Contents

[Model: 2](#_Toc420568794)

[ Business 2](#_Toc420568795)

[ Comment 3](#_Toc420568796)

[ Friend 4](#_Toc420568797)

[ Post 4](#_Toc420568798)

[ Review 5](#_Toc420568799)

[ Search 6](#_Toc420568800)

[ User 7](#_Toc420568801)

[View 9](#_Toc420568802)

[ Activity 9](#_Toc420568803)

[ Adapter 10](#_Toc420568804)

[ Dialog 11](#_Toc420568805)

[ Fragment 12](#_Toc420568806)

Layers:

* Model
* View
* Controller

## Model:

* Business:
  + BlockUser: It is an HttpGet webservice. A business owner blocks a user. Blocked user cannot write a comment on a post which belongs to the business.

Input: BUSINESS\_ID: int (business’ id who owns the post)

Output: ResultStatus

* + DeleteBusiness: It is an HttpGet webservice. A business owner can delete his own business.

Input: BUSINESS\_ID: int (the deleting business), USER\_ID: int (the business owner)

Output: ResultStatus

* + GetBlockedUsers: It is an HttpGet webservice. It returns blocked users of a business.

Input: BUSINESS\_ID: int (the business’ id), BEFORE\_THIS\_ID: int (is used to load more data), LIMITATION: int (is used to load more data)

Output: ArrayList<BaseAdapterItem>

* + GetBusinessContactInfo: It is an HttpGet webservice. It returns a business contact info.

Input: BUSINESS\_ID: int (the business’ id)

Output: Business

* + GetBusinessFollowers: It is an HttpGet webservice. It returns the list of a business’ followers.

Input: BUSINESS\_ID: int (the business’s id)

Output: ArrayList<BaseAdapterItem>

* + GetBusinessGategories: It is an HttpGet webservice. It returns categories of the business.

Input: null

Output: ArrayList<Category>

* + GetBusinessHomeInfo: It is an HttpGet webservice. It returns a business’ home information e.g. profile picture id.

Input: BUSINESS\_ID: int (the business’ id), USER\_ID (the business owner’s id)

Output: Business

* + GetBusinessProfileInfo: It is an HttpGet webservice. It returns a business’ profile information e.g. address.

Input: BUSINESS\_ID: int (the business’ id)

Output: Business

* + GetBusinessSubcategories: It is an HttpGet webservice. It returns subcategories of the business.

Input: CATEGROY\_ID: int (the selected category’s id)

Output: ArrayList<SubCategory>

* + RateBusiness: It is an HttpGet webservice. A user can rate a business with an integer value between 1 and 5.

Input: BUSINESS\_ID: int (the business’ id), USER\_ID: int (the user’s id), RATE: int

Output: ResultStatus

* + RegisterBusiness: It is an HttpPost webservice. A user can create a new business.

Input: Business

Output: BUSINESS\_ID: int (registered business’ id)

* + UnblockUser: It is an HttpGet webservice. A business owner can unblock a blocked user.

Input: BUSINESS\_ID: int (the business’ id), USER\_ID: int (the business owner’s id)

Output: ResultStatus

* + UpdateBusinessProfileInfo: It is an HttpPost webservice. A business owner can edit his business profile information.

Input: Business

Output: ResultStatus

## Comment

* + DeleteComment: It is an HttpGet webservice. A business owner can delete a user’s comment on his own business’ posts. The business owner is responsible to control comments on his posts.

Input: BUISNESS\_ID: int (the business’ id), COMMENT\_ID: int (the comment’s id)

Output: ResultStatus

* + GetAllCommentNotifications: It is an HttpGet webservice. It returns all comments of all posts which belongs to a business. If a user have more than one business, this webservice returns all his businesses’ notifications.

Input: USER\_ID: int (the user’s id), BEFORE\_THIS\_ID: int (is used to load more data), LIMITAION: int (is used to load more data)

Output: ArrayList<CommentNotification>

* + GetLastCommentNotification: It is an HttpGet webservice. It returns last comment of all posts which belongs to one of the user’s businesses.

