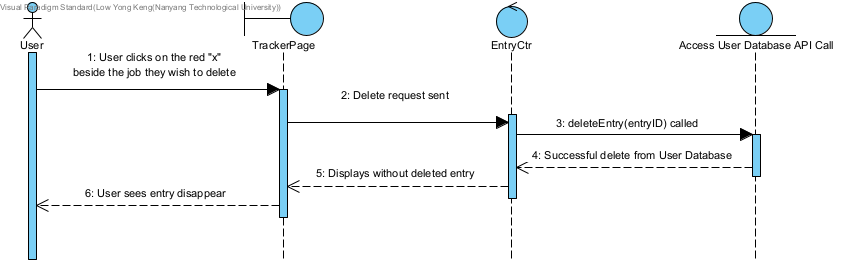
## 6.4.1 View Tracker Entry



## 6.4.2 Create Tracker Entry



## 6.4.3 Delete Tracker Entry

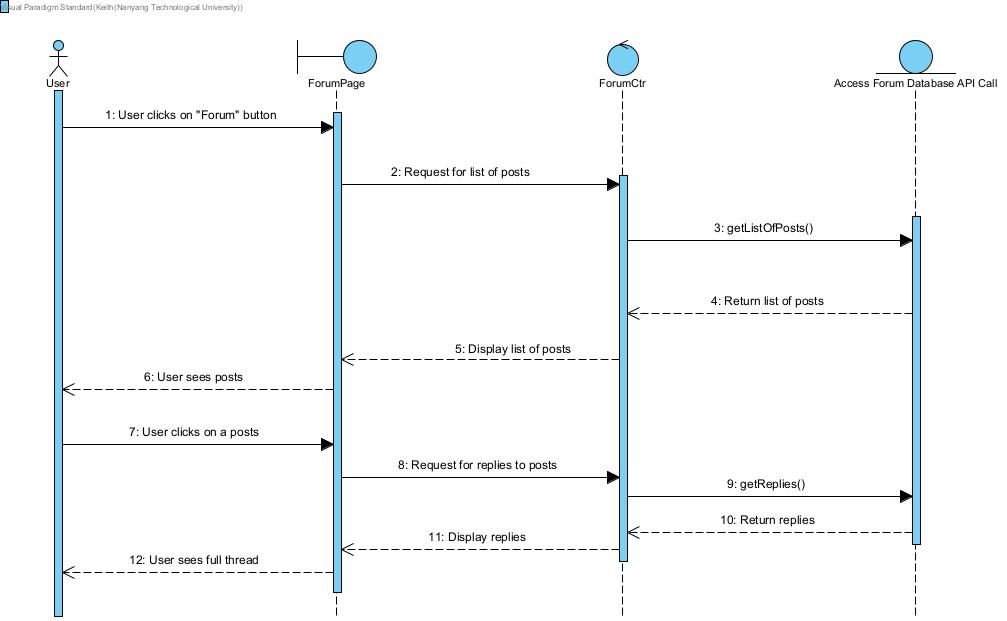


## 6.4.4 Update Tracker Entry

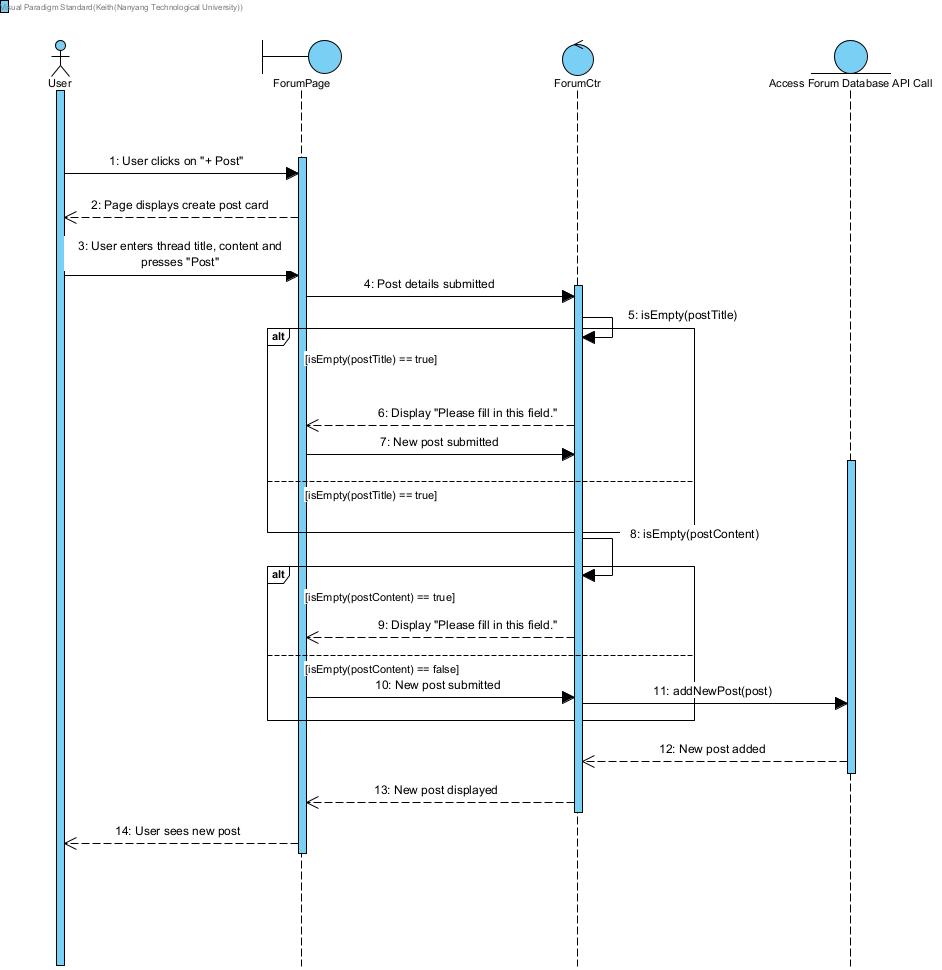


## 

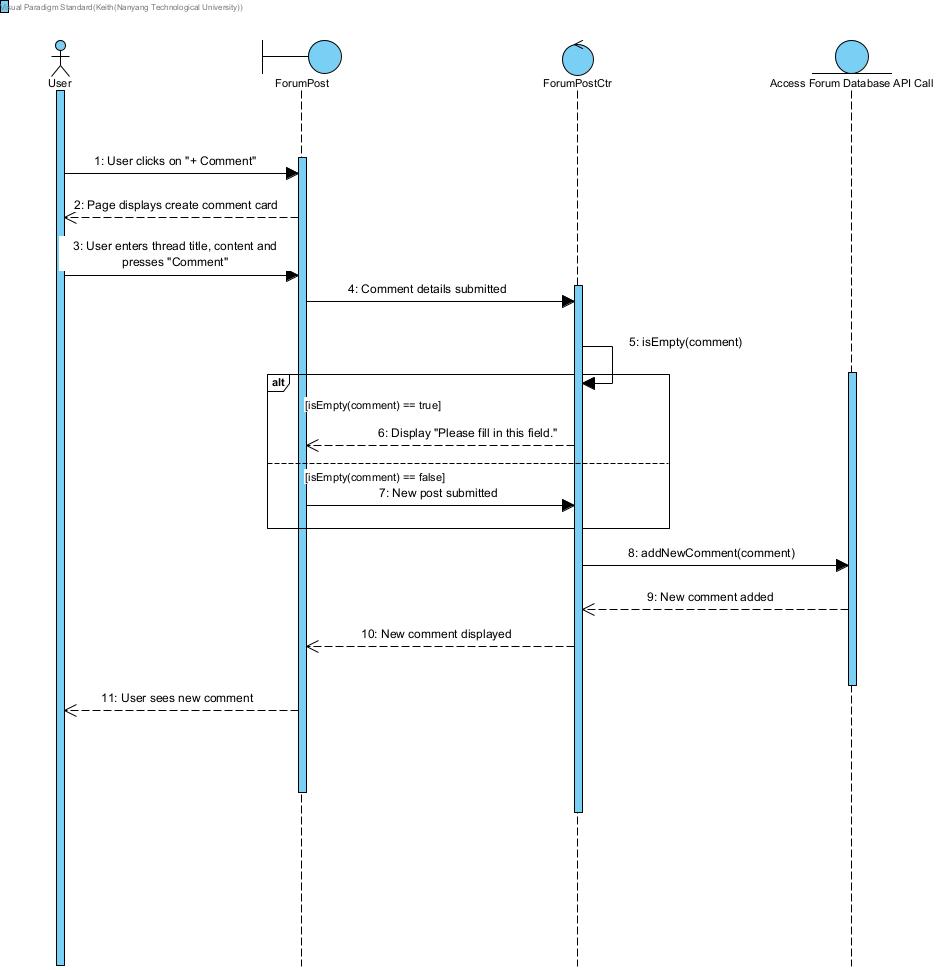
