

MIDDLE EAST TECHNICAL UNIVERSITY
COMPUTER ENGINEERING DEPARTMENT

ONLINE BARTER MARKET

SOFTWARE DESIGN DESCRIPTIONS (v 1.0)

LONESOME CODEBOYS

Ali Can BATUR	1745793
Donny Irawan BULHADIE	1702240
Emre DENIZ	1745876
Ismail Sarp DIKKAYA	1745884

Preface

This document includes the system design of online barter market project. The document is prepared according to the

IEEE Std 1016 – 2009 [1]

This Software Design Descriptions Documentation provides a complete description of all the system design and views.

The first, second and third sections of this document include the purpose, scope of the document and the references used though out the document.

The third section includes the design descriptions information content.

The forth section includes the design viewpoints of the whole system.

Table of Contents

1 Overview	5
1.1 Scope	5
1.2 Purpose	5
1.3 Intended Audience	6
2 Definitions	6
3 Conceptual Model for Software Design Descriptions	7
3.1 Software Design in Context	7
3.2 Software Design Descriptions within the Life Cycle	8
3.2.1 Influences on SDD Preparation	8
3.2.2 Influences on Software Life Cycle Products	8
3.2.3 Design Verification and Design Role in Validation	8
4 Design Description Information Content	9
4.1 Introduction	9
4.2 SDD Identification	9
4.2.1 References	9
4.3 Design Stakeholders and Their Concerns	9
4.4 Design Views	10
4.5 Design Viewpoints	10
4.6 Design Elements	11
4.7 Design Overlays	11
4.8 Design Rationale	12
4.9 Design Languages	12
5 Design Viewpoints	12
5.1 Introduction	12
5.2 Context Viewpoint	12
5.3 Composition Viewpoint	18
5.4 Logical Viewpoint	21
5.5 Information Viewpoint	24
5.6 Dependency Viewpoint	26
5.7 Interaction Viewpoint	26
5.8 Interface Viewpoint	33
5.9 Patterns Use Viewpoint	37
5.10 State Dynamics Viewpoint	38
5.11 Structure Viewpoint	40
5.12 Algorithm Viewpoint	40
5.13 Resource Viewpoint	40
6 Planning	40
7 Conclusion	42

List of Tables

Table 1: Terms and Definitions	6
Table 2: Login Case	15
Table 3: Add Item Case	15
Table 4: Remove Item Case	15
Table 5: Search for Items Case	16
Table 6: Message Case	16
Table 7: Edit Profile Case	16
Table 8: Thanks Case	17
Table 9: Search for another User Case	17
Table 10: Create Account Case	17
Table 11: Seeing Categories Case	17
Table 12: Database Design	25

List of Figures

Figure 1: Use Case Diagram for User	13
Figure 2: Guest Case Diagram	14
Figure 3: Overall Component Diagram	19
Figure 4: Detailed Component Diagram	20
Figure 5: Deployment Diagram	21
Figure 6: Class Diagram	22
Figure 7: ER Diagram	24
Figure 8: Login/Register View	25
Figure 9: Add Item View	26
Figure 10: Remove Item View	27
Figure 11: Search Item View	28
Figure 12: Send/Get Message View	29
Figure 13: Search User View	30
Figure 14: View/Update Profile View	31
Figure 15: UI - Register Page	32
Figure 16: UI - Login Page	33
Figure 17: UI - Home Page	34
Figure 18: UI - Profile Page	34
Figure 19: UI - Search Page	35
Figure 20: UI - My Items Page	35
Figure 21: MVC	36
Figure 22: State Diagram for Login and Register	37
Figure 23: State Transition Diagram for Whole System	38

1 Overview

1.1 Scope

The web based application project that we are trying to develop is called “Online Barter Market”. The application will base on barter method where money does not involve in any purchasing made. That is the reason why this application is intended for people who is willing to purchase something when they are short of money. So, the application will help the User to get an item by exchanging or borrowing his or her belonging with someone else.

This SDD defines and describes the use of each view, the architectural constraints of the system, the functional requirements with a significant impact on the architecture, use-case realization, concurrency aspects, the layers and subsystems of the application, performance issues and constraints. Thus, this document will serve as a guideline through the implementation phase. In the design process, additional requirements were added as needed for the implementation, based on the internal requirements.

1.2 Purpose

This Software Design Description (SDD) provides full or complete details of the design project. The purpose of this document is to provide the developers a guidance of the architecture of the software. In fact, it also can help the instructor in order to understand the implementation of our software.

This document will explain how to implement the system based on the requirements that mention in Software Requirement Specification. It will also explain the component of the application and their properties. Furthermore, it describes the additional classes, attributes and methods to be implemented.

1.3 Intended Audience

This Software Design Description is intended for programmers or developer who is going to develop this application as guidance while implementing the software requirement. So, this will become a standard design of the software before any upgrade.

This project is also a final project of computer engineering department of METU before graduate. Thus, the benefit for the instructors is that they can refer to this document when they are checking the software and also to understand whether it is well implemented or not.

For validation and verification purpose, tester can also use this SDD as a reference to find the bug and build a test case. However, they may also refer to SRS too for further information about the software.

2. Definitions, Acronyms, Abbreviations

Terms	Definitions
IEEE	Institute of Electrical and Electronics Engineers
SDD	Software Design Descriptions
ER	Entity Relationship Diagram
GUI	Graphical User Interface
Stuff/Good/Item	People's possessions that are shared, given or exchanged on the system
SRS	Software Requirements Specification
OOP	Object Oriented Programming
App	Application
UML	Unified Modeling Language

Table 1: Terms and Definitions

3. Conceptual Model for Software Design Description

In order to understand deeply about the application and picture it well of how it is going to be implemented, as a developer, we need to know about basic knowledge of web-based programming.

In this section, we will present a conceptual model for the SDD. This conceptual model mainly explains the context, description, influence and validation of the application in which SDD is prepared.

3.1 Software Design in Context

This application will be used by the user that already mention in the SRS. This application is design for all Turkish people who need to exchange their belonging. The stakeholder of the software will be user and the developer of the application.

The User of this application will be classified as Registered User or Guest User. All this type of users is in a relation with the general system according to their privilege or permissions. For instance, both of the Users have an access to see the item in the post section or search for it. Separately for Registered Users can use their account to login to the system. After that, they can access to the postings page where they can post the item that they want to trade and thanking the other user for their service.

Subsystems also have a relation among themselves. Changes in some subsystems affect other system automatically. For instance, posts that Registered User erases in their page trigger all other subsystems. Therefore, the item that was deleted will be erased from the database.

The whole data that is stored in the application will be kept in the database. In other word whole system has a structural relation with database.

3.2 Software Design Descriptions within the Life Cycle

3.2.1 Influence on SDD Preparation

The SRS is the main influence of this SDD preparation. Because all design made in this SDD is the implantation of the specification mention in SRS. It will include product perspective, functional and non-functional requirement and interface requirement implementation. Furthermore, the demand that stakeholders mention on the proposal of the project will be specified as the source of the design of the project.

3.2.2 Influences on Software Life Cycle Product

In this project, database mechanism should be designed first in order to picture how the data would be kept on the database. After that, we can start writing the code to build the web and connect it to the database, so that we can send the query through it. And finally the design of the application will be done after the system between the database and the code is well connected and tested.

3.2.3 Design Verification and Design Role in Validation

Verification and validation will be done after all test scenarios and cases are completed. Test cases are implemented on the system in order to check designs satisfy all requirements. Also tests could be done while developing the application. In fact, all the system will be tested in order to verify whether the app fulfills the requirements or not. Furthermore, the test will probably trigger an upgrade of the system especially when there is a bug found or a need to enhance the system so that it can work well as expected.

4 Design Description Information Content

4.1 Introduction

This document clarifies everything related with design and future implementation. Design of the project will be object oriented with modules, class files and interface will be designed.

4.2 SDD Identification

After testing for the verification and validation, at the 2013 final version of our project will be released. Users will be able to use our web site by using any Internet browser. Also users can reach our web site by using their computers', smart phones' and tablets' Internet browsers. It will be open to the new changes so that we can add some additional features.

4.2.1 References

[1] *IEEE STD 1016-2009, IEEE Standard for Information Technology – System Design – Software Design Descriptions. IEEE Computer Society, 2009.*

[2] *StarUML 5.0 User Guide. (2005).*

[3] *Booch, G., Rumbaugh J., Jacobson I., The Unified Modeling Language User Guide Addison Wesley*

4.3 Design Stakeholders and Their Concerns

Our web sites stakeholders' are our lecturers and other developers. With the help of this report, our lecturers will understand our project in detail and what we will do in implementation. Also other developers can benefit from this document.

4.4 Design Views

We use modular structure design in our project so that any stakeholder can add or remove any feature in our project easily if they want. Beside modular structure, we also use object oriented design. OOP design will allow us to contribute to our project maintainability feature.

For instance, when user enters to our system, main home page will be displayed. Users can see the items, item categories and search items. However if they want reach their personal account, they need to fill the login form in login page. If she/he has registered before, after entering username and password home page will show up. Not registered users have to register in order to create personal account. Users will have everything about using our web site in their hand. Context of using our site is explained in our SRS document.

Interface views let users see outputs clearly. A logical view is made clear by the help of diagrams. Team organization and composition is also explained. It will also declare what can be done with information they show scheduling and estimated cost of our project.

Design views are explained in design viewpoints section in detail.

4.5 Design Viewpoints

Context viewpoints will be explained in the next section, Context Viewpoints. Here there are short introduction explanations about each viewpoint;

- *Context Viewpoint* explains all about between user and system. Use case diagrams also will be used for understanding relations between user and system. In this viewpoint, every single event is explained.
- *Composition Viewpoint* explains logical and physical component of the system in detail and to explain overall system architecture from perspective of each component.
- *Logical Viewpoint* explains whole project structure, classes and methods in detail

- *Information viewpoint* shows the relationships in the system. The entity relationship diagram is used for a visual representation of entities and how they relate to each other.
- *Patterns Use Viewpoint* explains the design pattern we will use in the project. The design pattern we used is MVC (Model View Controller). MVC is an architectural pattern which separates the application into three main components.
- *Interface Viewpoint* describes a contract among designers, programmers, customers, and testers. The user interface will be explained with the just simple drawings to demonstrate how the user interface looks like. The images in these sections are for just simple demonstration.
- *Interaction Viewpoint* explains the dynamic structure and relations between objects. UML sequence diagrams are used for better representation.
- *State Dynamic Viewpoint* also explains the dynamic structure and relations between objects. However, this time UML state diagrams are used for representation.

4.6 Design Elements

Just not to divide the design views into two parts for section 4 and section 5, we explained all design elements, views and structures in the section 5 design viewpoints. Design elements can be analyzed in information viewpoint, logical viewpoint and compositions viewpoint.

4.7 Design Overlays

All existing and additional design information is described in the design viewpoints section.

4.8 Design Rationale

Sustainability, maintainability, reliability, availability and security principles are concerned during designing process. This project is maintainable which means that it can be modified by developers and stakeholders in the future. For explaining what the parts in implementation does, there will be comments on top of each file and near each function definition. By the help of these comments, future developers can understand the entire software in detail and can change it easily.