Input: USER\_ID: int (the user’s id)

Output: CommentNotification

* + GetPostAllComments: It is an HttpGet webservice. It returns all comments of a post.

Input: POST\_ID: int (the post’s id), BEFORE\_THIS\_ID: int (is used to load more data), LIMITATION: int (is used to load more data)

Output: ArrayList<Comment>

* + SendComment: It is an HttpGet webservice. A user can comment on a post.

Input: USER\_ID: int (the user’s id), POST\_ID: int (the post’s id), COMMENT: string (the comment’s text)

Output: COMMENT\_ID: int (the sent comment’s id)

* + UpdateComment: It is an HttpGet webservice. A user can edit his own comment

Input: USER\_ID: int (the user’s id), COMMENT\_ID: int (the comment’s id), COMMENT\_TEXT: string (the new comment’s text)

Output: ResultStatus

## Friend

* + AnswerRequestFriendship: It is an HttpGet webservice. A user can answer another user’s friend request with Yes or No.

Input: REQUEST\_USER\_ID: int (the id of the user whom the request is sent for him), APPLICATOR\_USER\_ID: int (the id of the user whom sent the request),

ANSWER: boolean (true means Yes and false means No)

* + GetUserFriendRequests: It is an HttpGet webservice. It returns all the user’s friend request.

Input: USER\_ID: int (the user’s id)

Output: ArrayList<BaseAdapterItem>

* + GetUserFriends: It is an HttpGet webservice. It returns a list of the user’s friends.

Input: USER\_ID: int (the user’s id)

Output: ArrayList<BaseAdapterItem>

* + RequestCancelFriendship: It is an HttpGet webservice. A user can cancel the friendship relation with another user.

Input: APPLICATOR\_USER\_ID: int (the user’s id who wants to cancel the friendship), REQUESTED\_USER\_ID: int (the user’s id who’s the other user wants to cancel his friendship with him)

Output: ResultStatus

* + RequestFriendship: It is an HttpGet webservice. A user can send a friendship request for another user. A user can see his friend shared post in his Time Line or he can see his friend’s following businesses and etc.

Input: APPLICATOR\_USER\_ID: int (the user’s id who wants to send the friendship request), REQUESTED\_USER\_ID: int (the user’s id who’s the other user wants to be friend with)

Output: ResultStatus

## Post

* + AddPost: It is an HttpPost webservice. A business owner can add a new post.

Input: Post

Output: Post (includes the new post information plus post’ id)

* + CancelShare: It is an HttpGet webservice. A user can cancel the sharing of a post.

Input: USER\_ID: int (the user’s id), POST\_ID: int (the post’s id)

Output: ResultStatus

* + DeletePost: It is an HttpGet webservice. A business owner can delete one of his own posts.

Input: BUSINESS\_ID: int (the business’ id), POST\_ID: int (the post’s id)

Output: ResultStatus

* + GetBusinessPosts: It is an HttpGet webservice. It returns all posts of a business.

Input: USER\_ID: int (the id of the business owner), BUSINESS\_ID: int (the id of the business), POST\_ID: int (the post’s id)

Output: ArrayList<Post>

* + GetPost: It is an HttpGet post. It returns a post’s information e.g. post’s title. If the user who sees the post is the owner of the post, he can delete or edit the post or watch the comments and then delete some comments. But if not, he can like or share the post or just watch the comments.

Input: USER\_ID: int (the id of the business owner), POST\_ID: int (the post’s id)

Output: Post

* + GetSharedPosts: It is an HttpGet webservice. It returns posts which a user shared.

Input: BEFORE\_THIS\_ID: int (is used to load more data), LIMITATION: int (is used to load more data)

Output: ArrayList<Post>

* + GetTimeLinePosts: It is an HttpGet webservice. It returns a user’s Time Line posts.

Input: USER\_ID: int (the user’s id), BEFORE\_THIS\_ID: int (is used to load more data), LIMITAION: int (is used to load more data)

Output: ArrayList<Post>

* + Like: It is an HttpGet webservice. A user can like a post.