## 6.5.1 View Forum



## 6.5.2 Create Forum Post



## 6.5.3 Create Forum Comment

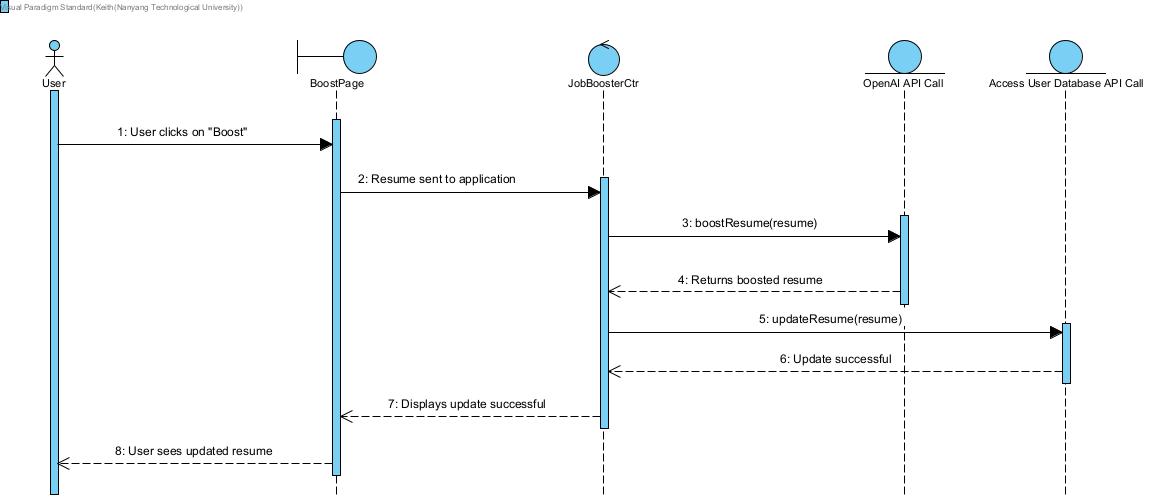


## 6.6.1 Set Resume

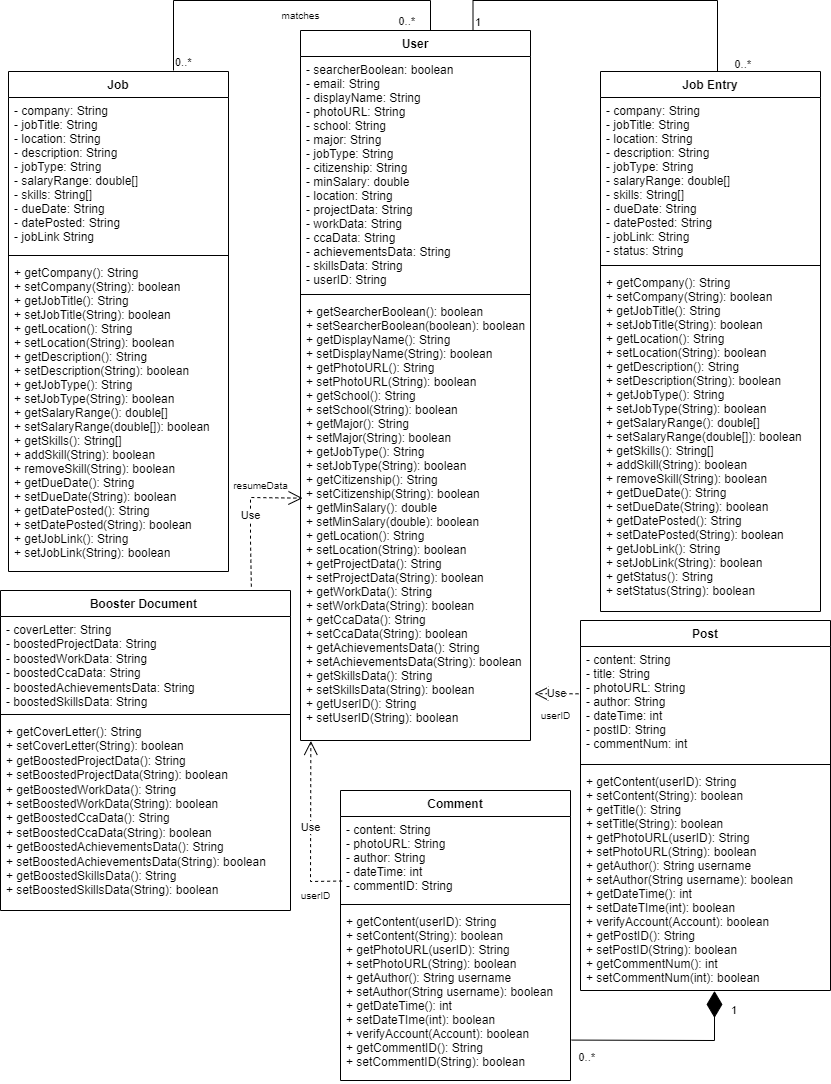
## 

## 

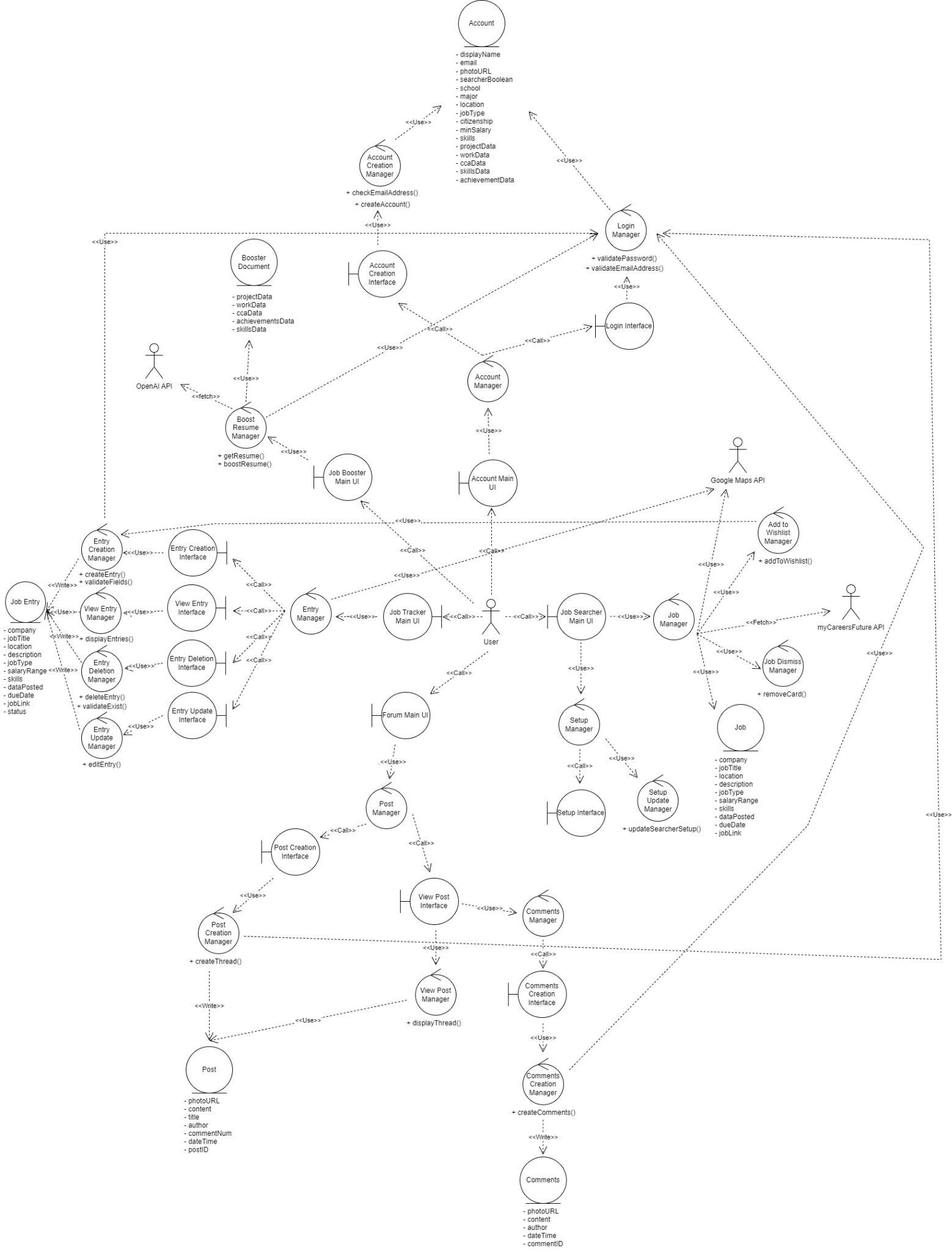
## 6.6.2 Boost Resume



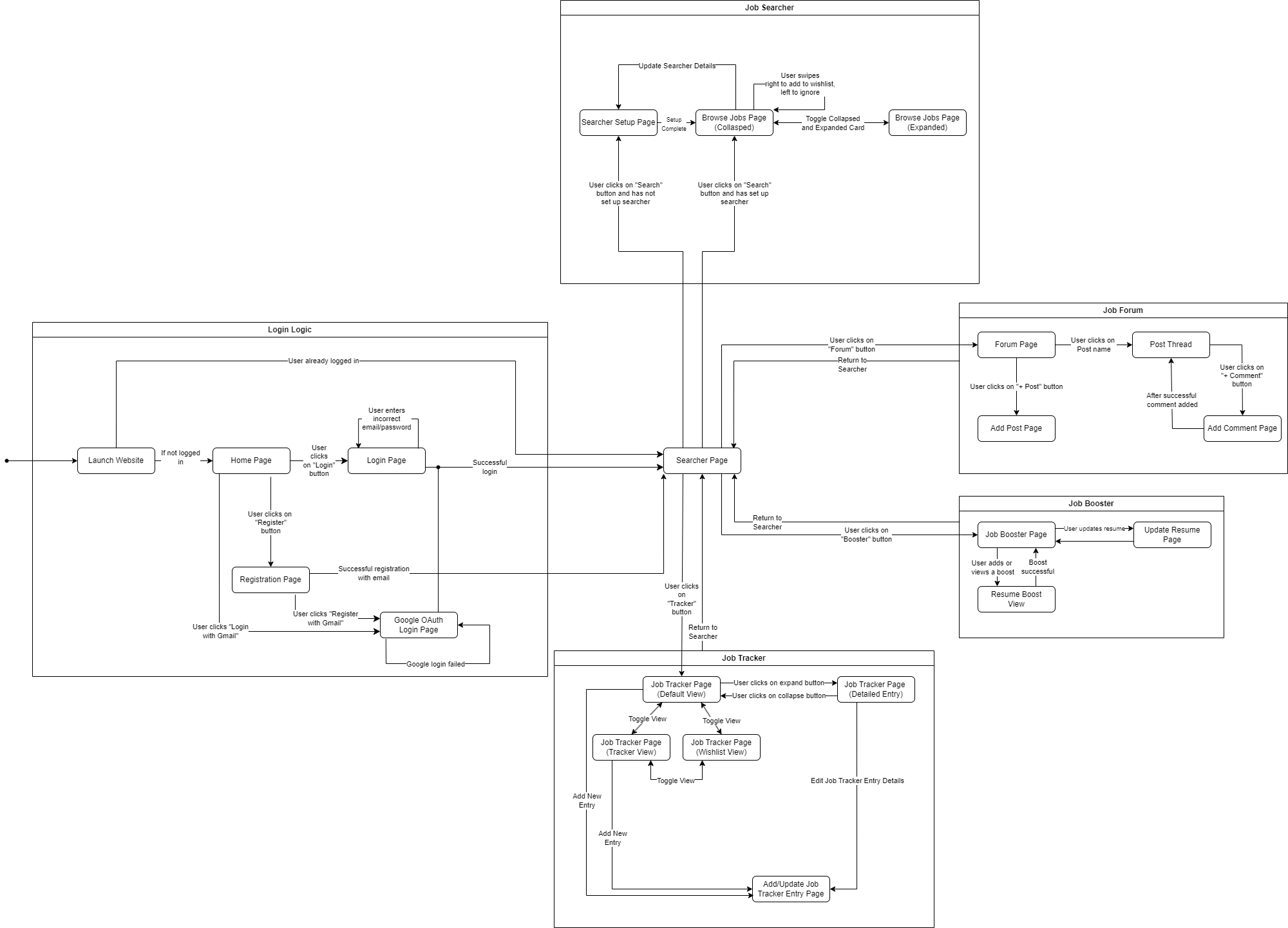
