# Software Requirements Specification

**Prepared by $anal Tragedy**

**for the project Tradegy**

*METU - Department of Computer Engineering*

*CENG 491 Senior Design Project I*

*Fall 2015-2016*

# Contents

# 1. Introduction

This section gives a scope description and overview of everything included in this SRS document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.

## 1.1 Problem Definition

The primary problem this project focuses on is the problem about re-usage of the products we have. Everyday people buying stuff they will use only a few times maybe only once and these products are forgotten after that. In this situation there are two parts Tradegy will take place. People may want to buy that product from someone else for cheaper than it will cost if s/he wants to but from a store. In this way people can use Tradegy and find used items and they can save both their own and natural resources. Also there is also necessity about a platform to sell the products we already have but we do not need. The lack of functionalities and amusement in the existing products about this area is another problem this project focuses on.

## 1.2 System Overview

Tradegy is an android based mobile application. In this application people will be able to create their profiles, put items in store to sell, see items in store to buy, offer for the items they are interested in, rate the user according to their previous shopping experiences and all these processes will be notified to user in case they does not check the application consistently and all these can be used location based. User will be able to see location information of the items they are interested in, and the user they are in contact.

The major parts we are going to produce in this project are:

- Application part
- Database part

The database part will be holding the information about user , items. The information user provides about themselves and the items they put on store will be stocked in database also application will provide some information and they will be also stocked in the database like location information.

In the application there will be functionalities like, registering , log in, putting item in store, visiting store, contacting user in case of an accepted offer. User will be receiving notifications in case of one of their items got an offer or one of their offers to another item got accepted.

## 1.3 Definitions, acronyms, and abbreviations

**Swipe** : is a domain-specific, declarative language for non-developers (such as designers, animators, illustrators, musicians, videographers and comic writers) to create media-rich/animated documents that contain photos, videos, images, vector graphics, animations, voices, musics and sound effects, which will be consumed on touch-enabled

devices such as smartphones, tablets and touch-enabled set-top-boxes (such as iPhone and Apple TV).

**AWS** : Amazon Web Server
**ADT** : Android Developer Tools
**API** :Application Programming Interface
**Sync** :Synchronization
**GPS** :Global Positioning System
**HTTP/S**: Hyper-Text Transfer Protocol / Secure Hypertext Transfer Protocol

### 1.4 Assumptions and dependencies

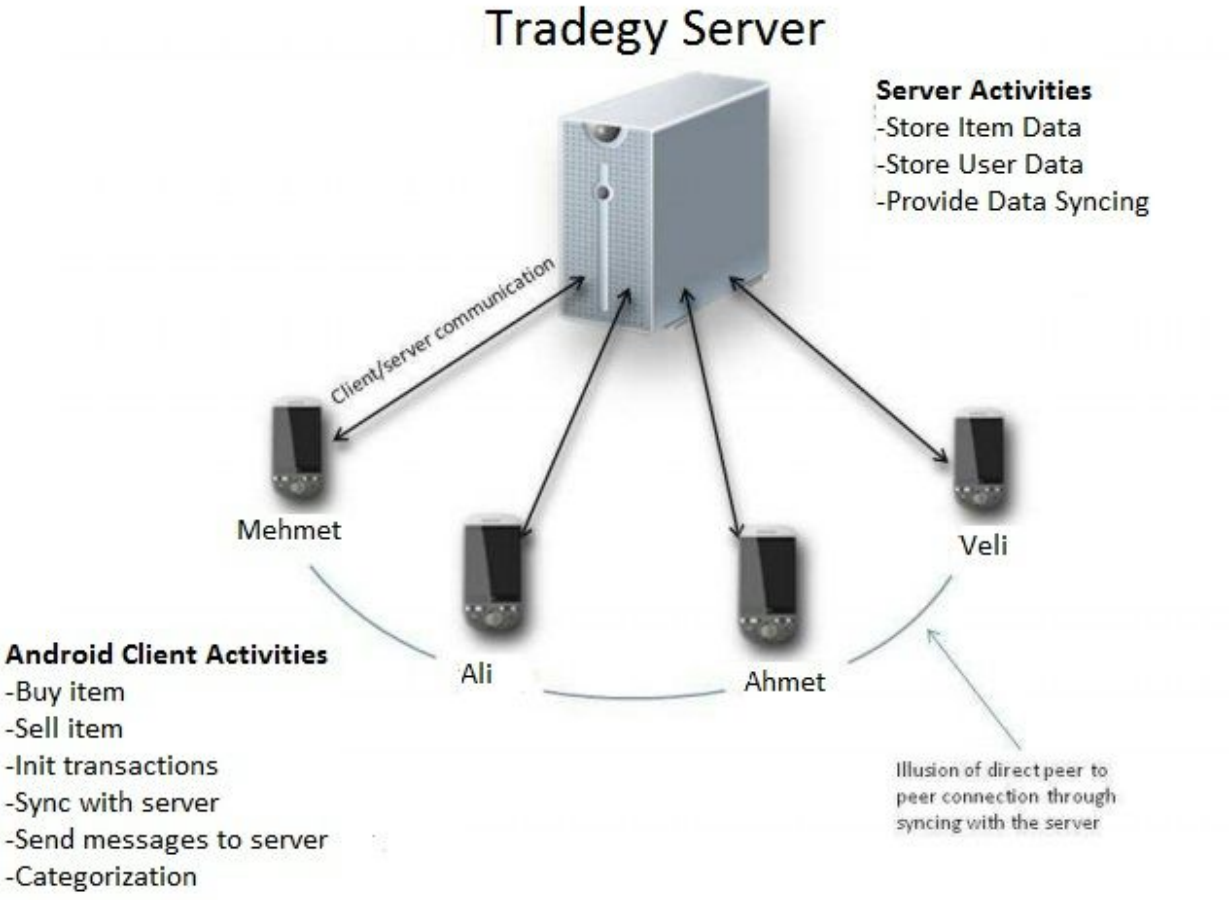
- Team members have enough experience and ability to fulfill this project.
- Team members have time to make discussions and meetings about this project.
- A smart phone with minimum accessories will be enough to use the application.
- Internet connection will be necessary to use the application and usage speed of this algorithm will be changing according to quality of the internet connection.
- The finalized user interface will be simple one and anyone will be able to use the application easily.
- There will be no limit of time or place to use this application.