In our project, we use MVC as the design pattern. MVC stands for model, view and controller. We chose this design pattern because it provides clean separation of concerns, enables full control over HTML and easy integration. Also MVC supports the web applications. Why we chose this design pattern is explained in the patterns use viewpoint section in design viewpoints.

4.9 Design Languages

Unified Modeling Language (UML) [2], [3] is used for creating design viewpoints in this document.

5 Design Viewpoints

5.1 Introduction

This and the following sections will give a brief outline on the design viewpoints. The following design viewpoints explain the project design. In some of them, there is UML diagrams to describe its point in detail.

5.2 Context Viewpoint

The IEEE 1016-2009 document says “That context is defined by reference to actors that include users and other stakeholders, who interact with the design subject in its environment” [2]. The context viewpoint explains the relationships, dependencies, and interactions in the system. It also describes the actors and stakeholders’ role. Below UML use case diagrams shows the relationships and actor’s role in the system. The use case diagrams used in this section are created according to UML use case diagram rules.

5.2.1 Design Concerns

- **User Use Case Diagrams:** The User has the following sets of use cases. Here user means registered user. The following is the user use case diagram of the project that we are going to build;

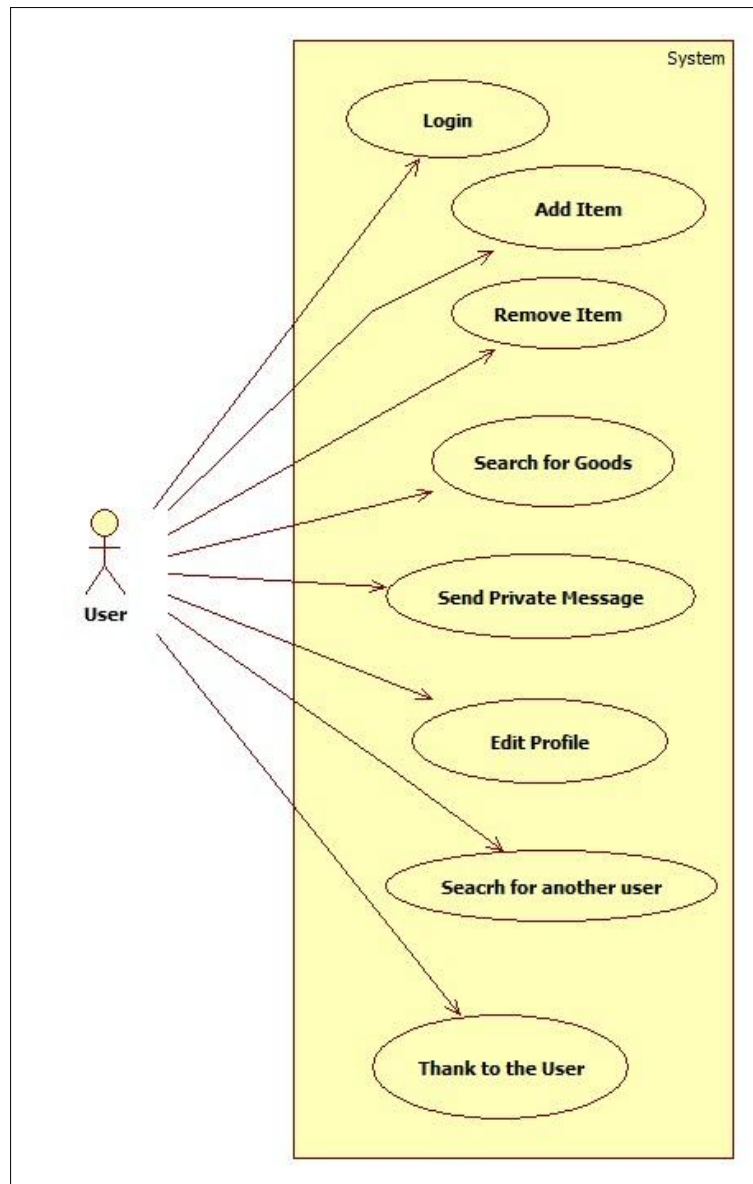


Figure 1: Use Case Diagram for User

- **Guest Use Case Diagrams:** Here is a guest use case diagram of the product that we are going to build:

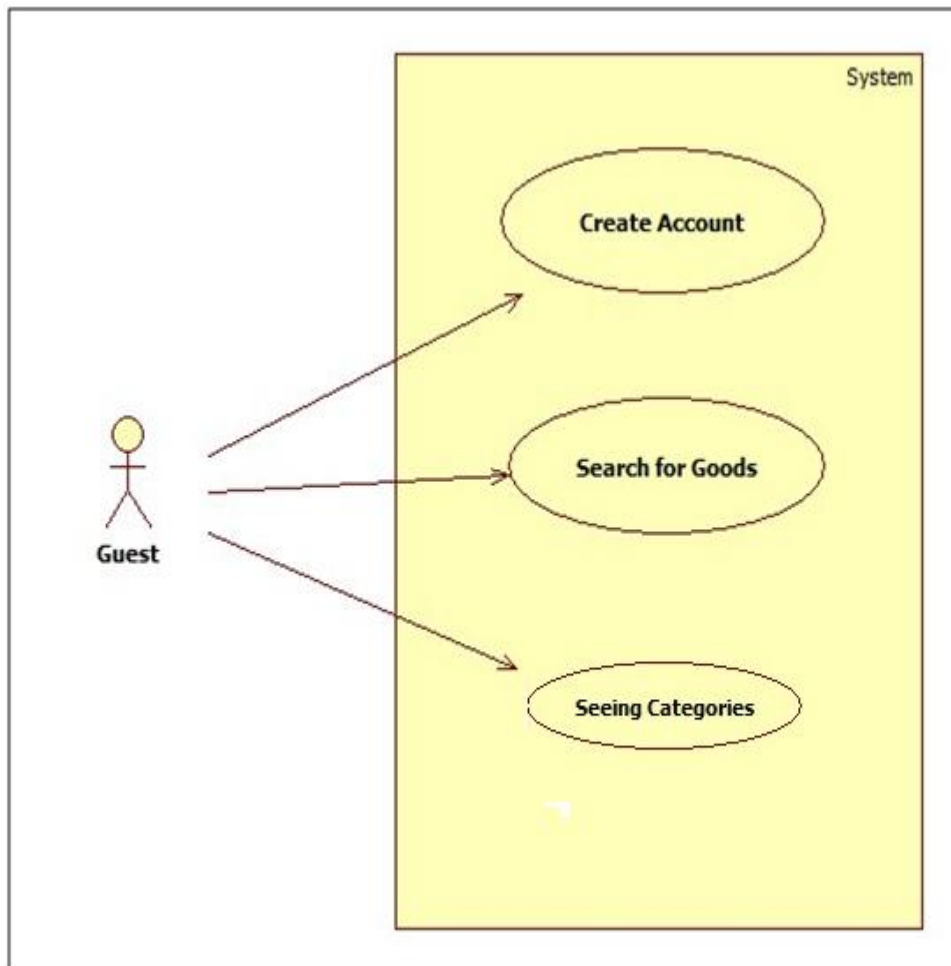


Figure 2: Guest Use Case Diagram

5.2.2 Design Elements

Use Case Name	Login
Actor	User
Description	Registered users are asked to login to the system before posting any goods. By login to the system, registered users can maintain their posts such as deleting post, creating a new post and also taking a look at the rating the guest give.

Table 2: Login Case

Use Case Name	Add Item
Actor	User
Description	This function will allow registered users to add some items to the system. The status of items could be “lend” or “exchange”. If lend option is selected, another user can borrow this item by giving nothing in return. If exchange option is selected, another user can take this item by giving an item, which supplier accepts; in return to the supplier. The supplier needs to give a name to the item. Also he/she can put a picture and give short information. Moreover he/she needs to pick the category, city and county from the list.

Table 3: Add Item Case

Use Case Name	Remove Item
Actor	User
Description	Registered users are able to remove their added items in any time

Table 4: Remove Item Case

Use Case Name	Search For Item
Actor	User
Description	This function allows users to search goods. Users can find items by just clicking the category names from the category menu or they can search an item with some keywords or picking the category, city and county.

Table 5: Search Item Case

Use Case Name	Send Private Message
Actor	User
Description	This function will allow users to send or reply private messages. With this property, users can contact with each other.

Table 6: Send Private Message Case

Use Case Name	Edit Profile
Actor	User
Description	This function will allow registered users to change their personal info such as profile picture, password, name, phone number, city or county information.

Table 7: Edit Profile Case

Use Case Name	Thank to the User
Actor	User
Description	This function enables registered users to thank the other users. For example, if a user gets an item from another user and get benefit from it. Then he/she have a chance to thank to supplier user.

Table 8: Thanks Case

Use Case Name	Search for another User
Actor	User
Description	This method enables users to search for other users with just picking a city and county from the list.

Table 9: Search for another User Case

Use Case Name	Create Account
Actor	Guest
Description	Guests need to create an account before they can post any goods that he/she wants to exchange or borrow. Otherwise he will only be treated like a visitor by a system. This “create account” process is called register in our system. Through register procedure, users have to write their email address, username, real name, password, country, city and county information.

Table 10: Create Account Case

Use Case Name	Seeing Categories
Actor	Guest
Description	Guest users can see the categories and items belong to these categories but they cannot see the detailed information about these items

Table 11: Seeing Categories Case

5.3 Composition Viewpoint

This viewpoint mainly aims to explain logical and physical component of the system in detail and to explain overall system architecture from perspective of each component. UML component diagram is used under this title.

We have mentioned our system will be built on MVC (Model-view-controller) system architecture in system requirements specifications document. The system will consist of three basic components which are GUI (View and controller parts will be on this component), business logic (model part will be on this component) and database. These components can be seen from the overall component diagram below:

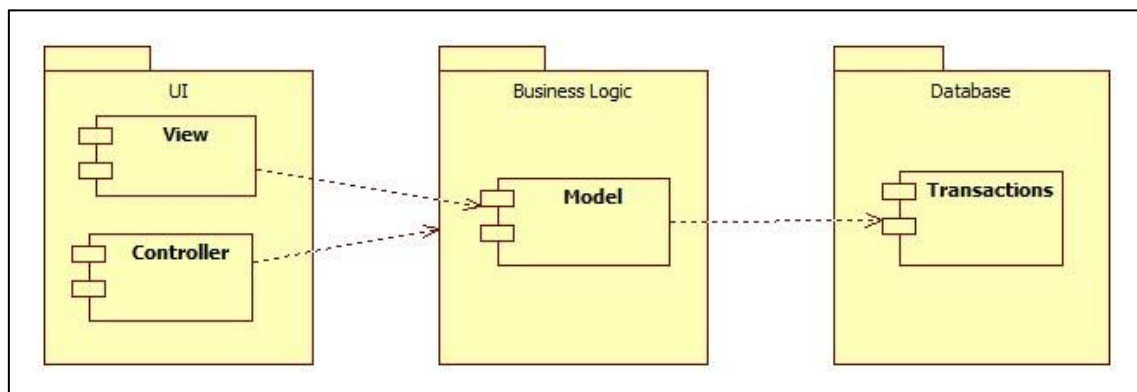


Figure 3: Overall Component Diagram

The detailed UML components diagram with critical user operations and their results included is shown below: (The subcomponents in the diagram will not represent classes of the system architecture; they represent the actual users of the system)