# Entity Class Diagram



# Boundary and Control Class Diagram



# Dialog Map



# UI Mockups (Initial and Final)

## 10.1. Landing Page

**Initial Mockup**



**Final UI**



## 10.2. Signup Page

**Initial Mockup**



**Final UI**

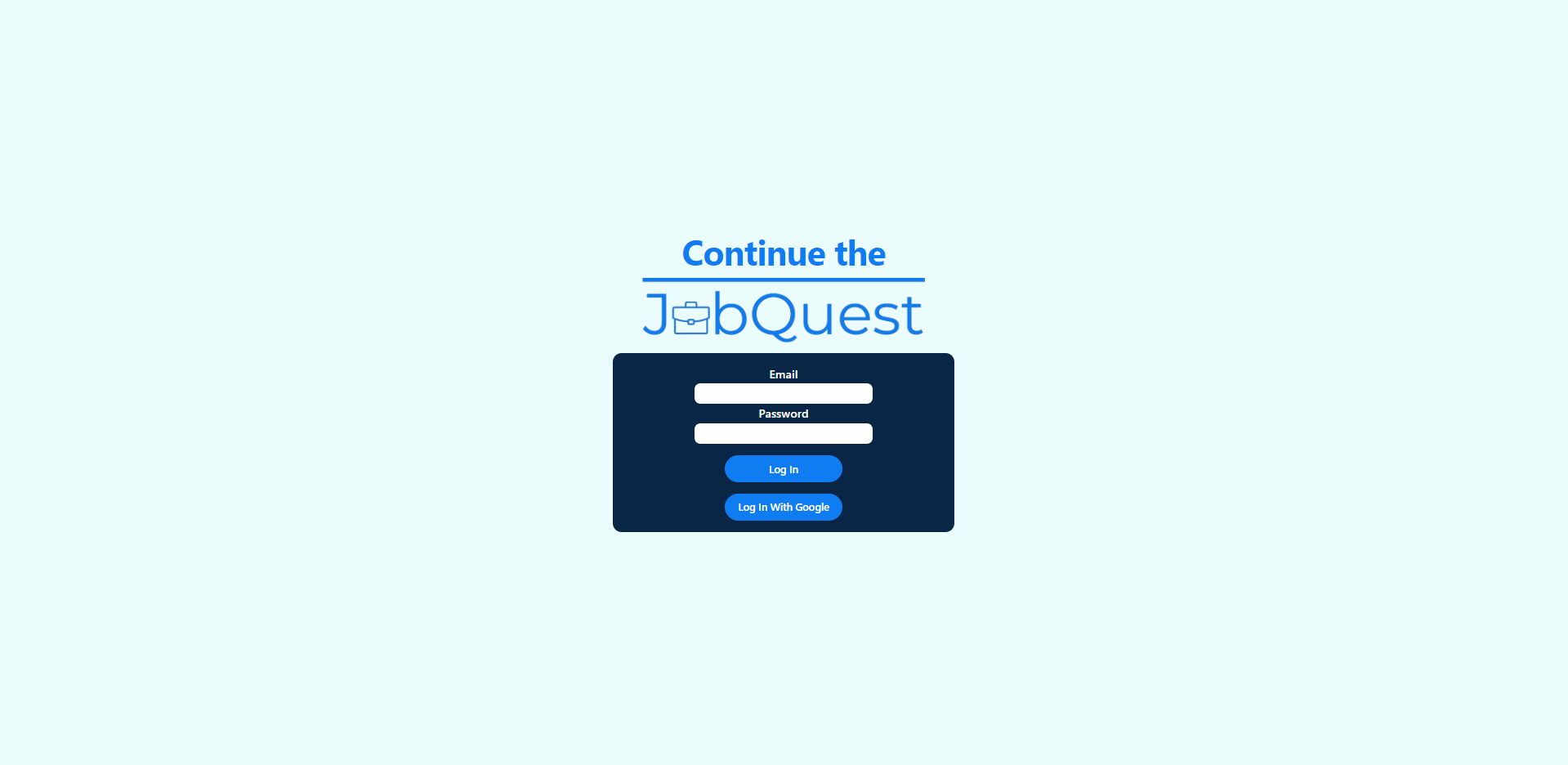


## 10.3. Login Page

**Initial Mockup**

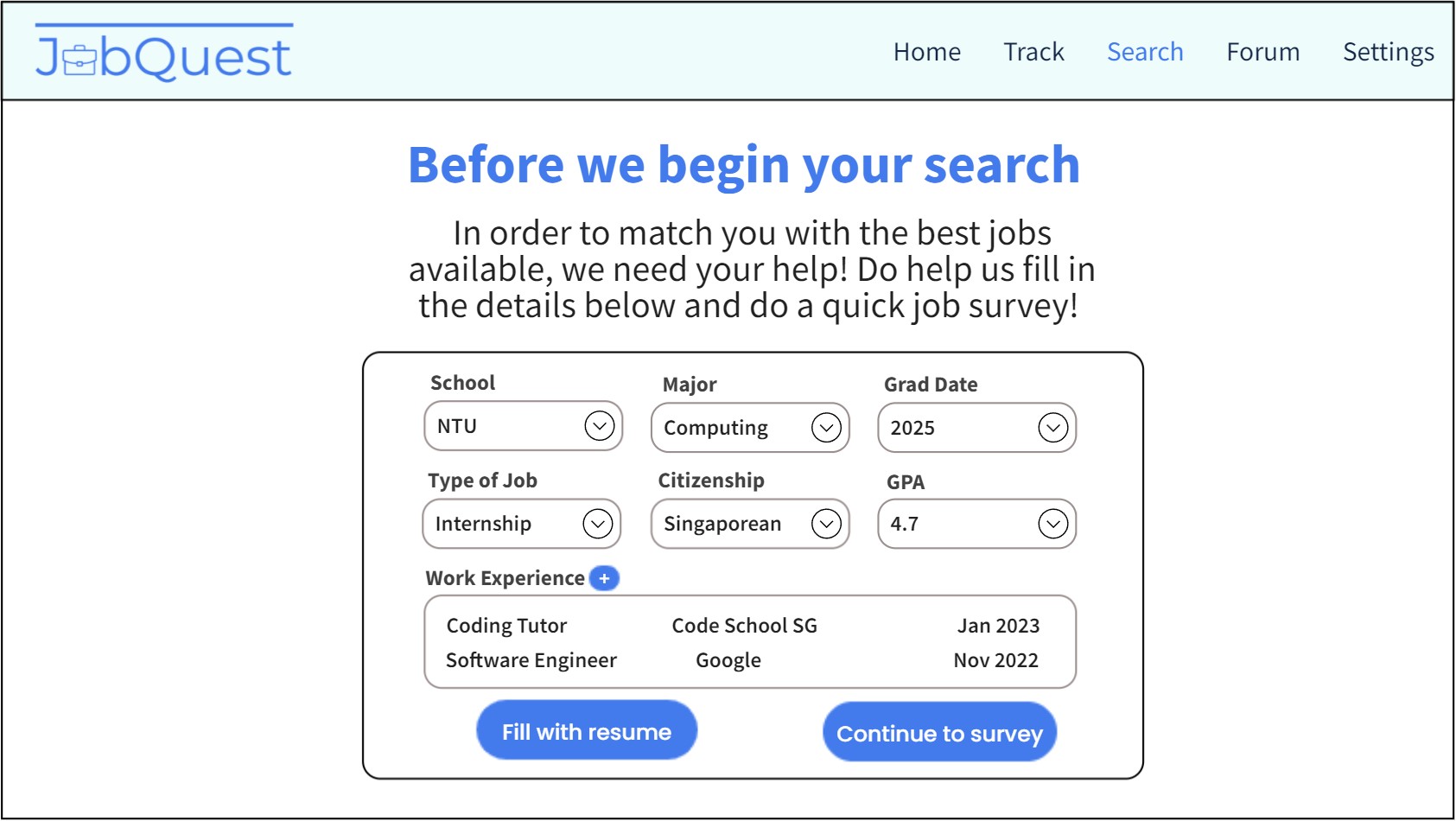


