**Final Document**

|  |  |
| --- | --- |
| Course | CIS 4911 |
| System | (IBM) Track and keep score of and compute a group's sustainability efforts |
| Team | Jorge McGarry  Monica Del Prado |
| Date | 12-10-2014 |
| Instructor | Masoud Sadjadi |
| Mentor | Juan Caraballo |

Table of Contents

1. Introduction 5

1.1 Problem Definition 5

1.2 Scope of the system 5

1.3 Overall development methodology 5

1.4 Definitions, Acronyms, and Abbreviations 6

1.5 Overview of document 6

2. Feasibility Study 8

2.1 Description of Current System 8

2.2 Alternative Solutions 8

Green Bean Recycle (https://gbrecycle.com/) 8

2.3 Recommendation 13

3. Project Plan 15

3.1 Project Organization 15

3.1.1 Project Personnel Organization 15

3.1.2 Hardware and Software Resources 15

3.2 Identification of Tasks, Milestones and Deliverables 15

3.3 Cost of the Project 16

4. System Requirements 17

4.1 Functional Requirements 17

4.2 Requirement Analysis 20

5. System Design 21

5.1 Overview 21

5.2 Subsystem Decomposition 22

5.3 Hardware and Software Mapping 23

5.4 Persistent Data Management 23

5.5 Security/Privacy 24

6. Detailed Design 26

6.1 Overview 26

6.2 Static model 27

6.4 Dynamic model 28

6.5 Code Specification 34

7. System Validation 35

7.1 Subsystem Tests 35

7.2 System Tests 35

7.3 Evaluation of Tests 37

8. Glossary 38

9. Appendix 39

9.1 Appendix A - Project schedule 39

9.2 Appendix B – Use Cases 39

9.4 Appendix D – Analysis models 56

9.5 Appendix E – Design Models 56

9.6 Appendix F – Documented Class interfaces 64

9.7 Appendix G – Documented code for test drivers and stubs 75

9.8 Appendix H – Diary of meeting and tasks 81

10. References 93

**Executive Summary**

This document begins with an introduction to the ECO Sustainability Efforts system.. Following, is the description of the current system that is being implemented. The next section is all about system design and the methodology used. Furthermore, the requirements of the system are outlined. Additionally, the overall system design, including the primary and secondary architectural patterns, is discussed in detail. The detailed design follows with visual diagrams for better understanding. Testing is explained in the succeeding section with a description of the system and subsystem tests. The end of the document includes a glossary, references, and an appendix.

# Introduction

This chapter explains the purpose and scope of the system. It also includes the definitions, acronyms, and abbreviations used in this document. And finally, it presents an overview of the next chapters.

## 1.1 Problem Definition

Currently, there is no software application that tracks and compares people’s recycling efforts. We want to build an application that tracks recycling efforts and encourages people to use it by providing a competitive approach.

Since recycling and keeping track are completely voluntary we also want to build in some sort of reward aspect into the application, to entice people to keep track of their efforts.

Our clients, FPL and IBM, want to increase recycling efforts and awareness. They want to build a platform that validates, tracks and rewards recycling initiatives. They not only want to do this at the individual level, but also at the corporate, college and community levels.

## 1.2 Scope of the system

The purpose of the system is to be help environmentally conscious users or groups of users such as communities, universities or any company to keep track of their green fingerprints. It will allow them to analyze and visualize data corresponding to their recycling activities.

## 1.3 Overall development methodology

|  |  |
| --- | --- |
| **Term** | **Definition** |
| AngularJS | MVC framework for web development |
| JavaScript | A client-side web development language that is used to create interactive websites. |
| IBM Worklight | Development platform used o create a web appplication |
| IBM Worklight Server | Web Server |
| Datatables | Used for UI tables |
| ChartJS | JavaScript chart library |
| MySQL | Open source relational database management system. |
| HTML | Markup language to design web pages |
| CSS | Used to style pages |
| Bootstrap | Framework to create responsive mobile applications |

## 1.4 Definitions, Acronyms, and Abbreviations

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Guest User | A user that has not created an account in the system |
| Registered User | A user that has created an account in the system |
| Group Owner | A user that created a group is automatically that group’s owner |
| Group Member | A registered user that joined a group. |
| UML | Unified Modeling Language |
| Class diagram | Describes the system in terms of classes, attributes, and their relationships. |
| Server-side architecture | An architectural pattern that consist of a server handling requests from clients . |

## 1.5 Overview of document

- Section 2 explains the current systems that exist that have the similar function as this system but fall off in one category or another.

- Section 3 details the organization of the project including the roles of team members, hardware and software requirements as well as milestones and deliverables.

- Section 4 specifies functional and non-functional requirements and create use cases to describe the interactions between actor(s) and system.

- Section 5 details the proposed software architecture of the whole project. This included the subsystem decomposition, hardware and software mapping, data management and Security and Privacy.

