**SC2006 Software Engineering**

****

**Lab#3 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**

[**2.0 Use Case Descriptions 4**](#_heading=h.3znysh7)

[2.1.1 Navigate to Tracker Page 4](#_heading=h.ht1hpwbkepbu)

[2.1.2 Navigate to Searcher Page 5](#_heading=h.18d44l6do6o5)

[2.1.3 Navigate to Booster Page 6](#_heading=h.hj1t5dxa3a5v)

[2.1.4 Navigate to Forum Page 7](#_heading=h.wsfravfipxwj)

[2.2.1 Create Account 8](#_heading=h.u9opbrr5wg6p)

[2.2.2 Login 10](#_heading=h.vl40c2o3byjj)

[2.2.3 Update Account Information 12](#_heading=h.sdc5a91bwhyd)

[2.2.4 Forget Password 13](#_heading=h.mhjl088rmhp3)

[2.3.1 Create Entry 14](#_heading=h.k6d0luuc3qs2)

[2.3.2 Delete Entry 15](#_heading=h.oq8wlblivt4x)

[2.3.3 View Entry 16](#_heading=h.hh1q362jni6g)

[2.3.4 Update Entry 17](#_heading=h.y6ee8vr7h2ye)

[2.4.1 Update Job Search Fields 18](#_heading=h.o1rphbl993er)

[2.4.2 View Jobs Available 19](#_heading=h.98kvxq3zert8)

[2.5.1 Make Forum Post 20](#_heading=h.wsypy65100t5)

[2.5.2 View Forum 21](#_heading=h.y6v588sde2in)

[2.6.1 Upload Resume 22](#_heading=h.yde2bxu6nhue)

[2.6.2 Boost Resume 23](#_heading=h.uzbxts1dljph)

[2.7.1 Access Account Database 24](#_heading=h.fpjsctywsmci)

[2.7.2 Access User Entry Database 25](#_heading=h.hmh8mr3wmad1)

[2.7.3 Suggest Resume Edit 26](#_heading=h.98gpbby3rhge)

[2.7.4 Fetch Job Data 27](#_heading=h.egfxeh4mzus5)

[2.7.5 Access Forum Posts 28](#_heading=h.rzjo0z6bsjps)

[**3.0 Entity Class Diagram 29**](#_heading=h.k34nbj45hk6a)

[**4.0 Key Boundary and Control Class Diagram 30**](#_heading=h.35nkun2)

[**5.0 Sequence Diagram of Some Use Cases 31**](#_heading=h.bvp7ff106zxv)

[5.1.1 Navigate to Tracker Page 31](#_heading=h.vz9nas8vs0p2)

[5.1.2 Navigate to Searcher Page 31](#_heading=h.2sange3txpt6)

[5.1.3 Navigate to Booster Page 32](#_heading=h.twfm1go42tok)

[5.1.4 Navigate to Forum Page 32](#_heading=h.qkkq4pwpm9gc)

[5.2.1 New User Sequence Diagram 33](#_heading=h.1quy1ggfei44)

[5.2.2 Login Sequence Diagram 34](#_heading=h.bll7z974gvxv)

[5.2.3 Update Account Sequence Diagram 34](#_heading=h.roggmj4zc2uq)

[5.2.4 Forgot Password Sequence Diagram 35](#_heading=h.og4d08jej0l9)

[5.3.1 Create Entry Sequence Diagram 36](#_heading=h.72c8lhib44ux)

[5.3.2 Delete Entry Sequence Diagram 37](#_heading=h.b6x5qkjv0c5)

[5.3.3 View Entry Sequence Diagram 38](#_heading=h.213emppdmdcu)

[5.3.4 Update Entry Sequence Diagram 39](#_heading=h.3mnm7i5vr50j)

[5.4.1 Update Job Search Fields Diagram 39](#_heading=h.ah9r7bj6wxu6)

[5.4.2 View Jobs Available Diagram 40](#_heading=h.mwusx45v7qn4)

[5.5.1 Make Forum Post Diagram 41](#_heading=h.qotz2bamep6r)

[5.5.2 View Forum Diagram 41](#_heading=h.phlydyx7wrbb)

[5.6.1 Upload Resume Sequence Diagram 42](#_heading=h.2ycuddusuwld)

[5.6.2 Boost Resume Diagram 42](#_heading=h.apep6yu5rmy7)

[**6.0 Initial Dialog Map 43**](#_heading=h.e0zj2tiq3tdx)

[**7.0 System Architecture 44**](#_heading=h.nzzdrn36pnx2)

# 2.0 Use Case Descriptions

## 2.1.1 Navigate to Tracker Page

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

| Actor: | **User** |
| --- | --- |
| Description: | **With the use of the router function, it allows the user to navigate from home page to dashboard page to searcher 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. **If isLoggedIn is true, user directed to Tracker page** |
| Alternative Flows: | **If isLoggedIn is false**   1. **User redirected to HomePage** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.1.2 Navigate to Searcher Page

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

| Actor: | **User** |
| --- | --- |
| Description: | **With the use of the router function, it allows the user to navigate from home page to dashboard page to 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. **If isLoggedIn is true, user directed to Searcher page** |
| Alternative Flows: | **isLoggedIn is false**   1. **User redirected to HomePage** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.1.3 Navigate to Booster Page

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

| Actor: | **User** |
| --- | --- |
| Description: | **With the use of the router function, it allows the user to navigate from home page to dashboard page to 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. **If isLoggedIn is true, user redirected to Booster page** |
| Alternative Flows: | **isLoggedIn is false**   1. **User redirected to HomePage** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.1.4 Navigate to Forum Page

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

| Actor: | **User** |
| --- | --- |
| Description: | **With the use of the router function, it allows the user to navigate from home page to dashboard page to searcher 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 directed to Forum page** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 

## 2.2.1 Create Account

| Use Case ID: | **ACC1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Create Account** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **20.03.2023** |

| Actor: | **New User** |
| --- | --- |
| Description: | **Create Account allows new users to create an account in our Account Database. To create an account, users must input Name, Username, Email, Password.** |
| Preconditions: | **(For email account creation)**   1. **Username must not be taken.** 2. **Passwords must meet the requirements:**     1. **At least 1 special character**    2. **At least 1 capital character**    3. **At least 8 characters.** 3. **Email must be unique.**   **(For Google OAuth Login)**   1. **Must have a Google account.** |
| Postconditions: | **(For email account creation)**   1. **System sends a confirmation link to the user's email account.** 2. **System redirects the user to the main page.**   **(For Google OAuth Login)**   1. **System now stores user data through Google API.** 2. **User can now Login using their Google account** |
| Priority: | **High** |
| Frequency of Use: | **Low** |
| Flow of Events: | **(Email)**   1. **User inputs required information: Name, Username, Email and Password.** 2. **System validates the information.** 3. **System creates the account.** 4. **System updates account information in database.** 5. **System directs the user to the main page.**   **(Google OAuth)**   1. **User clicks on Google Login.** 2. **Pop out window appears for user to select** |
| Alternative Flows: | **User account already exists: Email taken. (Email)**   1. **System displays “User account already exists”** 2. **System prompts users to use another email or log in.** 3. **If user selects log in, then system directs to log in page to continue from Login** 4. **Else, user has to use another email account.**   **Username already taken**   1. **System displays “Username already taken”** 2. **System prompts users to input a different username.** 3. **Continue from main flow step 1.**   **Passwords do not meet requirements.**   1. **System displays “Please input a password of at least 1 special character, mixed case and at least 8 characters long” message.** 2. **Continue from main flow step 1.** |
| Exceptions: |  |
| Includes: | **Access Account Database (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.2.2 Login