**Final UI**

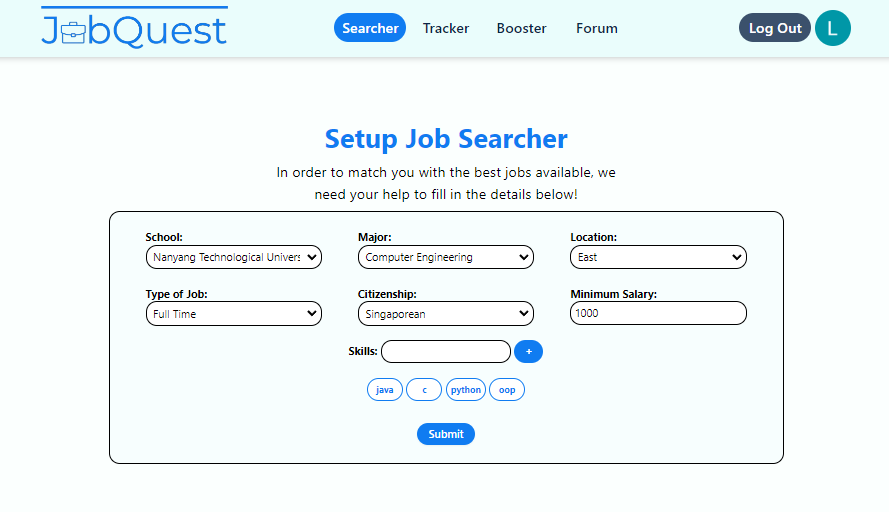


## 10.4. Searcher Set Up Page

**Initial Mockup**

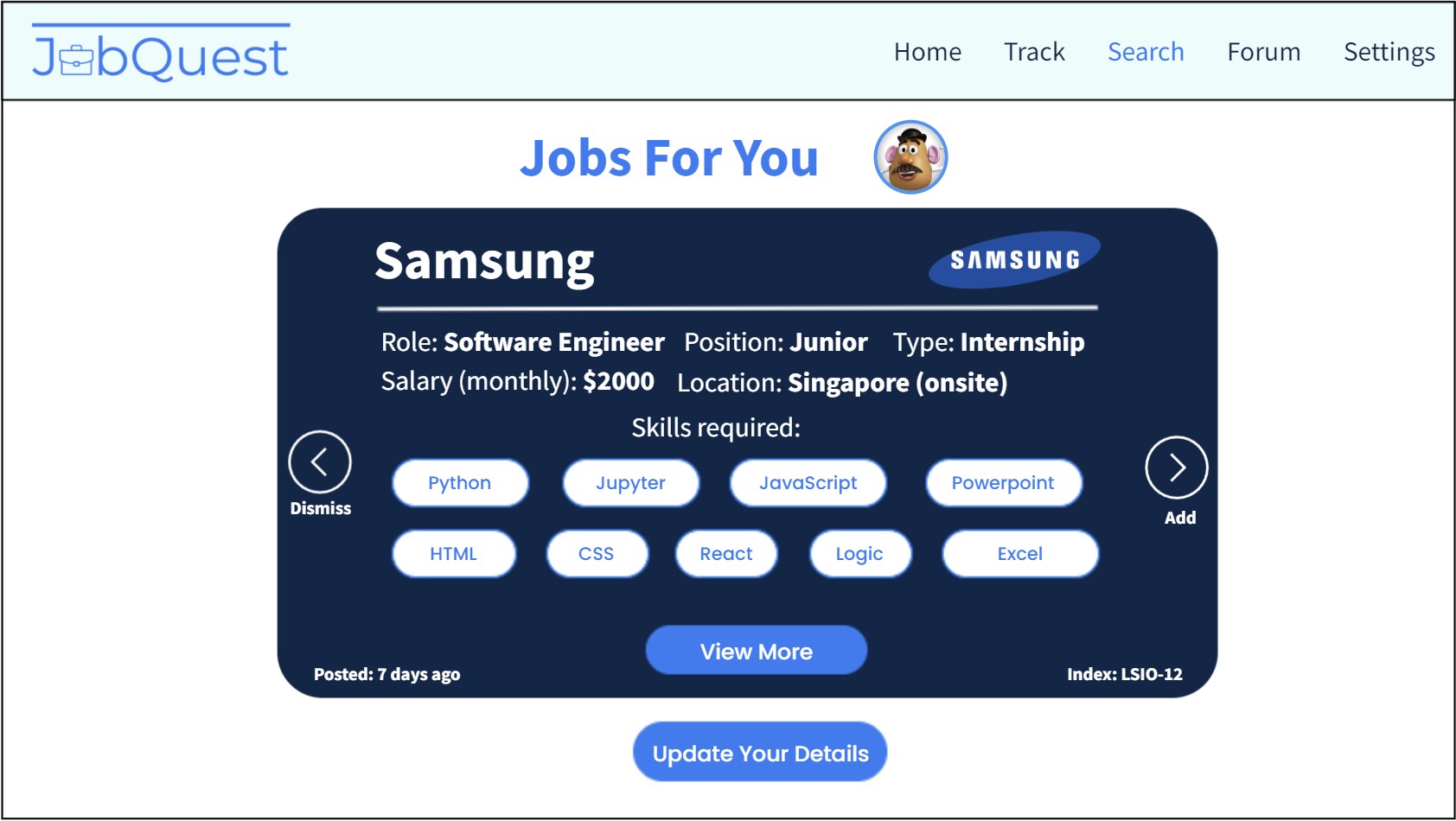


**Final UI**



## 10.5. Searcher Collapsed Card

**Initial Mockup**

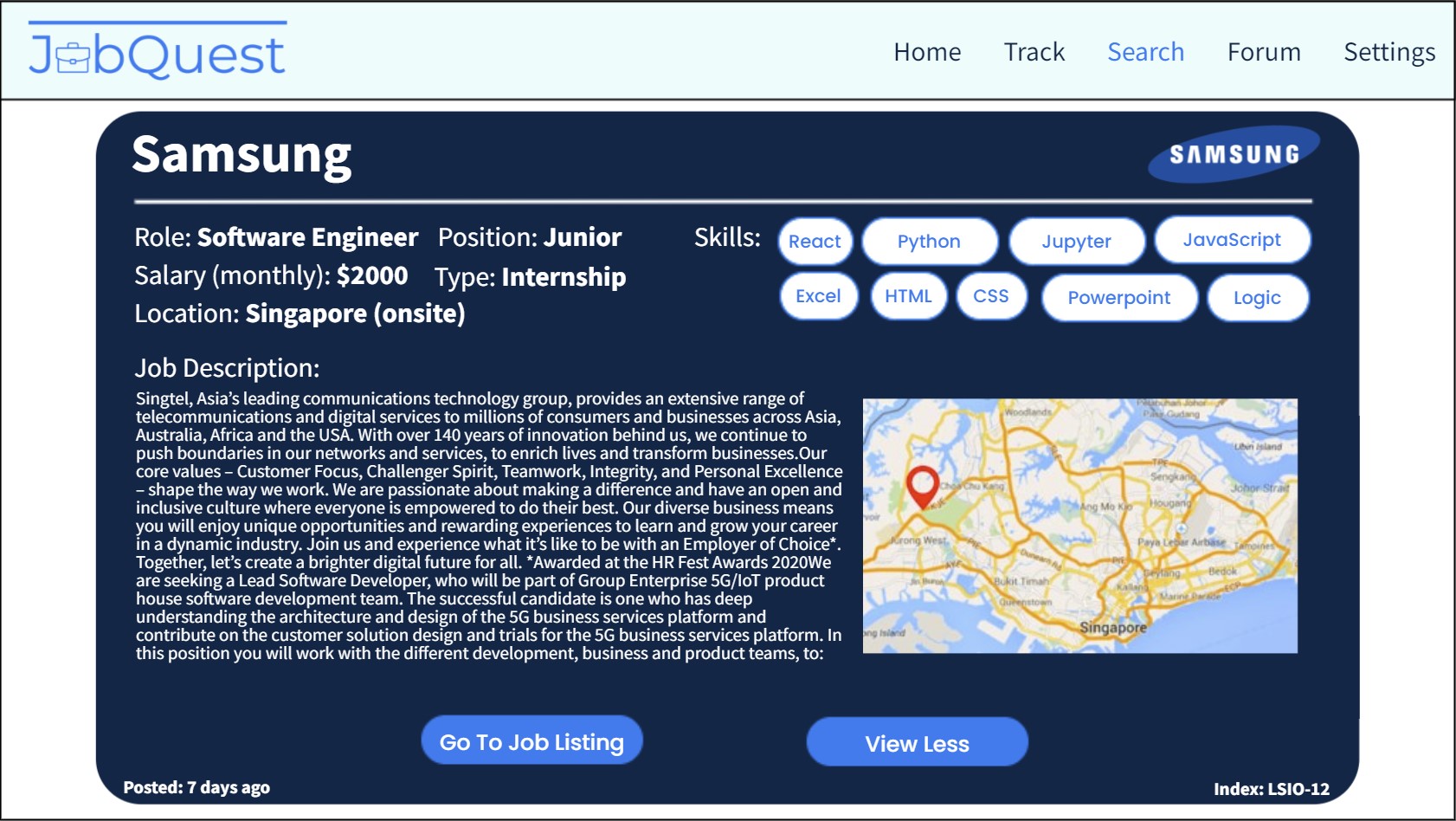


