**Software Engineering 2016**



Foodpass mobile application

Version 1.0

By

LAN company

Team name: LAN

**4) Component-level design**

**4.1) Description component (User Management by user)**

**4.1.1) Processing narrative (PSPEC) for component**

This component uses to manage the user’s information. It will control the action that user do with this application about their own information which are registration to FoodPass application via Facebook, the way to login to the application, manage all information that related to that user for example, history about the thing that user already access or use.

**4.1.2) User management processing detail**

**4.1.2.1) Design class hierarchy for user management by user**

|  |
| --- |
| **User Management by user** |
| - facebookName  - userID |
| + connectFacebook()  + readUserInformation |

**4.1.2.2) Restriction/ limitations for User management**

- When user want to sign in to this application, Facebook account must be the one that haven’t been use before.

- Sign in process will be success only when user do the correct step.

- When sign in process, all the information will be store in the correct database only.

- User can access and look at only their own information.

- The system need internet access to be able to view and edit personal information.

**4.1.2.3) Performance issues for component user management by user**

- If there are a lot of user using the application, the system may be delay or error because the system may slow down when it going into database.

- The system may slow down if that user has a bad internet for example when the user want to sign in, login, logout from this application.

**4.1.2.4) Design constraints for component user management by user**

**4.1.2.5) Processing detail for each operation of component user management**

**4.1.2.5.1) Processing narrative (PSPEC) for each operation**

|  |  |
| --- | --- |
| Operation | Description |
| connectFacebook() | Show the information that use to sign in/ login to this application which are facebook name and store that information in the database. |
| readUserInformation() | Show the information of that user which is facebook account/name. |

**4.1.2.5.1) Algorithmic model (e.g., PDL) for each operation**

|  |
| --- |
| connectFacebook()    SAVE TO DATABASE |

*Table: pseudo code for method connectFacebook()*

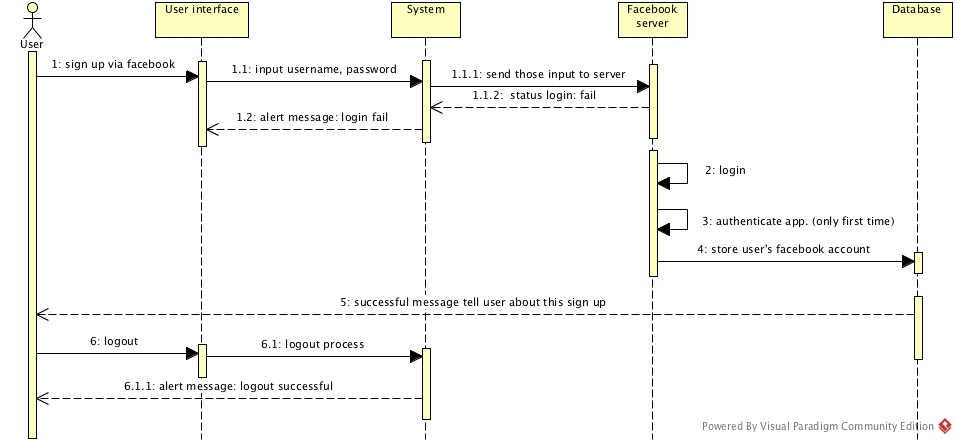
|  |
| --- |
| readUserInformation()  connectFacebook()             IF connect to facebook success             SHOW facebook name  ENDIF |

*Table: pseudo code for method readUserFacebook()*

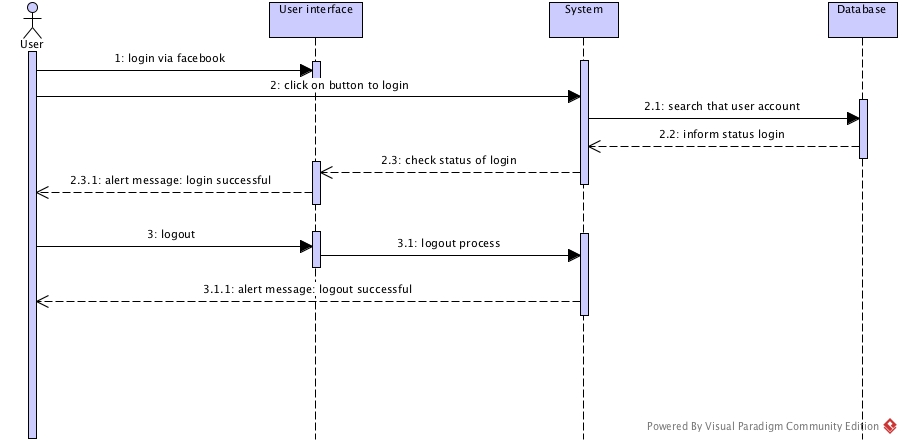
**4.1.3) User management by user test points last and description**

|  |  |  |
| --- | --- | --- |
| No. | Test Description | Expected Result |
| 1 | When user click on profile page | It will show user’s facebook account/ name. |
| 2 | If user success in connect facebook. | It will automatically generate userID and store user’s information in database. |
| 3 |  |  |

**4.1.4) Component user management by user dynamic behavior**



*Figure: Sequence diagram show a signup process to access in the application*

****

*Figure: Sequence diagram show a sign in process to access in the application*

**4.1.5) Component user management interfaces**

For the user interfaces of this component will be show in User interfaces design (page ).

**4.2) Description component (Restaurant Management)**

**4.2.1) Processing narrative (PSPEC) for component**

This component uses to manage the restaurant’s information. It will manage the information by including function which are delete restaurant/restaurant’s information, edit restaurant/restaurant’s information, add restaurant/restaurant’s information.

**4.2.2) Restaurant management processing detail**

**4.2.2.1) Component of restaurant management detail**

|  |
| --- |
| **Restaurant Management** |
| - restaurantName  - restaurantId  - restaurantAddress  - restaurantTelephone  - restaurantDetail  - rateOfRestaurant |
| + addRestaurantInfo()  + getRestaurantId()  + readRestaurantInfo()  + editRrestaurantInfo()  + deleteRestaurantInfo() |

**4.2.2.2) Restriction/ limitations for restaurant management**

- If administrator didn’t type restaurant’s name, an adding process won’t be run.

- When there is no internet connection, cannot use database so it lead to can’t add restaurant detail.

- When there is no internet connection, cannot use database so it lead to can’t delete restaurant data.

- When there is no internet connection, cannot use database so it lead to can’t edit restaurant data.

- When there is no internet connection, cannot use database so it lead to can’t get restaurant info.

- When there is no internet connection, cannot use database so it lead to can’t get restaurant ID.

- After submit all data requirement, it will store on database.

-This part need to do by admin only.

**4.2.2.3) Performance issues for component restaurant management by admin**

* This application need to connect to internet. Otherwise it will slow down and can’t connect to database.
* If admin try to add a lot of information of data into database, the application may be error and delay.

**4.2.3) Processing detail for each operation of component restaurant Management**

**4.2.3.1. Processing narrative (PSPEC) for each operation**

|  |  |
| --- | --- |
| **Operation** | **Description** |
| addRestaurantInfo() | Add restaurant information which include restaurant name, restaurant address, restaurant telephone and other restaurant detail to database. |
| getRestaurantId() | Get restaurant Id from database |
| readRestaurantInfo() | Get restaurant information from database |
| editRrestaurantInfo() | Edit restaurant detail in database |
| deleteRestaurantInfo() | Delete restaurant detail in database |

**4.2.3.2) Algorithmic model for each operation**

|  |
| --- |
| addRestaurantInfo( restaurantName ,restaurantAddress, restaurantTelephone,restaurantDetail)  TEXT restaurantName  TEXT restaurantAddress  TEXT restaurantTelephone  TEXT restaurantDetail                   INCREMENT restaurantId                  INSERT restaurantName                  INSERT restaurantAddress                  INSERT restaurantTelephone                  INSERT restaurantDetail |

*Table: pseudo code for method addRestaurant()*

|  |
| --- |
| getRestaurantId()              SEARCH restaurantName     IF(FIND restaurantName OR restaurantId)              GET restaurantId; |

Table: pseudo code for method getRestaurantId()

|  |
| --- |
| editRestaurantInfo()             SEARCH restaurantName OR restaurantId          IF(FIND restaurantName OR restaurantId)                  EDIT restaurantName                  EDIT restaurantAddress                  EDIT restaurantTelephone                  EDIT restaurantDetail |

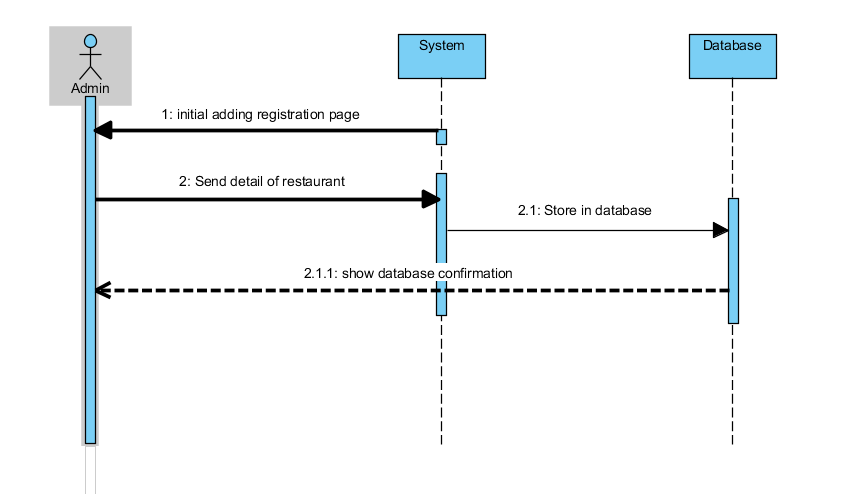
*Table: pseudo code for method editRestaurantInfo()*

|  |
| --- |
| deleteRestaurantInfo()  SEARCH restaurantName OR restaurantId          IF(FIND restaurantName OR restaurantId)                  DELETE restaurantName                  DELETE restaurantAddress                  DELETE restaurantTelephone                  DELETE restaurantDetail |

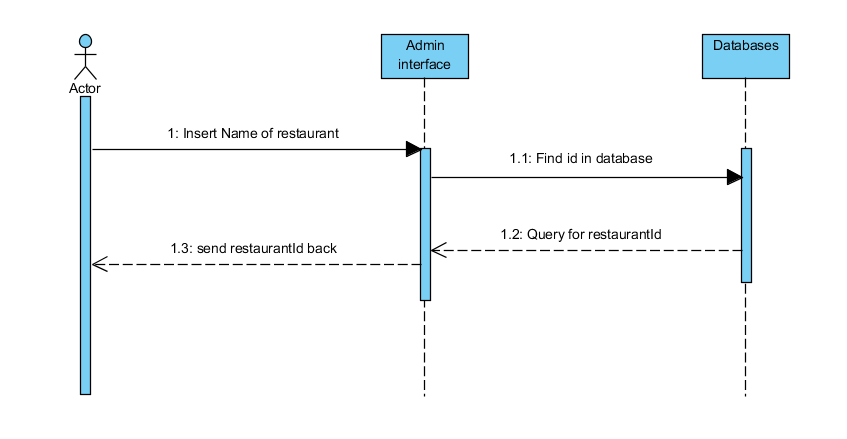
*Table: pseudo code for method deleteRestaurantInfo()*

**4.2.3.3. Component restaurant management test points list and description**

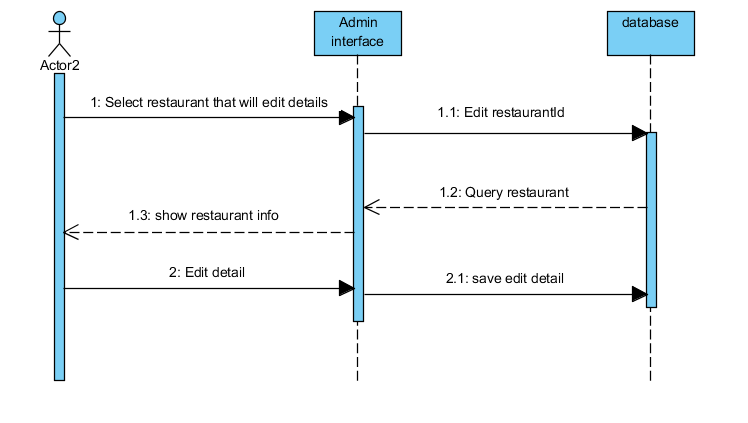
|  |  |  |
| --- | --- | --- |
| **No.** | **Test description** | **Expected result** |
| **1** | **Adding restaurant information** | **Be able to add information to database correctly.** |
| **2** | **Get restaurantId from restaurantName** | **Be able to get restaurantId from database correctly** |
| **3** | **Edit restaurant information** | **Be able to edit restaurant detail successfully.** |
| **4** | **Delete restaurant information** | **Be able to delete restaurant detail successfully.** |