| Use Case ID: | **ACC2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Login** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **11.02.2023** |

| 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 a username and password.**  **User can also login using Google OAuth with their existing Google account.** |
| Preconditions: | **(Email)**   1. **Must have an existing account in the entry database.**   **(Google OAuth)**   1. **Must have a Google account.** |
| Postconditions: | **(Email)**   1. **System directs user to main page.**   **(Google OAuth)**  **(If user has not register an account)**   1. **System will create and store user account into account database and directs user to main page.**   **(If user has created an account using Google OAuth)**   1. **System directs user to main page.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | **(Email)**   1. **User enters username and password on login form.** 2. **User submits details.** 3. **System validates the user’s username and password.** 4. **System directs user to dashboard page.**   **(Google OAuth)**   1. **User clicks on login with Google.** 2. **Pop up window shows lists of google accounts to select.** 3. **User select his/her account to login.** 4. **System directs user to dashboard page.** |
| Alternative Flows: | **(Email)**  **Invalid Username**   1. **System cannot find the username in the database.** 2. **System displays an error message.** 3. **System prompts user to input username and password.** 4. **Use case resumes at main flow step 1.**   **Invalid Password**   1. **Password does not match for unique username.** 2. **System displays an error message.** 3. **System prompts user to input username and password.** 4. **Use case resumes at main flow step 1.**   **Missing Username/ Password**   1. **System displays an error message.** 2. **System prompts user to input username and password.** 3. **Use case resumes at main flow step 1.** |
| Exceptions: |  |
| Includes: | **Access Account Database (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.2.3 Update Account Information

| Use Case ID: | **ACC3** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Update Account Information** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **09.02.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Update Account Information allows users to edit account information such as username, name, profile picture, password.** |
| Preconditions: | 1. **User must login first.** |
| Postconditions: | 1. **User will have updated their account information.** |
| Priority: | **Moderate** |
| Frequency of Use: | **Low** |
| Flow of Events: | 1. **User selects “Settings” button on the Dashboard Page.** 2. **User selects the field to update.** 3. **User inputs updated details.** 4. **System updates information in the database.** |
| Alternative Flows: | **User selects to change passwords.**   1. **If the user selects the password field to update, the user must input the old password and new password.**   **User enters new password same as old password**   1. **Error message shows “New password cannot be the same as old password”** |
| Exceptions: |  |
| Includes: | **Access Account Database (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.2.4 Forget Password

| Use Case ID: | **ACC4** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Forget Password** | | |
| Created By: | **Low Yong Keng** | Last Updated By: | **Low Yong Keng** |
| Date Created: | **12.02.2023** | Date Last Updated: | **12.02.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **User is allowed to reset their password if they forgot their password.** |
| Preconditions: | 1. **User must have an existing account binded to their email.** 2. **User must enter their email account.** 3. **A link will be sent to their email account for user to change password.** |
| Postconditions: | 1. **User account password has been changed to a new password.** |
| Priority: | **High** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on “Forgot Password”** 2. **resetPassword(UserID)** 3. **PasswordResetCtr calls getEmail(userID) from Account Database** 4. **Account Database returns user email** 5. **PasswordResetCtr sends reset code to user email** 6. **Asks user for reset code** 7. **codeVerify(code) runs to verify code** 8. **If codeVerified is true, ask user for new password** 9. **newPassword(password) submits to PasswordResetCtr** 10. **isValid(password) runs to verify if password is valid** 11. **if isValid is true, updatePassword(password) to update password in Account Database** 12. **Account Database sends password update successful** 13. **Display password successfully updated** |
| Alternative Flows: | **codeVerified is false**   1. **Wait for correct code**   **isValid(password) is false**   1. **Display message “Password invalid”** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.3.1 Create Entry

| Use Case ID: | **TRA1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Create Entry** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **12.02.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **In order for users to look for desired jobs, they have to first create an entry indicating their fields of interests so that the system can run its algorithm and show the user the lists of jobs available for them.** |
| Preconditions: | 1. **Users must be logged in.** |
| Postconditions: | 1. **Entry is created.** |
| Priority: | **High** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on “Create Entry”** 2. **Entry details sent to the application** 3. **createEntry(details)** 4. **Returns entry created message** 5. **Displays “Entry Created”** |
| Alternative Flows: | **Fields not fully filled up.**   1. **Display “Please fill in all fields”** 2. **User details submitted again** |
| Exceptions: | **User cancels entry creation, entry is not created.** |
| Includes: | **Access User Entry Database (API2)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.3.2 Delete Entry

| Use Case ID: | **TRA2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Delete Entry** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **27.01.2023** | Date Last Updated: | **27.01.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **Delete Entry allows users to remove an existing application that the user created previously.** |
| Preconditions: | 1. **Entry must exist before allowing users to delete** |
| Postconditions: | 1. **Entry is deleted from the account database.** |
| Priority: | **Moderate** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User selects Delete Entry.** 2. **Entry ID sent to application** 3. **deleteEntry(entryID)** 4. **Return entry deleted message** 5. **Displays “Entry Deleted”** |
| Alternative Flows: | **Entry does not exist.**   1. **Display “Please select valid entry to delete”** 2. **User submits another entry to delete** |
| Exceptions: |  |
| Includes: | **Access User Entry Database (API2)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.3.3 View Entry

| Use Case ID: | **TRA3** | | |
| --- | --- | --- | --- |
| Use Case Name: | **View 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 view all the entries that the user has created.** |
| Preconditions: | 1. **User must have created entries.** 2. **User must be logged in.** |
| Postconditions: | 1. **System displays all user entries.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User clicks on “View Entry”** 2. **userID sent to application** 3. **viewEntry(userID)** 4. **Return list of entries** 5. **Displays entries** |
| Alternative Flows: | **The user does not have any entry which will display an empty list.**   1. **Display “No entries exist”** |
| Exceptions: | **No entries created by the user yet.** |
| Includes: | **Access User Entry Database (API2)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.3.4 Update Entry

| Use Case ID: | **TRA4** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Update 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 “Update Entry”.** 2. **entryID sent to application.** 3. **updateEntry(entryID).** 4. **Return update successful message.** 5. **Displays “Update Successful”.** |
| Alternative Flows: | **entryID does not exist**   1. **Display “Entry does not exist”** 2. **User submits another entryID to update** |
| Exceptions: |  |
| Includes: | **Access User Entry Database (API2)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.4.1 Update Job Search Fields

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

| Actor: | **User** |
| --- | --- |
| Description: | **Allows users to update their survey results used to search for jobs.** |
| Preconditions: | 1. **User must be logged in to use personalised search results** |
| Postconditions: | 1. **Job search fields updated.** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **User goes to SearcherSetupPage** 2. **setupSearcher(surveyResults) is called** 3. **updateSurvey(survey) called to Account Database** 4. **Account Database replies update successful** 5. **Displays that update was successful** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: | **Access Account Database (API1)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.4.2 View Jobs Available

| Use Case ID: | **SEA2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **View Jobs Available** | | |
| Created By: |  | Last Updated By: |  |
| Date Created: | **12.02.2023** | Date Last Updated: | **12.02.2023** |

| Actor: | **User** |
| --- | --- |
| Description: | **For user to see all the jobs available to the user based on the Job Search Fields filled by the user.** |
| Preconditions: | 1. **User must 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 “Searcher” button** 2. **searchJobs(userID) called** 3. **ifExistsSearcherSetup(userID)** 4. **If Account Database returns exists, surveyExists is true and process continues** 5. **retrieveSearcherProfile(userID) called to Account Database** 6. **Account Database returns searcher profile** 7. **While browsing is true, getJob(userProfile) from Jobs API (API4)** 8. **Return Job from API** 9. **Job displayed to user** 10. **If swipe is right, addJob(job, userID)** 11. **Job added to user tracker** |
| Alternative Flows: | **If suveryExists is false**   1. **Redirect to SearchSetupPage**   **if Swipe is left**   1. **Shows next job** |
| Exceptions: |  |
| Includes: | **Fetch Job Data (API4)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.5.1 Make Forum Post