**Final UI**

## 

## 10.6. Searcher Expanded Card

**Initial Mockup**

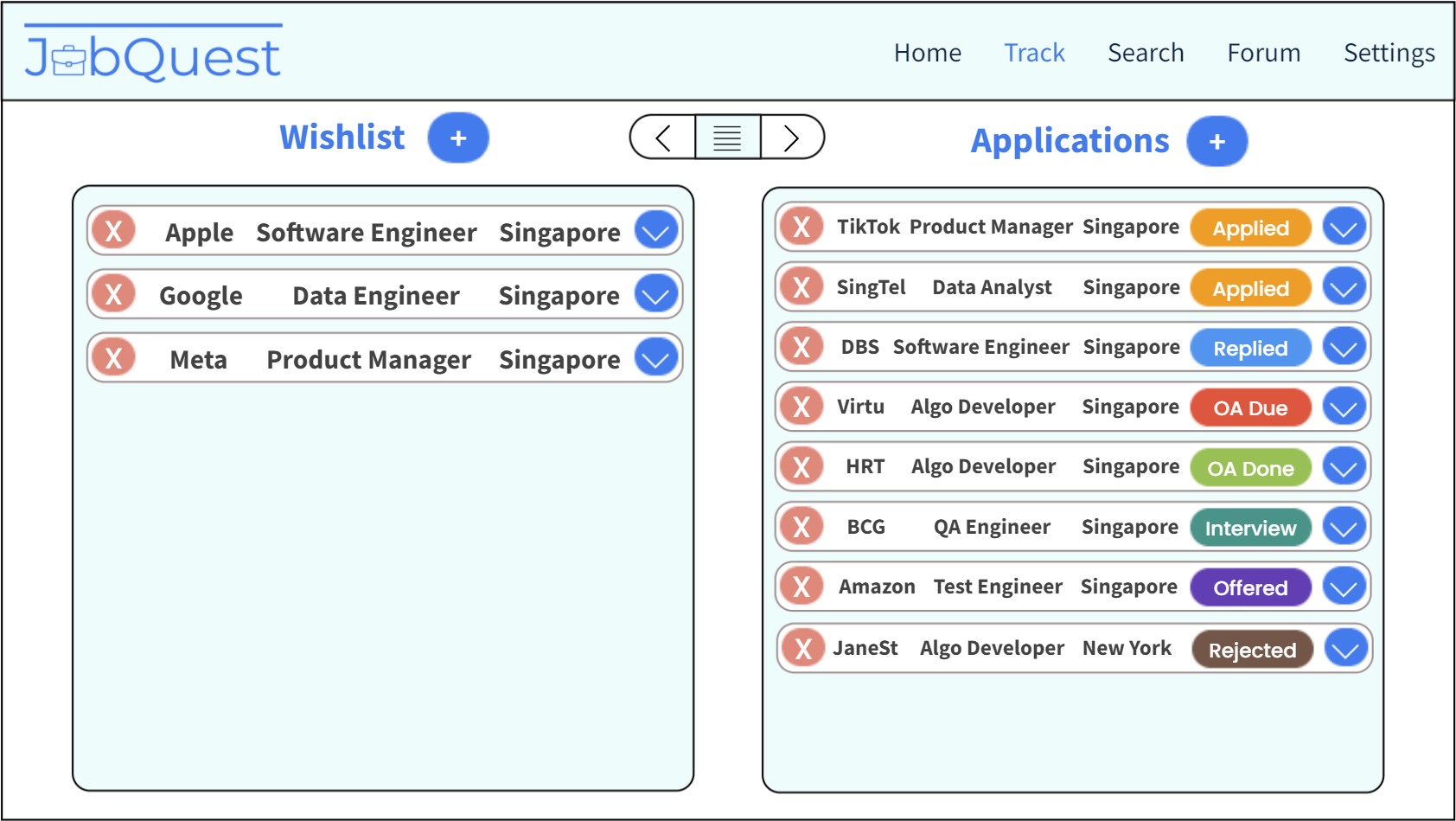


**Final UI**

## 

## 10.7. Tracker Page (Collapsed Entry)

**Initial Mockup**

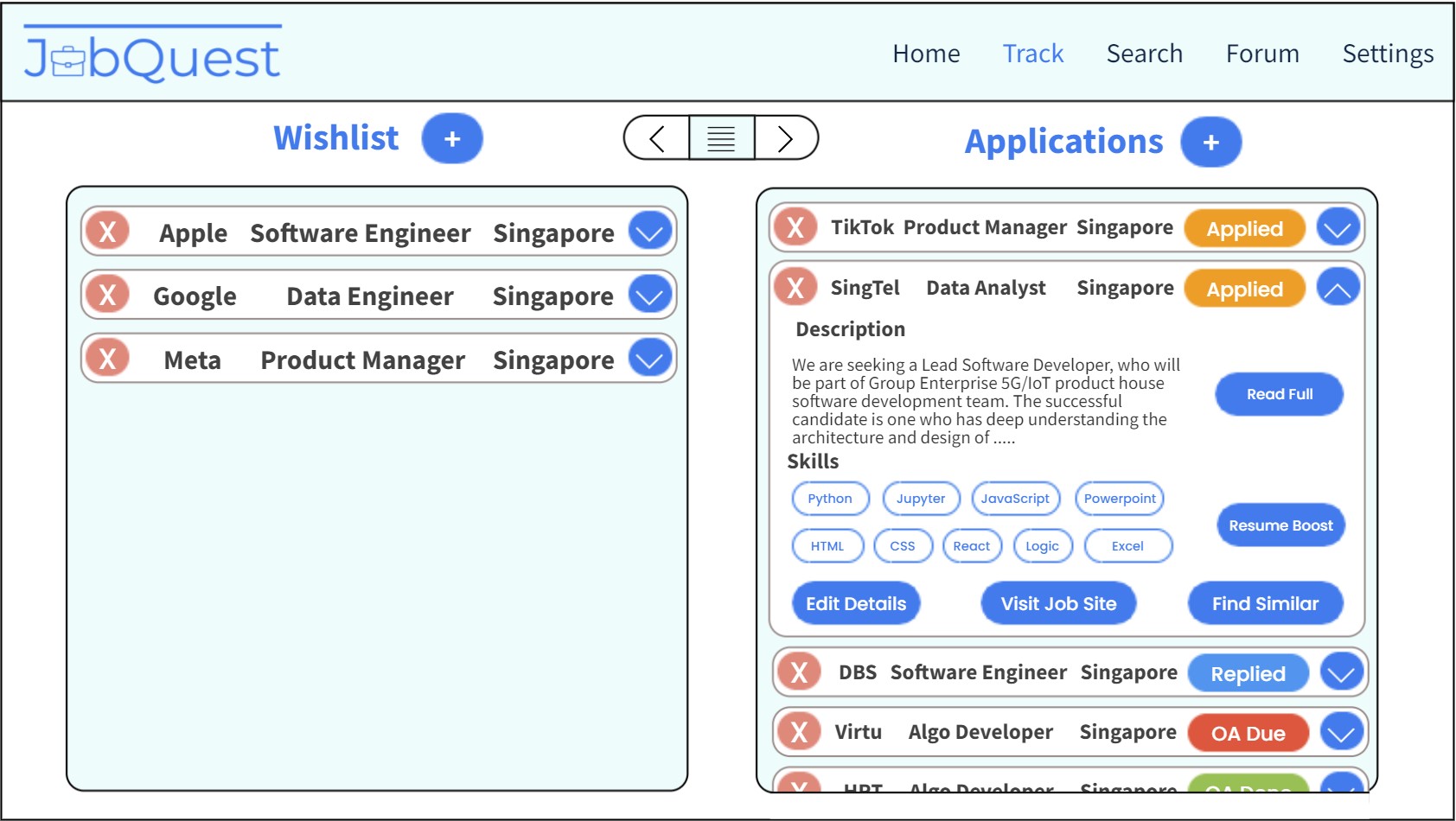


**Final UI**

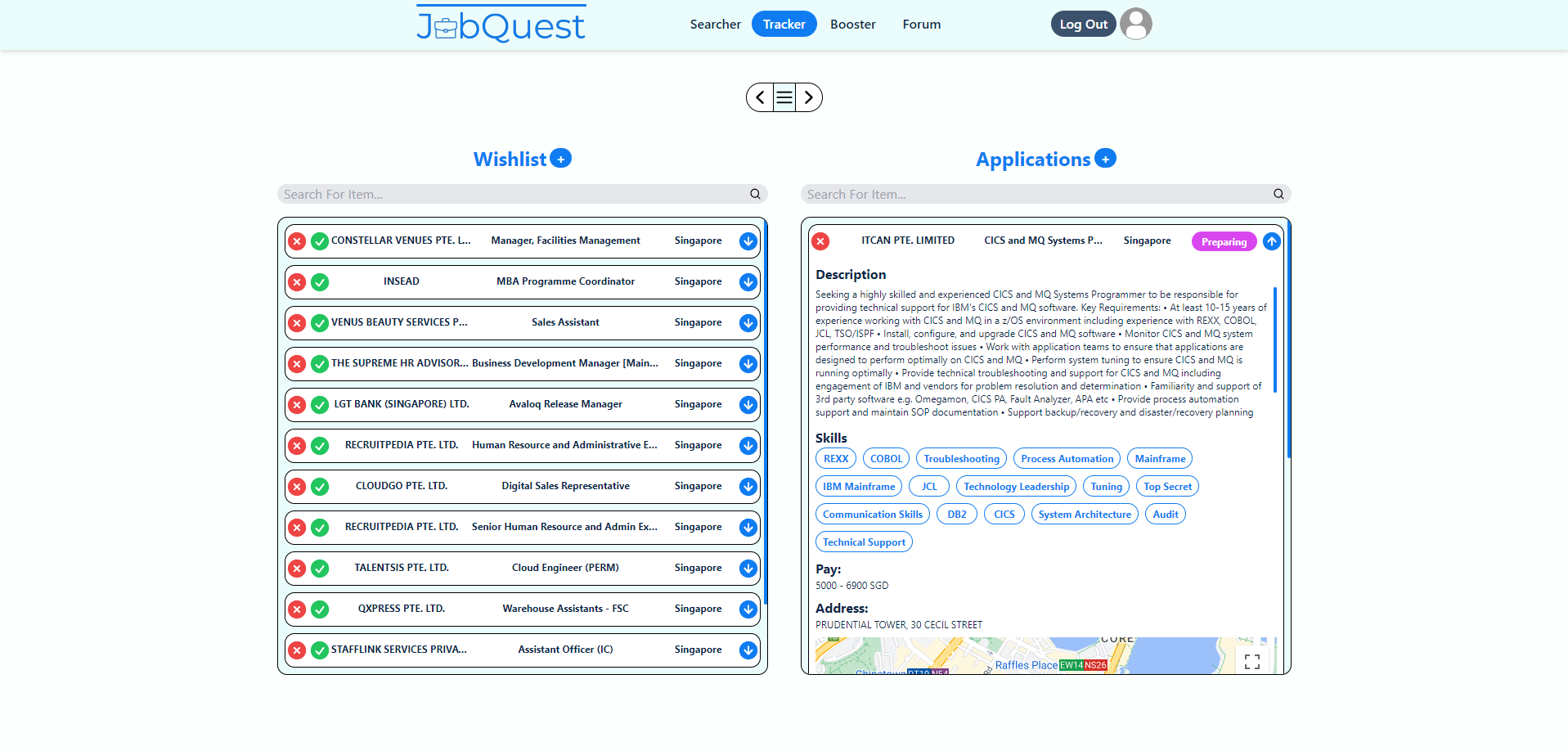