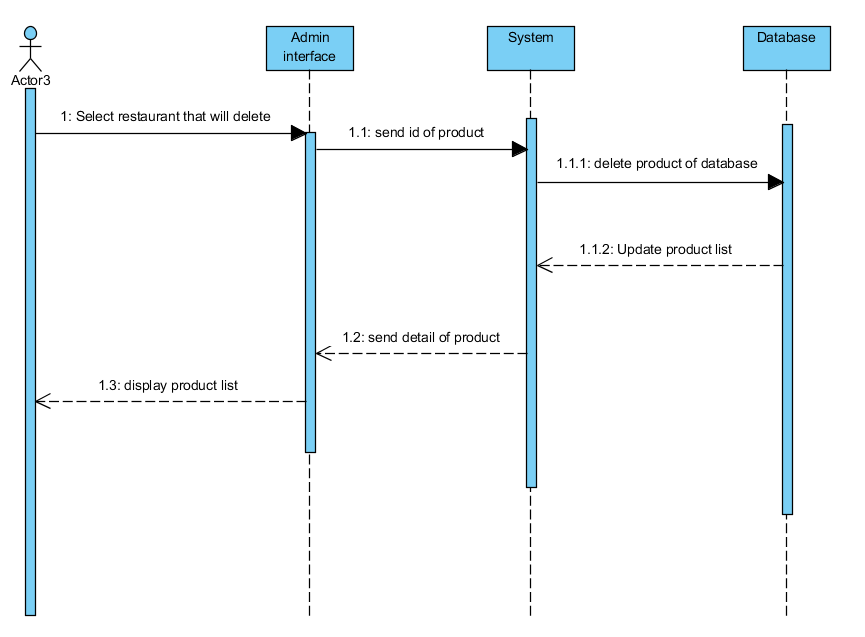
**4.2.4) Component restaurant management by user dynamic behavior**

*Figure: Sequence diagram of Adding restaurant information*



*Figure: Sequence diagram of GetId*

  
  
*Figure: Sequence diagram of Edit restaurant information*



*Figure: Sequence diagram of delete restaurant information*

**4.2.5) Component user management interfaces**

For the user interfaces of this component will be show in User interfaces design (page ).

**4.3) Description component (MyList Management)**

**4.3.1) Processing narrative (PSPEC) for component**

This component uses to manage the favorite restaurant that the user adds to his/ her account. It will control the action that user do with the restaurants that the user picks. The user can add the restaurant into the list, delete it from the list, and show all restaurants in the list.

**4.3.2) MyList Management processing detail**

**4.3.2.1) Component of MyList Management detail**

|  |
| --- |
| **MyList Management** |
| - userId  - restaurantId  - favoriteList[ ] |
| + addFavorite()  + readFavorie()  + deleteFavorite() |

**4.3.2.2) Restriction/ limitations for MyList Management**

- If the query from the database is failed, the user can’t add the restaurant to MyList

- This component won’t work if the user don’t pick any restaurant.

**4.3.2.3) Performance issues for component MyList Management**

- If the number of the restaurants in the MyList is too much, it may cause the delay.

**4.3.2.4) Processing detail for each operation of component MyList Management**

**4.3.2.4.1) Processing narrative (PSPEC) for each operation**

|  |  |
| --- | --- |
| Operation | Description |
| addFavorite() | Add the restaurant from selectRestaurant(), then check whether this restaurant is in MyList or not, if there is no, it will be added. |
| readFavorite() | Show the restaurant in MyList of the user |
| deleteFavorite() | Delete restaurant from MyList. |

**4.3.2.4.2) Algorithmic model for each operation**

|  |
| --- |
| readFavorite() INTEGER i=0    WHILE available in list (or favoriteList[ ]) (i<number of restaurant in favorite list) check favoriteList[i] in database then show detail  ELSE SHOW ERROR  i=i+1;    ENDWHILE |

*Table: pseudo code for method readFavorite()*

|  |
| --- |
| deleteFavorite()  INTEGER favoriteList [ ]    INTEGER restaurantId INTEGER i = 0 WHILE NOT SUBMIT    INPUT restaurantId favoriteList[I] = restaurantId i = i+ 1    ENDWHILE  WHILE available in list (or favoriteList[ ]) (I<number of select list in favorite list)    check(IF) favoriteList[i] in favorite list then delete ELSE SHOW DELETE ERROR at favoriteList[i] i=i+1    ENDWHILE |

*Table: pseudo code for method deleteFavorite()*

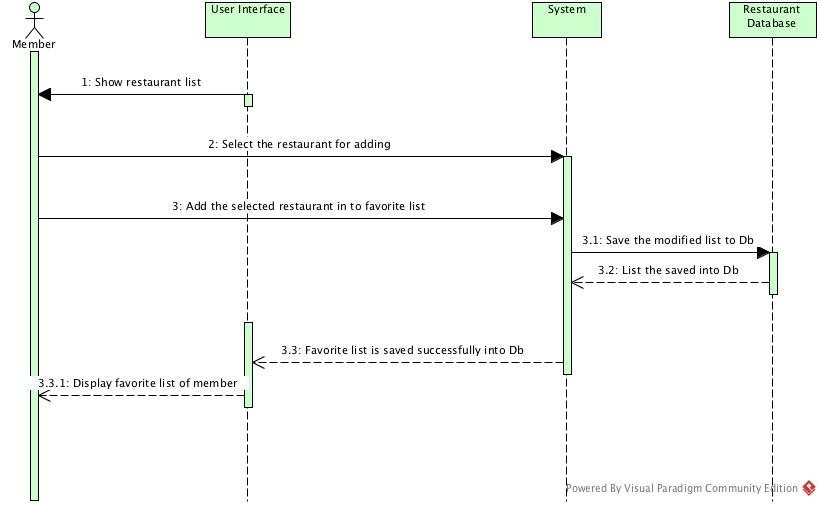
|  |
| --- |
| addFavorite()  STRING favoriteList[ ]    INTEGER i = 0  INTEGER j = 0  selectProduct()  WHILE available in list (or restaurantId[ ]) (i<number of restaurant)    check restaurant[i] in database then favoriteList[J] = restaurasntId[I] and j=j+1  i=i+1    ENDWHILE |

*Table: pseudo code for method addFavorite()*

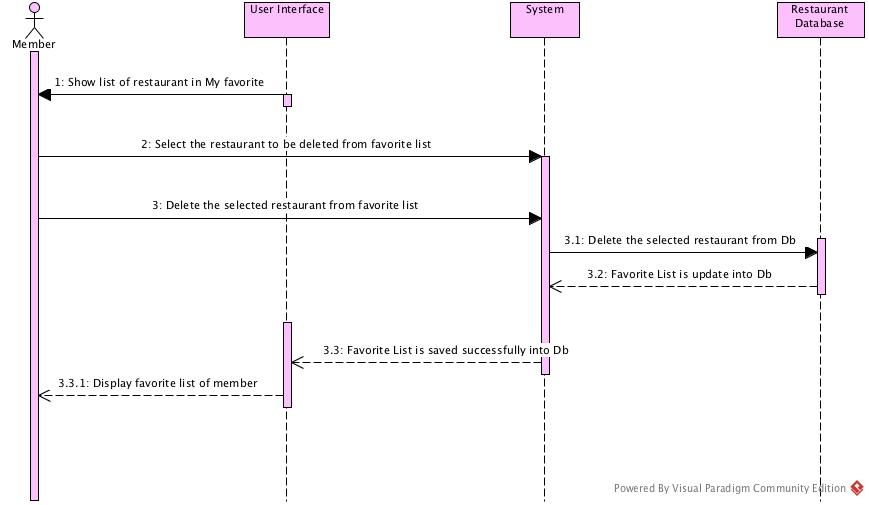
**4.3.3) MyList management by user test points last and description**

|  |  |  |
| --- | --- | --- |
| No. | Test Description | Expected Result |
| 1 | When user click on button for add restaurant in MyList | It will add that restaurant in database. |
| 2 | Click delete in MyList | Restaurant is deleted from the database |
| 3 | Click to see MyList | Show restaurant in MyList for each member that is stored in database |

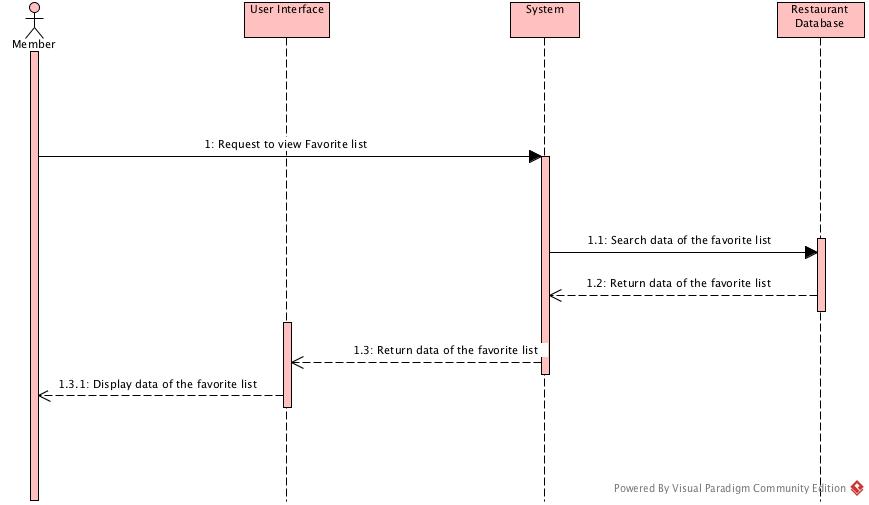
**4.3.4) Component MyList management by user dynamic behavior**



*Figure: This Sequence diagram show an add restaurant into MyList (Favorite list).*



*Figure: This Sequence diagram show delete restaurant into MyList (Favorite list).*



*Figue: This sequence diagram show delete restaurant into MyList (Favorite list)*

**4.3.5) Component user management interfaces**

For the user interfaces of this component will be show in User interfaces design (page ).

**5) User interface design**

**5.1) User interface design rules**

In this part, we need to have a rule name ‘The golden rules for user interface design’

The golden rules for user interface design are:

1. Strive for consistency.

Consistent sequences of actions should be required in similar situations; identical terminology should be used in prompts, menus, and help screens; and consistent color, layout, capitalization, fonts, and so on should be employed throughout.

1. Cater to universal usability

Recognize the needs of diverse users and design for novice to expert differences, age ranges, disabilities, and technological diversity. Adding features for novices, such as explanations, and features for experts, such as shortcuts and faster pacing, can enrich the interface design and improve perceived system quality.

1. Offer informative feedback

For every user action, there should be system feedback. Visual presentation of the objects of interest provides a convenient environment for showing changes explicitly.

1. Design dialogs to yield closure.

Sequences of actions should be organized into groups with a beginning, middle, and end. Informative feedback at the completion of a group of actions gives operators satisfaction. For example, e-commerce web sites move users from selecting products to the checkout, ending with a clear confirmation page that completes the transaction.

1. Prevent errors

As much as possible, design the system such that users cannot make serious errors; for example, gray out menu items that are not appropriate and do not allow alphabetic characters in numeric entry fields. If a user makes an error, the interface should detect the error and offer specific instructions for recovery.

1. Permit easy reversal of actions

As much as possible, actions should be reversible.

1. Support internal locus of control

Experienced users strongly desire the sense that they are in charge of the interface and that the interface responds to their actions. They don’t want surprises or changes in familiar behavior. They are annoyed by too-long data-entry sequences, difficulty in obtaining necessary information.