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

| 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 posted is created and corresponding entry inserted into the Forum Database.** |
| Priority: | **Moderate** |
| Frequency of Use: | **Moderate** |
| Flow of Events: | 1. **User clicks on “Start New Thread”** 2. **Enters thread details and presses “MakeNewThread”** 3. **MakeNewThread(thread)** 4. **isThreadValid(thread)** 5. **If isThreadValid is false, addNewThread(thread)** 6. **Thread added returned** 7. **User redirected to newly created thread** |
| Alternative Flows: | **isThreadValid is false**   1. **Display Message “Thread post invalid”** 2. **User submits new thread** |
| Exceptions: |  |
| Includes: | **Access Forum Posts (API5)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.5.2 View Forum

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

| 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. **getThreads() called** 3. **Get list of threads from Forum Database** 4. **Forum Database returns list of threads** 5. **Displays list of threads** |
| Alternative Flows: |  |
| Exceptions: |  |
| Includes: | **Access Forum Posts (API5)** |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.6.1 Upload Resume

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

| Actor: | **User** |
| --- | --- |
| Description: | **User uploads resume for the System to add in more details suited for the job selected by the user.** |
| Preconditions: | 1. **User 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. **uploadResume(resume) called** 3. **If isEmpty is false, createResume(resume) called to Account Database** 4. **Account Database returns confirmation resume created** 5. **Displays “Resume uploaded”** |
| Alternative Flows: | **If isEmpty is true**   1. **Display message “Resume empty”** 2. **uploadResume(resume) called when user resubmits resume** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 

## 

## 2.6.2 Boost Resume

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

| 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. **At least one resume must have been uploaded.** 2. **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 “Resume Boost” button** 2. **Chooses or enters resume to boost** 3. **boostResume(resume)** 4. **JobBoosterCtr sends resume to OpenAI API** 5. **OpenAI returns boosted resume** 6. **Displays boosted resume, asks to save or not** 7. **If save, updateResume(boostedResume)** 8. **Resume updated to boosted resume** 9. **Database returns update successful** 10. **Displays update successful** |
| Alternative Flows: | **If user chooses not to save**   1. **Remains at ResumeBoostPage** |
| Exceptions: |  |
| Includes: |  |
| Extends: | **Suggest Resume Edit (API3)** |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

## 2.7.1 Access Account Database

| Use Case ID: | **API1** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Access Account Database** | | |
| Created By: | **Keith** | Last Updated By: | **Keith** |
| Date Created: | **27.01.2023** | Date Last Updated: | **27.01.2023** |

| Actor: | **Account 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 account database for account information** 2. **Account 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: |  |

## 2.7.2 Access User Entry Database

| Use Case ID: | **API2** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Access User Entry Database** | | |
| Created By: | **Keith** | Last Updated By: | **Keith** |
| Date Created: | **27.01.2023** | Date Last Updated: | **27.01.2023** |

| Actor: | **User Entry Database** |
| --- | --- |
| Description: | **API that allows access to user entries for job applications currently tracked.** |
| Preconditions: | 1. **User entry data must exist** |
| Postconditions: | 1. **Frontend receives user entry information** |
| Priority: | **High** |
| Frequency of Use: | **High** |
| Flow of Events: | 1. **Frontend sends JSON requests to the account database for account entries information.** 2. **Account 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: |  |

## 2.7.3 Suggest Resume Edit

| Use Case ID: | **API3** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Suggest Resume Edit** | | |
| Created By: | **Keith** | Last Updated By: | **Keith** |
| Date Created: | **27.01.2023** | Date Last Updated: | **27.01.2023** |

| 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: |  |
| Notes and Issues: |  |

## 2.7.4 Fetch Job Data

| Use Case ID: | **API4** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Fetch Job Data** | | |
| Created By: | **Keith** | Last Updated By: | **Keith** |
| Date Created: | **27.01.2023** | Date Last Updated: | **27.01.2023** |

| Actor: | **Google Jobs API** |
| --- | --- |
| Description: | **Fetches a list of jobs from API based on user job search query and/or survey.** |
| Preconditions: | 1. **To support personalised search survey 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 Google 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: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

# 

## 

## 2.7.5 Access Forum Posts

| Use Case ID: | **API5** | | |
| --- | --- | --- | --- |
| Use Case Name: | **Access Forum Posts** | | |
| Created By: | **Yong Keng** | Last Updated By: | **Yong Keng** |
| Date Created: | **12.02.2023** | Date Last Updated: | **20.03.2023** |

| Actor: | **Forum Database** |
| --- | --- |
| Description: | **Create, read, update, delete (CRUD) operations for the post replies in the forum database.** |
| Preconditions: | 1. **User must be logged in** |
| Postconditions: | 1. **Corresponding CRUD 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 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 entry.** 2. **Forum database returns the post entry.**   **Update operation:**   1. **Application sends update operation with ID of post entry to update and new data.** 2. **Entry is updated.** 3. **Forum database responds with message confirming successful update.**   **Delete operation:**   1. **Application sends delete operation and ID of entry to delete.** 2. **Entry is deleted from the database.** 3. **Forum database sends message confirming successful delete.** |
| Exceptions: |  |
| Includes: |  |
| Extends: |  |
| Special Requirements: |  |
| Assumptions: |  |
| Notes and Issues: |  |