Input: USER\_ID: int (the id of the business owner), POST\_ID: int (the post’s id)

Output: ResultStatus

* + Report: It is an HttpGet webservice. A user can report a post as inappropriate post.

Input: USER\_ID: int (the id of the business owner), POST\_ID: int (the post’s id)

Output: ResultStatus

* + Share: It is an HttpGet webservice. A user can like a post.

Input: USER\_ID: int (the id of the business owner), POST\_ID: int (the post’s id)

Output: ResultStatus

* + Unlike: It is an HttpGet webservice. A user can unlike a liked post.

Input: USER\_ID: int (the id of the business owner), POST\_ID: int (the post’s id)

Output: ResultStatus

* + UpdatePost: It is an HttpPost webservice. A business owner can edit a post’s information.

Input: Post

Output: Post

## Review

* + DeleteReview: It is an HttpGet webservice. A user how wrote a review for a business, can delete his own review.

Input: REVIEW\_ID: int (the review’s id), USER\_ID: int (the user’s id)

Output: ResultStatus

* + GetBusinessReviews: It is an HttpGet webservice. It returns a list of a business’ reviews.

Input: BUSINESS\_ID: int (the business’ id), BEFORE\_THIS\_ID: int (is used to load more data), LIMITAION: int (is used to load more data)

Output: ArrayList<Review>

* + GetUserReviews: It is an HttpGet webservice. It returns a list of reviews which a user wrote.

Input: USER\_ID: int (the user’s id), BEFORE\_THIS\_ID: int (is used to load more data), LIMITAION: int (is used to load more data)

Output: ArrayList<Review>

* + ReviewBusiness: It is an HttpGet webservice. A user can review a business. He can write a text and rate the business with a value between 1 and 5.

Input: USER\_ID: int (the user’s id), BUSINESS\_ID: int (the business’ id), REVIEW: string (the review’s text), RATE: int (the review’s rate)

Output: ResultStatus

* + UpdateReview: It is an HttpGet webservice. A user can edit his own reviews.

Input: USER\_ID: int (the user’s id), BUSINESS\_ID: int (the business’ id), REVIEW: string (the review’s text), RATE: int (the review’s rate)

Output: ResultStatus

## Search

* + SearchBusinessesLocation: It is an HttpGet webservice. A user can search a business by specify its name, subcategory and a location. The result will be sorted base on the distance of the specified location. The nearest businesses will be on the top.

Input: SEARCH\_TEXT: string (the business’ name), SUB\_CATEGORY\_ID: int (the business’ subcategory id), LOCATION\_LATITUDE: string (the location’s latitude), LOCATION\_LONGITUDE: string (the location’s longitude), BEFORE\_THIS\_ID: int (is used to load more data), LIMITAION: int (is used to load more data)

Output: ArrayList< SearchItemBusiness>

* + SearchBusinessesNearBy: It is an HttpPost webservice. A user can search a business by specify its name, and a place like a city. This webservice has NOT used.

Input: USER\_ID: int (the user’s id), SEARCH\_TEXT: string (the business’ name), NEAR\_BY: string (the place name)

Output: ArrayList< BaseAdapterItem>

* + SearchPost: It is an HttpGet webservice. A user can search a post by its title or hashtags. The hashtags will help the system for better searching.

Input: SEARCH\_TEXT: string (the post’s title or hashtag), BEFORE\_THIS\_ID: int (is used to load more data), LIMITAION: int (is used to load more data)

Output: ArrayList< SearchItemPost>

* + SearchUser: It is an HttpGet webservice. A user can search other users by their identifier. The identifier is an English phrase e.g. “ali124Hooo”

Input: SEARCH\_TEXT: string (the other user’s identifier), BEFORE\_THIS\_ID: int (is used to load more data), LIMITAION: int (is used to load more data)

Output: ArrayList< BaseAdapterItem>

## User

* + FollowBusiness: It is an HttpGet webservice. A user can follow a business. He will see the business’ future posts in his Time Line.