- Section 6 elaborates on the detailed design of the PaLS system. This chapter includes an overview of the process as well as object interactions and class design. This chapter also includes minimal class diagrams for the subsystems being implemented as well as sequence diagrams for object interactions to be implemented.

- Section 7 describes different types of software testing.This chapter includes an overview of testing tools,subsystem test and evaluation of test.

- Section 8 is the glossary of terms used in the document, with an emphasis on domain specific terms.

- Section 9 is the appendix. This chapter contains the use case diagrams (both for functional and non-functional requirements). Detailed interface diagrams are shown for each class. Also in this chapter code for the subsystems being implemented is included. The document ends with a diary of meeting and tasks.

# Feasibility Study



## 2.1 Description of Current System

There are not previous versions of this system. This project will be the initial code base of the system.

## Alternative Solutions

**Description of Alternatives**

### Green Bean Recycle (https://gbrecycle.com/)

This a company that created software and hardware centered on recycling. The basic way it works is, you buy a specialized recycling machine they built and this machine is tied to software. End users register themselves online or through a mobile application and as they recycle containers the “Redemption Value” of the container they recycled is automatically given to their paypal, student account or donated to their favorite charity.

The software has a game concept built in, based on a Point system, see Figure 1. Users, teams and locations can compete against each other based on this point system.



**Figure 1:** Game Concept



**Figure 2:** Game Concept



**Figure 3:** Game Concept

Lastly, the company has sponsored itself with several local merchants and now they offer “Prizes” for a certain amount of recycling effort. Prizes include a free “Burrito”, “IPad Mini”, or gift cards ranging in $10 - $30 from various places.

One form of challenges is based on being the top 5 or 2 recycling person on a certain day. They also have a challenge where they specify the dates ranges, start to end date, for the challenge.

Another form of a challenge is they have is a weekly challenge where you could enter a “Pot”. The benefit of this is, that at the end of the week, its winner take all. The incentive is that you can increase your winnings if you enter the pot.

They also provide incentives, such as doubling your “Redemption Value” for each container between certain dates.

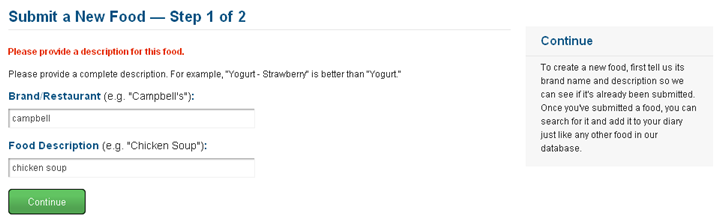
All of the challenges and incentives are based on location. Since they are only mainly on school campuses now, they will have a disclaimer where it says “Participating schools are ..” which means they base it on location.

#### MyFitnessPal (http://www.myfitnesspal.com)

FitnessPal is an application for tracking daily food in order to reach a desired weight based on the user’s current height, weight profile.

It has one of the largest and most complete food databases to choose from. Each food item also has accurate and detailed specifics of the makeup of the food. This information is usually the information you would find on the back of a food packaging. The items are created and kept up to date by the community at large. Any user can create food items which can be used immediately by the user, but have to be approved to be used by the community at large.

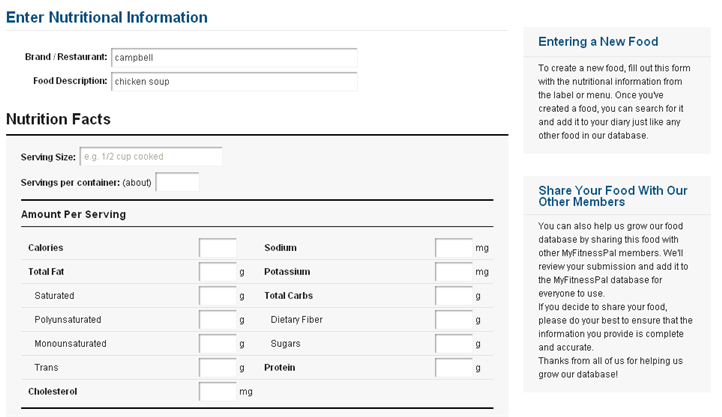
The food creation process is detailed in the below figures.