# 3.0 Entity Class Diagram

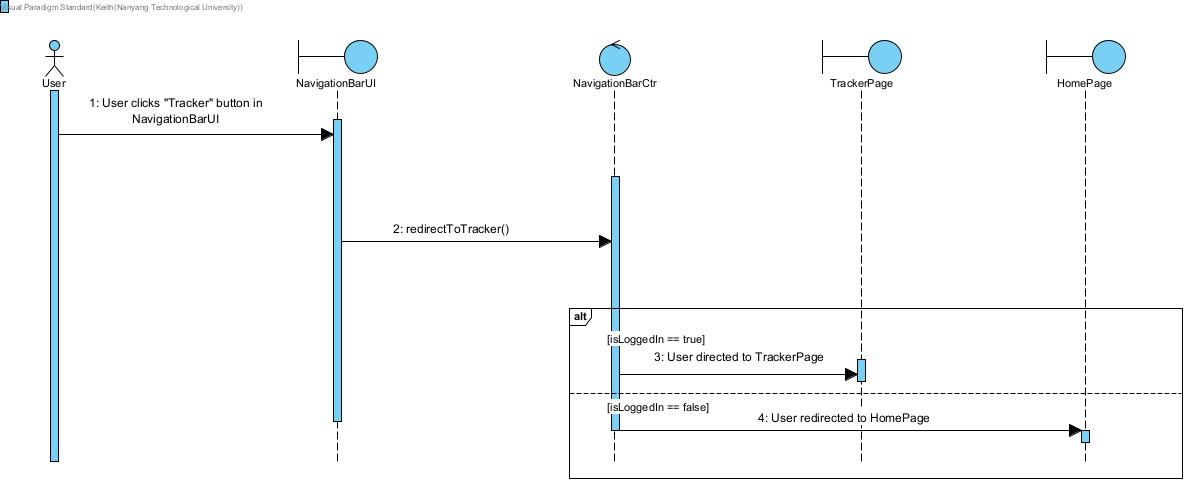
## 

## **4.0 Key Boundary and Control Class Diagram**

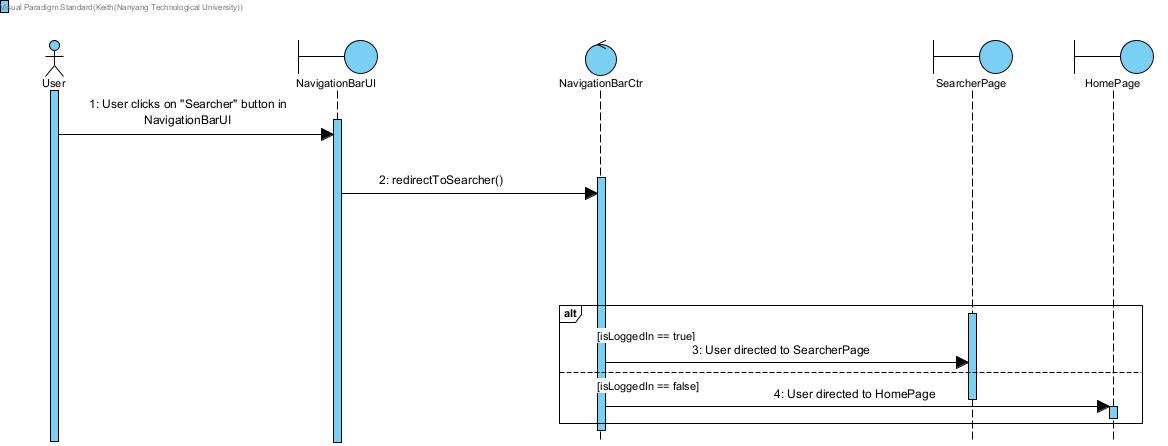
# 

# 5.0 Sequence Diagram of Some Use Cases

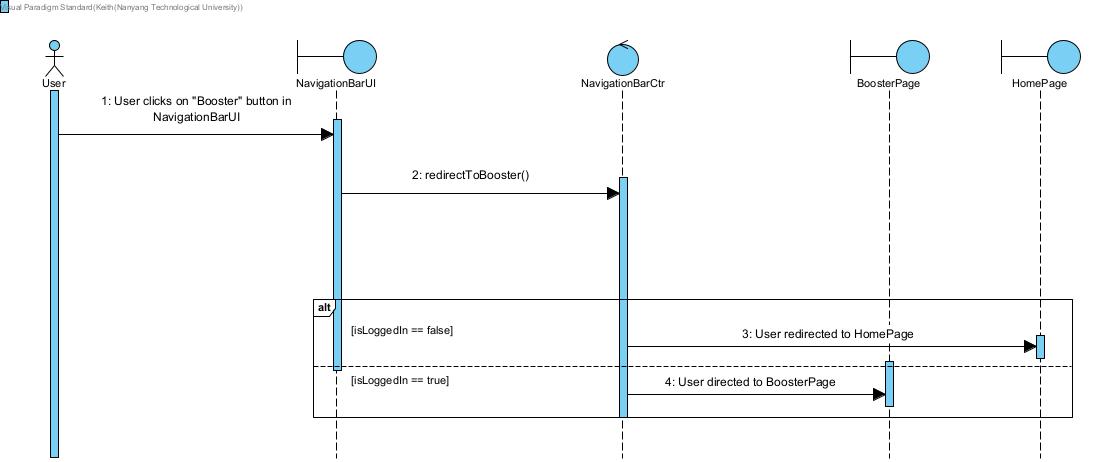