## 

## 10.8. Tracker Page (Expanded Entry)

**Initial Mockup**



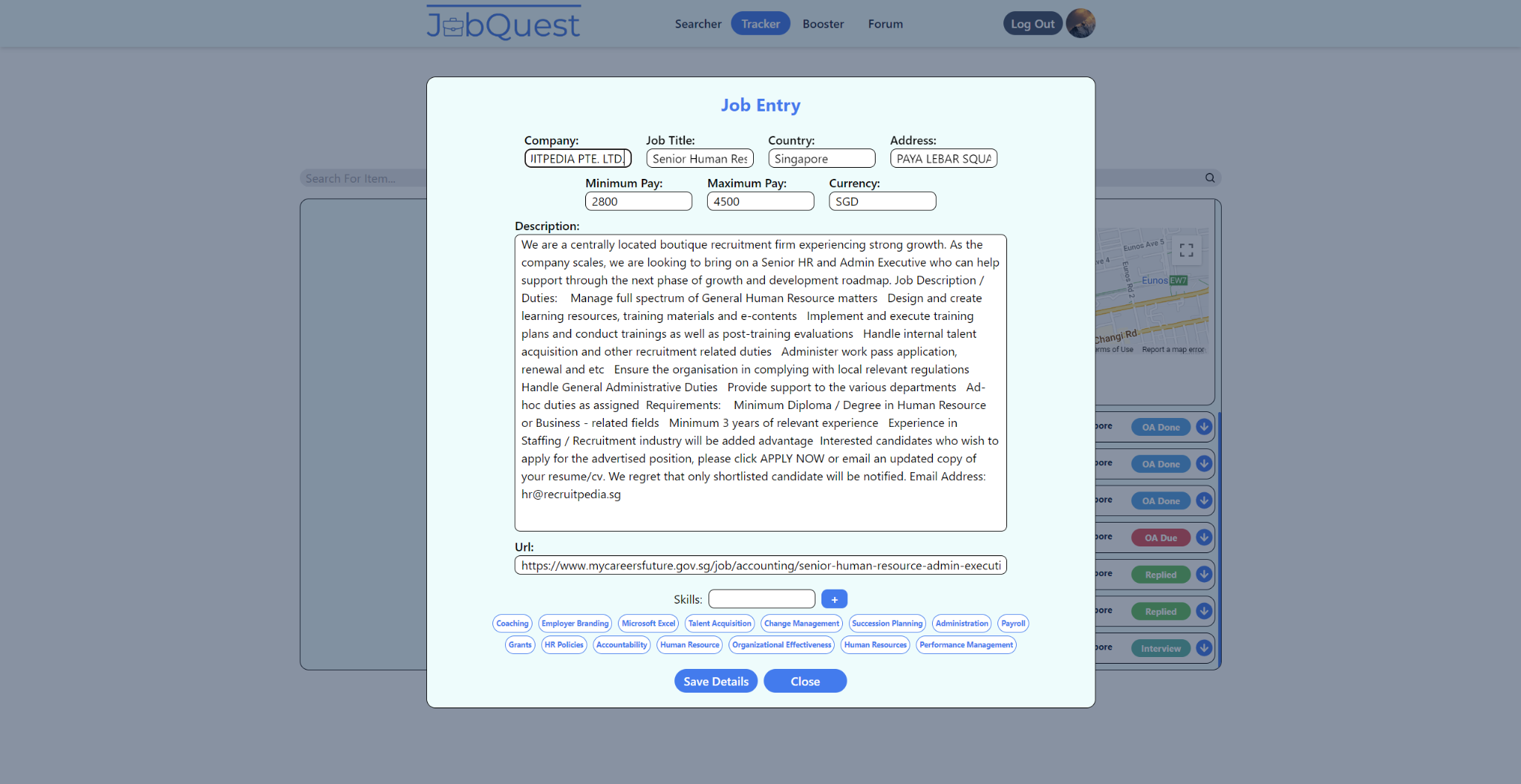
**Final UI**

****

## 10.9. Add Tracker Entry Page

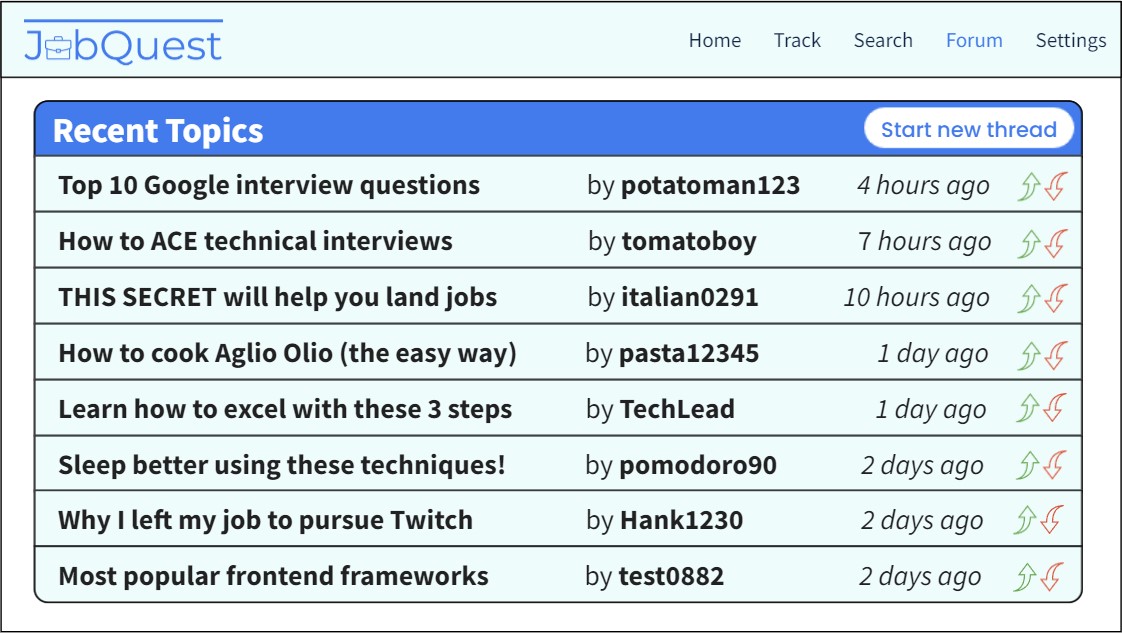
## **Initial Mockup**

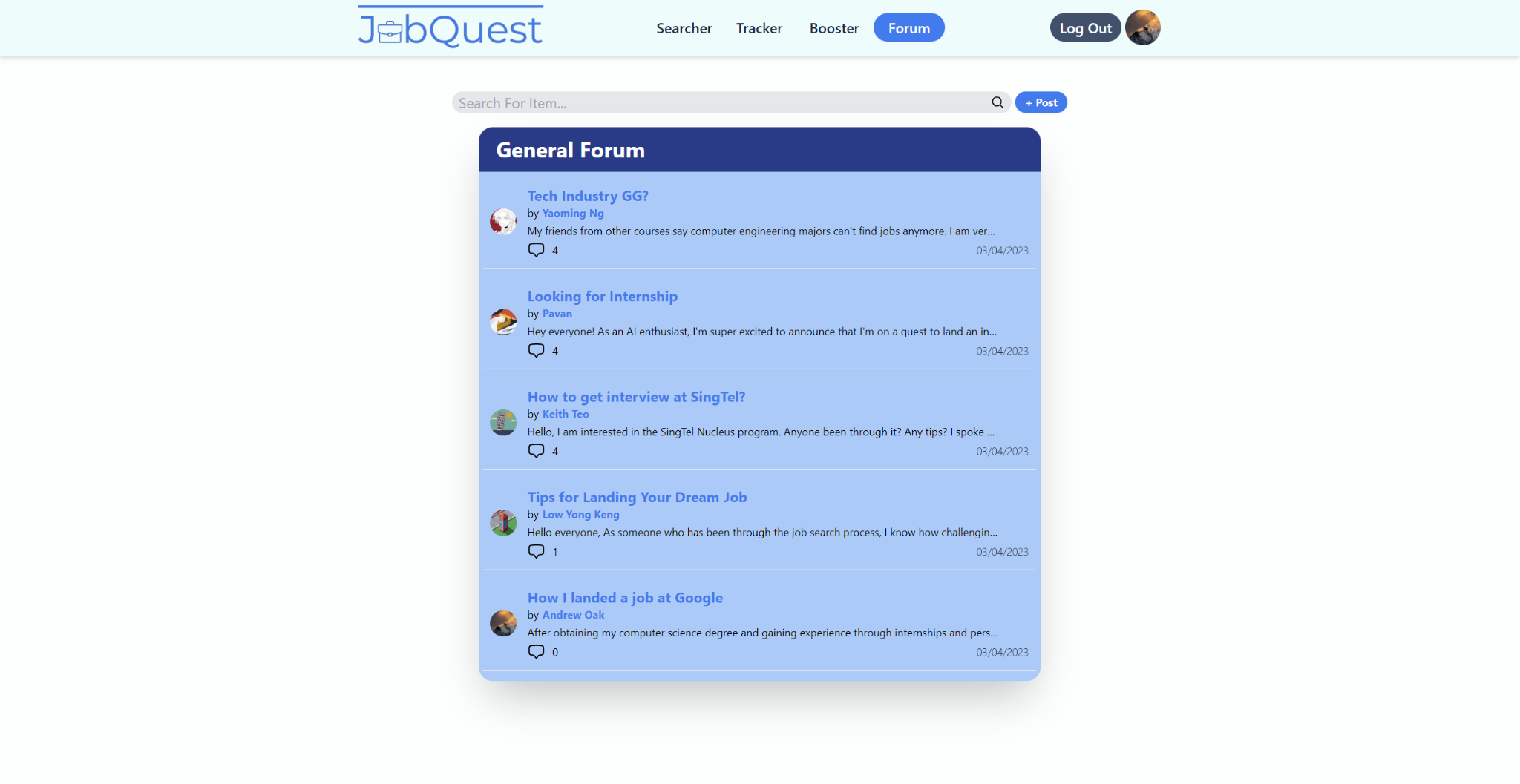
**Final UI**

****

## 10.10. Forum Page

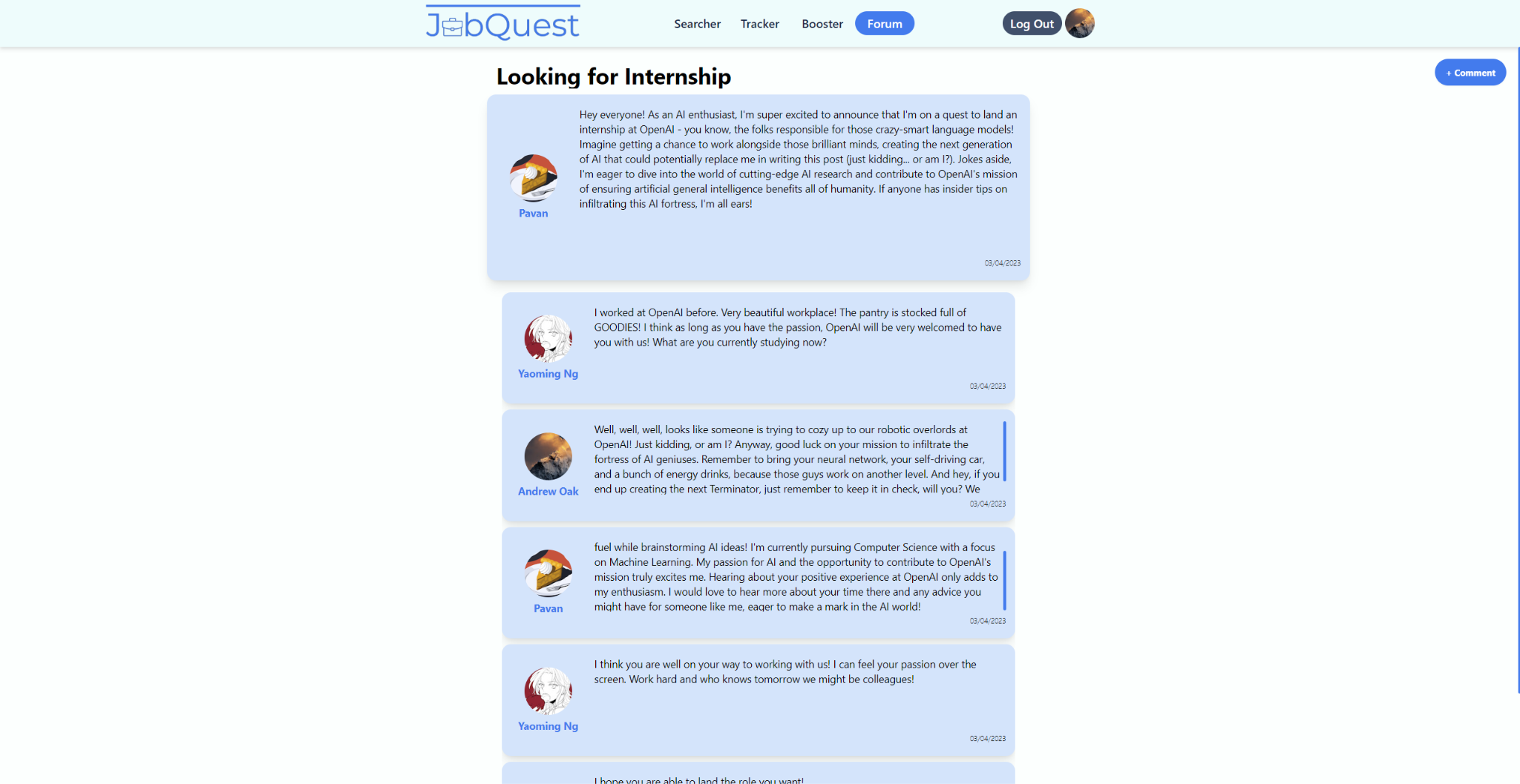