1. Reduce short-term memory load

Humans’ limited capacity for information processing in short-term memory requires that designers avoid interfaces in which users must remember information from one screen and then use that information on another screen. It means that cell phones should not require re- entry of phone numbers, web-site locations should remain visible, multiple-page displays should be consolidated, and sufficient training time should be allotted for complex sequences of actions.

**5.2) Components and development tools used**

- Justinmind program, use to create/ design user interface for example, each page in FoodPass application.

- MySQL program, use for create database for this application

- Apache

-

**5.3) Screen images and description**

1. Application login for both admin and user

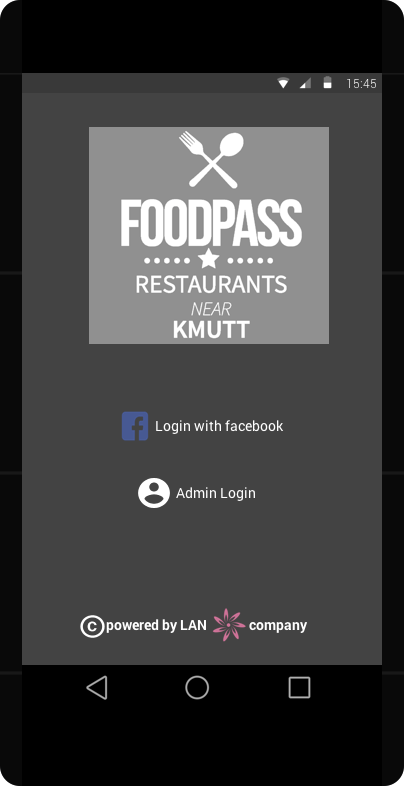


Figure1.1 This user interface show the first page of application log.

2. Login for user and admin

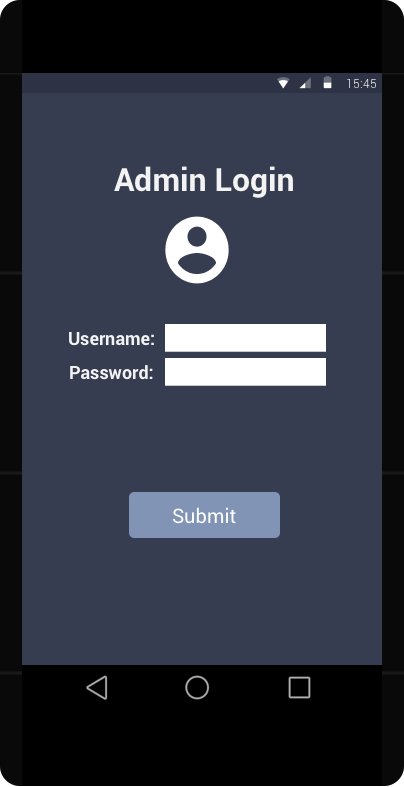


Figure2.1 This user interface show the admin login page.

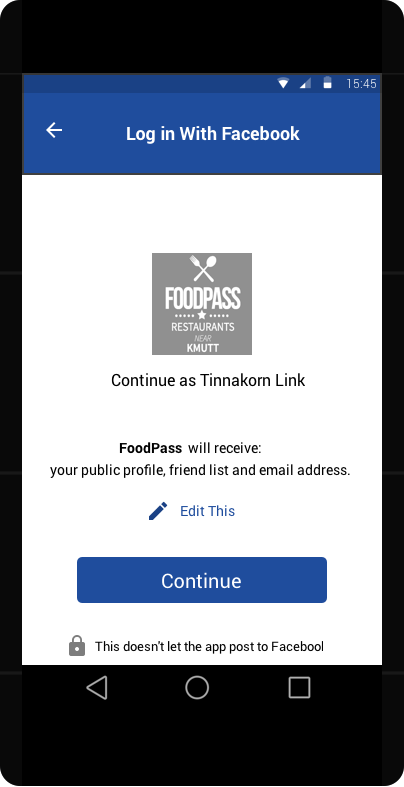


Figure2.2 This user interface show authentication to Facebook (user already login Facebook).

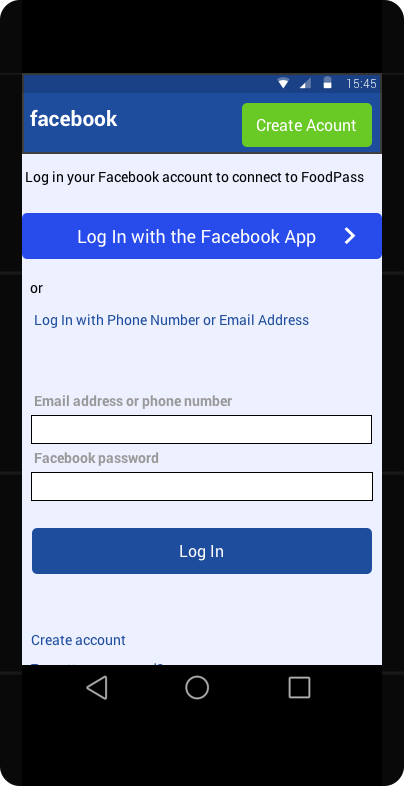


Figure2.3 This user interface show Facebook login page which the user hasn’t login on Facebook yet.

 3. Profile page (To access Profile page user have to click button in Figure5.)

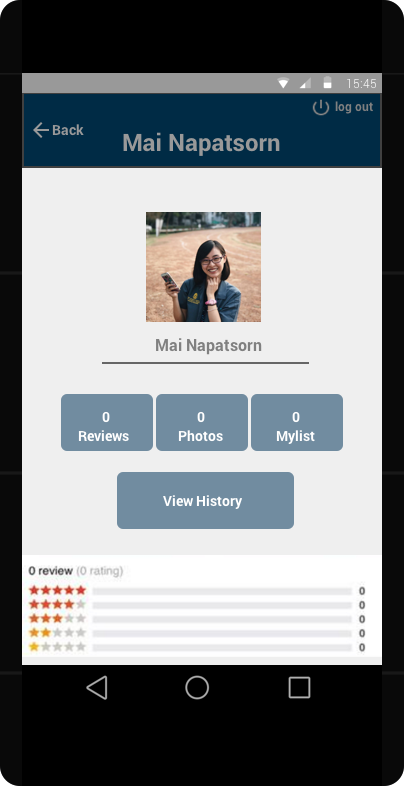


Figure3.1 This user interface show a profile page of user.

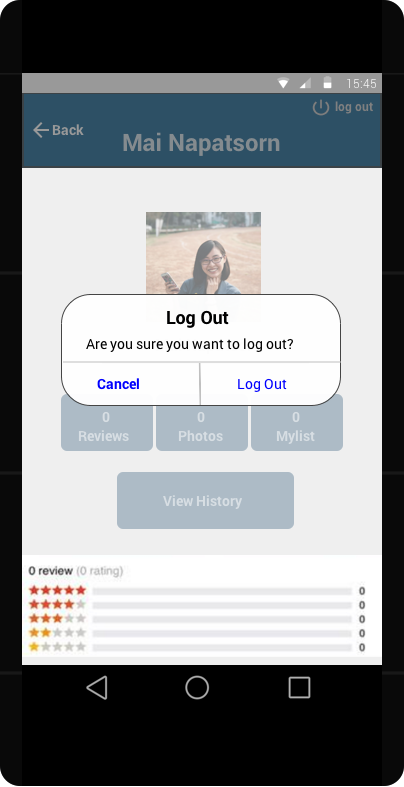


Figure3.2 This user interface show a logout confirmation. If user want to confirm logout

they have to click ‘Log Out’ button.

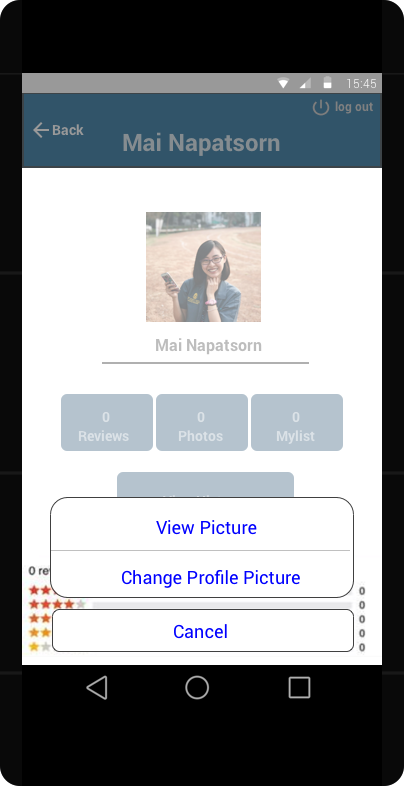


Figure3.3 This user interface use for edit profile picture of user.

4. Category page which show the category of all restaurant

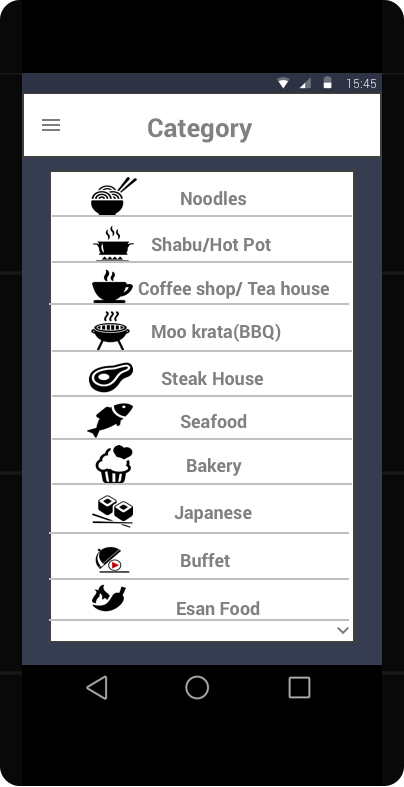


Figure4 This user interface show category of all restaurant.

5. Recommend page which is the first page after user login.

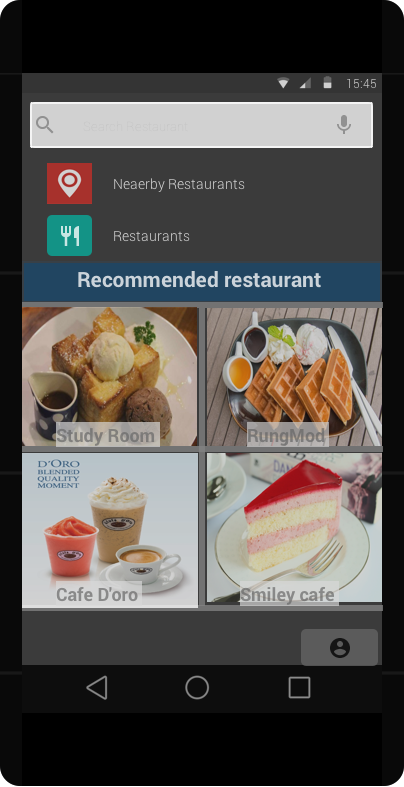


Figure5 This user interface show recommend page after user login.

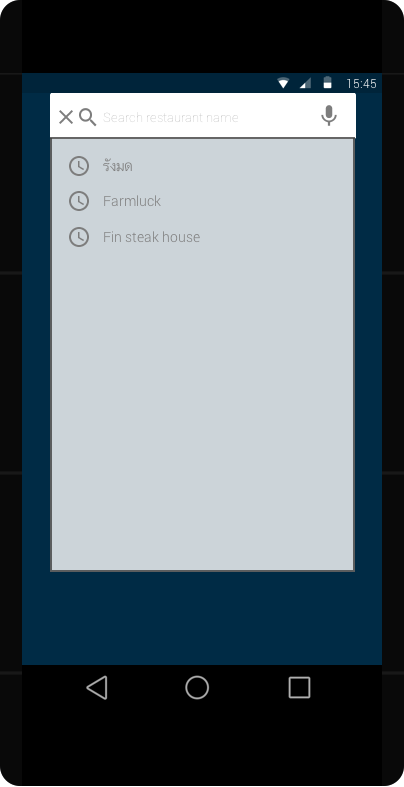
6. Search restaurant (After clicking on search bar on the top of Figure5, the search feature will appear to user like in Figure6.)

Figure6 This user interface show a search page when user want

to find restaurant by using restaurant name.

7. Nearby restaurant (Show a nearby restaurant, with their rate and their open/close time, the distance from yourself to that restaurant.)