Input: USER\_ID: int (the user’s id), BUSINESS\_ID: int (the business’ id),

Output: ResultStatus

* + ForgetPassword: It is an HttpGet webservice. If a user forget his password, he can recover his password by his email.

Input: EMAIL: string (the user’s email address)

Output: ResultStatus

* + GetFollowingBusinesses: It is an HttpGet webservice. It returns a list of businesses which a user is following.

Input: USER\_ID: int (the user’s id)

Output: ArrayList< BaseAdapterItem>

* + GetUserHomeInfo: It is an HttpGet webservice. It returns a user’s home information. The information will be displayed differently if user watches his own home page or another user.

Input: VISITED\_USER\_ID: int (The id of a user who the user is watching his home), VISITOR\_USER\_ID: int (The id of the user who is watching a user home)

Output: User

* + GetUserProfileInfo: It is an HttpGet webservice. It returns a user’s profile information e.g. “about me”

Input: USER\_ID: int (the user’s id)

Output: User

* + Login: It is an HttpGet webservice.

Input: EMAIL: string (the user’s email address), PASSWORD: string (the user’s account password)

Output: ResultStatus

* + RegisterUser: It is an HttpPost webservice. The user’s identifier and his email address should be unique. After the successful registration, a confirmation request will send to the user’s email address.

Input: User

Output: ResultStatus

* + RequestConfirmation: It is an HttpGet webservice. If a user didn’t receive the confirmation email, he can request to receive confirmation email again. This webservice has not used.

Input: USER\_ID: int (the user’s id)

Output: ResultStatus

* + UnFollowBusiness: It is an HttpGet webservice.

Input: USER\_ID: int (the user’s id), BUSINESS\_ID: int (the business’ id)

Output: ResultStatus

* + UpdatePassword: It is an HttpPost webservice. This webservice has not used. The password will be updated with another webservice.

Input: USER\_ID: int (the user’s id), NEW\_PASSWORD: string (the new password)

Output: ResultStatus

* + UpdateSetting: It is an HttpGet webservice. There is 3 permission for a user which the user can change them:
    - Followed businesses: the user can let his friends be able to see his following businesses.
    - Friends: the user can let his friends be able to see his friends.
    - Reviews: the user can let his friends be able to see his reviews on businesses.

Input: Permission

Output: ResultStatus

* + UpdateUserProfile: It is an HttpPost webservice. A user can edit his own profile info e.g. “name”.

Input: User

Output: ResultStatus

* DownloadCoverImage: this class uses an HttpGet webservice to download a user’s cover image or a business’ cover image. It takes an ImageView and it will displays the download image in the ImageView. After downloading an image, the image will be cropped to make it a triangle shape image. At the end, the image will be saved into local storage. Next time, the class uses the saved image to display cover. Actually it caches images. If the class failed to download an image, it will display a default image depending on it is a user’s cover or business’ cover.

Input: IMAGE\_ID: int (the image’s id), IMAGE\_VIEW: ImageView, IMAGE\_TYPE: Image\_M.ImageType

* DownloadImages: this class woks like DownloadCoverImage without cropping the image. This class can round the image if it is required. This class can download the image in 3 different sizes.

Input: IMAGE\_ID: int (the image’s id), IMAGE\_SIZE: int (1=large, 2=medium, 3= small), IMAGE\_TYPE: Image\_M.ImageType, IMAGE\_VIEW: ImageView, IS\_ROUNDED: Boolean

* WebserviceGET: all upper HttpGet classes use this class to execute HttpGet requests.

Functions:

* + public ServerAnswer execute(Context context);

It executes the HttpGet URL and returns the HttpResponse result as a ServerAnswer object.

* + public ServerAnswer executeList(Context context)

It works like execute () except we use it to get a list of objects e.g. GetUserFriends

* WebservicePOST: all upper HttpPost classes use this class to execute HttpPos requests.