## **Initial Mockup**



**Final UI**

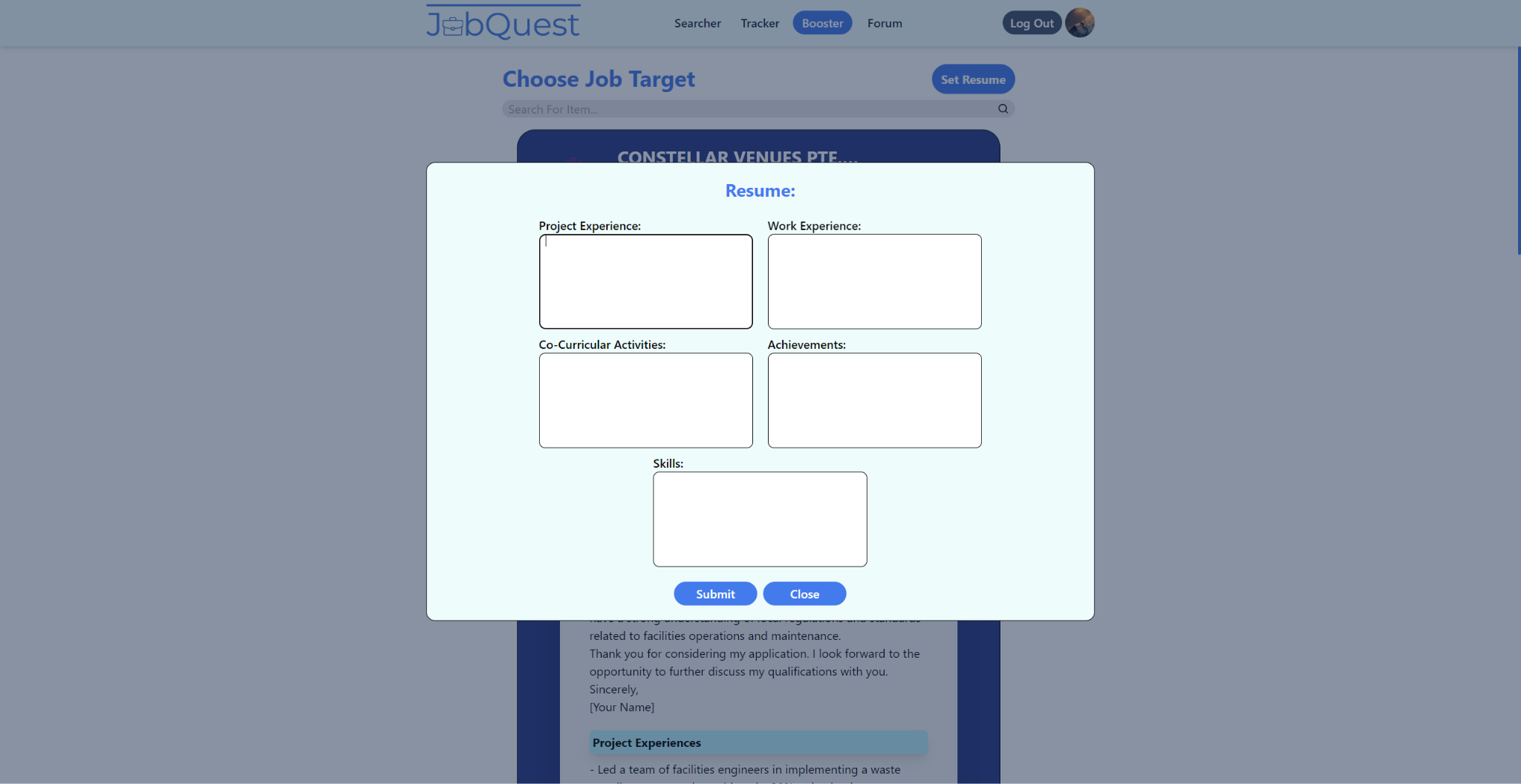
## 10.11. Individual Forum Thread Page

**Final UI**

****

## 10.12. Update Resume Details Page

**Final UI**

****

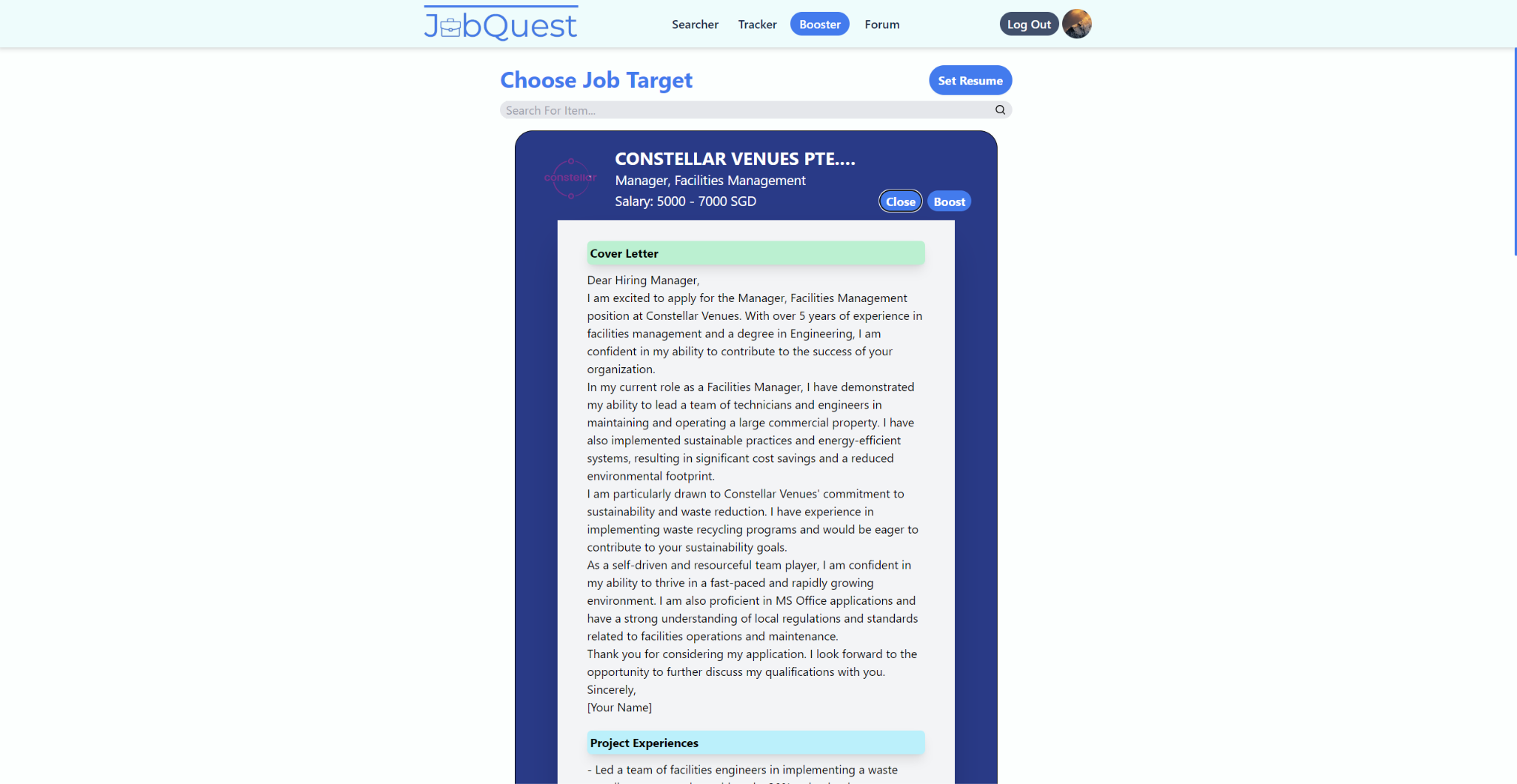
## 10.13. Resume Booster Page (Collapsed View)