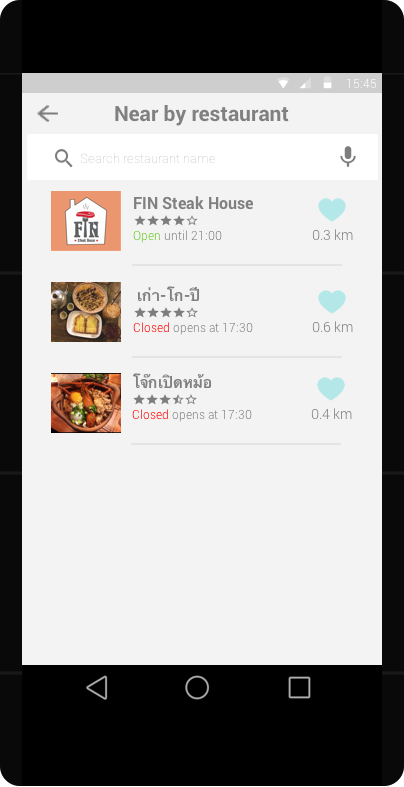


Figure7 This user interface shows a nearby page which show a nearby restaurant.

8. Restaurant page

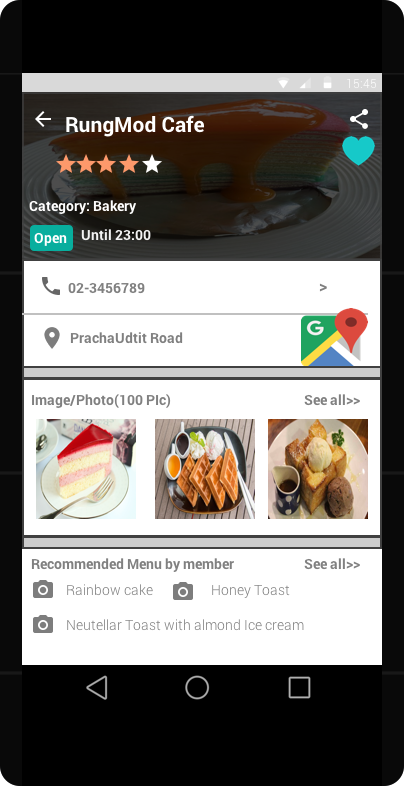


Figure8.1 This user interface show first page of restaurant which will show a restaurant information

for examples restaurant name, telephone number, Google Map, photo.

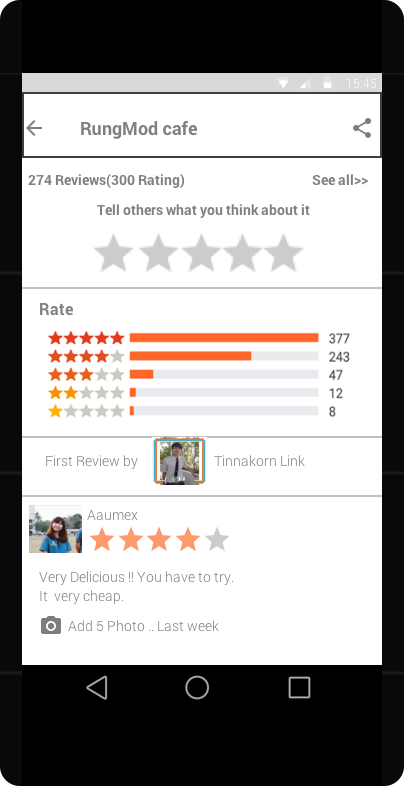


Figure8.2 This user interface show a page for rate the restaurant and also show the rating information that other people have been voted.

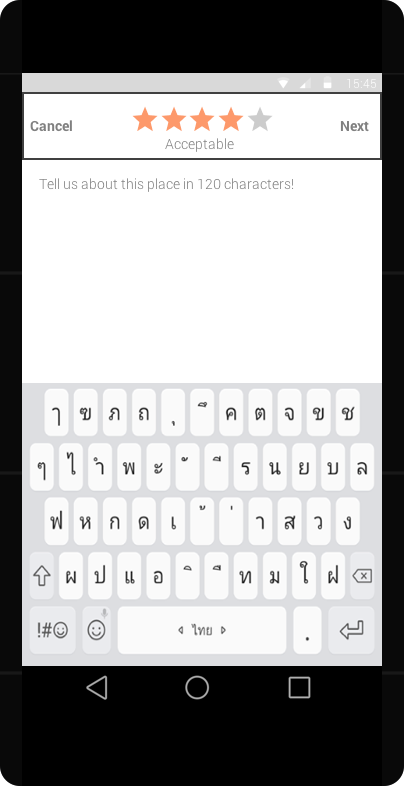


Figure8.3 This user interface show a page for typing a comment that user what to say with a rate that user has already rated.

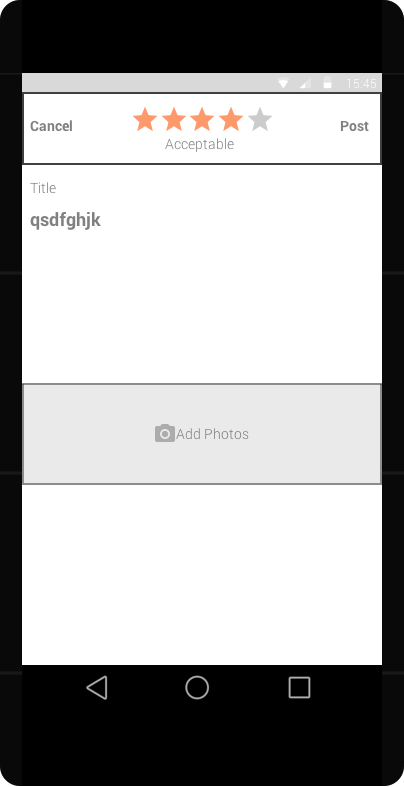


Figure8.4 This user interface show a add photos feature which consist in comment page. After select photo and comment user can post that comment by clicking on post button at top right corner.

9. Show map of restaurant via Google Map

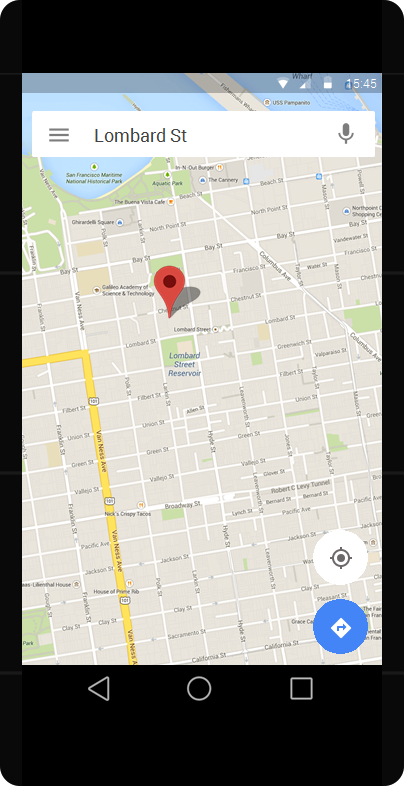


Figure9 This user interface show the page that we link the address of a restaurant to Google Map.

10. Admin management feature which will separate into user management and restaurant management.

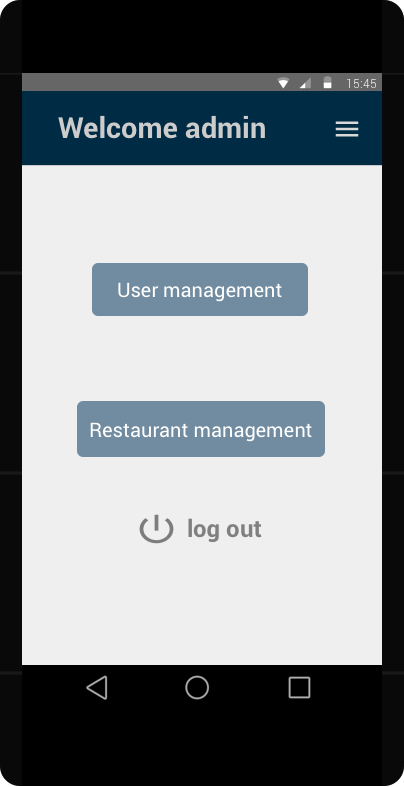


Figure10.1 This user interface shows the first page that admin will see after login as admin already.

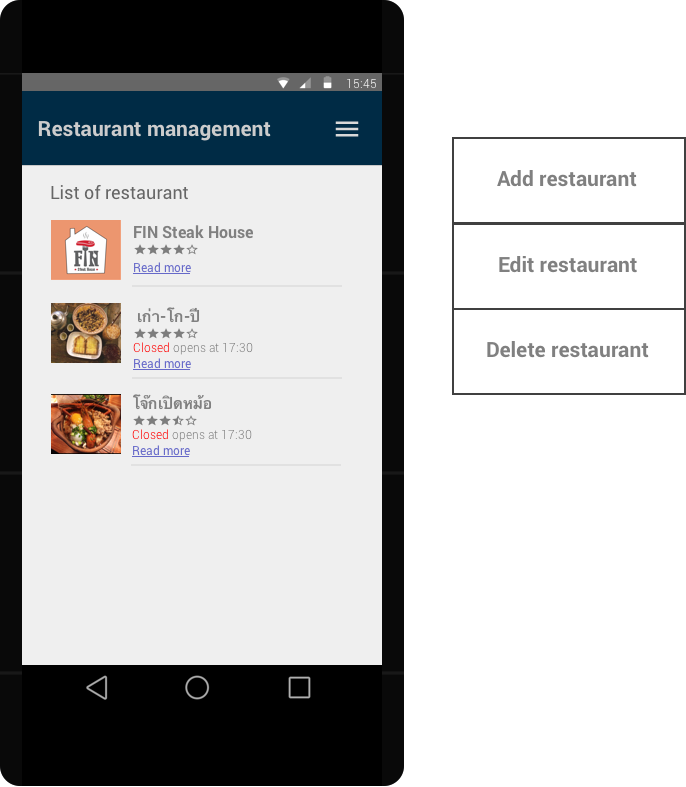


Figure10.2 This user interface show a list of restaurant after admin click on ‘Restaurant management’ button on Figure10.1 and on a menu bar (red circle) on a top right corner will be a list of feature that admin can do with restaurant.

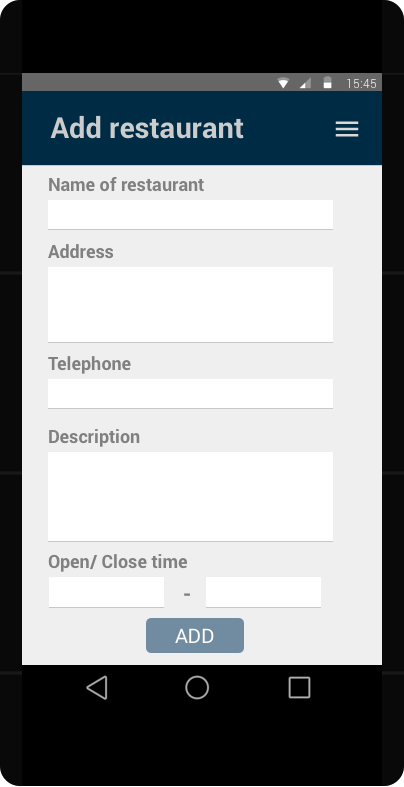


Figure10.3 This user interface show an add restaurant page.

After user finish insert all the information, user be able to click ‘ADD’ button to complete adding process.

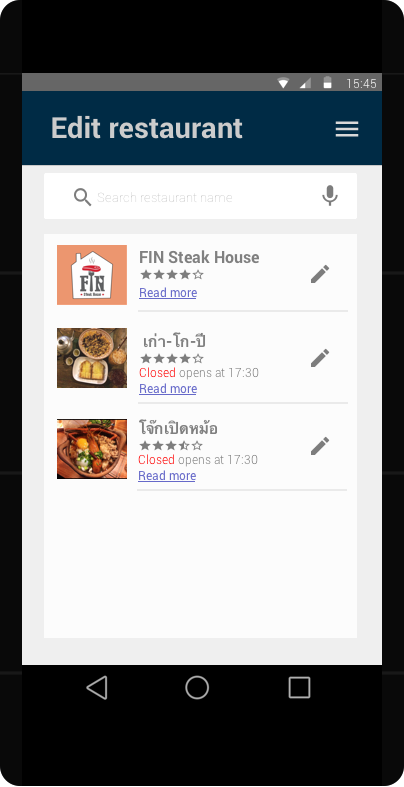


Figure10.4 This user interface show a list of restaurant that let a user choose and edit them.

