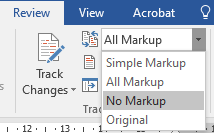
**Notes to the Team**

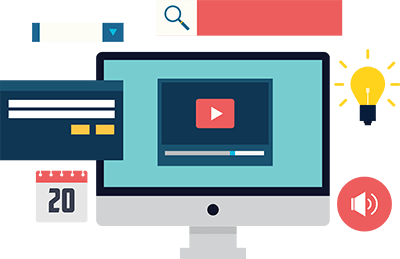
This is a template for the **Elaboration Document** for your project with explanatory notes included.

When using this template, the following steps are recommended:

1. Replace all text enclosed in angle brackets (e.g., <Team Name>) with the correct values. These angle brackets appear either in the body of the document or in headers and footers.
2. To update the Table of Contents, right-click on it and select “Update field” and choose the option - “Update entire table”.
3. Turn off Tracking. Go to the Review tab; make sure Display for Review is set to “No Markup”.



Before submission of this document, delete this instruction section “Notes to the Team”.



Le Fleur King

Elaboration Document

Rosy Outlook

Team Members:

|  |
| --- |
| 224078321, Ganeefa Sirkhotte  223126756, Junaid Brooks |

Document Date: <dd MMMM yyyy>

**TABLE OF CONTENTS**

[1 FUNCTIONAL REQUIREMENTS 4](#_Toc132630173)

[1.1 Analysis Use Case Model 4](#_Toc132630174)

[1.2 Use Case Glossary and Responsibilities 5](#_Toc132630175)

[2 UI Prototypes 7](#_Toc132630176)

[2.1 Team UI Guidelines 7](#_Toc132630177)

[2.2 Team Design for 8](#_Toc132630178)

[2.2.1 Login / Logout 8](#_Toc132630179)

[2.2.2 Dashboard / Landing Page / Main Page 8](#_Toc132630180)

[2.3 UI Designs & Updated Analysis Use Case Narratives 9](#_Toc132630181)

[2.3.1 Designed by <Insert team member name> 9](#_Toc132630182)

[2.3.2 Designed by <Insert team member name> 10](#_Toc132630183)

[2.3.3 Designed by <Insert team member name> 11](#_Toc132630184)

[2.3.4 Designed by <Insert team member name> 12](#_Toc132630185)

[3 DATA REQUIREMENTS 13](#_Toc132630186)

[Implementation Ready Class Diagram 13](#_Toc132630187)

# FUNCTIONAL REQUIREMENTS

## Analysis Use Case Model

[Replace this text with the Analysis Use Case diagram – this must be drawn in Visio.]

## Use Case Glossary and Responsibilities

|  |  |
| --- | --- |
| **Team Member Responsible: Junaid Brooks** | |
| **Use Case Id** | **Use Case Name** |
|  |  |
|  |  |
|  |  |
|  |  |
| **Queries/Reports** | |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Team Member Responsible: Ganeefa Sirkhotte** | |
| **Use Case Id** | **Use Case Name** |
|  |  |
|  |  |
|  |  |
|  |  |
| **Queries/Reports** | |
|  |  |
|  |  |

# UI Prototypes

## Team UI Guidelines

Just a general thing about naviagation mention using the phone/ mobile devices back button as nativation

ARCCK

Navigation is easy, our home and gardens button current will be highlighted, and menu ebing in the top left

Each interface/ page would be designed with one mani functionality and may possibly have other minor functionality

Fonts: only font that would be different is our logos font consisntent font

Pastelcolour scheme

Users will be able to navigate to any page within three actions by using the navigation menu. Navigating through the system and completing tasks will be intuitive.

Logout : Errors will be reduced by making use of confirmation messages

ABACUS Redacted

The consistently placed buttons and screen layout, minimalist features and recognizable icons promote recognition. This consistency ensures that the user only needs to learn how to use the system once and having more features to remember will decrease the user’s memorability of using the website.

Our aim is to keep the layout consisntent throughout the entire app. Our minimalistic design would have features, button and icons recognizable to promote recognition. The consistency combined with the design being similar to many other modern social media apps, ensures that users wouldn’t need to learn anything new.

Uni-Eats user interface is designed in such a way so that it resembles other more recognizable food delivery websites. If our users have had experience with other delivery systems and understand how they work, then it will be easier for users to recognize and understand how the system works. This capitalizes on the user’s mental model on how they understand food delivery websites. Familiar interface metaphors such as the cart symbol are utilized, a cart symbol is used to resemble a shopping cart and the user adds items to this cart just which is the same flow of buying from a physical store.

BinaryBeats

Errors will be handled in the code. A user will not be able to complete an action if the details are not filled in correctly. An error message will be displayed if the input is invalid.