## 5.1.1 Navigate to Tracker Page



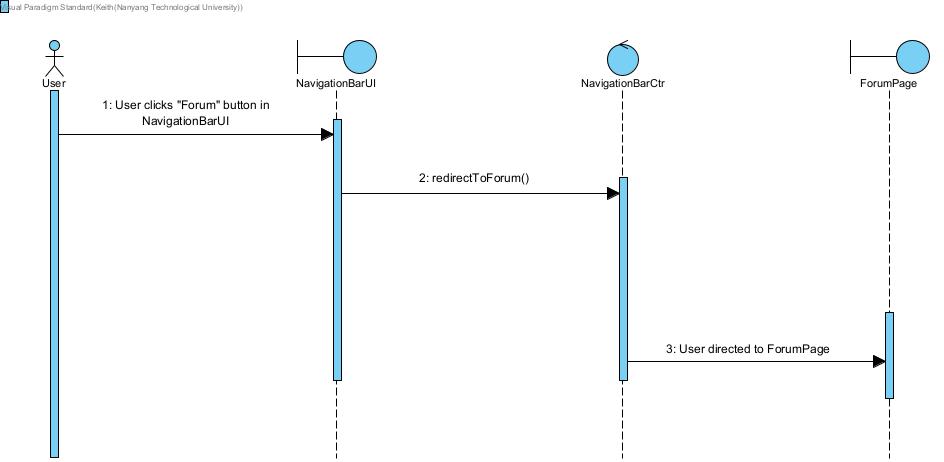
## 5.1.2 Navigate to Searcher Page



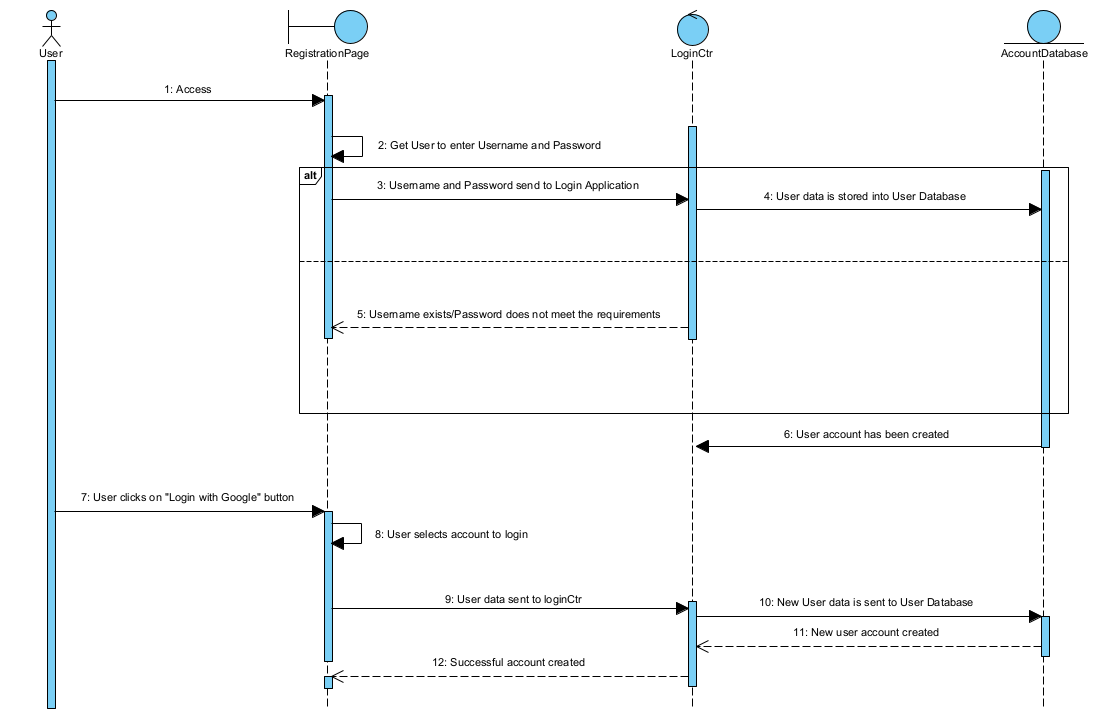
## 5.1.3 Navigate to Booster Page



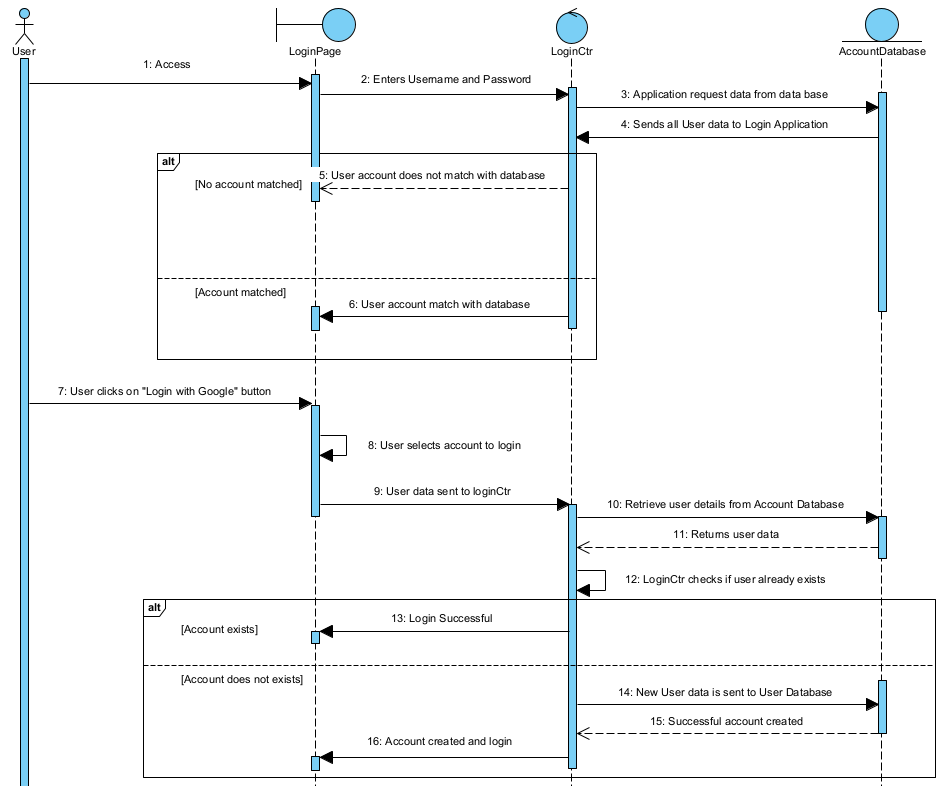
## 5.1.4 Navigate to Forum Page



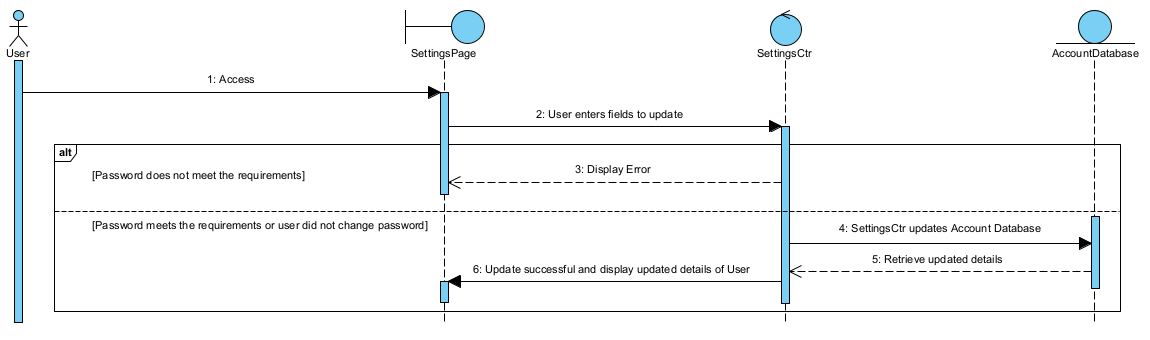