## 2. Overall description

This section of the SRS should describe the general factors that affect the product and its requirements.

### 2.1 Product functions

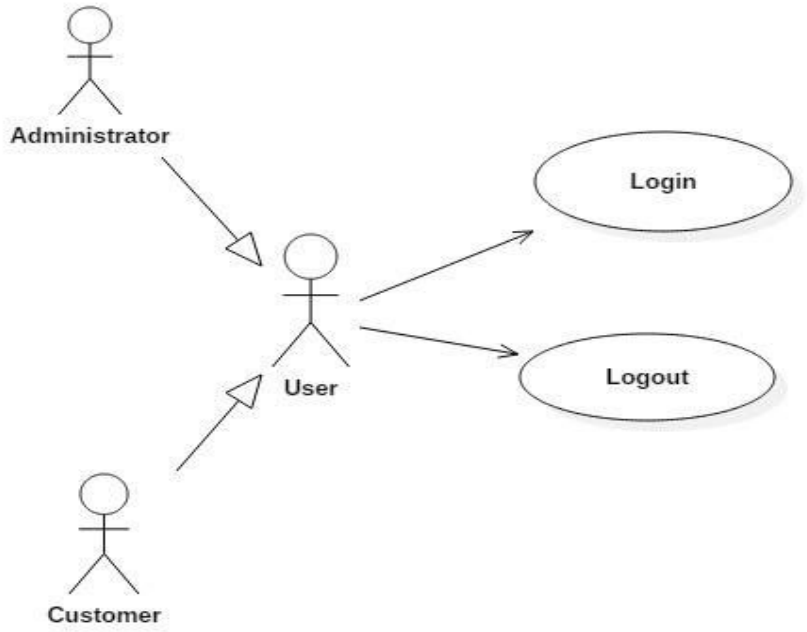
The Tradegy project is a new, self-contained product intended for use on the Android platform. While the Tradegy mobile application is the main focus of the project, there is also a server-side component which will be responsible for database and synchronization services. The scope of the project encompasses both server and client-side functionalities, so both aspects are covered in detail within this document. Below is a diagram of the Tradegy system which illustrates the interactions between the server and client applications.




### 2.1.1 Use-case model survey

{This section provides an overview of the use-case model. The survey is used by people interested in the behavior of the system, such as the customer, users, architects, use case authors, designers, use case designers, testers, managers, reviewers, and writers. This section lists for each use case

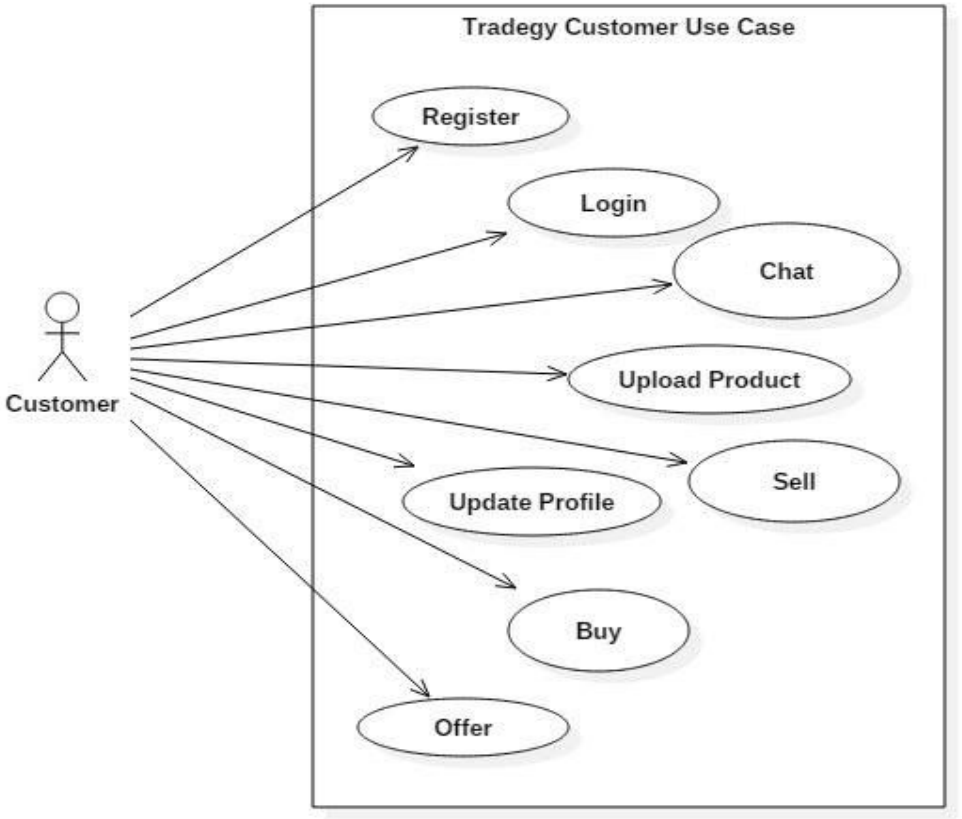
- The use case name.
- A brief description explaining the use case's function and role in the system.
- A list of actors for the use case (Primary and secondary, active and passive). The aggregation of these actors is further defined in the accompanying actor survey.
- Diagram of the use-case model. A diagram of the entire use-case model is included here.}




In our top-down approach user-case system , users may be categorized as administrator or customer with simple login and logout user functions.