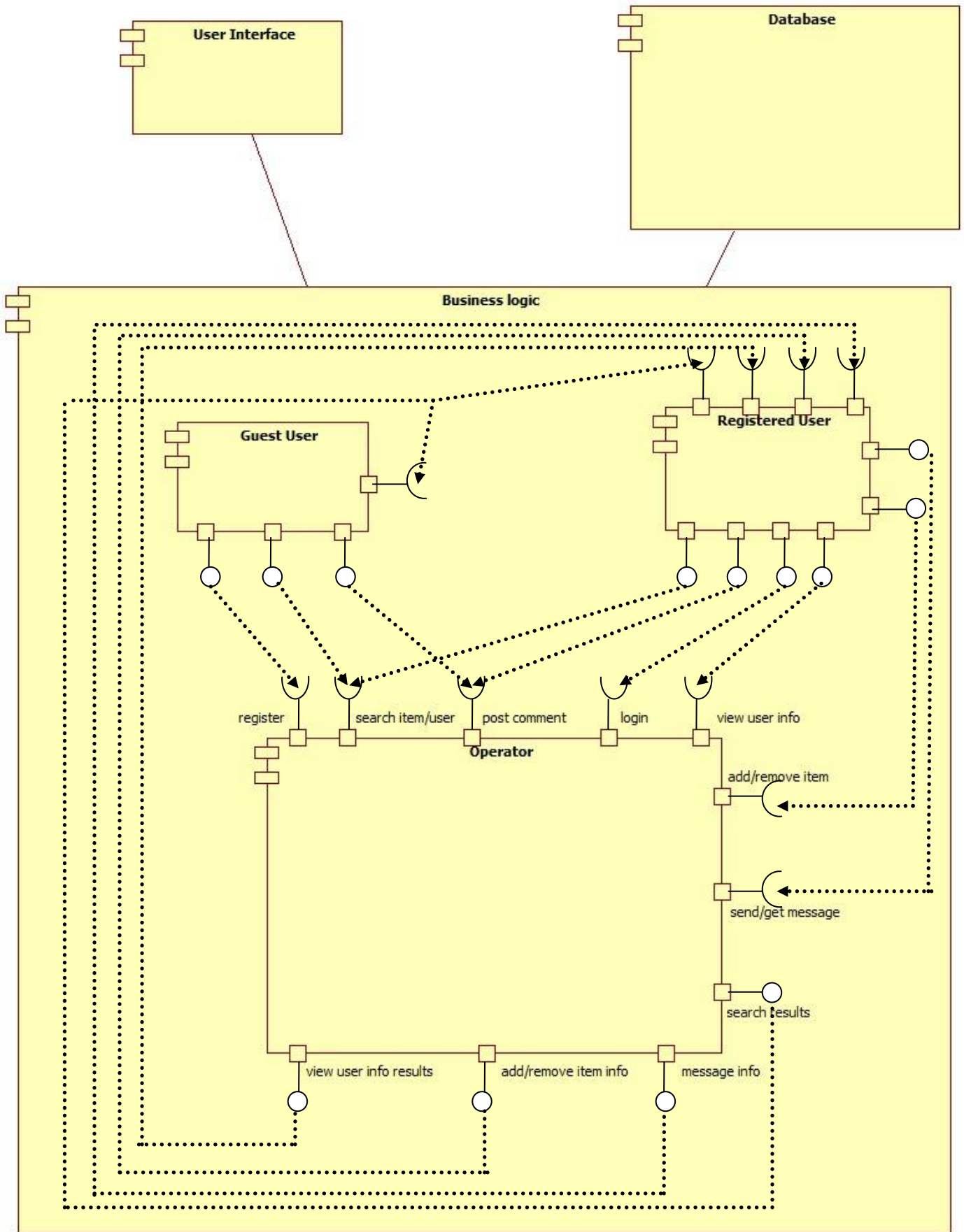


Figure 4: Detailed Component Diagram

As it can be seen from the diagram, there are three main components in business logic. Operator which acts like organizing all the operations on business logic, two different user types which are guest user and registered user and they represent the users of the system and provide some queries to get some results from the operator.

Physical component of the system can be seen from UML deployment diagram as follows:

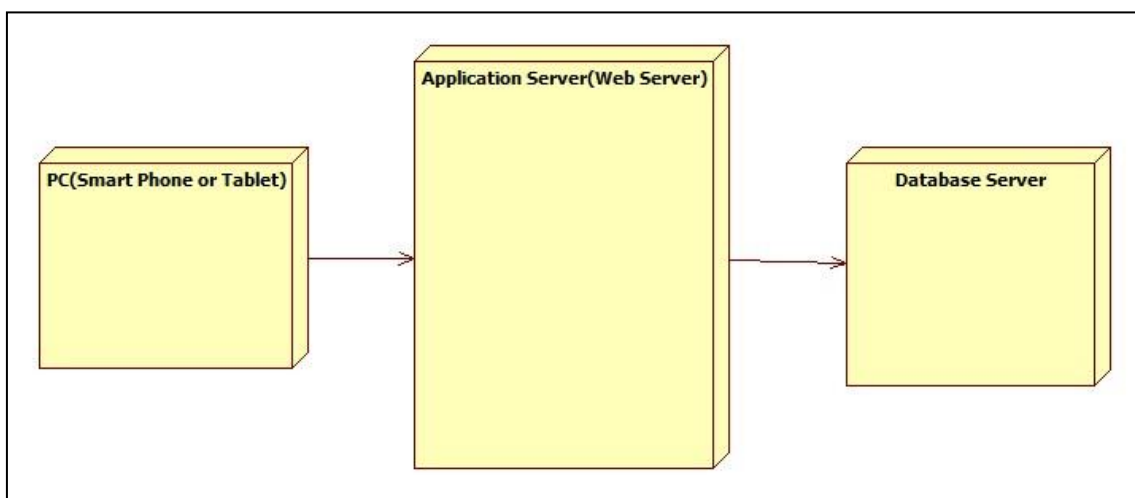


Figure 5: Deployment Diagram

From this diagram we can see that there will be three physical components of our system. PC (Smart phone or tablet) will be the devices people reach our application by using web browsers. Application servers will store our business logic and program information which we discussed earlier and database servers will respond to queries sent by our application program and data flow will be between application server and database server using PHP-MySQL database connectivity.

5.4 Logical Viewpoint

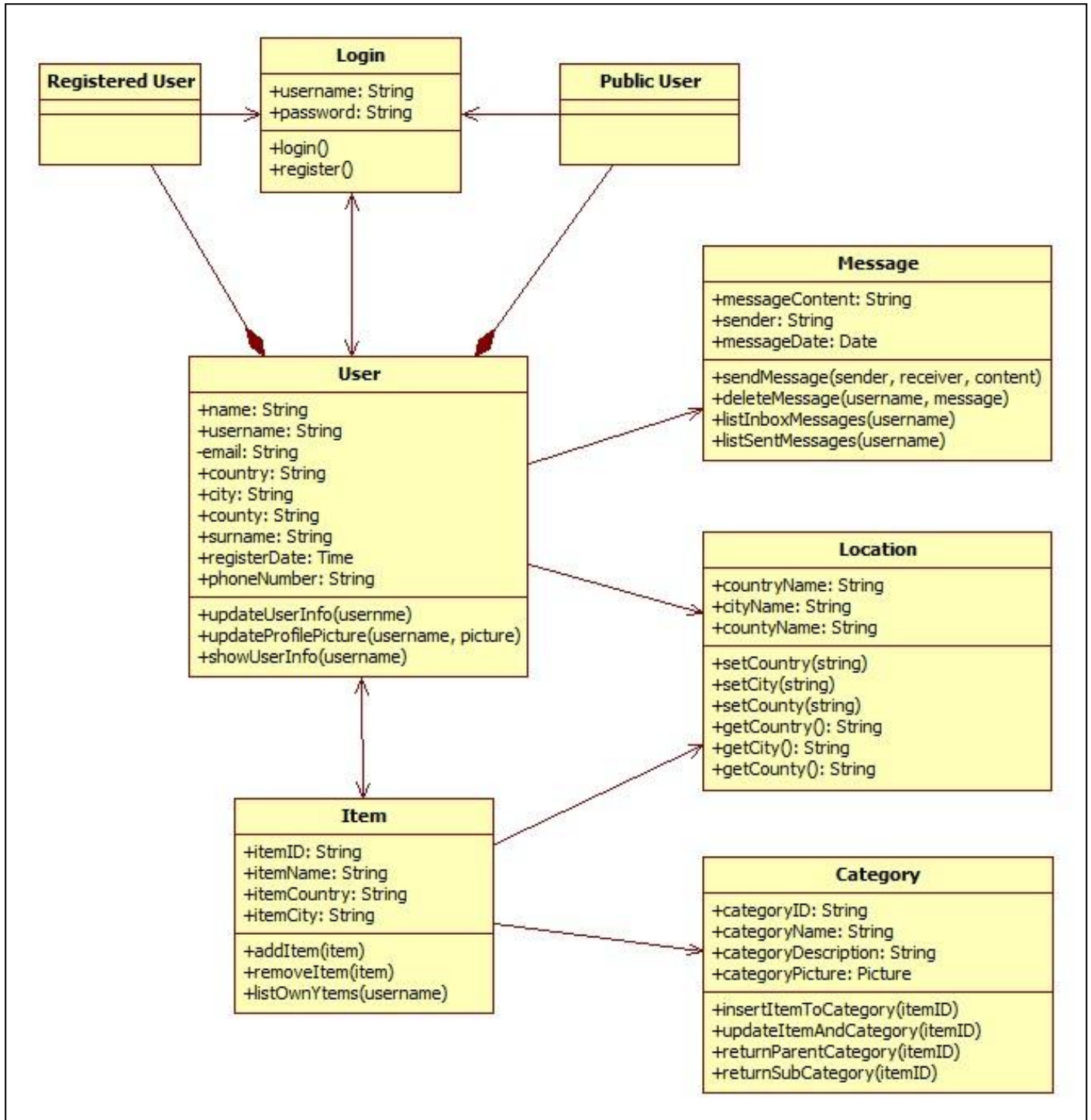


Figure 6: Class Diagram

Method Name	addItem(item)
Description	Registered user adds item

Method Name	removeItem(item)
Description	Registered user removes item

Method Name	listOwnItems(username)
Description	Registered user sees his/her item list

Method Name	updateUserInfo(username)
Description	Registered user updates personal info

Method Name	updateProfilePicture(username, picture)
Description	Registered user updates his/her picture

Method Name	showUserInfo(username)
Description	Registered user sees his/her personal information

Method Name	sendMessage(sender, receiver, content)
Description	Registered user sends a message

Method Name	deleteMessage(username)
Description	Registered user deletes a message

Method Name	listInboxMessages(username)
Description	Registered user sees his/her inbox messages

Method Name	listSentMessages(username)
Description	Registered user sees his/her sent messages

Method Name	Get methods of location class
Description	Registered user sees his/her location info

Method Name	Set methods of location class
Description	Registered user sets his/her location info

Method Name	insertItemToCagegory(itemID)
Description	Item is inserted to category

Method Name	updateItemAndCategory(itemID)
Description	Item is updated in category class

Method Name	returnParentCagegory(itemID)
Description	Return item's parent category