**Final UI**

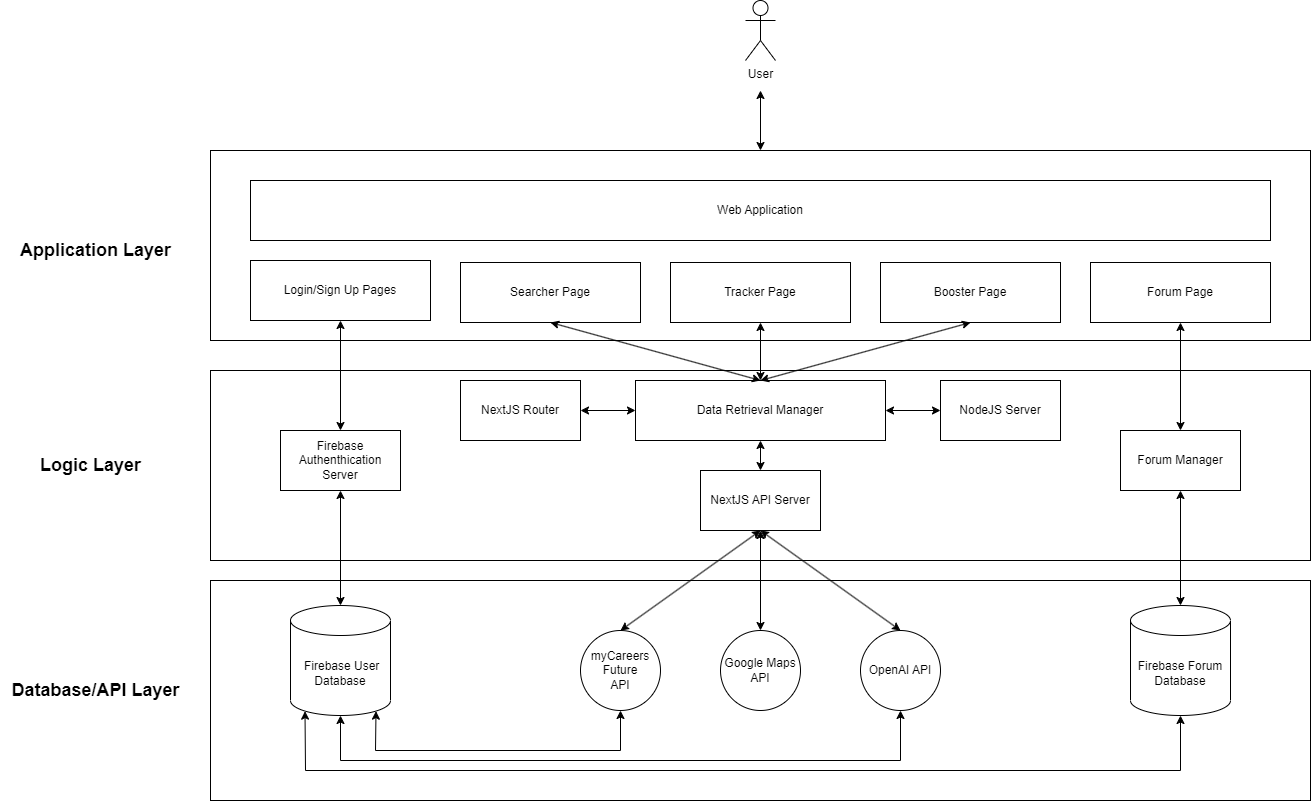
****

## 10.14. Resume Booster Page (Expanded View)

**Final UI**

****

# System Architecture



# Unit Testing

## 12.1 Black Box Testing

Black Box testing is used to test equivalences classes and boundary value testing.

Test Case 1: Creating Account

Checking if all the user details that a user inputs is valid before signing the user up and creating a new account in the database.

## 12.1.1 Black Box Testing (Sign Up)

| Email | Username | Password | Confirm Password | Expected Output | Manual Unit Test Results |
| --- | --- | --- | --- | --- | --- |
| Empty | Keith | Password1! | Password1! | Display “Please enter a valid email.” | Pass |
| [KTEO024@e.ntu.edu.sg](mailto:KTEO024@e.ntu.edu.sg) (Assuming email already in use) | Keith | Password1! | Password1! | Display “The current email is already in use. Please use another email or log in instead.” | Pass |
| [KTEO024@e.ntu.edu.sg](mailto:KTEO024@e.ntu.edu.sg) (Assuming email not yet used for sign up) | Empty | Password1! | Password1! | Display “Username cannot be empty.” | Pass |
| [KTEO024@e.ntu.edu.sg](mailto:KTEO024@e.ntu.edu.sg) (Assuming email not yet used for sign up) | Keith | Empty | Password1! | Display “Password should be minimum 8 characters, containing at least 1 letter, number and special character.” | Pass |
| [KTEO024@e.ntu.edu.sg](mailto:KTEO024@e.ntu.edu.sg) (Assuming email not yet used for sign up) | Keith | Password1! | Empty | Display “Two passwords do not match.” | Pass |
| [KTEO024@e.ntu.edu.sg](mailto:KTEO024@e.ntu.edu.sg) (Assuming email not yet used for sign up) | Keith | Password1! | Password2! | Display “Two passwords do not match.” | Pass |
| [KTEO024@e.ntu.edu.sg](mailto:KTEO024@e.ntu.edu.sg) (Assuming email not yet used for sign up) | Keith | Password1! | Password1! | User redirected to searcher page. | Pass |

## 

## 12.1.2 Black Box Testing (Searcher Set Up Page)

Test Case 2: Setting up Searcher SetUp Page Information

Checking if all the Searcher SetUp Page Information required fields are inputted , minSalary is in the range of 0 to 20000 and the skills added does not comprise only whitespaces before allowing the form to be submitted and the database to be updated.

| School | Major | Location | jobType | Citizenship | minSalary | Skills | Expected Output | Manual Unit Test Results |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Empty | Computer Engineering | West | Internship | Singaporean | 1000 | [“Java”, “C”] | Display “Please select an item in the list.” | Pass |
| National University of Singapore | Empty | East | Full-Time | PR | 2000 | [] | Display “Please select an item in the list.” | Pass |
| Nanyang Technological University | Computer Science | Empty | Part-Time | Others | 3000 | [“Public Speaking”] | Display “Please select an item in the list.” | Pass |
| Singapore Management University | Business | South | Empty | Singaporean | 4000 | [“C”, “C++”, “C#”] | Display “Please select an item in the list.” | Pass |
| Singapore Management University | Economics | North | Contract | Empty | 20000 | [“Teamwork”] | Display “Please select an item in the list.” | Pass |
| Singapore University of Technology and Design | Chinese | Central | Part-Time | Singaporean | Empty | [] | Display “Please fill out this field” | Pass |
| Singapore University of Technology and Design | Chinese | Central | Part-Time | Singaporean | -1 | [] | Display “Value must be greater than or equal to 0.” | Pass |
| Singapore University of Technology and Design | Chinese | Central | Part-Time | Singaporean | 20001 | [] | Display “Value must be less than or equal to 20000.” | Pass |
| Singapore University of Technology and Design | Chinese | Central | Part-Time | Singaporean | 0 | Empty | Display “Please fill out this field.” | Pass |
| Singapore University of Technology and Design | Chinese | Central | Part-Time | Singaporean | 6000 | [“”] | Does not allow skill to be added. | Pass |
| Singapore Management University | Chinese | Central | Part-Time | Singaporean | 7000 | [“ “] | Does not allow skill to be added. | Pass |
| Nanyang Technological University | Computer Engineering | West | Internship | Singaporean | 6969 | [“Java”, “C”] | Searcher Set Up Information saved to database. User redirected to Searcher Page | Pass |

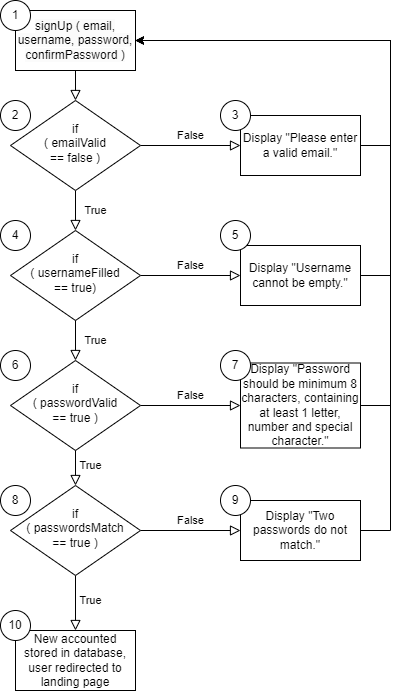
## 

## 

## 12.2 White Box Testing

White box testing was done to ensure that the flow of our application is correct as well as to check for alternative flows in our application.

## 12.2.1 Sign Up White Box Testing



Test Cases:

1. User creates an account with email not taken, username not taken, password meets requirements and both passwords match.

Execution path: 1, 2, 4, 6, 8, 10

1. User creates an account with email taken, username not taken, password meets requirements and both passwords match.

Execution path: 1, 2, 3

1. User creates an account with email not taken, username taken, password meets requirements and both passwords match.

Execution path: 1, 2, 4, 5

1. User creates an account with email taken, username not taken, password does not meet requirements and both passwords match.

Execution path: 1, 2, 4, 6, 7

1. User creates an account with email not taken, username not taken, password meets requirements and both passwords do not match.

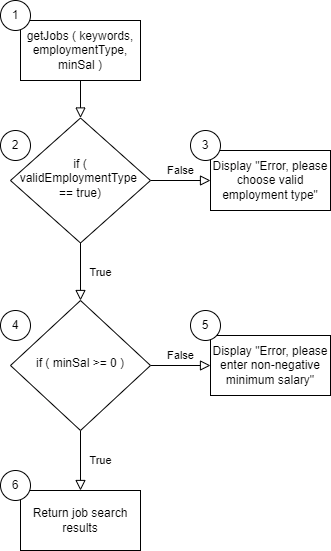
Execution path: 1, 2, 4, 6, 8, 9

Cyclomatic Complexity:

[Edges - Nodes] + 2 = [9 - 6] + 2 = 3

[Conditions] + 1 = 4 + 1 = 5

## 11.2.2 getJobs White Box Testing



Test Cases:

1. User sends job request with keywords, valid employment type and non-negative minimum salary.

Execution path: 1, 2, 4, 6

1. User sends job request with keywords, invalid employment type and non-negative minimum salary.

Execution path: 1, 2, 3

1. User sends job request with keywords, valid employment type and negative minimum salary.

Execution path: 1, 2, 4, 5

Cyclomatic Complexity:

[Edges - Nodes] + 2 = [5 - 4] + 2 = 3

[Conditions] + 1 = 2 + 1 = 3