#### 2.1.1.1 Customer Use Case

**Diagram:**



**Brief Description: Register**

Customer type users who haven't signed up yet may click on sign up button and register themselves to our app database.

**Initial Step-By-Step Description:**

1. Customer type user opens application.
2. Welcome screen pops up.
3. If user is not signed up yet, s/he clicks the sign up button.
4. Sign up screen opens, user fills his/her information on blanks.
5. System checks if user information is valid
6. If the information user filled is valid, system sends an e-mail to user's mail address for verification.
7. If the user opens the mail and clicks the given link,system accepts user and registers him/her to database, after that system lets user know that registration process is successful.
8. If the information is not valid, system gives a response to user that the given information is not valid for the registration process.

**Brief Description: Login**

Customer type user who already signed up into system may login and use the application with their specific username and password.

**Initial Step-By-Step Description:**

1. Customer type user opens application.
2. Welcome screen pops up.
3. If the user is signed up, s/he fills username and password fields and clicks login button.
4. System checks the information user entered.
5. If the information is valid, system accepts the login request of users and forwards him to application main page.
6. If the information is not valid, system gives an error message to user and lets him/her know that s/he entered the wrong information in the given fields.

**Brief Description:Chat**

Customer type user may chat with another customer for their trading process progression.

**Initial Step-By-Step Description:**

1. If a customer type user gives an offer for another user's selling product, notification will be sent to that customer.
2. After getting notification, user accepts offer of other user.
3. In this case, chat button will appear on both user's interface for letting them chat with each other and make an agreement.
4. One of users clicks chat button and sends a message to other one.
5. Other one gets the message and may reply to him/her.

**Brief Description:Upload Product**

User who wants to sell his/her good may want to upload photos and information of the product s/he wants to sell.

**Initial Step-By-Step Description:**

1. Customer clicks sell button for selling his/her good.
2. Selling screen pops up.
3. In selling screen, there is an upload button, user clicks upload button.
4. Application will ask user's permission for accessing his/her camera and gallery.

5. If user gives permission to application, user may upload a photo s/he selected for his selling good.
6. If the size is valid and has appropriate content for selling, system accepts user's photo.
7. Photo information is uploaded to database with attached to user's information.
8. If the size is not valid or has inappropriate content for selling, system gives an error to user and asks him/her to upload another photo.

**Brief Description:Sell**

Customer type user can sell his/her good with sell button on interface and add necessary information and images attached to it.

**Initial Step-By-Step Description:**

1. Customer clicks sell button on application interface.
2. Selling screen pops up.
3. User fills the given necessary fields with valid information and may upload a photo or photos of his/her product.
4. System checks all the information user entered for selling process.
5. If the given information is valid , system uploads the particular selling good to database with given information.
6. If there exists an invalid information in given fields, system gives an error message to user and asks him/her to refill the given fields.

**Brief Description:Update Profile**

Users may update their profile information by clicking on their profile button.

**Initial Step-By-Step Description:**

1. Users can click on profile button after they logged in application interface.
2. After clicking on profile button , profile settings pops up.
3. In profile settings view, user may do operations such as changing his username, password, phone number, email address or any other information.
4. If user changes any information, system checks if the new information is valid or not.
5. If the given information is aptly for system, system updates the information of user.
6. In case of attempting changing mail address , system sends user a verification mail.
7. If the user opens verification mail and clicks the give link, mail address gets updated.
8. If there exists any inappropriate information for system, system gives an error message to user and requests him/her to refill necessary fields again.

**Brief Description: Buy**

Users may want to buy products online via our application, in order to do that users should click "Buy" button look for goods they search.

**Initial Step-By-Step Description:**

1. Users can click on buy button after they logged in application system.
2. Buying screen pops up.
3. Users go to search screen and type the kind of good they are looking for by the help of our search engine.
4. Result numbers and information pops up as button for users.
5. If the user clicks the button buying screen will pop up
6. In buying screen products comes up randomly(considering the chosen category)
7. If user likes the product s/he may swipe right and offer for it.
8. If user does not like the product s/he may swipe left and pass to another product for viewing.

**Brief Description: Offer**

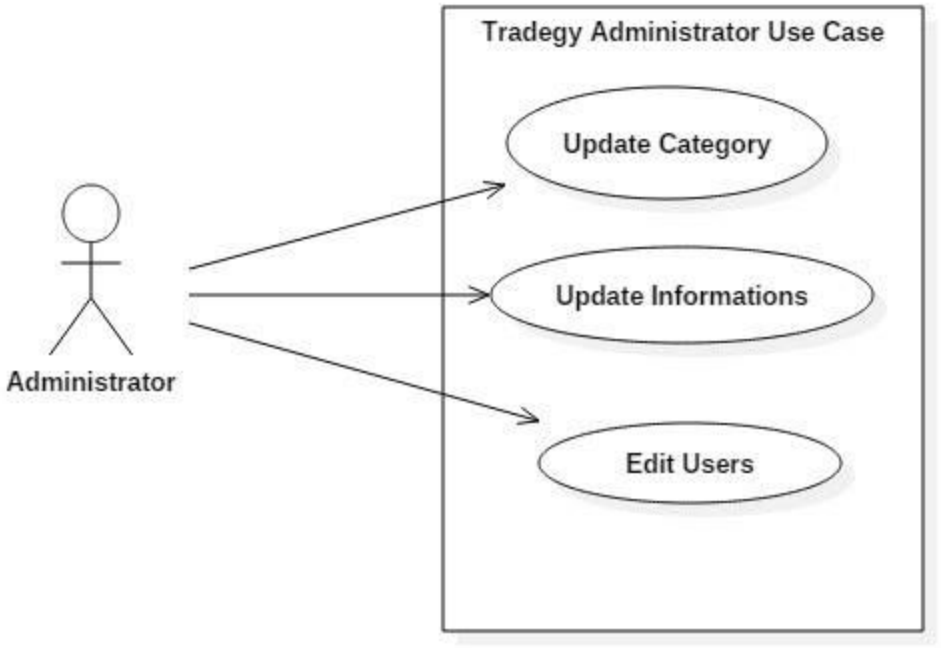
User may want to offer for particular product s/he liked on buying screen.