Method Name	returnSubCagegory(itemID)
Description	Return item's sub category

5.5 Information Viewpoint

This viewpoint shows the relationships in the system. The entity relationship diagram is a visual representation of entities and how they relate to each other. The entity relationship (ER) diagram of this system is shown as below;

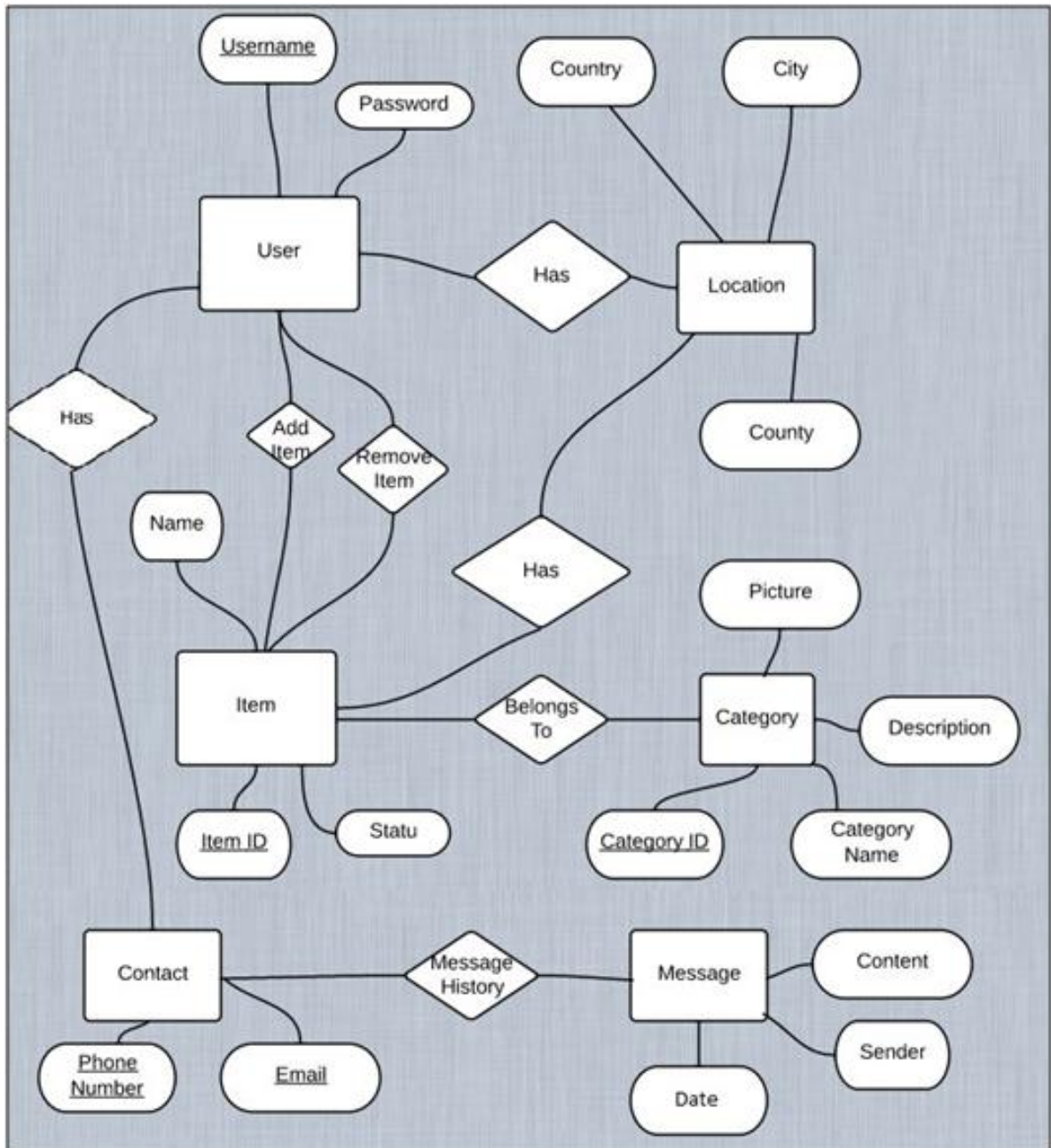


Figure 7: ER Diagram

We created our MySQL database according to this viewpoint. Our database design information is like the following table;

User	<i>User_Number: Int (Auto Increment)</i> <i>Real_Name: Varchar</i> <i>Username: Varchar (Unique)</i> <i>Password: Varchar</i> <i>Email: Varchar (Unique)</i> <i>Phone_Number: Varchar</i> <i>Profile_Picture: Longblob</i> <i>Country: Varchar</i> <i>City: Varchar</i> <i>County: Varchar</i> <i>Register_Date: Timestamp</i>
Item	<i>Item_ID: Int (Auto Increment)</i> <i>Username: Varchar</i> <i>Item_Name: Varchar</i> <i>Status: ENUM("FREE" or "EXCHANGE")</i> <i>Country: Varchar</i> <i>City: Varchar</i> <i>County: Varchar</i> <i>Item_Picture: Longblob</i> <i>Item_Information: Varchar</i> <i>Add_Date: Timestamp</i>
Message	<i>Sender_Username: Varchar</i> <i>Receiver_Username: Varchar</i> <i>Message_Content: Varchar</i> <i>Message_Date: Timestamp</i>
Category	<i>Item_ID: Varchar</i> <i>Main_Category: Varchar</i> <i>Sub_Category: Varchar</i>
City	<i>City_ID: Int (Unique)</i> <i>City_Name: Text</i>
County	<i>County_ID: Int (Unique)</i> <i>City_ID: Int</i> <i>County_Name: Text</i>

Table 12: Database Design

5.6 Dependency Viewpoint

We do not write a dependency viewpoint because in the composition and logical viewpoints, this viewpoint is explained.

5.7 Interaction Viewpoint

Interaction between system components are visualized by using UML sequence diagrams as follows:

5.7.1 Login/Register View

When the user browses the login page there will be two options which are login to the system or register to the system. Following sequence diagram explains which operations will be taken on this situation.

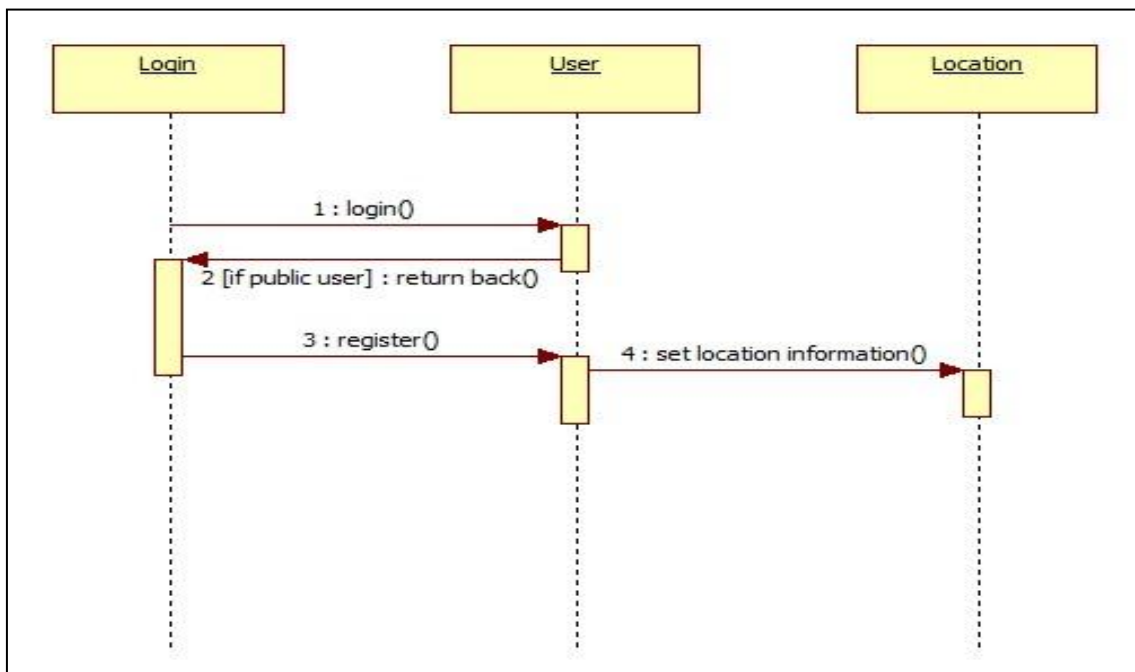


Figure 8: Login/Register View

5.7.2 Add Item View

On the add item view it is provided that only registered users can add items and set their features accordingly (These features are setting location information, item category etc). Following UML sequence diagram shows this.

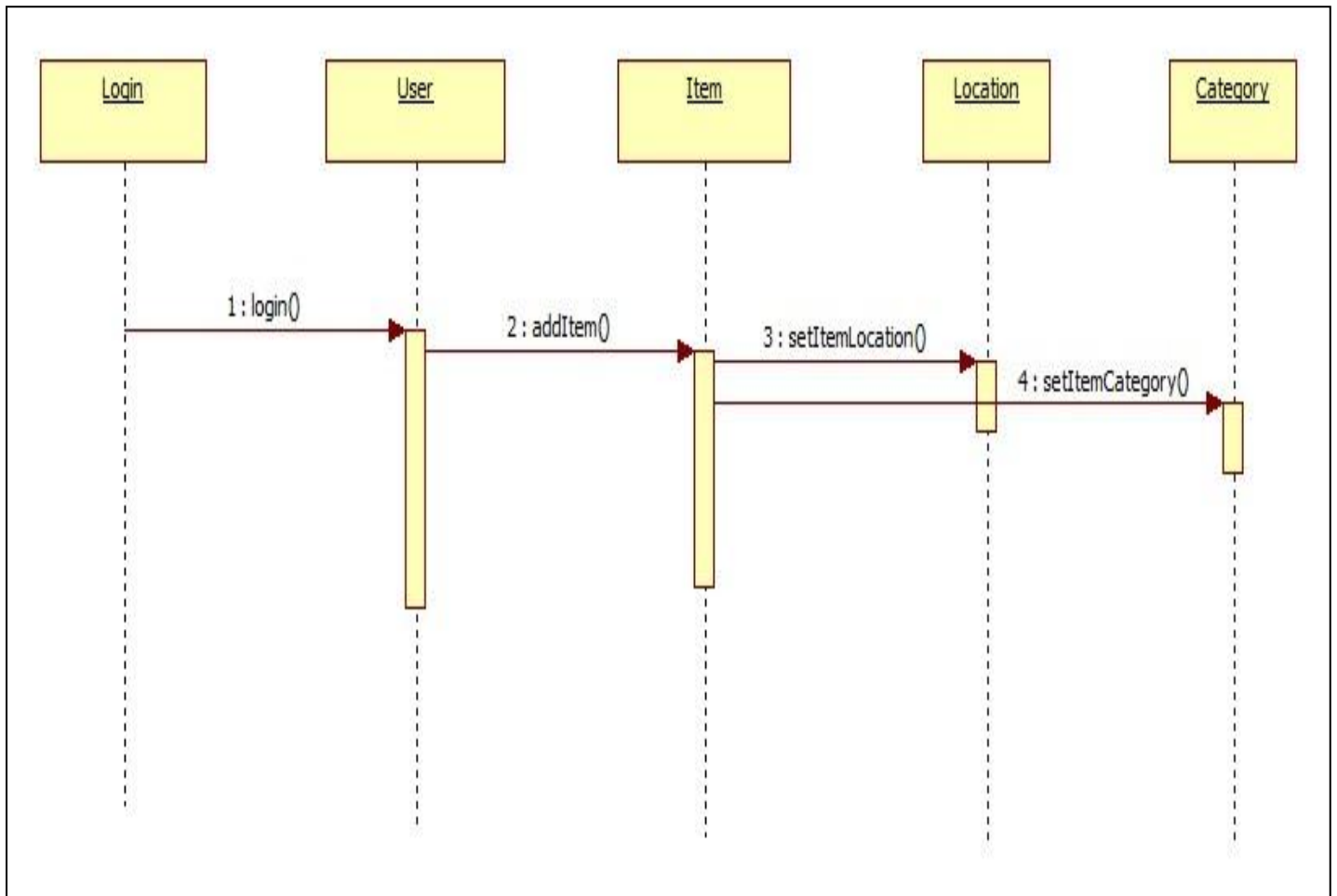


Figure 9: Add Item View

5.7.3 Remove Item View

In this viewpoint registered user can delete an item if the provided item is selected from item list with corresponding location and category information, at the end item deleted message will be sent to user.