## 5.2.1 New User Sequence Diagram



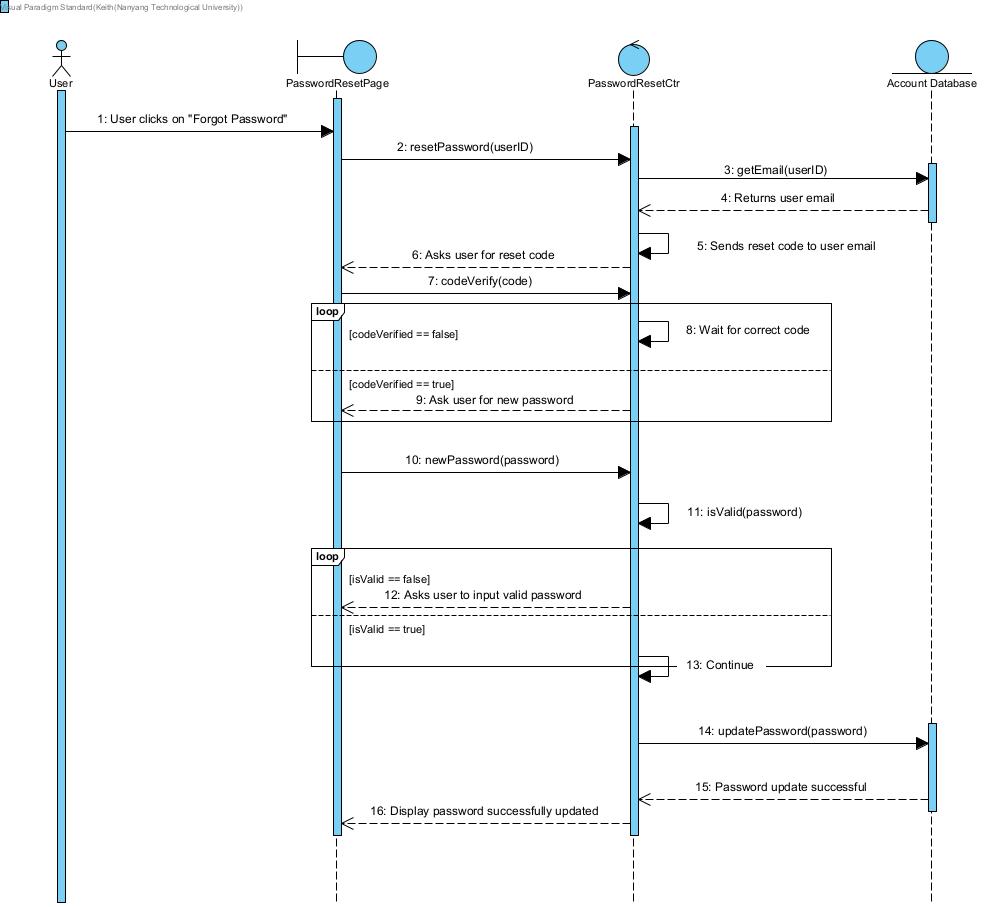
## 5.2.2 Login Sequence Diagram



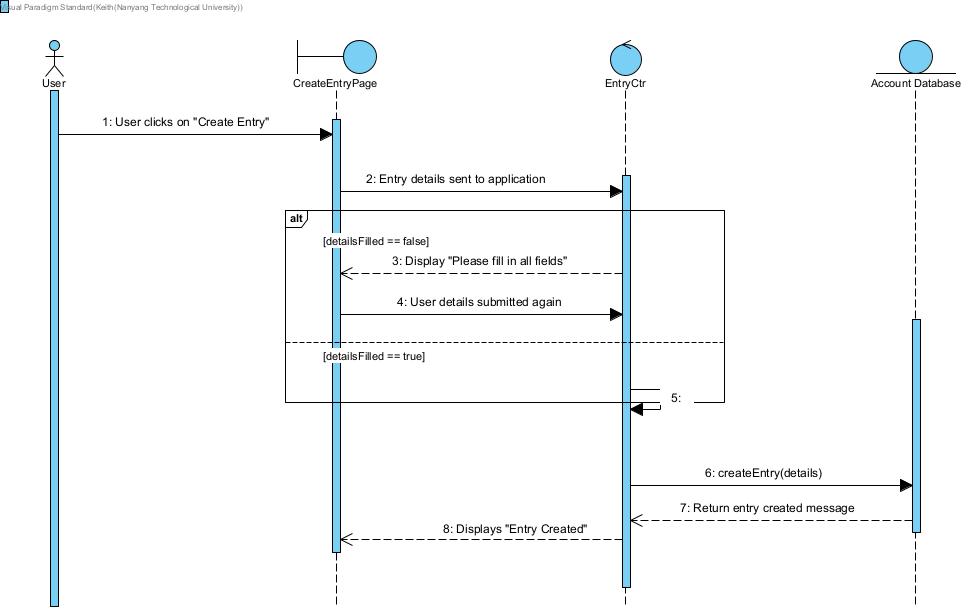
## 5.2.3 Update Account Sequence Diagram



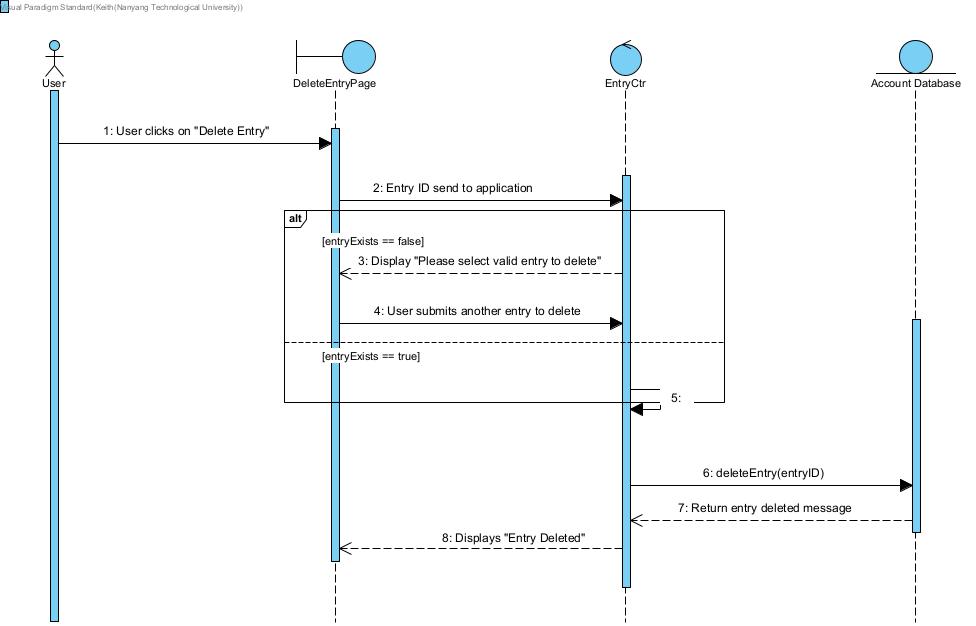
## 5.2.4 Forgot Password Sequence Diagram