**Initial Step-By-Step Description:**

1. Users can swipe right for the product they liked on buying screen
2. Offering screen pops up after they swipe right.
3. Users can offer at any amount of money for the product and send their request to our server by application interface
4. If seller does not have any kind filter(like location, or minimum offering limit ) excluding buyer's request, the request will be sent to seller as notification.
5. If the request is caught by any kind of filter, it will not be sent to seller as notification.

#### 2.1.1.2 Admin Use Case

**Diagram:**




**Brief Description:Update category**

Administrator type of user may logged into application and update the categories for products.

**Initial Step-By-Step Description:**

1. Administrator type of users can click on categories button after they logged in.
2. In categories section , they can click add categories or remove categories button.
3. If Administrator clicks add categories button, he can name a new category and try to add it into categories field.
4. System checks if the name is valid and does not already exist in categories database.
5. If the name is valid and does not already exist in categories database, request gets accepted by server and aptly changes are made.
6. If the name is not valid or already exists in categories database, an error message will be given to administrator.

**Brief Description: Update Informations**

Administrator can update informations on welcome screen messages or

notifications and events.

**Initial Step-By-Step Description:**

1. Administrator can change the information on welcome screen, main page by clicking buttons which are designed for him on the interface.
2. On main page , he can add information or update information by clicking edit info button on main page.
3. Administrator may also update information such as profile settings fields by clicking profile button.

**Brief Description: Edit Users**

Administrator type users can edit users in terms of operations such as blocking them, sending warning to them when they violate terms and agreement of application manifest or they can edit their information according customer users wishes when it is necessary.

**Initial Step-By-Step Description:**

1. Administrator can click edit users button when they are in main screen.
2. Administrator can view users in a list view, search them on search field, see their status and click on them.
3. If administrator click on customer type user name, s/he can edit his/her information .
4. Updating request goes to server .
5. If the update request is valid, user information will be updated by administrator.
6. If the update request is not valid, user information will not be updated and error message will be sent to administrator by application interface.

### 2.1.2 Actor survey

{All of the actors mentioned in the use-case model survey are reported here. For each actor, you should list

- The actor's name
- A brief description of the actor}

There are two actors in our trading application system; customer and administrator.

**Customer:** Customer is a user type for our trading app, who can search the particular product type he/she wants, can view search results with our swiping technique and make offers for the goods he/she choose, or s/he can buy goods and review offers by the help of our Android application interface.

**Administrator:** Administrator is a user type for our trading app, who has special attributes other than customers such as updating categories, editing user information and updating content of application interface by small changes.

## 2.2 Interfaces

This section defines the interfaces that will be supported by the application. This section contains adequate specificity, protocols, ports, and logical addresses.

### 2.2.1 User Interfaces

A first time user on our mobile application should see the login page when s/he opens the application, in case s/he is not registered, s/he should be able to do that on the login page(see figure 1).

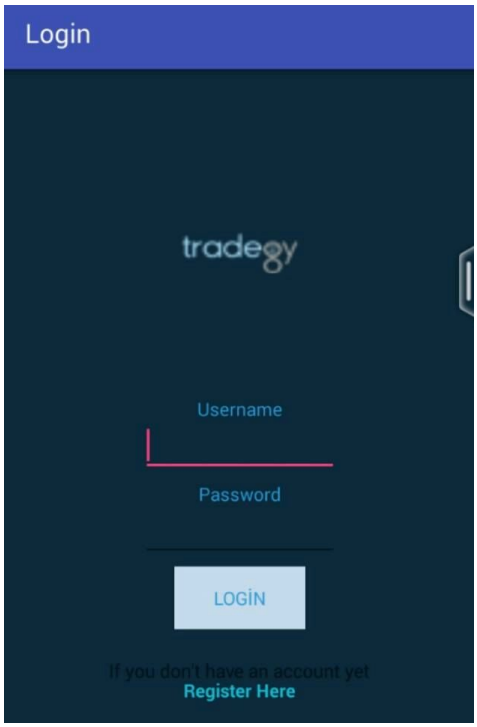



**Figure 1**

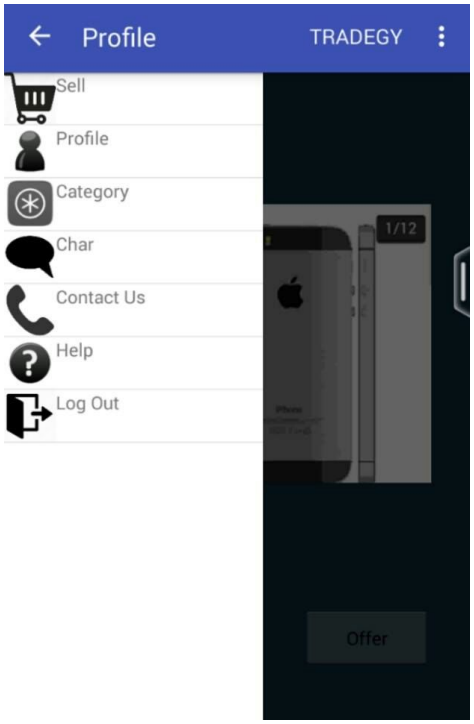



**Figure 2**

If the user is not a first-time user, s/he must be able to see the main page directly when s/he opens the page, in here user may do operations such as looking for profile settings or searching for any product s/he is looking for(see figure 2).