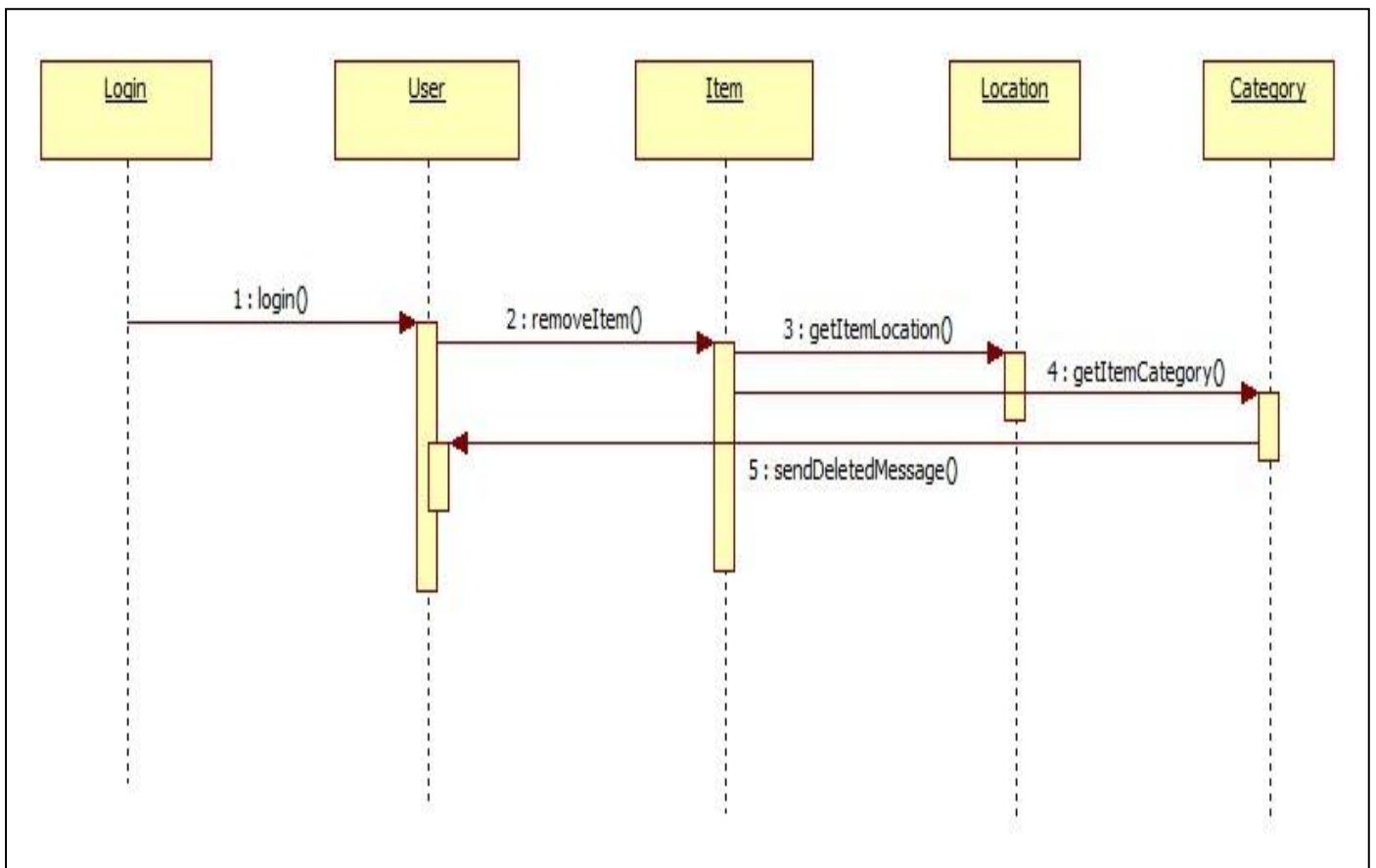


Figure 10: Remove Item View

5.7.4 Search Item View

In this view registered or public all users can search an item on the application after they provide corresponding location, category information about the item. Then, the application will search through the database and return whether the item is found or not as a message on the screen. This situation can be visualized in UML sequence diagram as follows:

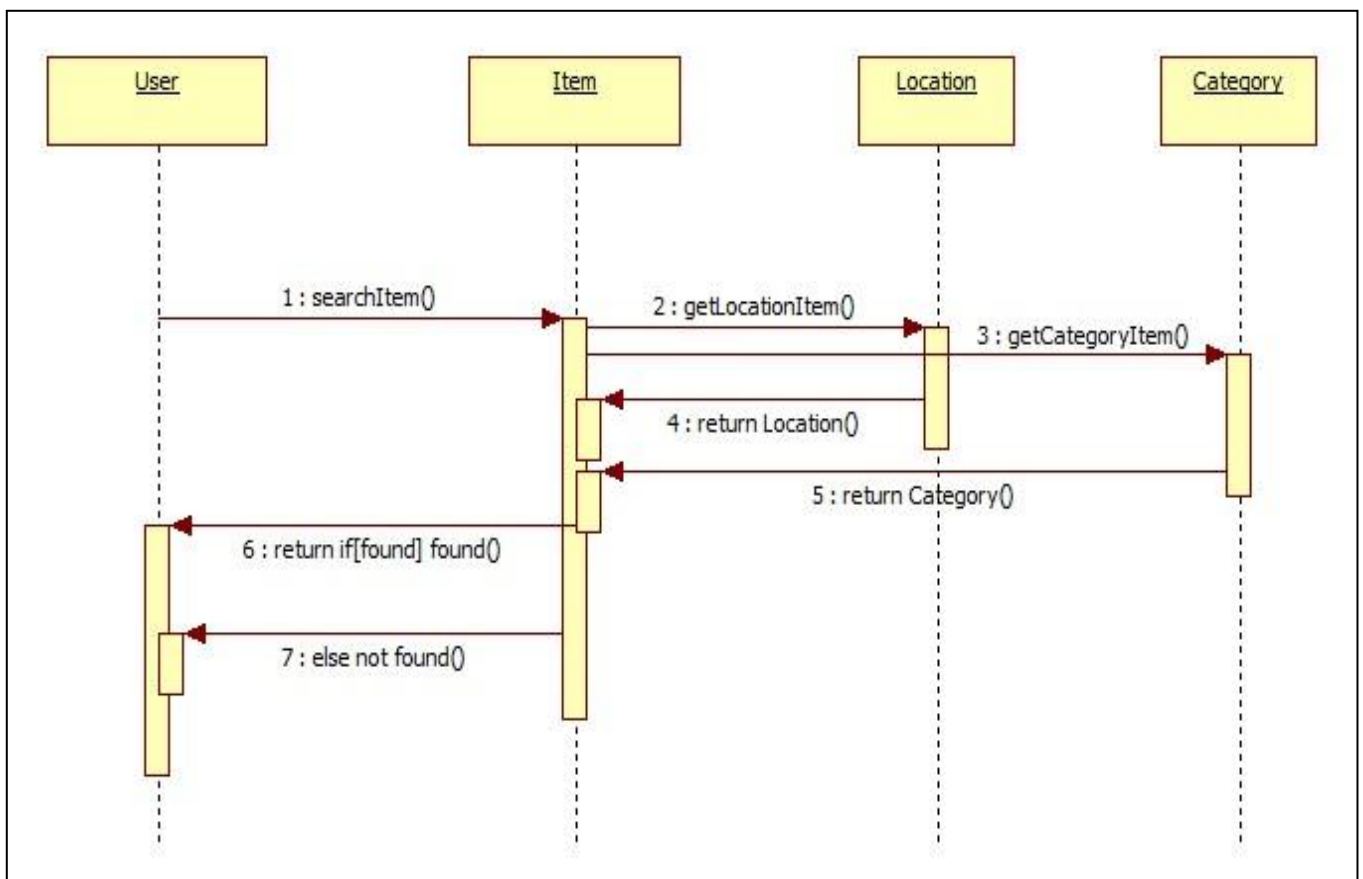


Figure 11: Search Item View

5.7.5 Send/Get Message View

Registered users can send and receive messages on our application. After a user sends and other user gets the message, message content, sender/receiver and message date information will be kept in database. In the following diagram, after two registered users' login to the system, they send messages to each other.