Redacyed

System will have a fixed head consoistnig of the menu/ account button and the logo.

The menu options offer a clear logout option for users. The system can further be navigated by means of the built in android system navigation

**Navigation & Control**

Navigation would be extremely simple. The main pages, the ‘My Friends’ and ‘Following’, would have a BANNER(tab?) on the main screen which high lights which of the two the user is currently on. The app will remain consistent with placements, all pages would have a header which will consist of the logo, which is mainly centered and an account/ menu button, which gives the user access to all the other sections of the application. Navigation through the system and performing any task, such as posting or watering, would be intuitive. The application can further be navigated by means of the built in android system navigation.

Our aim is to keep it consistent with other modern day social media applications, and have it easy to learn. This navigation scheme ensures usability and an enjoyable user experience.

**Support For User Efficiency**

Each interface/ page would be designed with one main functionality and some may have the possibility of having other minor functions included in them.

**Recognition rather than recall**

The user-interface of our app is designed in such a way that it resembles and is consistent with most social media apps, making it something that is recognizable and familiar to users. Our additional features to regular social media apps would have its features, buttons and icons recognizable in a similar way to promote recognition. This consistency ensures that new social media users only needs to learn the system once and users that have previously made use of other social media apps, would just have to learn the new features in our app.

**Aesthetic Design**

The app will have a fixed head consisting of the menu/account button and the logo throughout all the pages on the app. Throughout the app we keep our font and font colour consistent and the same. The only font that differs is that of the logo. It has a minimalistic design, with each page only having one main function. The app will have a warm pastel colour scheme. As this app is about positivity we try not to use any colours associated with negativity.

**Error prevention and recovery**

The errors will be dealt with in code. A user would not be able to complete certain actions (creating an account, logging in and influencer application) if they have not filled in the details correctly. An error message will be displayed if the input is invalid. These error messages would be detailed enough for the user to know where their error lies.

## Team Design for

### Login / Logout

### A screenshot of a phone Description automatically generated with low confidenceDashboard / Landing Page / Main Page

Answer Blooming Gratitude A0500

Post Petal B0100

View Rosebud A1300

Friends Blooming Gratitude A1400

Account Menu A1700

## UI Designs & Updated Analysis Use Case Narratives

### Designed by Junaid Brooks

|  |  |
| --- | --- |
| **Use Case ID** | **Use Case Name** |
|  |  |
| **Actors** | |
|  | |
| **Description** |  |
| **Pre-Conditions** |  |
| **Triggers** |  |
| **Post-Conditions** |  |
| **Basic Flow of Events** |  |
| **Alternate Flow of events** |  |
| **Initial UI design** |  |

|  |  |
| --- | --- |
| **Use Case ID** | **Use Case Name** |
| D0100 | View Application |
| **Actors** | |
| Admin | |
| **Description** | This use case describes the process of viewing all the current influencer applications that have been submitted (A0400). |
| **Pre-Conditions** | User has to be on an admin account. |
| **Triggers** | Admin clicks View Application button (Influencer Button). |
| **Post-Conditions** | A list of unprocessed influencer applications is displayed. |
| **Basic Flow of Events** | 1. The system displays all the influencer applications. 2. Admin selects to accept application 3. D0200 is called |
| **Alternate Flow of events** | 2.1 Admin selects to reject application  2.2 D0300 is called |
| **Initial UI design** | Use case D0100 is invoked.  View application details  Can be sorted from oldest to newest or vice versa.    Use case D0200 is called.  Influencer Application  Use case D0300 is called. |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0200 | Approve Influencer | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | Admin makes a regular user a new influencer. | |
| **Pre-Conditions** | A user submitted an application to be an influencer. | |
| **Triggers** | Called from D0100 | |
| **Post-Conditions** | There is now a new influencer. | |
| **Basic Flow of Events** | 1. Admin will review the application. 2. Admin approves the influencer. 3. Return to step 1 in D0100 | |
| **Initial UI Design** | There is now a potential new Influencer. Use case D0200 is invoked. | |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0300 | Reject Influencer | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | Admin denies a regular user becoming a new influencer. | |
| **Pre-Conditions** | A user submitted an application to be an influencer. | |
| **Triggers** | Called from D0100 | |
| **Post-Conditions** | Influencer application is rejected. | |
| **Basic Flow of Events** | 1. Admin would review the application. 2. Admin rejects the application. 3. Return to step 1 in D0100 | |
| **Initial UI Design** | User gets rejected from being an Influencer. Use case D0300 is invoked. | |