Another edit page after we click on edit symbol (red circle) will be show in this figure.

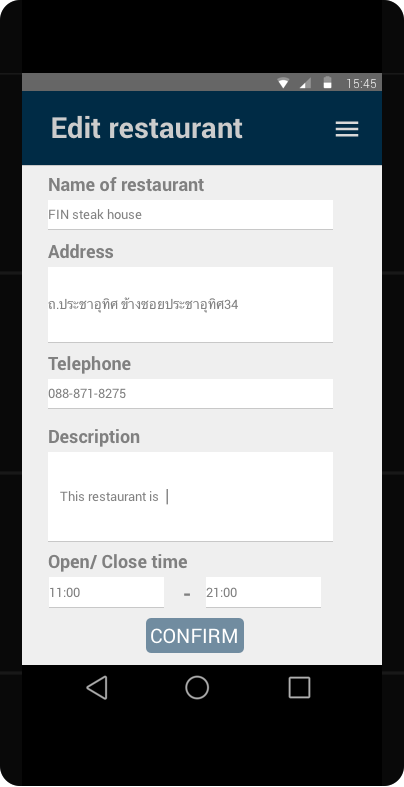


Figure10.5 This user interface show an edit page.

After finish edit restaurant, user have to click on ‘CONFIRM’ button to complete editing.

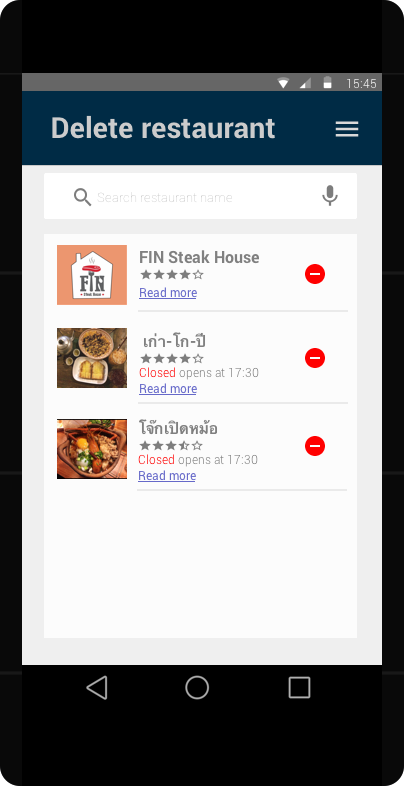


Figure10.6 This user interface show a list of restaurant that provide for admin to delete.

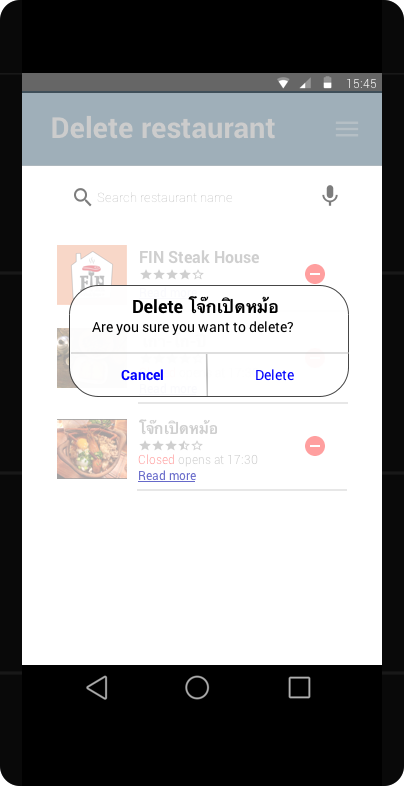


Figure10.7 This user interface show a delete confirmation. If user want to confirm delete process

they have to click ‘Delete’ in a popup message.

11. User management

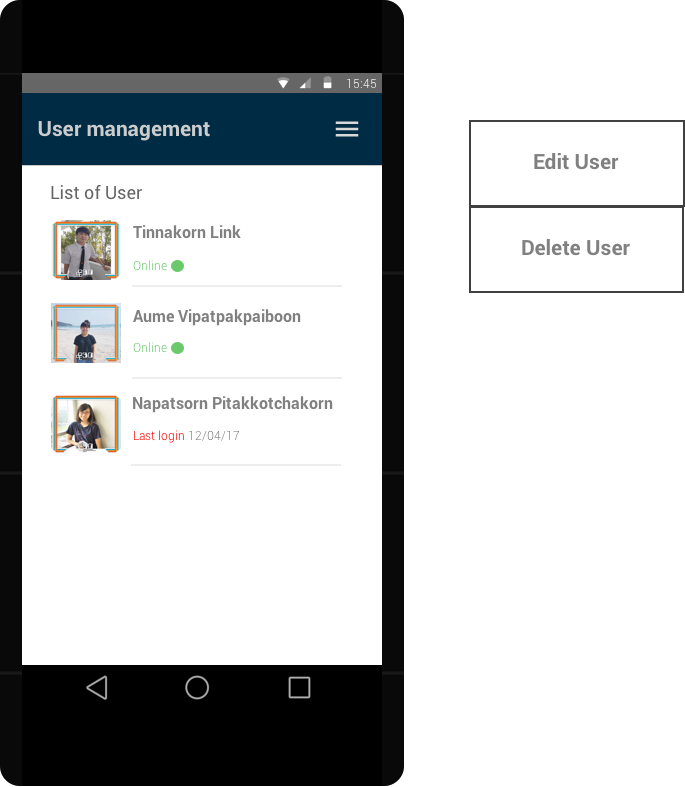


Figure11.1 This user interface show a list of user after admin click on ‘User management’ button on Figure10.1 and on a menu bar (red circle) on a top right corner will be a list of feature that admin can do with user.

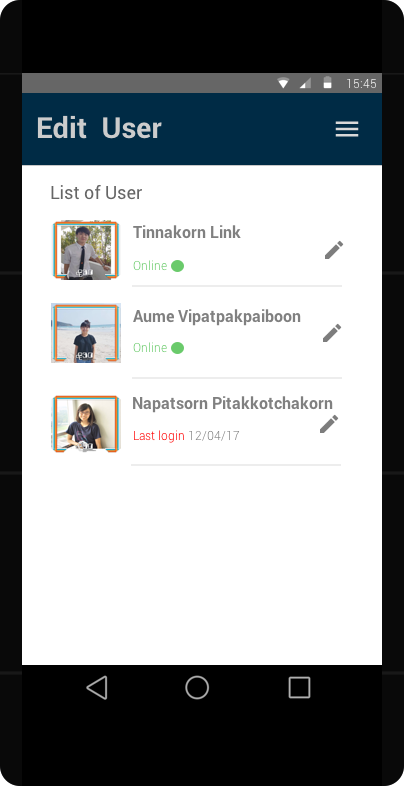


Figure11.2 This user interface show a list of user that let an admin choose and edit them.

Another edit page after we click on edit symbol (red circle) will be show in this figure.

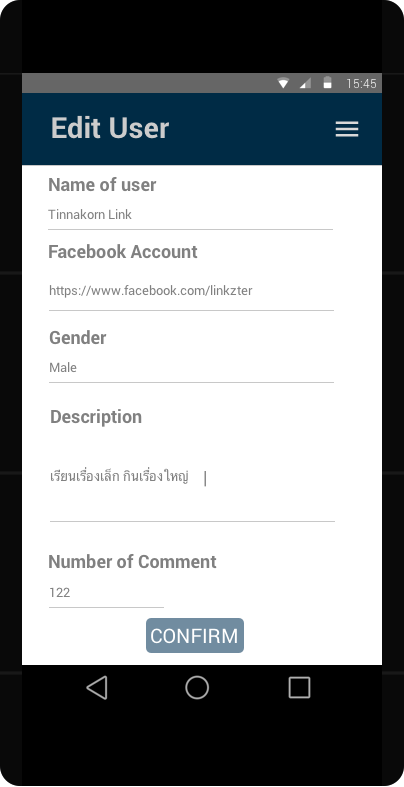


Figure11.3 This user interface show an edit page.

After finish edit user information, admin have to click on ‘CONFIRM’ button to complete editing.

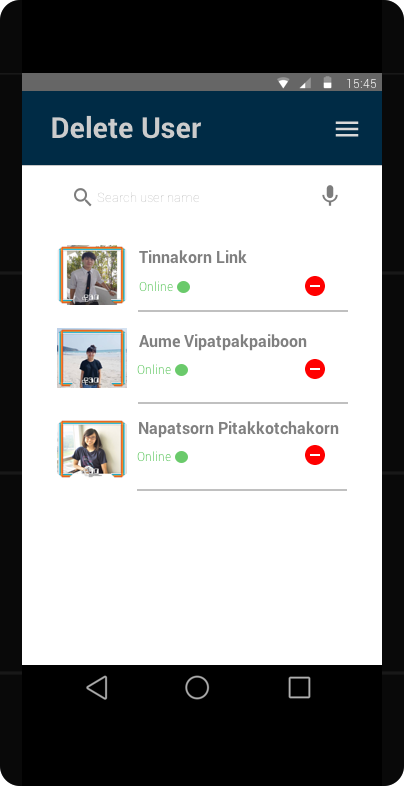


Figure11.4 This user interface show a list of user that provide for admin to delete.

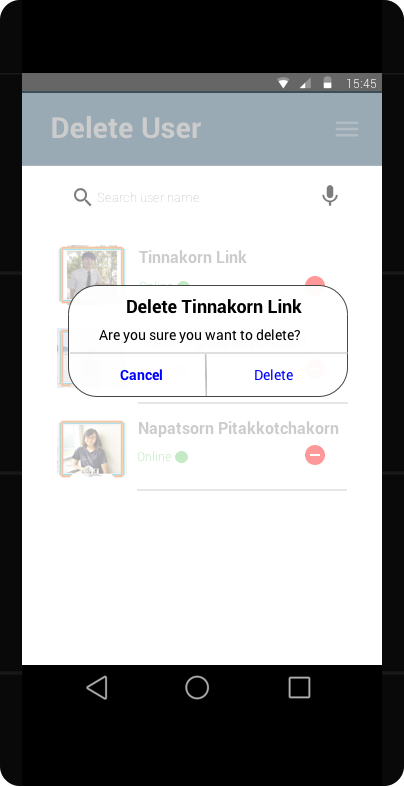


Figure115 This user interface show a delete confirmation. If user want to confirm delete process

they have to click ‘Delete’ in a popup message.

12. MyList (MyList can be separate into many group depend on each user.) To access into MyList user have to click ‘MyList’Button in Figure3.1

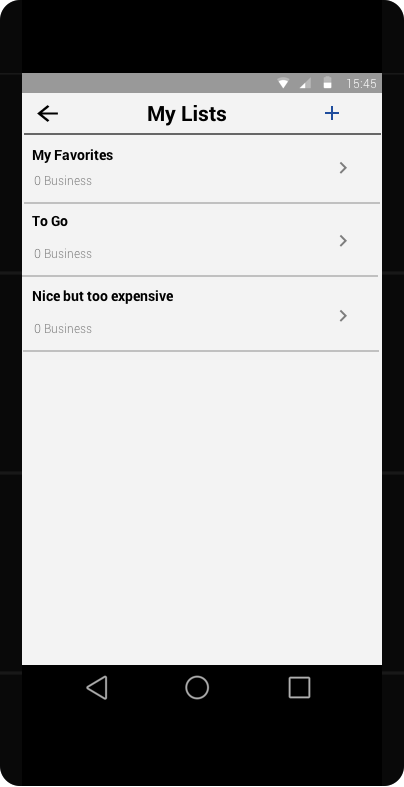


Figure12.1 This user interface show MyList which show all favorite restaurant that user have been added.