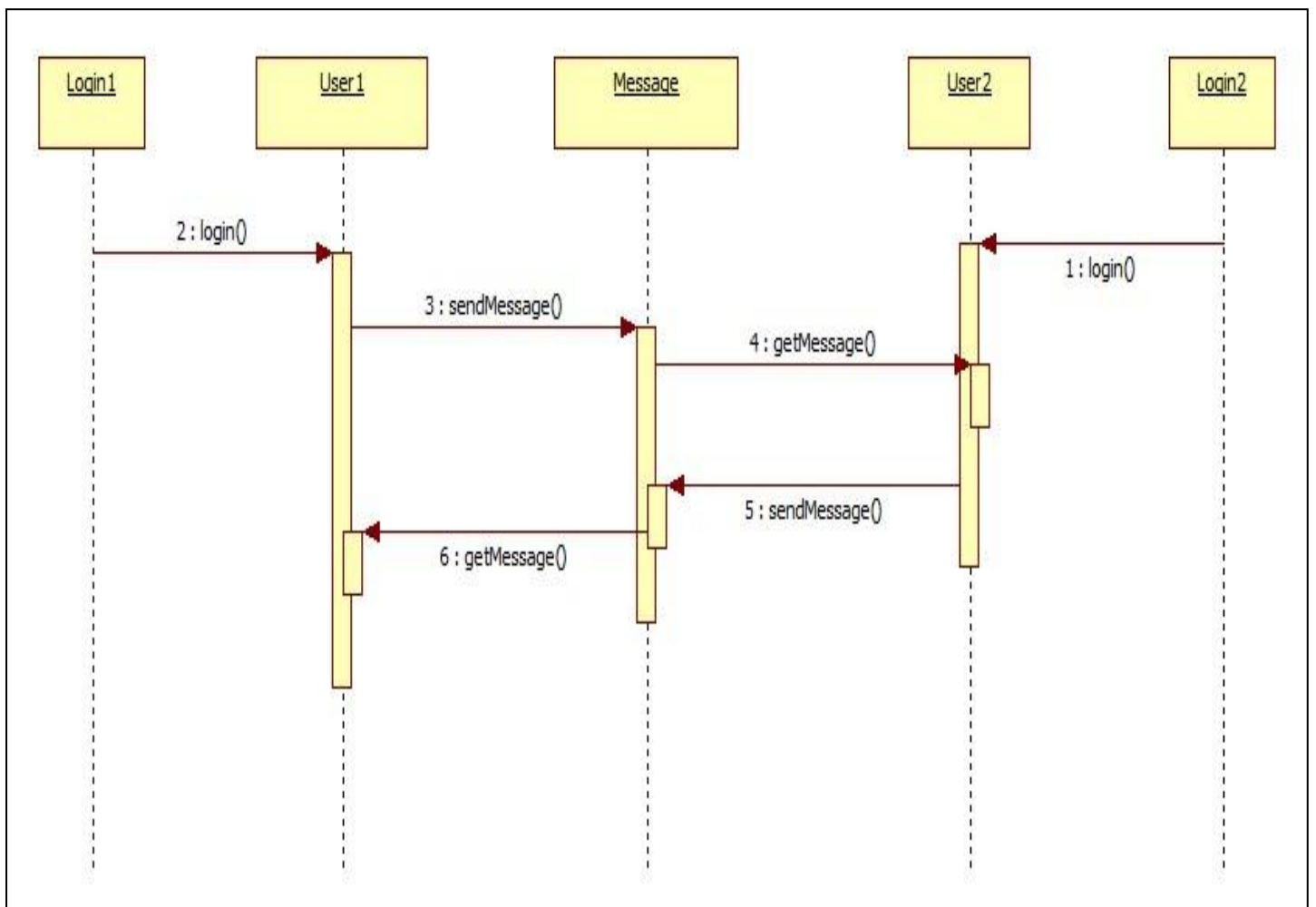


Figure 12: Send/Get Message View

5.7.6 Search User View

Registered or public users can search another user of the application by just providing necessary information about the user (name, surname, location information etc.). If the user searched is found, “found” will be displayed on the user interface, “not found” otherwise. Search user view can be modeled in UML sequence diagram as follows: (In the diagram, “User” object contains list of all users registered to the system)

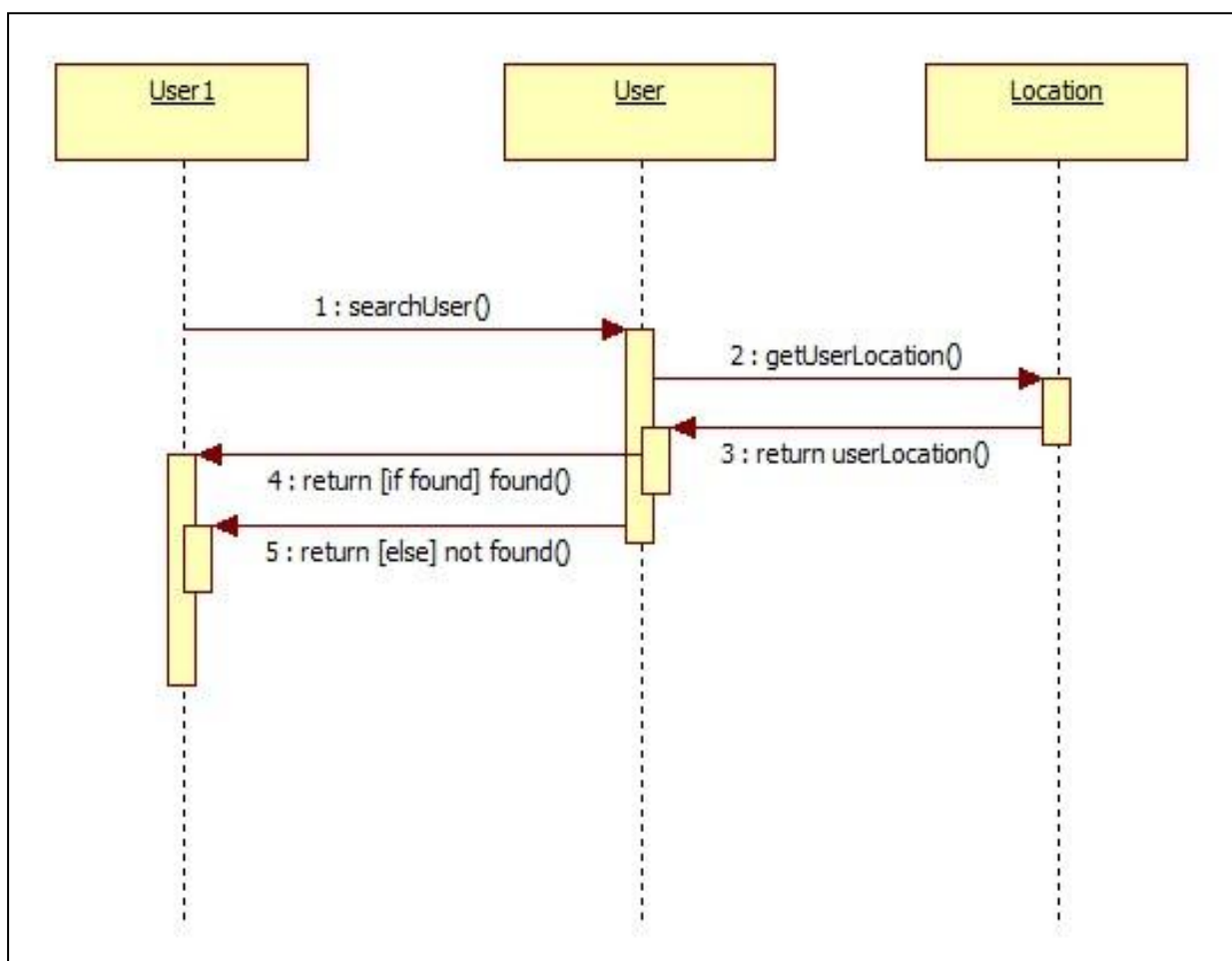


Figure 13: Search User View

5.7.7 View/Update Profile View

In this view, only registered users can access their profile they created and they can edit or update their profiles. They can change their profile information, picture, e-mail address, and password and location information with this function. This action in UML sequence diagram is as follows:

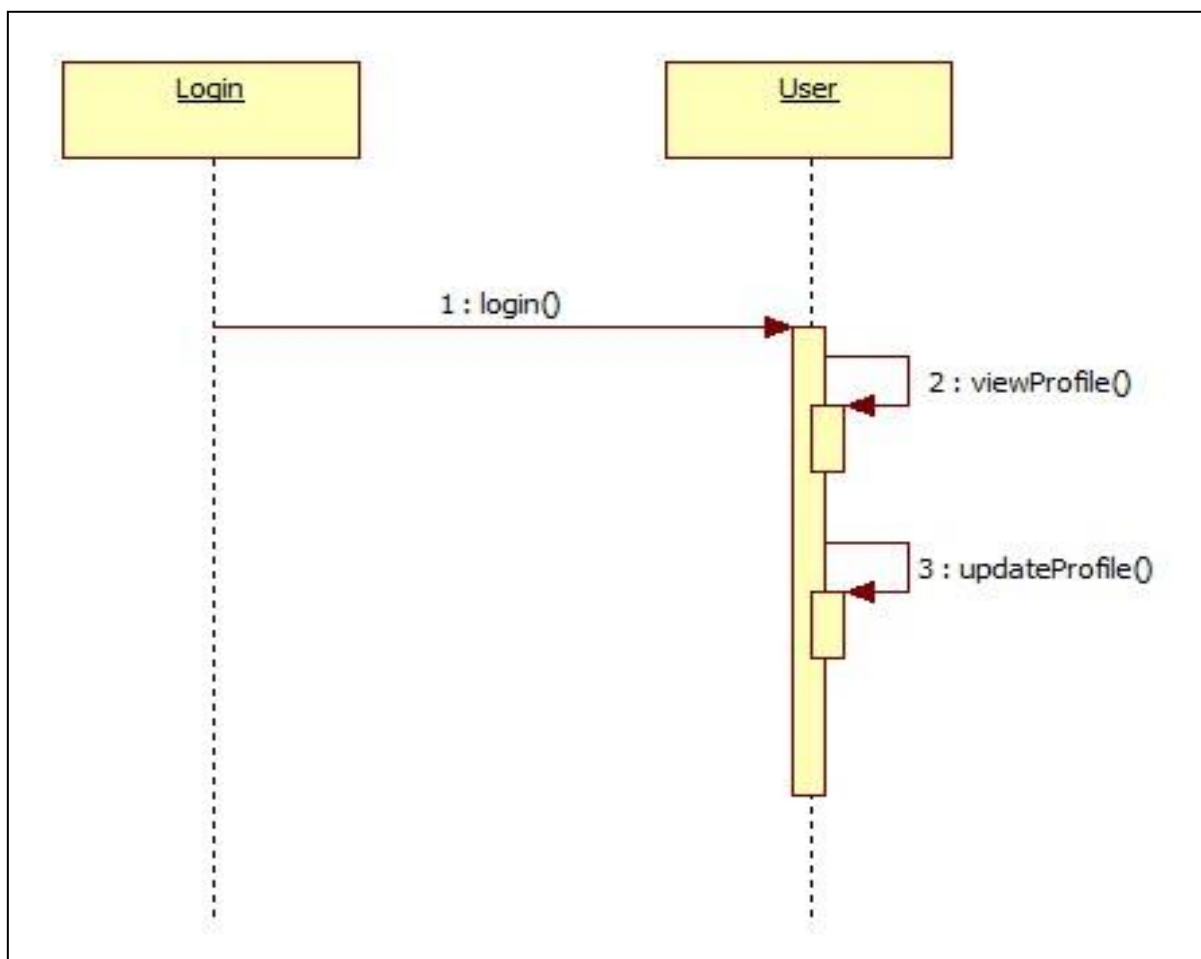
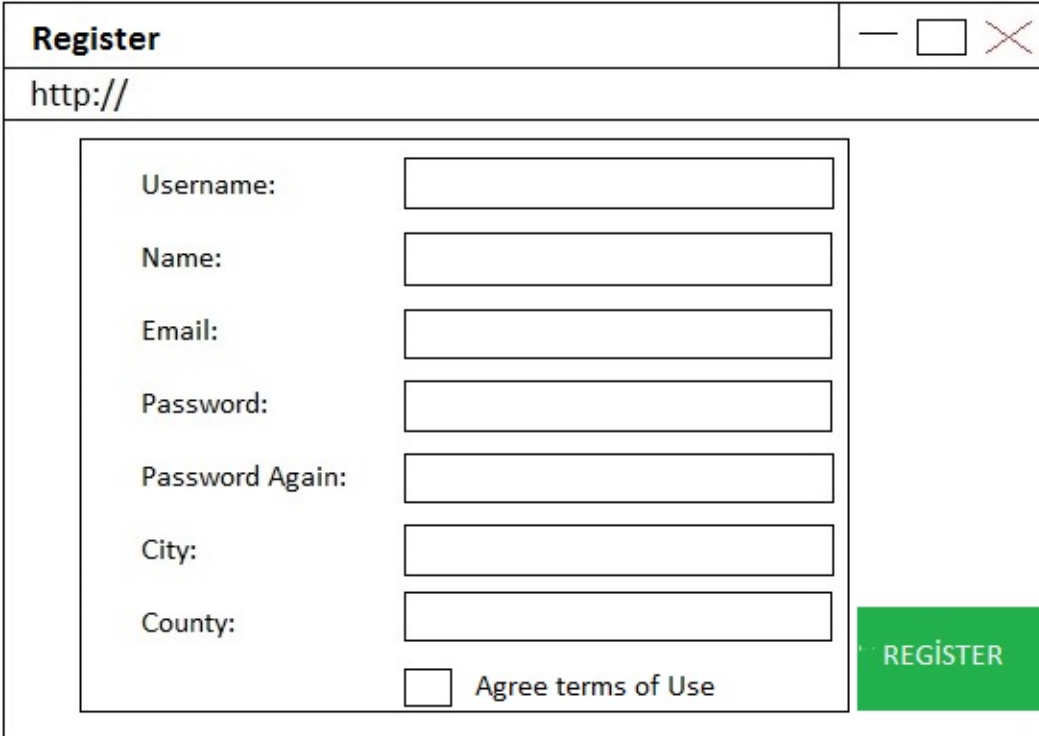