Functions:

* + public void addParam(String paramName, String paramValue)

It adds a parameter to the JSONObject.

* + public ServerAnswer execute(Context context)
  + public ServerAnswer executeList(Context context)
  + private HttpResponse run(HttpPost httpPost)

Two upper functions (execute () and executeList ()) use this function to execute the HttpPost request. It sets entity and adds headers to the HttpResponse before executing it.

# View

## Activity

* + ActivityBusiness: It displays the user’s business home information e.g. name.
  + ActivityBusinessBlockedUsers: It displays a business’ blocked users. The business owner can remove a user (unblock) from the blocked users list.
  + ActivityBusinessContactInfo: It displays a business’ contact information e.g. email address.
  + ActivityBusinessFollowers: It displays a business’ follower users.
  + ActivityBusinessOther: The user can sees other user’s businesses’ home information in this activity.
  + ActivityBusinessRegisterEdit: A user can register a new business or edit an existing business in this activity.
  + ActivityBusinessReviews: It displays a business reviews.
  + ActivityBusinessWorkTime: A user can set a business’ working days and working times in this activity. If the user is editing the business, he can edit the data.
  + ActivityCamera: A user can take a picture from the camera in this activity.
  + ActivityComments: It displays all comments of a post.
  + ActivityCommentsNotifications. It displays all comments of all posts that belongs to the user’s businesses.
  + ActivityGallery: A user can select an image from his device gallery via this activity.
  + ActivityLogin: A user can login to the app via this activity.
  + ActivityMain: It the main activity of the app. It handles 4 fragments: FragmentHome, FragmentSearch, FragmentBusinesses and FragmentUser.
  + ActivityMapChoose: A user can choose a location on the google map via this activity.
  + ActivityMapDisplay: It displays a location on the google map. It is used to display a business location via ActivityBusinessContactInfo
  + ActivityPost: It displays a post’s information e.g. price. By clicking on the post picture while the posts are displaying as the Gridview, the user will navigate to this activity.
  + ActivityPostAddEdit: A business owner can add a new post or edit an existing post.
  + ActivityProfileUser: It displays the user’s profile information e.g. name. The user can edit his profile information.
  + ActivitySearchBusinessResult: The SearchBusinessesLocation webservice runs in this activity and the result will be displayed here.
  + ActivitySearchPostResult: The SearchPost webservice runs in this activity and the result will be displayed here.
  + ActivitySearchUser: The SearchUser webservice runs in this activity and the result will be displayed here.
  + ActivityUserFollowingBusinesses: It displays a list of the user’s following businesses. If the user is watching his own following businesses, he is able to remove a business from the list.
  + ActivityUserFriendRequests: It displays friend requests of the user. The user can answer Yes or No to the request.
  + ActivityUserFriends: It displays a list of the user’s friends. If the user is watching his own friends, he is able to remove a friend from the list.
  + ActivityUserLoginSignUp: It is not implemented.
  + ActivityUserOther: It displays another user’s home information rather than the current user.
  + ActivityUserRegister: A user who wants to user the application should register first.
  + ActivityUserReviews: It displays a list of reviews which the user wrote. The user can edit or delete a review.
  + ActivityUserSetting: There are three setting options:
    - Permission for shared post: with this permission, the friends of a user can see his shared posts.
    - Permission for following businesses: with this permission, the friends of a user can see his following businesses.
    - Permission for reviews: with this permission, the friends of a user can see his reviews.