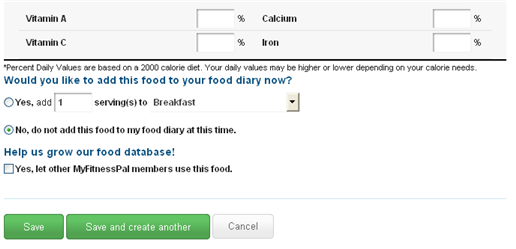


**Figure 4:** Create Food Item – Step 1



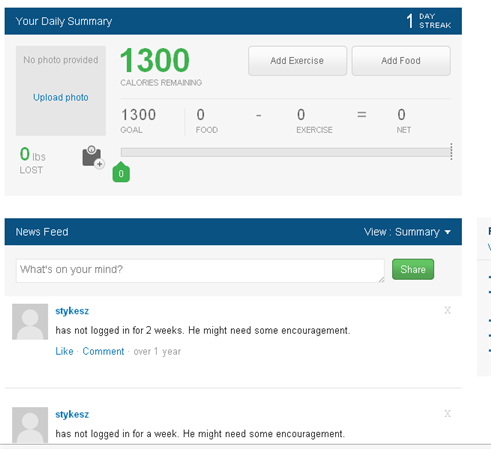
**Figure 5:** Create Food Item – Step 2





**Figure 6:** Create Food Item – Step 3

Aside from this it has a game concept as well, where calories act as points. It keeps tracks of your calories by giving you a very basic dash board showing you your current intake and what your goal is for the day, see Figure 5. If you don’t enter any information it gives you some passive-aggressive encouragement, as you can see in Figure 5 as well.



**Figure 5:** User Home Page

## 2.3 Recommendation

* Selection Criteria

Game component on application:

1. Keep point-based application where groups can keep track of their efforts. track of their recycling efforts.
2. Keep an additional level-based system where the user can advance among levels
3. Challenge other users or teams.

Validation of recycling action:

1. Taking pictures while recycling an item.
2. Scanning items’ barcode.
3. Using geo-tag location to assign recycling efforts to a specific group.

* Analysis of Alternatives

Game component on application:

Alternative 1: Extra features could be added without affecting the ability to add alternative 2 and 3.

Alternative 2 and 3: Given the time constraint on this project, these alternatives could be added on future releases if needed.

Validation of recycling action:

Alternative 1: It would still require manual input from the user and verification from the group administrator to approve the scanned item.

Alternative 2: The barcode will provide a validation step on the process of recycling an item. The limitation will be that only items with barcodes will be accounted for.

Alternative 3: The geo-tag location approach could be effective since it will allow to log items into the appropriate group, so individual actions will not count towards their group contribution.

* Recommendations

Game component on application:

Start with a point-based system.

Validation of recycling action:

Even though bar code approach has some limitation, it will be a suitable way of validating user’s recycling actions. Geo-tag location will provide a way of assigning items to appropriate groups.

# Project Plan



## 3.1 Project Organization

### 3.1.1 Project Personnel Organization

Monica Del Prado will be working on all the front-end website aspects of the application.

Jorge McGarry will be working on all the front-end mobile aspects of the application.

Monica Del Prado and Jorge McGarry will work jointly between the backend aspects of the application, for both the website and mobile platforms.

Monica Del Prado and Jorge McGarry will work jointly on the documentation of the application.

### 3.1.2 Hardware and Software Resources

Hardware:

* Windows, Mac, or Linux desktop or laptop computers

Software:

* Eclipse JEE (Kepler, Juno or Luna)
* Worklight plugin for Eclipse
* MySQL
* Apache

## 3.2 Identification of Tasks, Milestones and Deliverables

|  |
| --- |
| **Tasks** |
| Requirement Elicitation |
| Requirement Analysis |
| Feasibility Document (Deliverable 1) |
| Use cases, minimal class diagrams and sequence diagrams |
| Requirement Document (Deliverable 2) |
| System Architecture |
| Database Design and Setup |
| System Design |
| Implementation of General Use Features |
| Implementation of Individual Features (Website and Mobile application) |
| Design Document (Deliverable 3) |
| Test Cases Design |
| Test Implementation |
| Final Document (Deliverable 4) |

Table 3‑1. Project tasks

|  |  |
| --- | --- |
| **Milestone** | **Date** |
| Requirements Specification | 09/19/14 |
| System Design | 10/03/14 |
| Implementation | 11/15/14 |
| Testing | 12/08/14 |
| Project Completed | 12/11/14 |

Table 3‑2. Project Milestones

## 3.3 Cost of the Project

No cost has been associated for this project. Personal computers and open source software will be used for the development of the application.

# System Requirements

The following section provides information about the functional and non-functional requirements of the proposed system. These requirements were analyzed to create the non-functional requests with regards to: usability, reliability, performance, and supportability.