Figure 14: View/Update Profile View

5.8 Interface Viewpoint

An Interface view describes a contract among designers, programmers, customers, and testers. The user interface will be explained in this section with the just simple drawings to demonstrate how the user interface looks like. The images in these sections are for just simple demonstration. Also page styles can be change a little while implementation. Users are able to see the categories from the left panel under that category or search a specific item with some keywords.

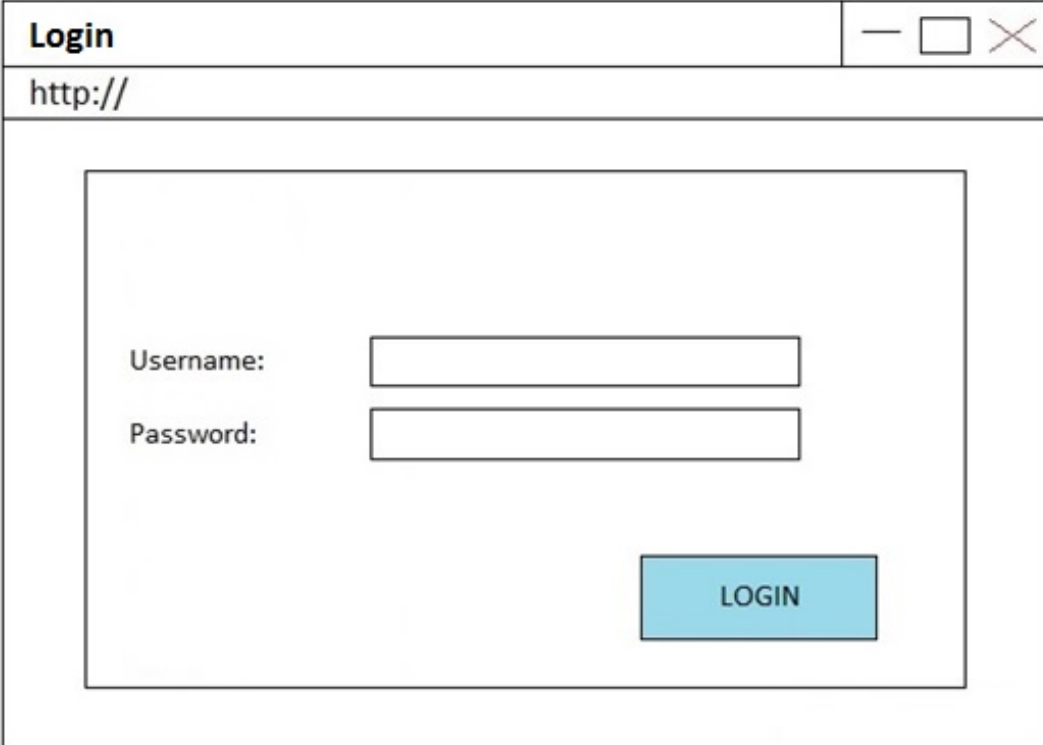
Firstly, if a user wants to post goods or contact with the other users, he/she needs to register the system. Over registration process user needs to enter his real name, email, password, city and county. The register interface will look like as the following figure;



Register		— □ ×
http://		
Username:	<input type="text"/>	
Name:	<input type="text"/>	
Email:	<input type="text"/>	
Password:	<input type="text"/>	
Password Again:	<input type="text"/>	
City:	<input type="text"/>	
County:	<input type="text"/>	
<input type="checkbox"/>	Agree terms of Use	
		REGISTER

Figure 15: UI - Register Page

After the registration, users will get an email for register validation. Then they will click the link in the email. Then login page will be displayed. Also registered users needs to login to the system for reaching their accounts;



The image shows a wireframe of a login page within a browser window. The window has a title bar with the text 'Login' and standard minimize, maximize, and close buttons. Below the title bar is an address bar containing 'http://'. The main content area is a large rectangle containing a smaller rectangle. Inside this inner rectangle, there are two labels: 'Username:' and 'Password:'. To the right of 'Username:' is a horizontal text input field. To the right of 'Password:' is another horizontal text input field. Below the password input field, centered horizontally, is a blue rectangular button with the text 'LOGIN' in white capital letters.

Figure 16: UI - Login Page

After login, the user will be directed to main home page. On the main home page there are;

- “Categories” on the left panel,
- “Home”, “Messages”, “Logout”, “Search” and “Edit My Profile” buttons on the top panel,
- Some random items on the middle

Left categories panel and top buttons panel will always be displayed on the any page in this system. Main home page will look like as the following;

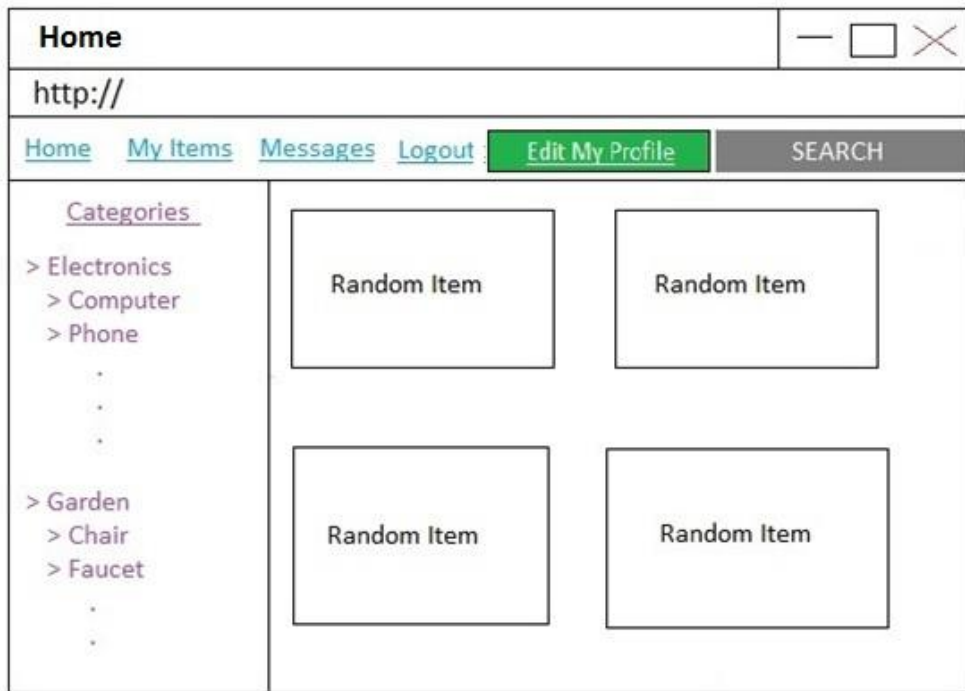


Figure 17: UI - Home Page

Users can reach their profile by just clicking the “Edit My Profile” button. In “Edit My Profile” page, users are able to update their personal information. He/she can add or change the profile picture, phone number, address and also password. This page will be like as follows;



Figure 18: UI - Profile Page

On the search page, users can search an item or other users with some keywords;

Figure 19: UI - Search Page

The interface of “my items page” could be change but the logic and usage will be like the following figure. There is a panel called “Add Item”. Users are able to add an item in this page. When adding an item, users have to enter the name, description, status (exchange or lend), city, county information or optionally a picture. In this page users also remove their items in any time. Lastly, users can see their added items in this page.

Figure 20: UI - My Items Page

5.9 Patterns Use Viewpoint

This viewpoint explains the design pattern we will use in the project. The design pattern is MVC (Model View Controller). MVC is an architectural pattern which separates the application into three main components.

Model: Model objects are the parts of the application that implement business logic. Mostly, model objects are related with the database. They store, remove or show objects of the database.

View: Views are responsible for displaying the application's user interface. This user interface is created from model objects. View is just a presentation of data in a particular format, formed by controller's requests.

Controller: Controllers are bridges between models and views. They handles user interface and make connection between UI and models. In MVC design pattern, the view only shows UI and information but the controller deal with the user input and interaction with model.

For example, when a user makes an action such as pressing a keyboard button to see something, this input is handled by the controller. Controller decides the correct model related that input. Then view displays the output. MVC's three main components and their relations are displayed in the following figure;

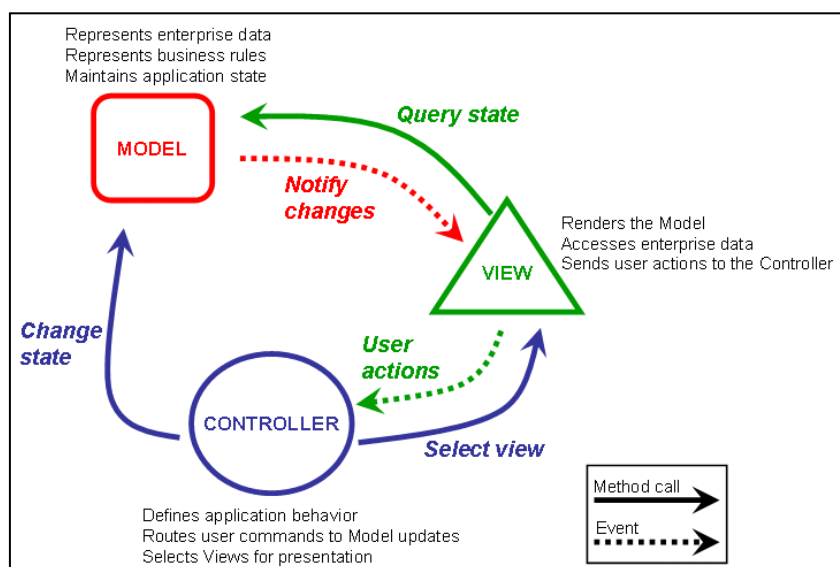


Figure 21: MVC

5.10 State Dynamics Viewpoint

State dynamics viewpoint classifies the software into sub elements and describes the relations between these sub elements. Also, it illustrates the order of execution process. Diagrams in section 5.8.1 and 5.8.2 are created by the UML state diagram rules.