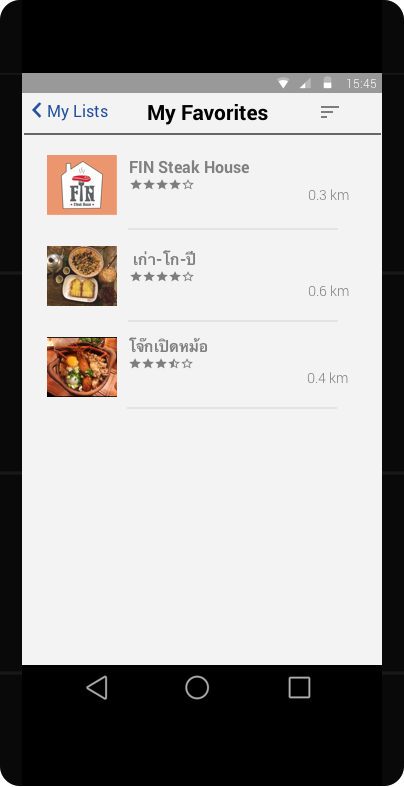


Figure12.2 This user interface show one of a group in MyList which is ‘My Favorites’ from Figure12.1.

In ‘My Favorites’ will show all restaurant that user like and add them in this group.

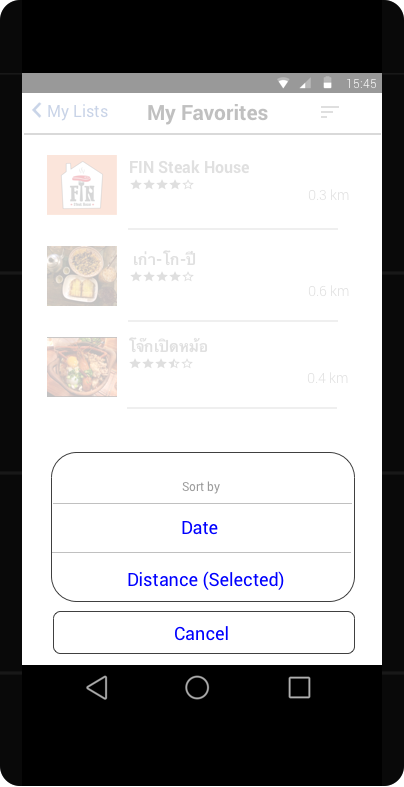


Figure12.3 This user interface show feature sort by the restaurant in ‘My Favorites’ which are Date and Distance.

**6) Other interfaces design**

**6.1) Hardware interfaces design**

- Smart phone & tablet to open application

- Database Server for service data and information by connecting to internet.

- Have to use on Android Platform only

**6.2) Software interfaces design**

- Facebook When user login has to use Facebook account.

- Google Maps for showing direction to restaurant.

**6.3) Communication interfaces design**

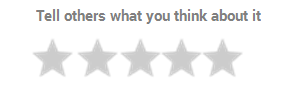
- Email used for reporting a problem of application to Admin and give suggestion

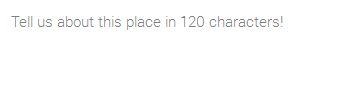
of new restaurant to admin so that admin will add to system.

**7) How to use for each scenario**

**Scenario 1: Rating with comment**

1. Tap on a star to rate

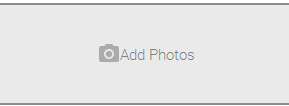




1. Add comment on



1. click button

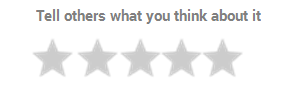
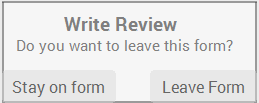


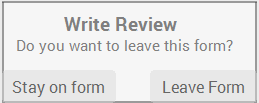
1. Give Title and Add Photo



1. Finally, click

**Scenario 2: Rating without comment**

1. Tap on a star to rate
2. It will go to comment page. If you want to rate only but not want to comment.
3. You have to press button.
4. They will show this pop up



1. You have to click button.
2. Finally, rate will be send.

 **Scenario 3: User Login if user has already login in Facebook app.**

1. Press on button.

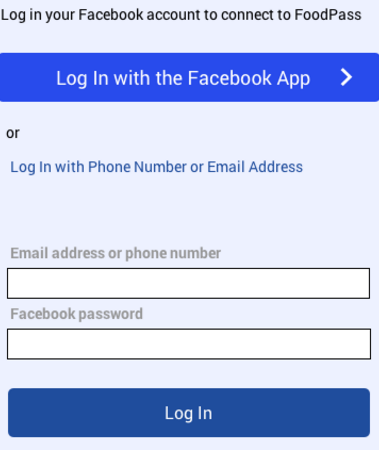
2. Use Facebook account as an application account by button.

**Scenario 4: User Login if user hasn’t login in Facebook app or don’t have an account.**

1. Press on button.

2. If user has an account but user hasn’t login in Facebook app, user have to login on below page.

3. If user didn’t have an account, user have to create an account first.



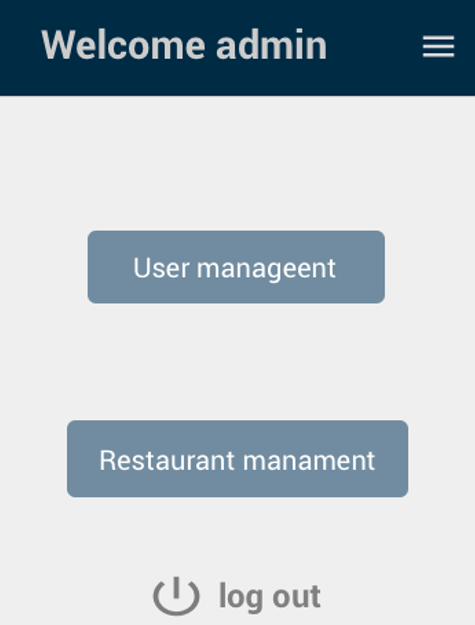
 **Scenario 5: Admin login**

1. Press on button.
2. Insert Username and password of admin to login





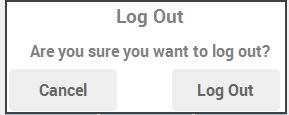
1. Finally, press button.
2. It will go to this page



**Scenario 6: User log out**

1. When user want to log out have to press  button.

2. Then user have to confirm that user want to logout or not.

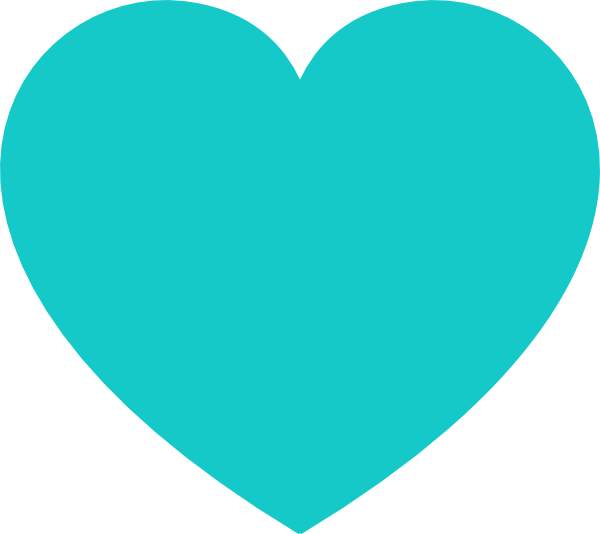




3. If press button mean user want to logout.

**Scenario 7: Add to MyList**

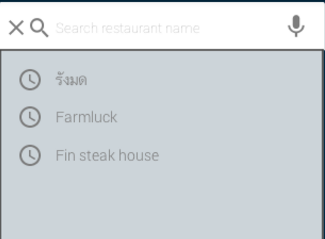
1. In the restaurant page, if user feel interesting on the restaurant they can save it on MyList page by press button.



1. After the restaurant has been press it will be

 **Scenario 8: Search for a restaurant**

1. In the recommend page, User have to tap on
2. It will lead to Search page so user can search for a restaurant on this page.



 **Scenario 9: Admin manage restaurant**

1. In admin welcome page, Press button.
2. Go to restaurant management page



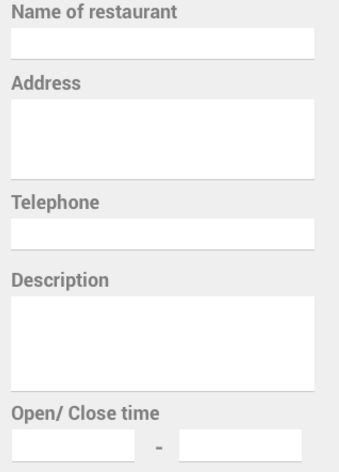
1. Press on the right corner of restaurant management page to see menu of management restaurant.



1. Admin will see

 **Scenario 10: Admin want to add restaurant.**

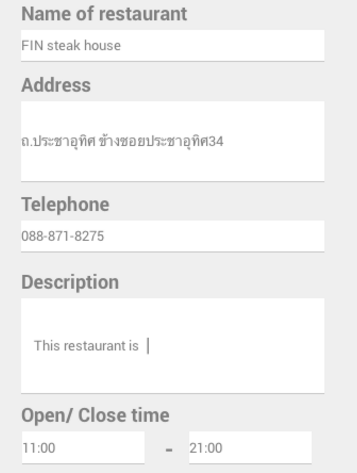
1. Press on restaurant management page
2. Add restaurant detail on add restaurant page.



3. After admin filled in all information, Press to complete.

**Scenario 11: Admin want to edit restaurant information.**

1. Press on restaurant management page
2. Go to edit restaurant page.
3. Press to edit a restaurant that you want to edit.
4. Go to that restaurant information page and edit it.



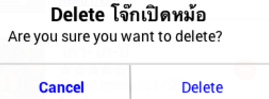
1. After you have already edit, press button to complete.

 **Scenario 12: Admin want to delete restaurant information.**

1. Press on restaurant management page
2. Go to delete restaurant page.



1. Press to delete a restaurant that you want to delete.
2. Then admin have to confirm that admin want to delete this restaurant or not.



 **Scenario 13: Admin manage User**

1. In admin welcome page, Press button.
2. Go to restaurant management page
3.  Press on the right corner of restaurant management page to see menu of

management restaurant.

Admin will see



**Scenario 14: Admin want to edit User information.**



1. Press on user management page

2. Go to edit user page.

3. Press to edit a user that you want to edit.

4. Go to that user information page and edit it.





5. After you have already edit, press button to complete.

** Scenario 15: Admin want to delete User information.**

1. Press on user management page
2. Go to delete user page.



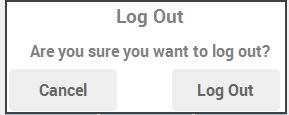
1. Press to delete user that you want to delete.
2. Then admin have to confirm that admin want to delete this restaurant or not.



1. If press delete button, User will be delete.

 **Scenario 16: Admin logout**

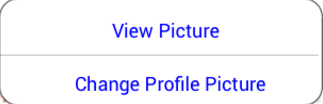
1. In admin welcome page, When admin want to log out have to press button.
2. Then Admin have to confirm that Admin want to logout or not.





1. If press button mean Admin want to logout.

**Scenario 17: User change photo**

1. User have to click a profile picture.
2. 

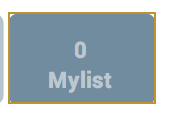
This menu will be pop up



1. If you press the application will automatically link to your photo library so you can choose your image.

**Scenario 18: Check your Mylist**

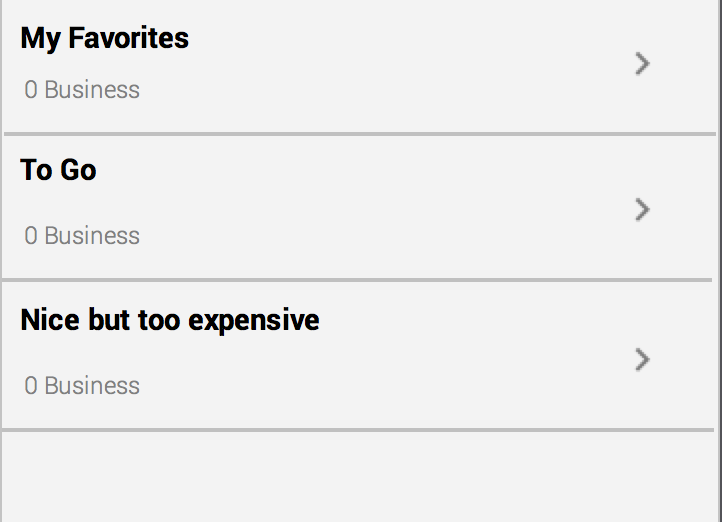
1. When the user press in the recommend page, it will redirect to profile page.