|  |  |
| --- | --- |
| **Use Case ID** | **Use Case Name** |
| D0400 | View Petal Addition |
| **Actors** | |
| Admin | |
| **Description** | This use case describes the process of viewing all the pending petal additions that have been submitted by influencers (B0100). |
| **Pre-Conditions** | User has to be on an admin account |
| **Triggers** | Admin clicks the View Petal Addition button (Petals button) |
| **Post-Conditions** | A list of unprocessed petals is displayed |
| **Basic Flow of Events** | 1. The system displays all the Petal Additions that admin needs to approve/deny. 2. Admin selects to accept the petal addition 3. D0500 is called |
| **Alternate Flow of events** | Admin selects to reject the petal additionD0600 is called |
| **Initial UI design** | Use case D0600 is called.  Use case D0500 is called.  Use case D0400 is invoked.  Can be sorted from oldest to newest or vice versa. |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0500 | Approve Petal | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | The admin accept/ adds the new petal’s addition to the influencers garden. | |
| **Pre-Conditions** | Influencer has uploaded a petal that’s waiting for approval or to be denied. | |
| **Triggers** | Called from D0400 | |
| **Post-Conditions** | Influencer has a new petal in their garden. | |
| **Basic Flow of Events** | 1. Admin reviews the Petal. 2. Admin accepts the Petal. 3. System removes Petal from list to be reviewed by admin. 4. Return to step 1 of D0400 | |
| **Initial UI Design** | Use case D0500 is invoked. | |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| D0600 | Reject Petal | |
| **Primary Business Actors** | | **Other participating Actors** |
| Admin | |  |
| **Description** | The admin denies the new petal’s addition to the influencers garden. | |
| **Pre-Conditions** | Influencer has uploaded a petal that’s waiting for approval or to be denied. | |
| **Triggers** | Called from D0400 | |
| **Post-Conditions** | New flower is not added to the Garden. | |
| **Basic Flow of Events** | 1. Admin reviews the Petal. 2. Admin rejects the addition. 3. System removes Petal from list to be reviewed by admin. 4. Return to step 1 of D0400 | |
| **Initial UI Design** | Use case D0600 is invoked. | |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| A0200 | Add Friend | |
| **Primary Business Actors** | | **Other participating Actors** |
| Public | | Influencer |
| **Description** | This use case describes the process of adding a new friend, or more specifically sending a friend request. | |
| **Pre-Conditions** | Public must have registered an account. Must be on the Friends tab of the Friends page. | |
| **Triggers** | Called from A1500 | |
| **Post-Conditions** | Friend request was sent. | |
| **Basic Flow of Events** | 1. User would click on the search/add friend button 2. User types in their friend’s username and hits the search button. 3. System checks if username exists. 4. User then selects friend’s account. 5. User then click on Add friend button. | |
| **Alternate Flow of Events** | 3.1 Check fails, system displays error.  3.2 Returns to step 2. | |
| **Initial UI Design** | System displays a list of accounts that the user could be looking for.  Use case A0200 is invoked. Friend request is sent.  Search bar. User enters their friend’s username.    Username not found System displays that no such user exists. | |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| A0600 | Accept Friend | |
| **Primary Business Actors** | | **Other participating Actors** |
| Public | | Influencer |
| **Description** | Users will receive friend requests from other users, when they try to add them as a friend. User could then accept the friend request. | |
| **Pre-Conditions** | User has a registered account and has a pending friend request.  Public has to be in the Friends tab of the Friends page, which is invoked by A1500. | |
| **Triggers** | Invoked by A1000. | |
| **Post-Conditions** | A new friend was added. | |
| **Basic Flow of Events** | 1. User clicks the Add/ accept button. | |
| **Initial UI Design** | Accept Friend Request. Use case A0600 is invoked.  Pending Friend Requests | |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| A0700 | Reject Friend | |
| **Primary Business Actors** | | **Other participating Actors** |
| Public | |  |
| **Description** | A User receives a friend request from someone they do not wish to add, so they reject the request. | |
| **Pre-Conditions** | User has a registered account and have a pending friend request.  Public has to be in the Friends tab of the Friends page, which is invoked by A1500. | |
| **Triggers** | Invoked by A1000. | |
| **Post-Conditions** | User has denied the friend request. | |
| **Basic Flow of Events** | 1. User clicks the Deny/ reject button. | |
| **Initial UI Design** | Reject Friend Request. Use case A0700 is invoked. | |

|  |  |
| --- | --- |
| **Use Case ID** | **Use Case Name** |
| A1000 | View Friend Request |
| **Actors** | |
| Public | |
| **Description** | This use case describes the process of viewing the friend requests that have been sent to a user. Both incoming and outgoing are visible to the user. |
| **Pre-Conditions** | User either needs to have a pending friend request or have sent out request that hasn’t been accepted nor rejected. |
| **Triggers** | Invoked by A1500 |
| **Post-Conditions** | The system displays the friend request that were sent to that user |
| **Basic Flow of Events** | 1. System displays a list of people that have sent friend requests to the user. There could also not be any new friend requests. 2. Select request to accept 3. Call A0600 |
| **Alternate Flow of events** | 2.1 Select request to reject  2.2 Call A0700 |
| **Initial UI design** | Use case A1000 will be invoked  Use case A0700 will be invoked  Use case A0600 will be invoked  Search for friend/ user |