5.10.1 Viewpoint for Login and Register

In order to exchange or borrow goods on our online market, users need to register. However, guests can only see the items, item categories and item locations. For example, these guests cannot add or borrow an item from our market without registration. After registration, every user has individual accounts. Here is the state diagram for login and register process;

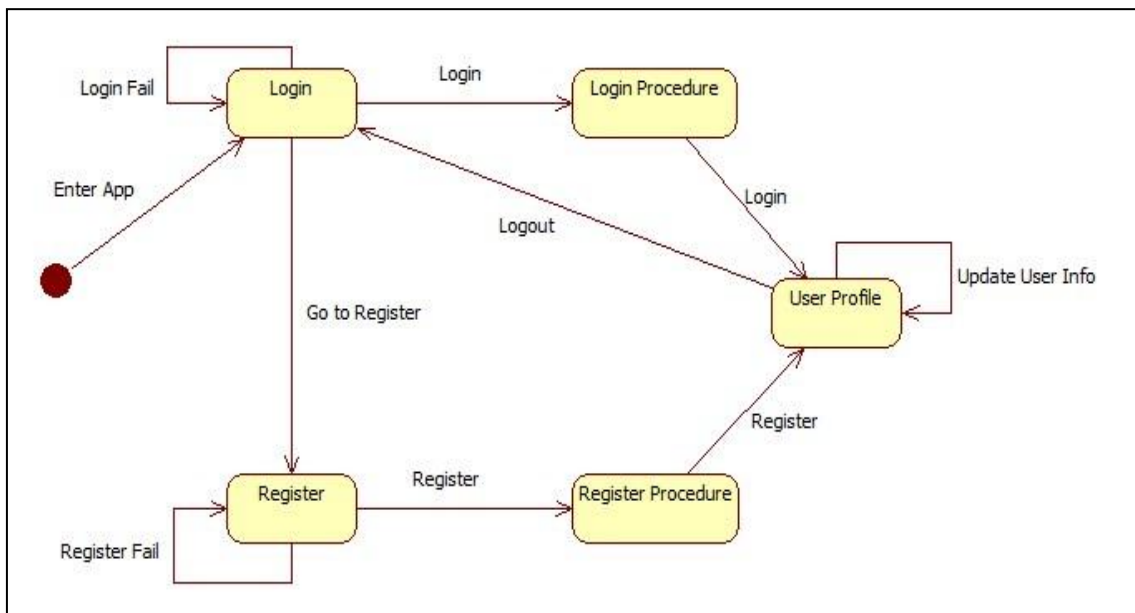


Figure 22: State Diagram for Login and Register

5.10.2 Viewpoint for Whole System

In our system, after registration, users have their own profile. They can edit their profile but they cannot change their username and email addresses. Also they can upload a profile picture. Registered users are able to add, edit or remove their items. Also they can search other items in the system and see the details of these items. Moreover, they can search the other users and see their profile and their added items. Here is the state diagram for the whole system usage;

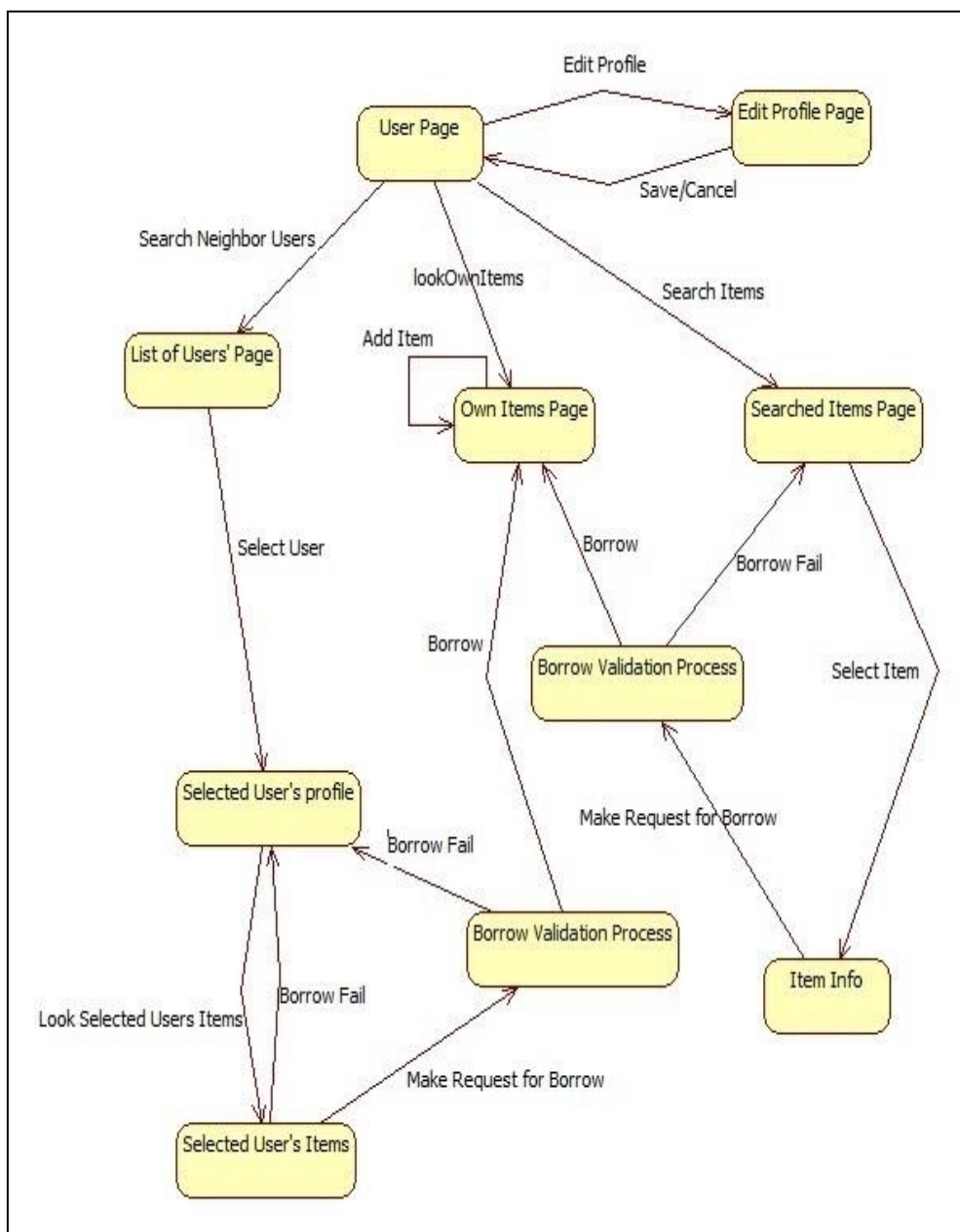


Figure 23: State Transition Diagram for Whole System

5.11 Structure Viewpoint

We do not write a dependency viewpoint because in the composition and logical viewpoints, this viewpoint is explained.

5.12 Algorithm Viewpoint

We do not write an algorithm viewpoint because we will not use a known special algorithm.

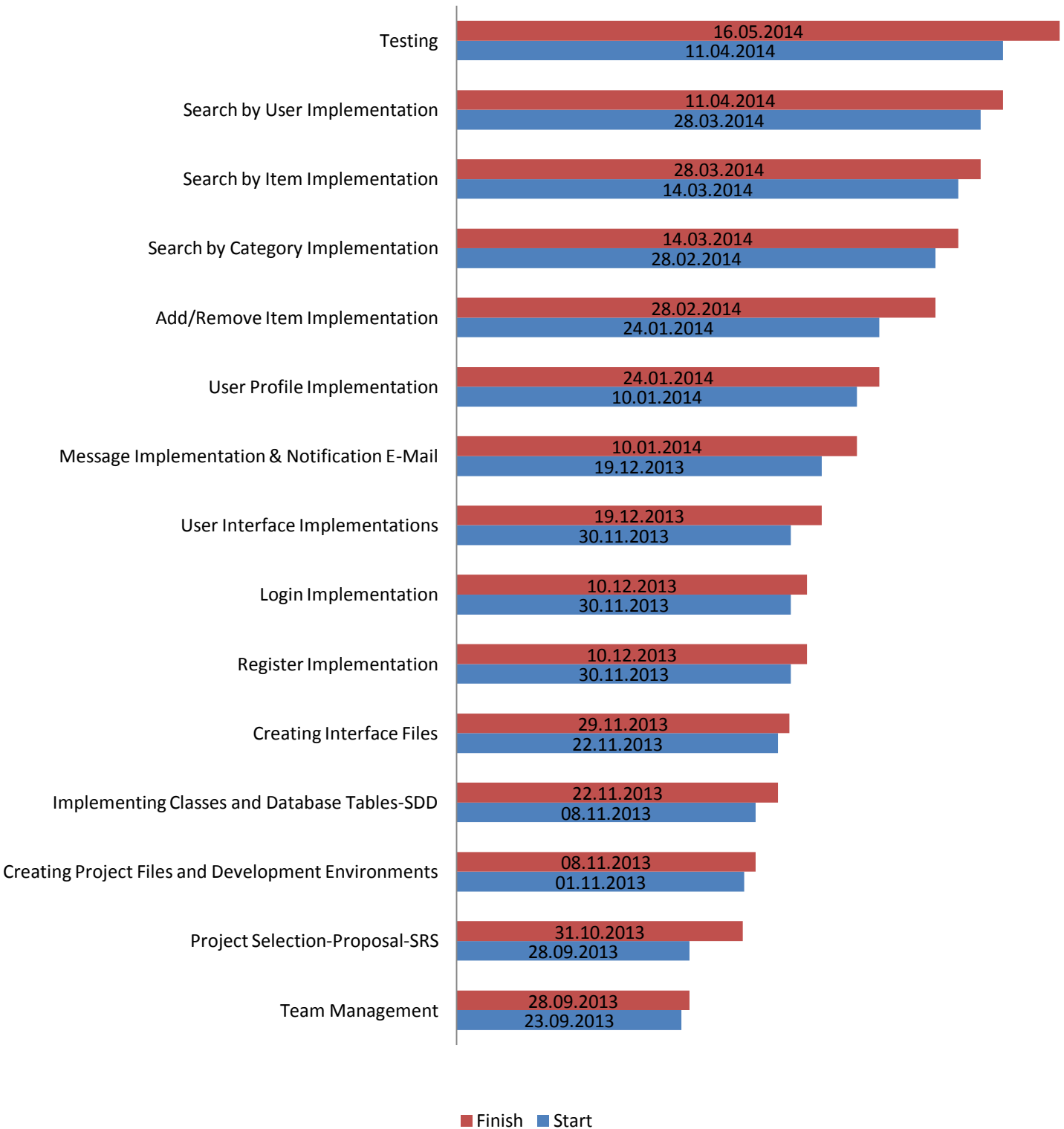
5.13 Resource Viewpoint

We do not write a resource viewpoint because we will not use a known special resource design.

6 Planning

Our planning for whole project is displayed on the following diagram;

Planning



7 Conclusion

This SDD document describes and makes visual representation of the use of each the viewpoint, architecture, requirements defined in SRS, use-case realization, subsystems of the application. Therefore, this document will be used as a guide for the implementation phase. In the design and implementation process, additional requirements can be added as needed for better implementation. If some changes are needed, this SDD document will be updated according to these changes.