Every user has a profile page where they can edit their mail addresses, passwords and phone numbers(see figure3).

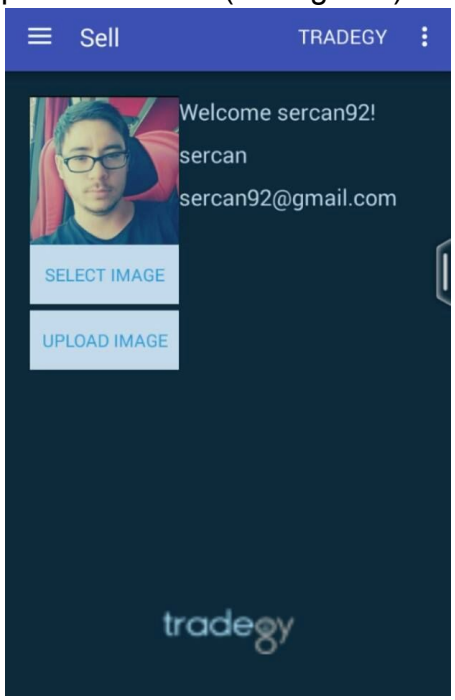



**Figure 3**

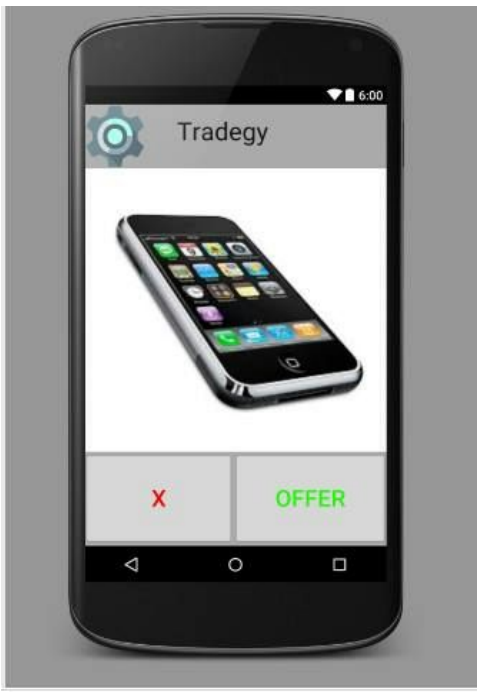



**Figure 4**

Users will be able to swipe right for the products they like and offer whereas they can swipe left for the products they don't like and skip to next one(see figure 4) on search screen.

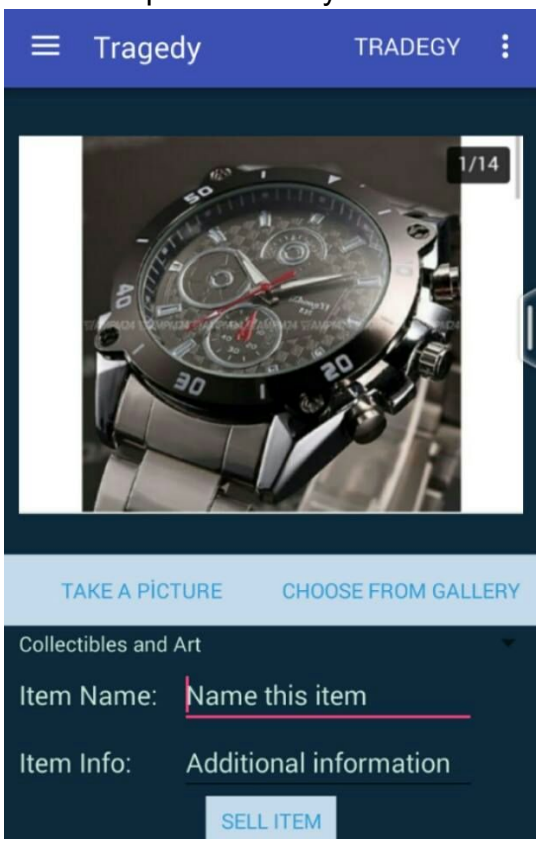



**Figure 5**

Users can also sell their goods and name them by help of application screen(see figure 5).

### 2.2.2 Hardware Interfaces

For Tradegy to run, you will need an Android phone, as it is not a resource hungry program any Android Device could do. Camera, GPS, and Internet connection hardware are default Android Device hardware, thus we have them already. Also our database at Amazon Web Service must be working without problem.

### 2.2.3 Software Interfaces

The system shall make use of the android operating system calls to the file management system to store and retrieve files. Also Google Maps software is required for our design. Any Android platform would do.

### 2.2.4 Communications Interfaces

Client on Internet will be using HTTP/HTTPS protocol.

## 2.3 Constraints

The primary design constraint is the mobile platform. Since the application is designated for mobile handsets, limited screen size and resolution will be a major design consideration. Creating a user interface which is both effective and easily navigable will pose a difficult challenge. Other constraints such as limited memory and processing power are also worth considering.
Tradegy is meant to be quick and responsive, even when dealing with large user pool and transactions, so each feature must be designed and implemented with efficiency in mind.
There is also internet connection constrain exist, application needs a constrain of 50+ kbps stabile network connection during session.
GUI is only in English. Login and password is used for the identification of users. Also only registered buyers or sellers will be authorized to use the services.