1. After press it will go to see My list page which is shown the folder of

your favorite restaurants.

1. You can pick one of the folder to see the list of restaurant inside.





1. Or you can press to add a new folder in MyList.
2. When you press one of the folder it will show the list of restaurant that you have saved.

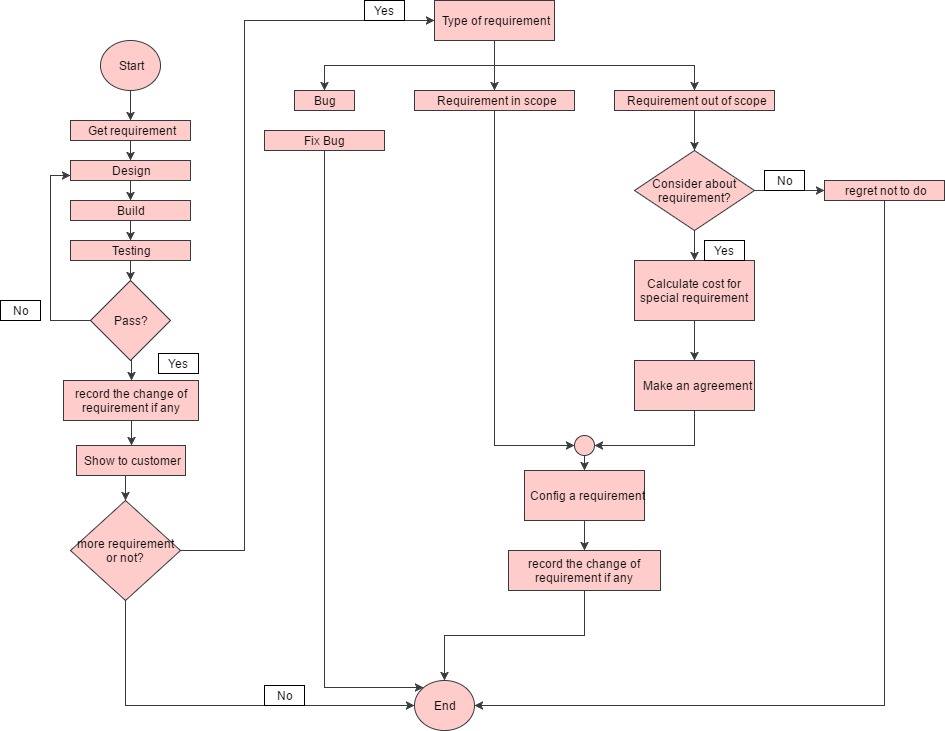


1. If you swipe to the left it will ask you to delete that specific restaurant out of the folder

 **Scenario 19: Go to google map to see location of a restaurant.**

1. In restaurant page, user have to press
2. It will link to google map.

**4) Requirements and change management process**



**Initial work –** The step after we get a requirement will be like this:

1. **Design** – In this step, we will bring a requirement that we already have into a design state to design every process about that requirement or about the application for example design the user interface (theme/style, background, icon in the application), component and function in the application, class, and also about the time management to complete design process in time.

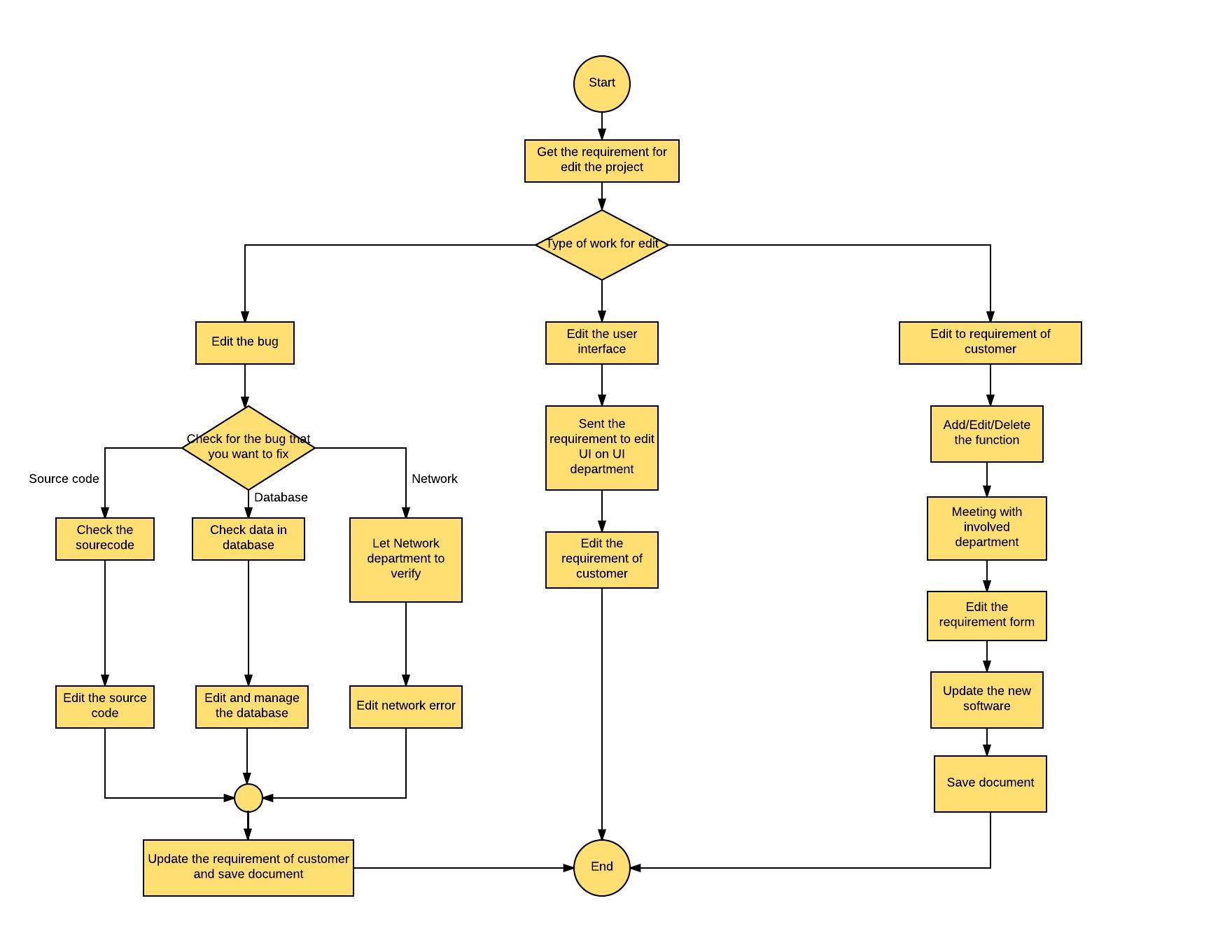
2. **Build –** After we finish design all user interface, component and class, we will bring all that information together and build them to be an application.

3. **Implement** – In this step, we will implement the application that we already build to. In this step, we also edit and solve the problem that has been sent to the developer too.

4. **Test** – In this state, we will test an application that we already build. This step is the step that use to check the correctness of the application. The tester will test from the test case that we already plan. If the test case is pass, the next step will be running. But if the test case is fail, we have to go back to step 2 and 3 to solve that problem from the test case situation.

5. **Record** – If the test case is pass, the record state will record the details about that test case. So, it will be easy in the future when we want to check the history of this application.

**5) Configuration management process**



Consider from Software requirement out of scope or not in this step is the step which our customer or employee have to filled in to improve software so that we can add new requirement or inform about problem of application. For example, Application does not work as good as customer aspect and Application does not work as in requirement. So, company will consider from a form or document that customer has been filled in and verify scope of requirement. Scope have been divided into two categories

1. In scope - After we have been considered a form which customer has been inform. Company have to check for error or Bug

1.1 Fix bug If there is a bug, have to fix immediately.

2.Out of scope - After we have been considered a form which customer has

been inform and the requirement is out of requirement document scope.

Company have to contact a customer and ask customer about adding a

function. If user have confirmed that they want to add a new function,

company will calculate a total expenditure which company have to spend and

increasing working day. Before Company confirm customer about adding new

function, Company have to make a meeting with a stakeholder who involved

for example, Project Manager, System Analyst, Developer and Tester to

confirmed about developing the application and calculate an expenditure that

will involve in this development. If user does not confirm, cancel the form.

**6) Measure for success in timely delivery**

**Table show master schedule (orange color) compare with actual schedule (green color**)

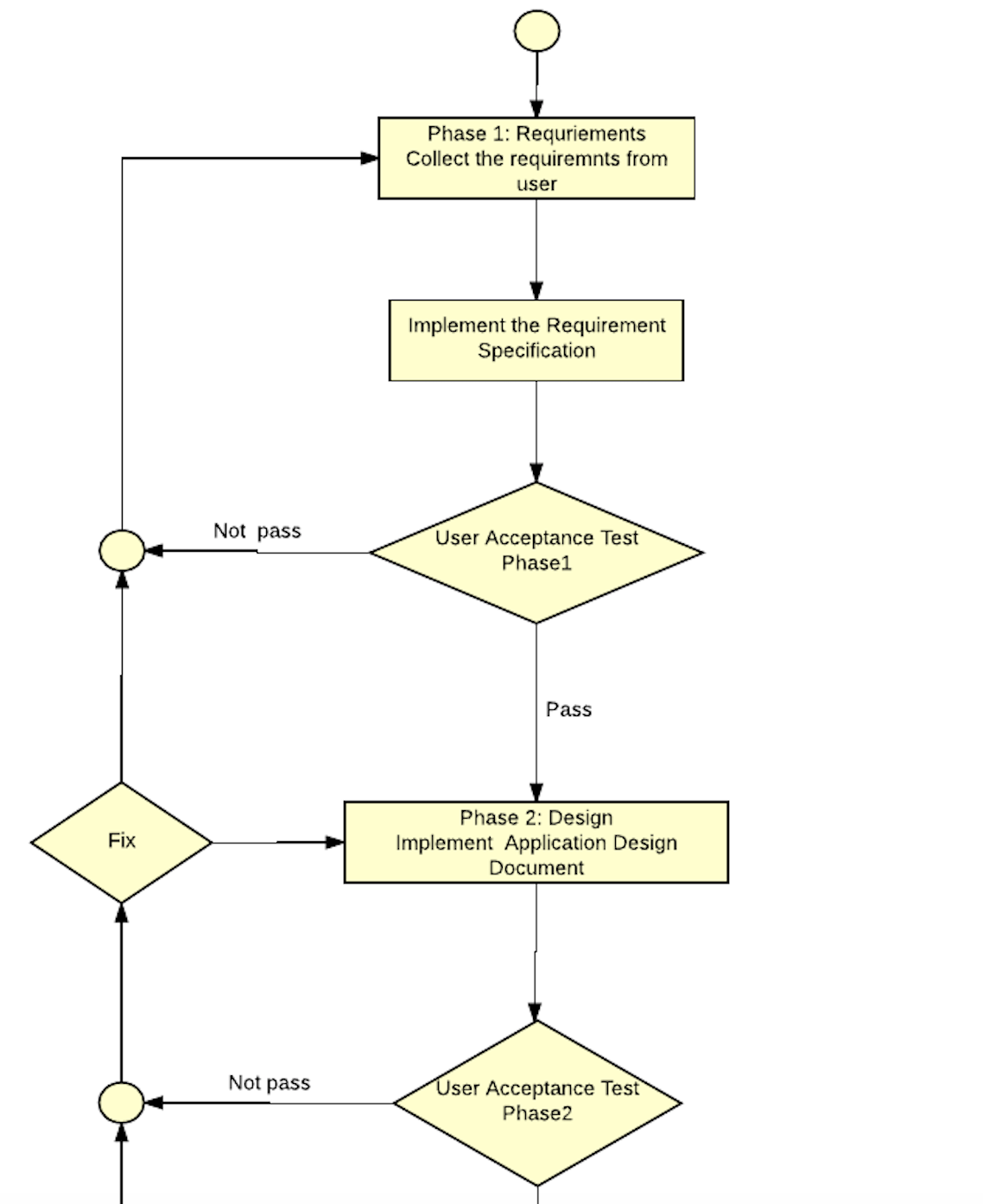
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Description** | **January** | | | | | **February** | | | | | **March** | | | | **April** | | |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** |
| **a) Requirements** | | | | | | | | | | | | | | | | | |
| Meet customer |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Get the requirements from customers. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Make a document from customer’s requirement and present to customers. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Check the document and let the customer make an acceptation. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| **b) Design** | | | | | | | | | | | | | | | | | |
| Create the prototype of this application (template, background, detail of each page, icon) and let the customer make an acceptation. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Design a document of an application design. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |

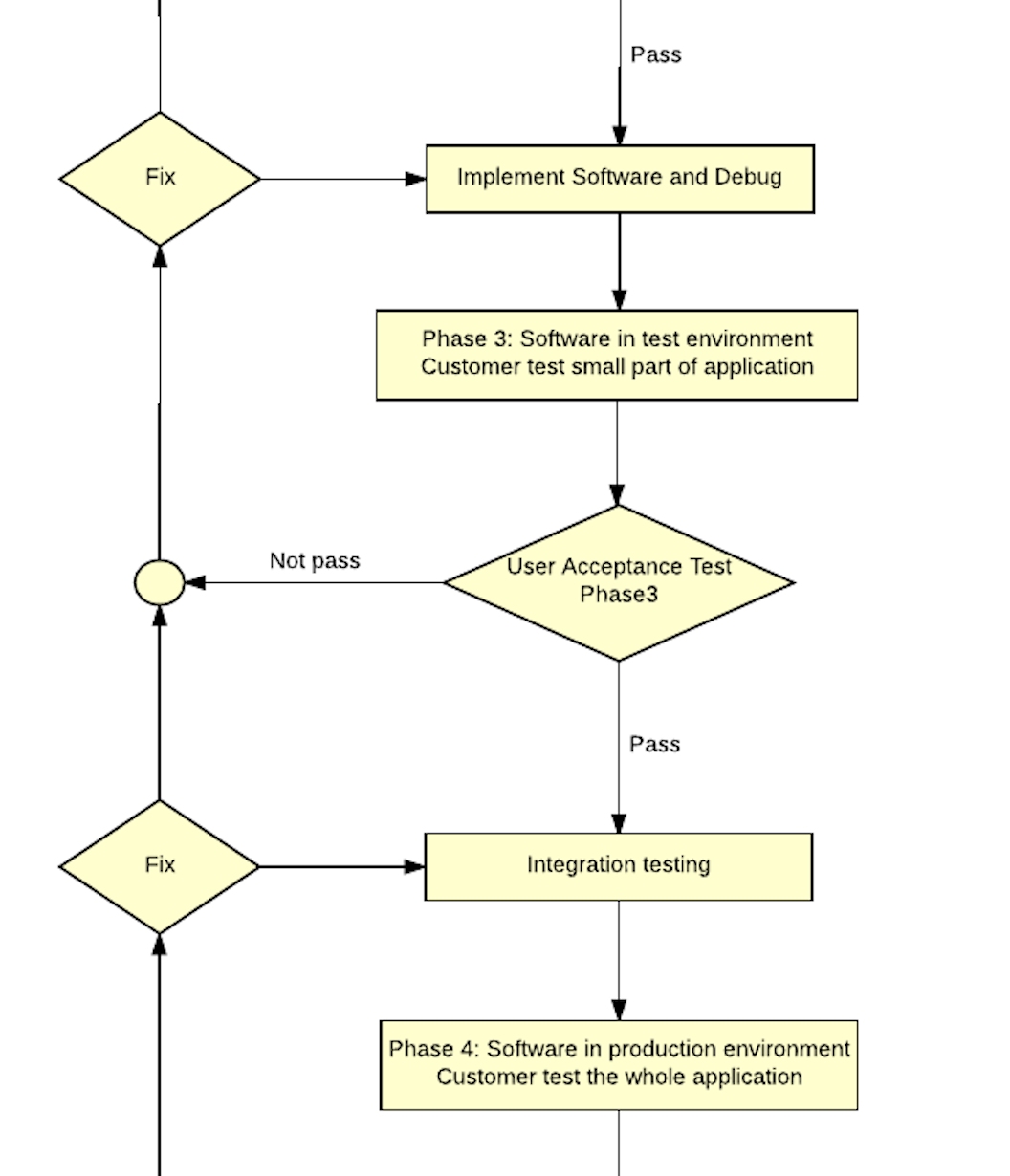
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Description** | **January** | | | | | **February** | | | | | | **March** | | | | **April** | | |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | | | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** |
| **b) Design** | | | | | | | | | | | | | | | | | | |
| Design a presentation (document presentation) of the application. |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| **c) Programming** | | | | | | | | | | | | | | | | | | |
| Create and link database with the application (store the restaurant’s information, user’s information, admin’s information) |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Sign up via Facebook (for user) |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Log in process via Facebook (for user) |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Log in process (for admin) |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Get data from source outside the application. (Facebook’s information) |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Page for admin to add/edit/delete user’s account. |  |  |  |  |  |  |  |  |  | | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | | |  |  |  |  |  |  |  |
| Page for admin to add/edit/delete restaurant’s information |  |  |  |  |  |  |  |  |  | | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |

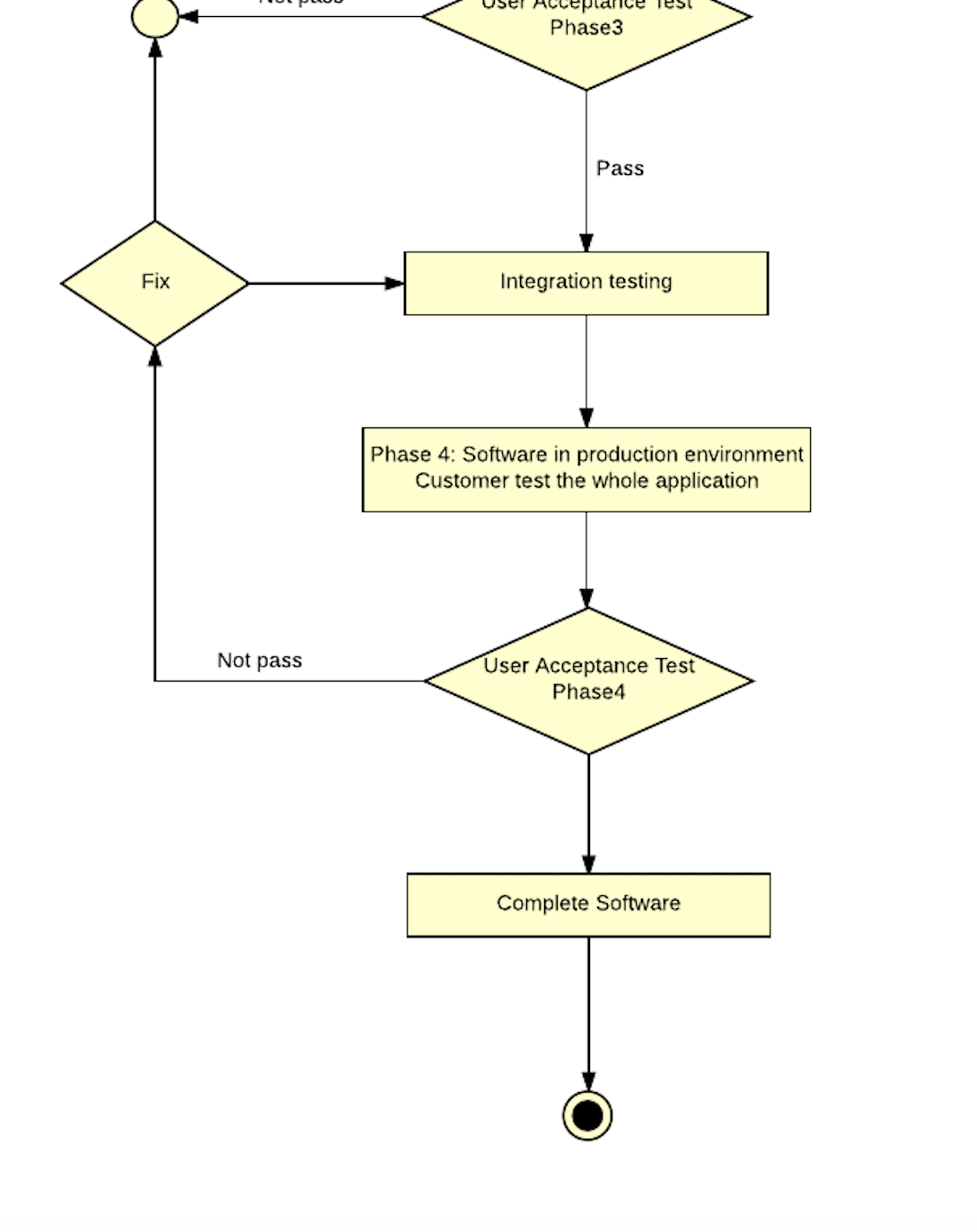
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Description** | **January** | | | | | **February** | | | | | **March** | | | | **April** | | |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** |
| **c) Programming** | | | | | | | | | | | | | | | | | |
| Page for show a list of user. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Page that show a list of restaurant. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Page to show each restaurant’s information |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Profile page for user. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Function for user to edit their photo (profile information). |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Recommendation page |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Find nearby restaurant function |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Function search restaurant by their name. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Categories for restaurant |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Link Google map with an address of a restaurant. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| Function for user to rate the restaurant. |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Description** | **January** | | | | | **February** | | | | | | **March** | | | | **April** | | |
| **1** | **2** | **3** | **4** | **5** | **6** | **7** | **8** | | | **9** | **10** | **11** | **12** | **13** | **14** | **15** | **16** |
| **c) Programming** | | | | | | | | | | | | | | | | | | |
| Function for user to write a comment and post photos. |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Function for user to look back to their history. |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | | |  |  |  |  |  |  |  |
| MyList to store the favorite restaurants |  |  |  |  |  |  |  |  |  | | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | | |  |  |  |  |  |  |  |
| Function to let user select their favorite restaurant. |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| **d) Test Preparation** | | | | | | | | | | | | | | | | | | |
| Create test plan |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Check test plan |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Edit/rewrite some test plan |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| **e) Test Execution** | | | | | | | | | | | | | | | | | | |
| Test a test plan |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Solve some bugs/ defects |  |  |  |  |  |  |  |  |  | | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  | | |  |  |  |  |  |  |  |
| Retest a test plan |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
| Deliver Beta Build |  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  | |  | |  |  |  |  |  |  |  |

**7) User acceptance process**

 The company needs to show every step of the implementation of the software exactly and it should be traceable which the company could be use it as the reference when the customer need it. For example, the customer doesn’t accept our software that is finished as same as the requirement specification that the customer wanted. The company can show the evidence (Document that the customer has signed) to ensure that I already agreed with this requirement. The method of doing the user acceptance process is shown below.





**Acceptance Test Form**

USER ACCEPTANCE TEST

Project’s Name:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Form’s Name/Report/Module:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Version:\_\_\_\_\_\_\_\_\_\_\_\_\_

Detail of Form/ Report/ Module

|  |  |  |  |
| --- | --- | --- | --- |
| Template of Form/ Report/ Module | | | |
| # | Detail of function | # | Detail of function |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Accept Reason:

Reject

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

( ………………………………….. ) ( ………………………………….. )

Project Owner Project Manager

….../……/…… ….../……/……

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

( ………………………………….. ) ( ………………………………….. )

Project Owner Project Manager

….../……/…… ….../……/……

An Example

USER ACCEPTANCE TEST

Project’s Name:\_\_\_\_\_\_\_\_\_FoodPass\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Date:\_\_\_\_20/04/17\_\_\_\_\_\_

Form’s Name/Report/Module:\_\_\_\_\_\_\_\_\_Design\_\_\_\_\_\_\_ Version:\_\_\_\_\_1.2\_\_\_\_\_\_

Detail of Form/ Report/ Module

|  |  |  |  |
| --- | --- | --- | --- |
| Template of Form/ Report/ Module | | | |
| # | Detail of function | # | Detail of function |
| 1 | Add “New” button in MyList page |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Accept Reason: To make it more reliable

Reject

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

( ……Sarah…Greyham….. ) ( …Tinnakorn…Marlaithong….. )

Project Owner Project Manager

….21../…04…/…17… ….21../…04…/…17…