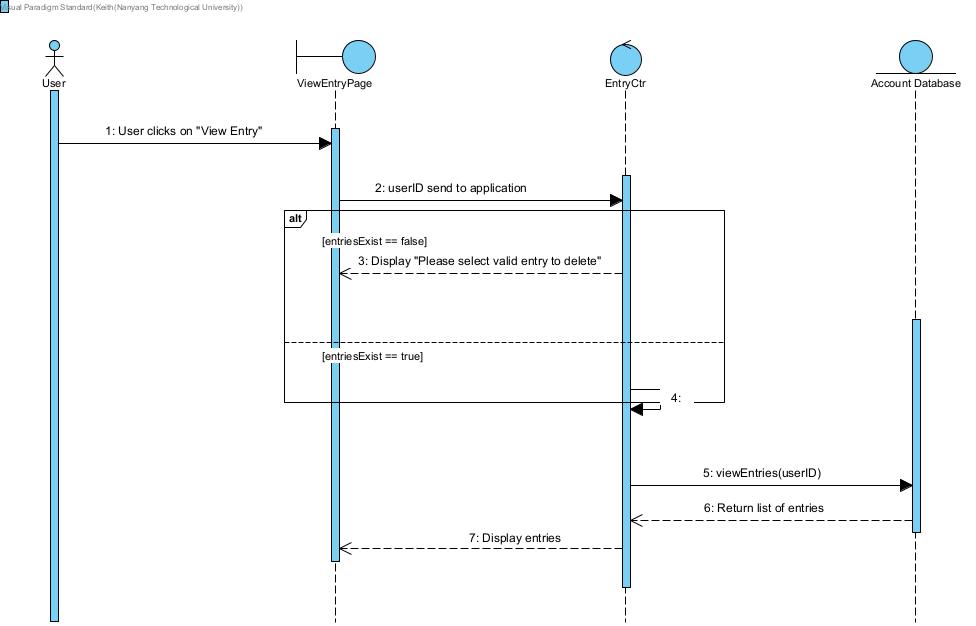
## 5.3.1 Create Entry Sequence Diagram



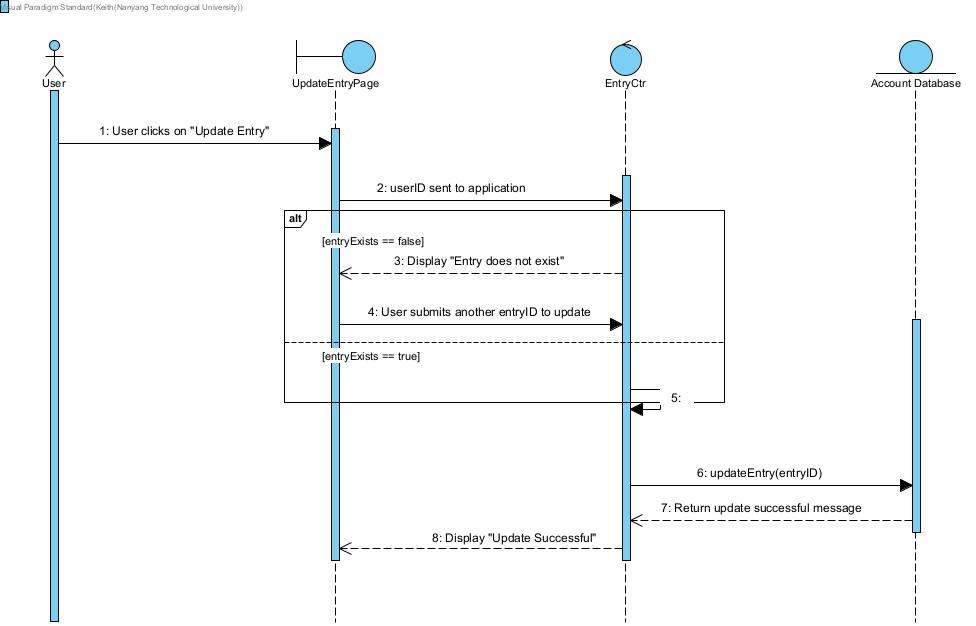
## 5.3.2 Delete Entry Sequence Diagram



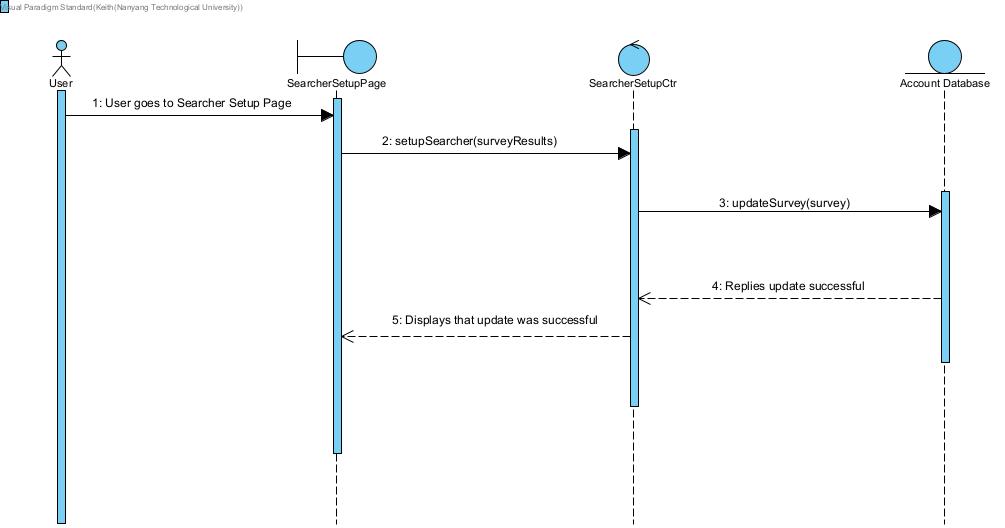
## 5.3.3 View Entry Sequence Diagram



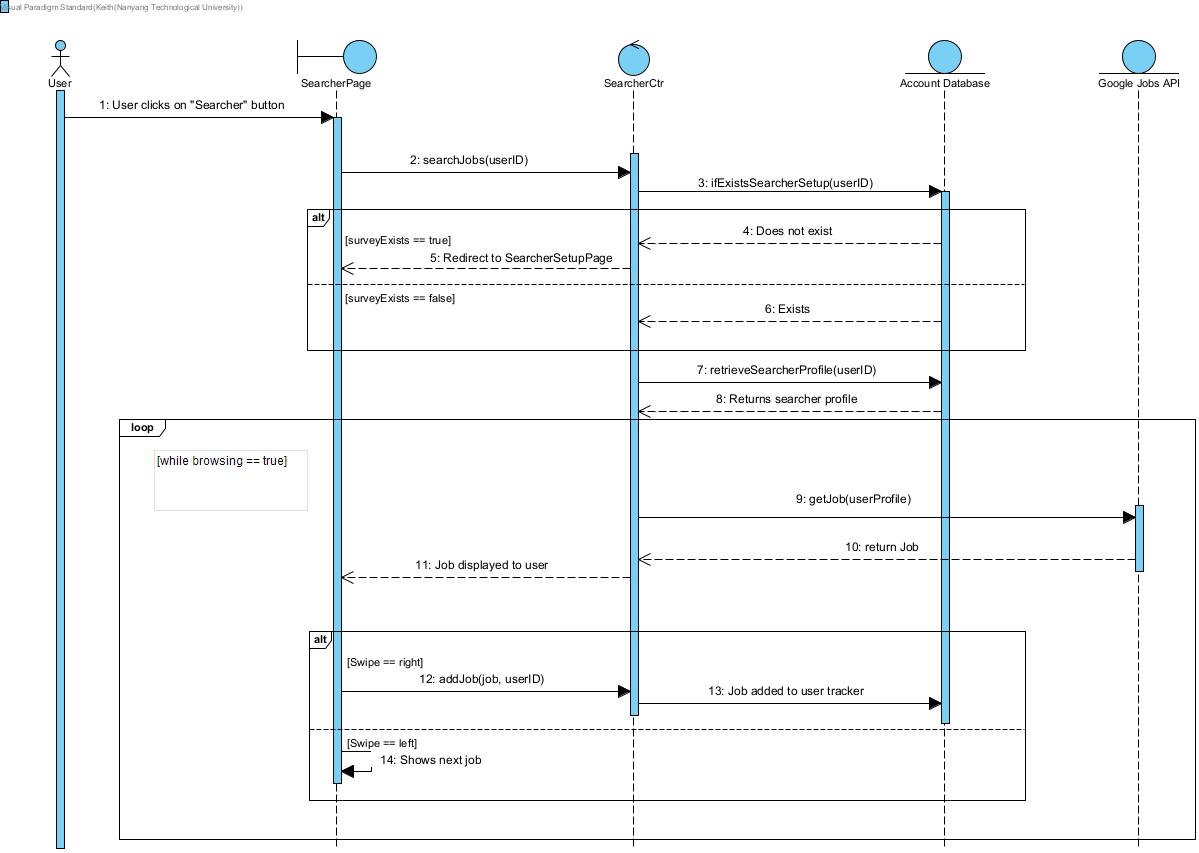
## 5.3.4 Update Entry Sequence Diagram



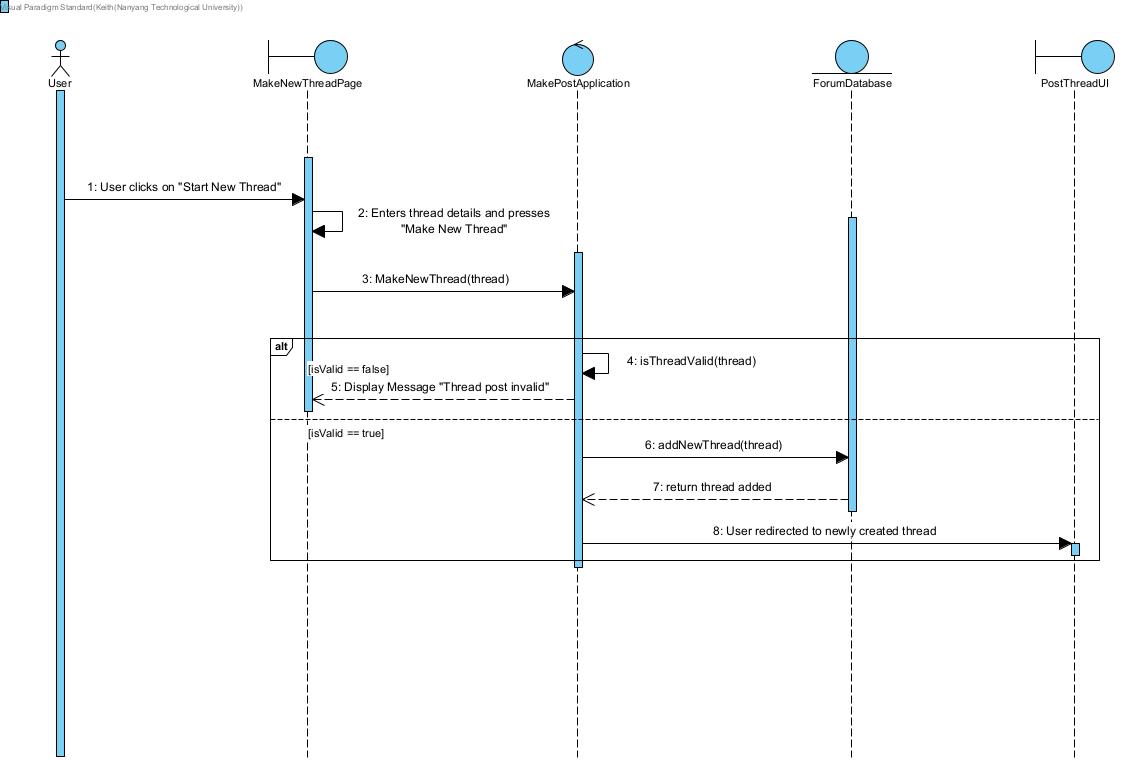
## 5.4.1 Update Job Search Fields Diagram