# 3. Specific requirements

## 3.1 Functional Requirements

### 3.1.1.Register

| **Use case name** | Register |
|---|---|
| **Trigger** | Customer type user clicks sign up button |
| **Precondition** | User mustn’t be logged on already |
| **Path** | 1. Customer type user opens application.<br>2. Welcome screen pops up.<br>3. If user is not signed up yet, s/he clicks the sign up button.<br>4. Sign up screen opens, user fills his/her information on blanks.<br>5. System checks if user information is valid<br>6. If the information user filled is valid, system sends an e-mail to user’s mail |

| | |
|---|---|
| | address for verification.<br>7. If the user opens the mail and clicks the given link,system accepts user and registers him/her to database, after that system lets user know that registration process is successful.<br>8. If the information is not valid, system gives a response to user that the given information is not valid for the registration process. |
| **Postcondition** | User is registered to system. |
| **Exception Path** | If the information is not valid, system gives a response to user that the given information is not valid for the registration process. |

### 3.1.2 Login

| | |
|---|---|
| **Use case name** | Login |
| **Trigger** | Customer types user fills the necessary fields and clicks log in button |
| **Precondition** | User must be open android application |
| **Path** | 1. Customer type user opens application.<br>2. Welcome screen pops up.<br>3. If the user is signed up, s/he fills username and password fields and clicks login button.<br>4.System checks the information user entered.<br>5. If the information is valid, system accepts the login request of users and forwards him to application main page. |
| **Postcondition** | The user logged into application |
| **Exception Path** | If the information is not valid, system gives an error message to user and lets him/her |

| | |
|---|---|
| | know that s/he entered the wrong information in the given fields. |

### 3.1.3 Chat

| | |
|---|---|
| **Use case name** | Chat |
| **Trigger** | Customer clicks chat button. |
| **Precondition** | User must have sent an offer to seller and seller must have already accepted that offer for chat function |
| **Path** | 1.If a customer type user gives an offer for another user's selling product, notification will be sent to that customer.<br>2.After getting notification, user accepts offer of other user.<br>3.In this case, chat button will appear on both user's interface for letting them chat with each other and make an agreement.<br>4.One of users clicks chat button and sends a message to other one.<br>5.Other one gets the message and may reply to him/her. |
| **Postcondition** | Users message is sent to other user on chat screen |
| **Exception Path** | None |

### 3.1.4 Upload Product

| | |
|---|---|
| **Use case name** | Upload Product |
| **Trigger** | Customer clicks upload button |
| **Precondition** | Customer must be already in sell screen in order to click upload button |
| **Path** | 1.Customer clicks sell button for selling his/her good.<br>2.Selling screen pops up.<br>3.In selling screen, there is an upload |

| | |
|---|---|
| | button, user clicks upload button.<br>4.Application will ask user's permission for accessing his/her camera and gallery.<br>5.If user gives permission to application, user may upload a photo s/he selected for his selling good.<br>6.If the size is valid and has appropriate content for selling, system accepts user's photo.<br>7.Photo information is uploaded to database with attached to user's information. |
| **Postcondition** | Product with photo is uploaded to server via application interface |
| **Exception Path** | If the size is not valid or has inappropriate content for selling, system gives an error to user and asks him/her to upload another photo. |

### 3.1.5 Sell

| **Use case name** | Sell |
|---|---|
| **Trigger** | User clicks sell button on application interface |
| **Precondition** | User must already logged onto system in order to click sell button |
| **Path** | 1.Customer clicks sell button on application interface.<br>2.Selling screen pops up.<br>3.User fills the given necessary fields with valid information and may upload a photo or photos of his/her product.<br>4.System checks all the information user entered for selling process.<br>5.If the given information is valid , system uploads the particular selling good to database with given information. |

| | |
|---|---|
| **Postcondition** | Product is uploaded to system with necessary information and images attached to it |
| **Exception path** | If there exists an invalid information in given fields, system gives an error message to user and asks him/her to refill the given fields. |

### 3.1.6 Buy

| | |
|---|---|
| **Use case name** | Buy |
| **Trigger** | User clicks the buy button |
| **Precondition** | User must be already logged in for buying operation |
| **Path** | 1.Users can click on buy button after they logged in application system.<br>2.Buying screen pops up.<br>3.Users go to search screen and type the kind of good they are looking for by the help of our search engine.<br>4.Result numbers and information pops up as button for users.<br>5.If the user clicks the button buying screen will pop up<br>6.In buying screen products comes up randomly(considering the chosen category)<br>7.If user likes the product s/he may swipe right and offer for it.<br>8.If user does not like the product s/he may swipe left and pass to another product for viewing. |
| **Postcondition** | User may offer for a product s/he likes if there exists any. |
| **Exception path** | None |

### 3.1.7 Offer

| **Use case name** | Offer |
|---|---|
| **Trigger** | User swipes right for giving an offer to product s/he likes |
| **Precondition** | User must be in buy screen already |
| **Path** | 1.Users can swipe right for the product they liked on buying screen<br>2.Offering screen pops up after they swipe right.<br>3.Users can offer at any amount of money for the product and send their request to our server by application interface<br>4.If seller does not have any kind filter(like location, or minimum offering limit ) excluding buyer's request, the request will be sent to seller as notification. |
| **Postcondition** | Seller got a notification by system since buyer has sent an offer for him/her |
| **Exception path** | If the request is caught by any kind of filter, it will not be sent to seller as notification. |

### 3.1.8 Update Category

| **Use case name** | Update Category |
|---|---|
| **Trigger** | Administrator must click on category button |
| **Precondition** | Administrator must logged on to system |
| **Path** | 1.Administrator type of users can click on categories button after they logged in.<br>2.In categories section , they can click add categories or remove categories button.<br>3.If Administrator clicks add categories button, he can name a new category and try to add it into categories field.<br>4.System checks if the name is valid and |

| | |
|---|---|
| | does not already exist in categories database.<br>5.If the name is valid and does not already exist in categories database, request gets accepted by server and aptly changes are made. |
| **Postcondition** | Category section is updated by administrator |
| **Exception path** | If the name is not valid or already exists in categories database, an error message will be given to administrator. |