### Designed by Ganeefa Sirkhotte

|  |  |
| --- | --- |
| **Use Case ID** | **Use Case Name** |
|  |  |
| **Actors** | |
|  | |
| **Description** |  |
| **Pre-Conditions** |  |
| **Triggers** |  |
| **Post-Conditions** |  |
| **Basic Flow of Events** |  |
| **Alternate Flow of events** |  |
| **Initial UI design** |  |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| A1500 | View Friend | |
| **Primary Business Actors** | | **Other participating Actors** |
| Public | | Influencer |
| **Description** | This use case describes the process of viewing the Friends section. | |
| **Pre-Conditions** | User must have an account. | |
| **Triggers** | Called from A1700. | |
| **Post-Conditions** | User’s friends are displayed. | |
| **Basic Flow of Events** | 1. The system displays the friends of the user. 2. User Selects to Add Friend 3. System calls A0200 | |
| **Alternate Flow of Events** | 2.1 Select Friend Requests  2.2 System calls A1000 | |
| **Initial UI Design** | Showing additional info of friends.  Displaying all the users friends.  Select to view friend requests. Use case A1000 is called.  Tapping the search button selects to add a friend. Use case A0200 is called.  View Friends. Use case A1500 is invoked. | |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| A1700 | View Account | |
| **Primary Business Actors** | | **Other participating Actors** |
| Public | | Influencers |
| **Description** | This use case describes the process of viewing the Account menu | |
| **Pre-Conditions** | User must have a registered account. | |
| **Triggers** | User clicks Account button. | |
| **Post-Conditions** | A menu of options is displayed. | |
| **Basic Flow of Events** | 1. System displays a menu. 2. User selects Friends. 3. System calls A1500 | |
| **Alternate Flow of Events** | 2.1.1 User selects Following.  2.1.2 System calls A1800.  2.2.1 User selects Gratitude History.  2.2.2 System calls A1100.  2.3.1 User Selects Influencer Application,  2.3.2 System calls A0400. | |

|  |  |
| --- | --- |
| **Initial UI Design** | Regular Users:  Menu being displayed.    Calls use case A0400.  Calls use case A1500.  Calls use case A1800.  Influencers:  Calls use case A1100.    User logs out of their account.  Calls use case A1200. |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| A0900 | Follow Influencer | |
| **Primary Business Actors** | | **Other participating Actors** |
| Public | | Influencer |
| **Description** | Follow a specific influencer to then have their flowers appear on your page. | |
| **Pre-Conditions** | User needs to have a registered account.  Public has to be in the Following tab of the Friends page. | |
| **Triggers** | Called from A1700. | |
| **Post-Conditions** | The User now follows the influencer. | |
| **Basic Flow of Events** | 1. User types in their influencers username and hits the search button. 2. System checks if username exists 3. User then selects influencer’s account. 4. User then click on follow button. | |

|  |  |
| --- | --- |
| **Alternate Flow of Events** | 2.1 Check fails, system displays error.  2.2 Returns to step 1. |
| **Initial UI Design** | Use case A0900 is invoked.  System displays the possible Influencer it may be.  User types in Influencers Username    Username not found. System displays that no such user exists. |

|  |  |  |
| --- | --- | --- |
| **Use Case ID** | **Use Case Name** | |
| A1800 | View Following | |
| **Primary Business Actors** | | **Other participating Actors** |
| Public | | Influencer |
| **Description** | This use case describes the process of viewing the Following section. | |
| **Pre-Conditions** | User must have an account. | |
| **Triggers** | Called from A1700. | |
| **Post-Conditions** | User’s following is displayed. | |
| **Basic Flow of Events** | 1.The system displays the influencers that the user is following.  2.User Selects to Follow Influencer  3.System calls A0900 | |
| **Initial UI Design** | A0900 is called.  Additional info.  Menu to view additional info.  List of Influencer that user follows. | |

# DATA REQUIREMENTS

## Implementation Ready Class Diagram

[Replace this text with the implementation ready diagram – this must be drawn in Visio.]