## Adapter

* + AdapterBlockedUsers: It is used to display the list of a business’ blocked users.
  + AdapterBusinessFollowers: It is used to display the list of users who follow a business.
  + AdapterBusinessReview: It is used to display a business’ reviews.
  + AdapterBusinessSearchResult: It is used to display the result of search about businesses.
  + AdapterCategories: It is used to display the list of the business’ categories.
  + AdapterCommentNotification: It is used to display the list of comment notifications – all comments of a user’s all posts.
  + AdapterPostBusiness: It is used to display the list of a business’ posts.
  + AdapterPostComments: It is used to display the list of all comments of a post.
  + AdapterPostGrid: It is used to display the list of posts’ pictures in a Gridview.
  + AdapterPostShared: It is used to display the list of a user’s shared posts.
  + AdapterPostTimeLine: It is used to display the list of user’s time line posts.
  + AdapterUserBusinesses: It is used to display the list of user’s businesses.
  + AdapterUserFollowingBusinesses: It is used to display the list of user’s following businesses.
  + AdapterUserFriends: It is used to display the list of a user’s friends.
  + AdapterUserFriendshipRequest: It is used to display the list of a user’s friend requests.
  + AdapterUserReview: It is used to display the list of a user’s review.
  + AdapterUserSearchResult: It is used to display the result of the search about users.

## Dialog

* + DialogAddReview: It extends the MyDialogOkCancel. It takes the user’s review which includes text and rate.
  + DialogBlockUserConfirmation: It extends the MyDialogOkCancel. It displays a message to confirm blocking a user.
  + DialogCancelFriendshipConfirmation: It extends the MyDialogOkCancel. It displays a message to confirm cutting the friendship with a user.
  + DialogChangePassword: It extends the MyDialogOkCancel. It takes the user’s current password, new password and confirm password.
  + DialogClearSearchHistoryConfirmation: It extends the MyDialogOkCancel. It displays a message to confirm clearing the search history.
  + DialogDeleteBusinessConfirmation: It extends the MyDialogOkCancel. It displays a message to confirm removing a business.
  + DialogDeleteCommentConfirmation: It extends the MyDialogOkCancel. It displays a message to confirm deleting a comment.
  + DialogDeletePostConfirmation: It extends the MyDialogOkCancel. It displays a message to confirm deleting a post.
  + DialogDeleteReviewConfirmation: It extends the MyDialogOkCancel. It displays a message to confirm deleting a review.
  + DialogEditComment: It extends the MyDialogOkCancel. It displays a user’s written comment. The user can edit the comment.
  + DialogEditReview It extends the MyDialogOkCancel. It displays a user’s review. The user can edit the review’s text and change the review’s rate.
  + DialogExit: It extends The MyDialogOkCancel. It displays a message to confirm deleting a review.
  + DialogForgetPassword: It extends the MyDialogOkCancel. It the user completely forgot his own account’s password, with using his email address, he can retrieves his forgotten password.
  + DialogMessage: It extends the MyDialogOkCancel. It is used to display a message.
  + DialogUnfollowBusinessConfirmation: It extends the MyDialogOkCancel. It displays a message to confirm unfollowing a business.
  + MyDialog: It extends MyPopup. All dialogs which extend this class have a title, body and a footer. This class adds a body and a footer to the MyPopup.
  + MyDialogOk: It extends MyDialog. All dialogs which extend this class have a title, body, and a button (OK) in the bottom of the dialog. With clicking on the button, the dialog will dismiss.
  + The MyDialogOkCancel: It extends MyDialog. All dialogs which extend this class have a title, body, and two buttons (OK and Cancel) in the bottom of the dialog. With clicking on the OK button, a defined action will execute and with clicking on the Cancel button, the dialog will dismiss.
  + MyPopup: It is the parent class for all classes in the Dialog directory. All dialogs which extend this class have a body.
  + PopupBlockUser: It has a button which is used to block a user.
  + PopupCancelFriendship: It has a button which is used to cancel a friendship relation with a user.
  + PopupCancelSharePost: It has a button which is used to cancel sharing a post by the user.
  + PopupCategories: It displays a list of business’ categories. A user can select one category. It is used in Search Business and Register Business.
  + PopupDeleteCommentBlockUser: It has two buttons, delete a comment and block a user. A business owner can delete user’s comment in his own posts. The business owner can block a user either. The blocked user won’t be able to comment on the business owner’s posts again.
  + PopupEditDeleteBusiness: It has two buttons, edit a business profile information and delete a business. A business owner can delete his own business or edit its profile information.
  + PopupEditDeleteComment: It has two buttons, edit a comment and delete the comment. A user can delete his own comment or edit it.
  + PopupEditDeleteReview: It has two buttons, edit a review and delete the review. A user can delete his own review or edit it.
  + PopupReportCancelSharePost: It has a button which is used to cancel sharing a post, or report the post.
  + PopupReportPostActivity: It has a button which is used to report a post inside ActivityPost.
  + PopupReportPostAdapter: It has a button which is used to report a post inside adapters.
  + PopupSelectCameraGallery: It has two buttons, select camera and select gallery. To choose a picture, user can uses camera to take a picture or opens his device gallery to select a picture.
  + PopupSubCategories: It displays a list of business’ sub categories. A user can select one sub category. It is used in Search Business and Register Business.
  + PopupUnfollowBusiness: It has a button which is used to unfollow a following business.