### 3.1.9 Update Information

| | |
|---|---|
| **Use case name** | Update Information |
| **Trigger** | Administrator clicks update button on main screen |
| **Precondition** | Administrator must be logged in |
| **Path** | 1.Administrator can change the information on welcome screen, main page by clicking buttons which are designed for him on the interface.<br>2.On main page , he can add information or update information by clicking edit info button on main page.<br>3.Administrator may also update information such as profile settings fields by clicking profile button. |
| **Postcondition** | Information such as events welcome screen text etc. is updated. |
| **Exception Path** | Administrator update request is not found valid by system . |

### 3.1.10 Edit User

| | |
|---|---|
| **Use case name** | Edit user |
| **Trigger** | Administrator clicks edit users button |
| **Precondition** | Administrator must be logged into system |
| **Path** | 1.Administrator can click edit user button when they are in main screen.<br>2.Administrator can view users in a list view, search them on search field, see their status and click on them.<br>3.If administrator click on customer type user name, s/he can edit his/her information .<br>4.Updating request goes to server .<br>5.If the update request is valid, user information will be updated by administrator. |
| **Postcondition** | User information is edited by administrator |
| **Exception Path** | If the update request is not valid, user information will not be updated and error message will be sent to administrator by application interface. |

## 3.2 Non-functional Requirements

### 3.2.1 Usability

Since Tradegy will have a real simple interface, any user that can use a smartphone already will be able to use this application easily. With a short tutorial after installing the application, user can be ready to go in minutes. We are not thinking about focusing on power user while testing the application since this is a mobile application.Our user tests will be focused on inexperienced and regular user.

A clear visual hierarchy will be included, so that the navigation through the application can be smooth and clear. Error and success messages will provide a meaningful feedback to users. User should improve his/her skills over time and should be able to learn easily.

### 3.2.2 Reliability

According to user characteristics system should provide high availability.98% of the time the system will be up and the maintenances will be filling the left. 7/24 availability is one of the essential things in this application. According to hosting service there may be downtimes also, but these must be rare times and they must be handling it in a quick manner.

The system is allowed to stay inaccessible for 30 minutes in maximum. System failure will be handling quickly in this time if there is any, also planned maintenances will be taking this long at most.

### 3.2.3 Performance

The system will be interactive and delays involved will be less. So in every action - response of the system there will be no immediate delays. Tradegy does not require a powerful processor or does not use ram that much. Our system will be responding to transactions in 0.50 ms in average and 2 s at most. Quality of the internet connection will affect the total process time of the functions. The system will be able to handle 100k transactions per second. Tradegy uses AWS as for hosting services now and it is able to handle 1k user but it will be upgraded when the current server can not handle the user.

### 3.2.4 Supportability

Supportability is not a concern for this project.

# 4.Data Model and Description

## 4.1 Data Description

**User**

A User can be Customer and Administrator.A user can login and logout.A user has a userID(username) and a password.Our system check the user's status(online or offline).When a user login,our application get a location from user.

**Administrator**

Admin has authority much more than a user.A admin can edit a user , change a user interface,update some informations etc.

**Customer**

Customer can sell or buy products or offer for products.If s/he want to chat dealers about price ,information or quality of products , s/he can do that. A customer can upload a product to sell now or wait to sell later.

**Shopping Cart**

A customer can have one more than Credit Cart and s/he can buy products by using cards.

**Category**

There are six categories now , it can be more than six.

**Product**

A customer can sell or buy a product or products, s/he can offer for an product.

**Orders**

A customer can buy a item by trading face to face or order for products.