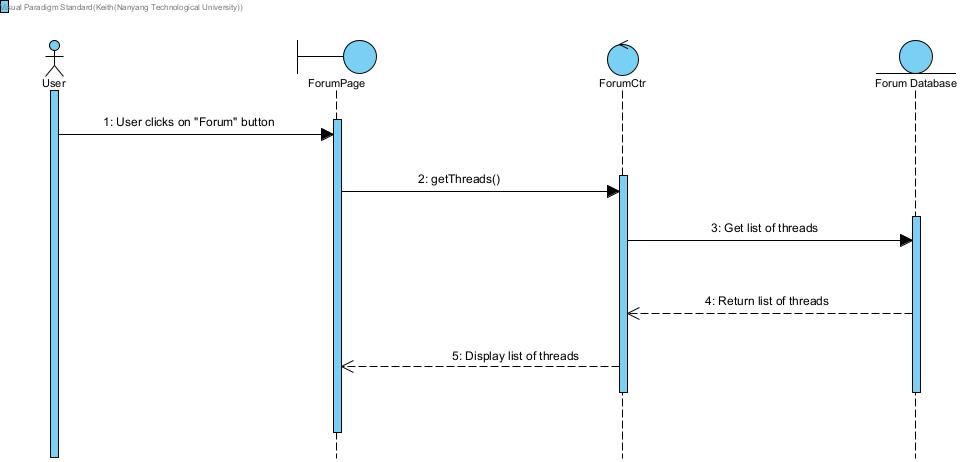
## 5.4.2 View Jobs Available Diagram



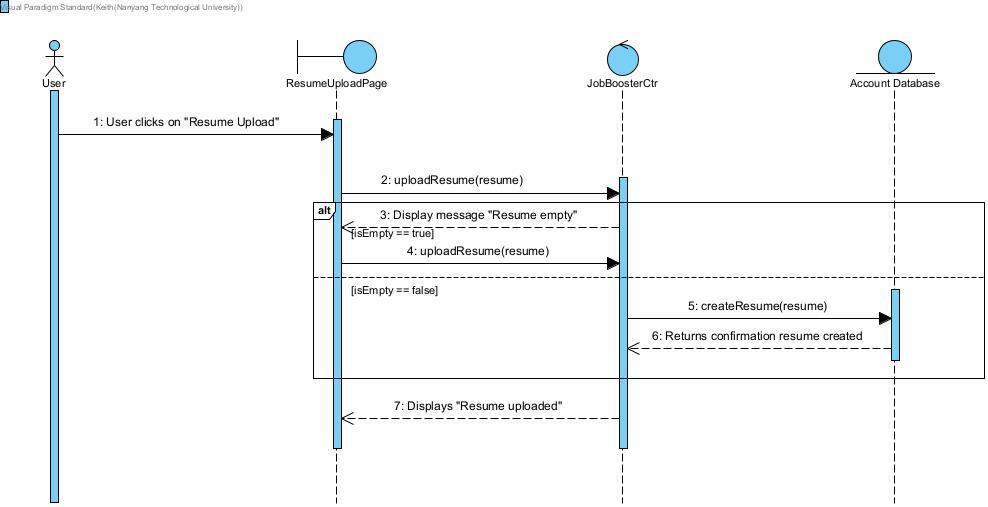
## 5.5.1 Make Forum Post Diagram



## 5.5.2 View Forum Diagram



## 5.6.1 Upload Resume Sequence Diagram



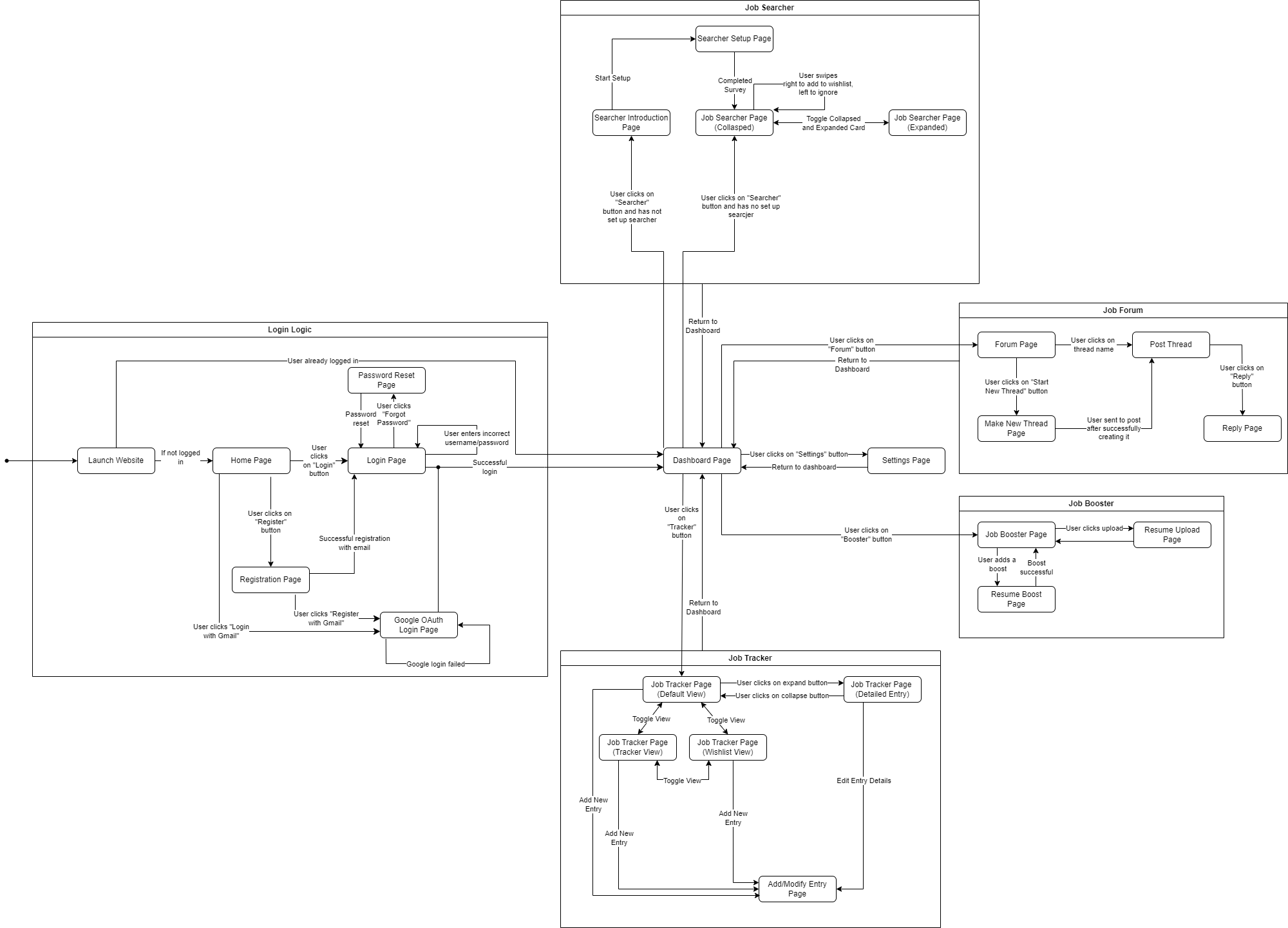
## 5.6.2 Boost Resume Diagram

## 

# 6.0 Initial Dialog Map

## 

## 



# 7.0 System Architecture