## Fragment

* + FragmentBusinessRegisterBaseInfo: It is first step of registering a business. In this fragment, a user enters business’s base information e.g. identifier, name, etc.
  + FragmentBusinessRegisterContactInfo: It is second step of registering a business. In this fragment, a user enters business’s contact information e.g. work time, cellphone number, email address or website URL. Except work time field, all other fields are optional.
  + FragmentBusinessRegisterLocationInfo: It is second step of registering a business. In this fragment, a user enters business’s location information e.g. province, city, address and business coordinates.
  + FragmentHome: The user’s Time Line will be displayed in this fragment.
  + FragmentSearch: The user can search about businesses, users and post in this fragment. To search businesses, it is necessary to determine business location.
  + FragmentUser: The user’s home information e.g. user’s shared post will be displayed in this fragment.
  + FragmentUserBusinesses: A list of user’s businesses will be displayed in this frgment.
  + FragmentUserLogin: This fragment is not implemented.
  + FragmentUserSignUpeEmailID: This fragment is not implemented.
  + FragmentUserSignUpeNamePassword: This fragment is not implemented.
  + FragmentUserSignUpePicture: This fragment is not implemented.

## Interface\_m

* + IAddReview: It is used to add a new review to a business’ reviews after successfully running the ReviewBusiness webservice.
  + IChangeBusiness: It is used to delete a business from user’s businesses list in FragmentUserBusinesses after successfully running the DeleteBusiness webservice.
  + IChangePassword: It is used to notify ActivityUserProfile if user changed his password.
  + IChangeTabs: It is used to notify ActivityMain to display 4 tabs when the user register a business or 3 tabs when the user delete all his businesses.
  + ICommentChange: It is used to notify AdapterPostComment to display an updated comment or remove a deleted comment.
  + ICropResult: It is used to notify ActivityCamera to return crop result.
  + IDeletePost: It is used to notify ActivityPost to return to the previous activity and it is used to notify AdapterPostBusiness to remove the deleted post.
  + IGetCallForTakePicture: It is used to notify ActivityBusinessRegisterEdit to display PopupSelectCameraGallery dialog.
  + IGetNewTimeLinePost: It is used to notify FragmentHome to update its adapter.
  + IPullToRefresh: It is used to notify all activities and fragments which use pull\_to\_refresh library to refresh their data.
  + IReportPost: It is used to notify ActivityPost, AdapterPostShared and AdapterPostTimeline to hide report option. If the user reports a post, he shouldn’t be able to report a post again.
  + ISelectCategory: It is used to return selected category from PopupCategories or return selected sub category from PopupSubCategories.
  + IUnfollowBusiness: It is used to notify AdapterUserFollowingBusinesses to remove an unfollowed business or notify GridViewBusinessOther to change follow status.
  + UpdateTimeLine: It is used to notify FragmentHome to update a post’s share status.
  + IUpdateUserPraofile: It is used to notify FragmentUser to display the user’s updated information after updating in
  + IWebserviceResponse: It is used to return a webservice’s result.

## Widget\_customized

* + Buttons
  + Charsoo\_activity
  + checkbox

Cached Images

Local Data Base and Share Preferences