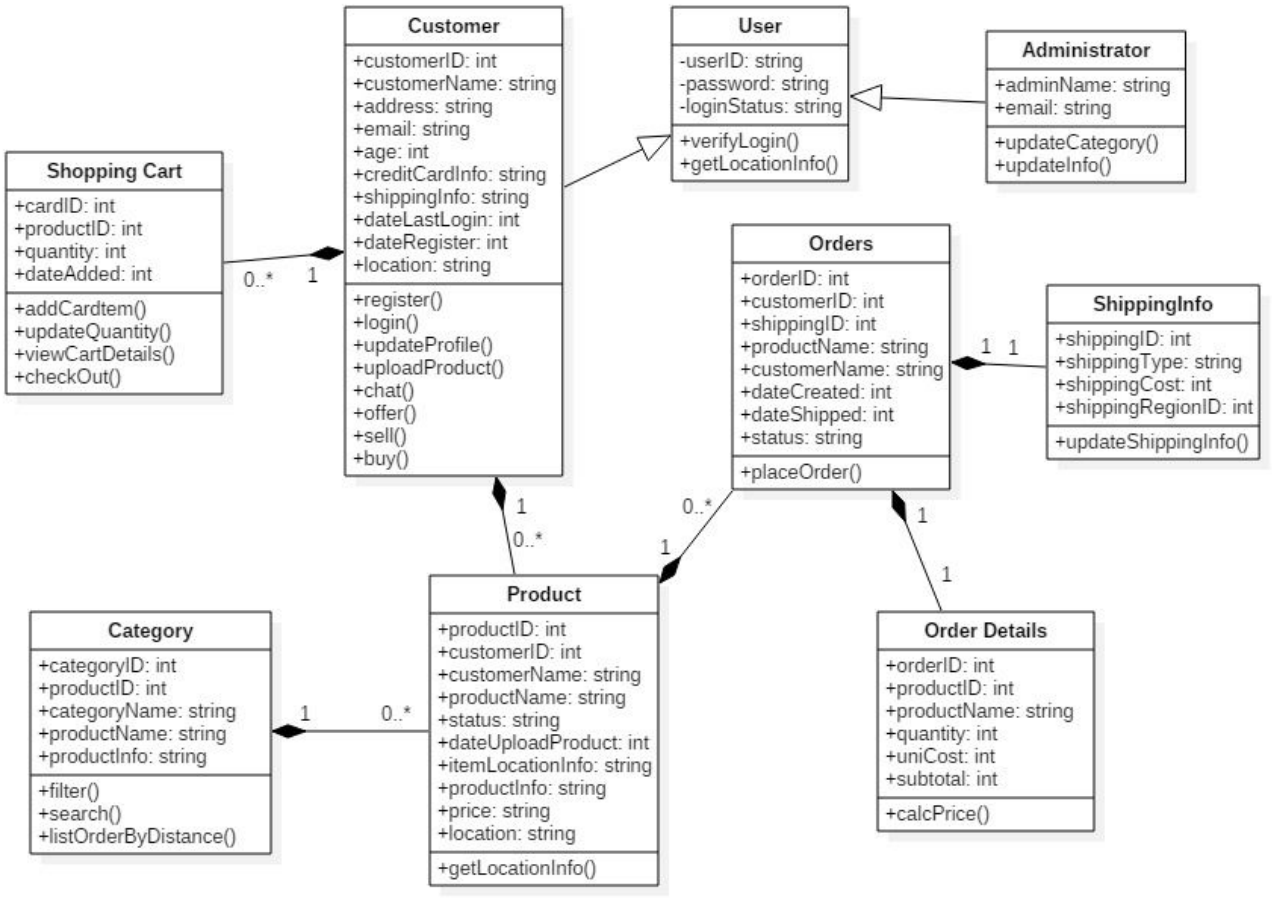
**OrderDetails**

It give us detail informations of orders.

**Shipping Info**

It's about the shipping informations.

### 4.1.1 Data objects




***verifLogin()****(return : bollean) : check to correct user entries*

***getLocationInfo()****(return : int) : get information of user's location*

***register()****(return : void) : sign up to the system*

***login()****(return : void) : login to the system*

***updateProfile()****(return : void) : change , edit or delete user' profile information.*

***uploadProduct()****(return : void) : upload a product to sell now or wait to sell later.*

***chat()****(return : string) : connection between customers*

***offer()****(return : int) : offer a price for a product or products*

***sell()****(return : void) : sell product*

***buy()****(return : void) : buy product*

***addCardItem()****(return : void) : add credit card information to buy a product.*

***viewCartDetails()****(return : void) : view detail informations of the credit card.*

***checkOut()****(return : boolean) : checkout for a product*

***filter()****(return : void) : filter the products to find searching products easy.*

***search()****(return : void) : search whatever product customer want it.*

***listOrderByDistance()****(return : void) : list products order by close distance.*

***placeOrder()****(return : void) : place order the orders.*

***calcPrice()****(return : int) : calculate total of shipped cost and product cost*

***updateShippingInfo()****(return : void) : update shipping information.*

### 4.1.2 Data dictionary

***cardID****(Data Type : int )* : System give an id number customers.it's primary key.

***productID****(Data Type : int )* : Every product has an id number.it's primary key.

***quantity****(Data Type : int )* : balance in the credit card.

***dateAdded****(Data Type : int )* : credit card Date Added

***categoryID****(Data Type : int )* : every category has an id.it's primary key.

***categoryName****(Data Type : string)* : Categories' names are Fashion,Electronics,Collectibles & Art,Home & Garden,Sporting Goods,Motors,Daily Deals.

***productName****(Data Type : string)* : When customer upload a product , s/he have to give a product name.it necessary for searching a product.

***productInfo****(Data Type : string)* : it also necessary for searching and need to know informations about the product.

***customerName****(Data Type : string)* : customers have a name when created accounts.

***status****(Data Type : string)* : status help us to know product's status.it could be selled,offered for some price or not selled.

***dateUploadProduct****(Data Type : int )* : its about date upload product.

***itemLocationInfo****(Data Type : int )* : when a customer login , s/he has a location information.If customer sell product , the product will has a location information as a customer location.

***price****(Data Type : int )* : product have to be a price because other customers need to know its price.

***OrderID*** *(Data Type : int )* : A order product must have a orderID.It must be primary.

***shippingID****(Data Type : int )* : To buy a product ,need to ship .So system give a primary id for orders in shipping status.it's primary key.

***dateCreated****(Data Type : int )* : date a order created.

***dateShipped****(Data Type : int )* : date a order shipped.

***uniCost****(Data Type : int )* : how much cost to order without shipped cost.

***subtotal****(Data Type : int )* : cost to product price and shipped cost.

***shippingType****(Data Type : string)* : information of shipping company.

***shippingCost****(Data Type : int )* : information of shipping cost.

***shippingRegionID****(Data Type : int )* : information of shipping region.

***adminName****(Data Type : string)* : information of admin name.

**About this template**

This template was adapted by Emre Akbas from two sources: the IEEE 830 [1] and the ``Modern SRS package" [2].

## 5 References

[1] IEEE Guide for Software Requirements Specifications," in IEEE Std 830-1984 , vol., no., pp.1-26, Feb. 10 1984, doi: 10.1109/IEEESTD.1984.119205,
URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=278253&isnumber=6883

[2] Appendix C of Don Widrig, Dean Leffingwell, "Managing Software Requirements: A Unified Approach," Addison-Wesley Professional, Release Date: October 1999, ISBN: 0201615932.