## 4.1 Functional Requirements

* **Allow user to register**
* Usability: The register form is simple and easy to follow.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 seconds.
* Supportability: The system should be easy to maintain
* **Allow users to edit their profile**
* Usability: Data displayed should be easy to follow. Each user can see only their profile.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow users to view recycling locations in a map**
* Usability: The interface should be simple.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow users to create groups**
* Usability: Any user should be able to create a group.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 2 seconds.
* Supportability: The system should be easy to maintain.
* **Allow users to edit group information**
* Usability: The edit form is simple and easy to follow.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1-5 seconds, depending on the file size.
* Supportability: The system should be easy to maintain.
* **Allow users to view the groups they have joined**
* Usability: Users are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow users to search groups to join**
* Usability: Users are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow users to join groups**
* Usability: Users are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow groups owners to reject requests**
* Usability: Group owners are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow groups owners to accept requests**
* Usability: Group owners are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow users to recycle items**
* Usability: Recycling item form is easy to follow.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow users to view group activity**
* Usability: Group activity is presented in tabular or graphical interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 3 seconds.
* Supportability: The system should be easy to maintain.
* **Allow site admin to create materials**
* Usability: Site admin is presented with material creation form. The data is validated to ensure consistency in the system.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second. It should take less than 1 minute to complete.
* Supportability: The system should be easy to maintain.
* **Allow site admin to create items**
* Usability: Site admin is presented with item creation form. The data is validated to ensure consistency in the system.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second. It should take less than 1 minute to complete.
* Supportability: The system should be easy to maintain.
* **Allow Contest admin to create contests**
* Usability: Contest admin is presented with contest creation form.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second. It should take less than 1 minute to complete.
* Supportability: The system should be easy to maintain.
* **Allow to view contests in the system**
* Usability: Users are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow to view materials in the system**
* Usability: Users are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow to view items in the system**
* Usability: Users are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow to search groups in the system**
* Usability: Users are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Allow to search users in the system**
* Usability: Users are presented with a simple interface.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Require a username and password to log into the system (security)**
* Usability: This is required for a user to log in.
* Reliability: The system should perform correctly 99% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.
* **Require login before viewing data (security)**
* Usability: Security measure for system.
* Reliability: The system should perform correctly 9% of the time.
* Performance: The system should respond within 1 second.
* Supportability: The system should be easy to maintain.

## 4.2 Requirement Analysis

The requirement analysis for the system consisted of determining what features the system should have. Since the application is a first version, basic functionalities have been selected to develop the use cases that will be implemented. The user must be able to log in, edit their profile information and log out. Once the user has registered, should be able to create groups, be part of other groups , accept or reject requests from other users that want to join their group. Create and access material and items in the system and also be able to recycle items.

An important feature of the system is the ability to visualize all the data and the usage of the application and the competitive aspect of the application. For this purpose, a dashboard will be created and users will be allowed to create contests and participate in them.

# System Design

In this chapter, the system design is explained in detail. Package diagrams are used to represent the major subsystems, which will be described in the next section along with the corresponding use cases, and then provide an overview of these subsystems. A deployment diagram will help to visualize the mapping between hardware and software.

The next two sections are related to stored data and security aspects of the system. The persistent data management and security requirements applicable to the storage of this data are described. Finally, security and privacy concerns and how they are managed on this system are explained.

## 5.1 Overview

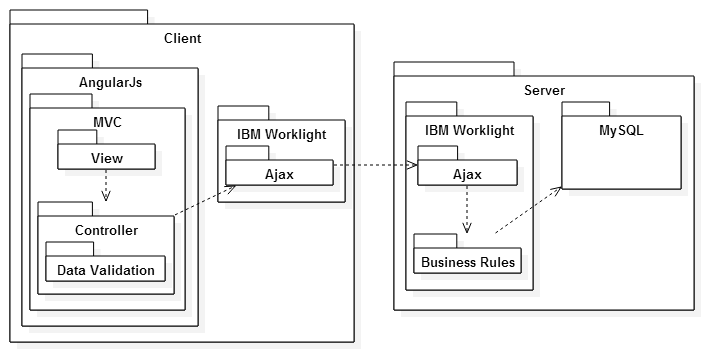
The primary system architecture for this system will be the Server-Client Architecture and the secondary system architecture will be the MVC used on the client side. Both are used in the creation of the system due to the use of IBM Worklight Server. Both architectures are discussed in greater details throughout this section.

*Server-Client Architecture*

The Server-side of the system is provided by IBM Worklight Adapter framework. The Adapter is in charge of communication with the database and in charge of business logic of the application.

*MVC Architecture*

The MVC architecture was used on the Client side to ensure separation of concerns and maintainable code. A simplified diagram is provided below to visualize the decomposition and system architecture or the application



Server-Client Package Diagram. MVC Architecture on client-side

## 5.2 Subsystem Decomposition

The application is split into group administration subsystem, contest subsystem, recycling subsystem, dashboard subsystem.

The user administration is in charge of user-required functionalities such as log in, authentication verification and basic functionalities such as edit user profile.

The group administration subsystem is in charge of allowing users to perform group related functions, such as creating groups, editing groups information. The information related to this subsystem is stored in the database and functions such as edit group profile and setting user roles is based on permission, which is stored in the database and assigned to each user.

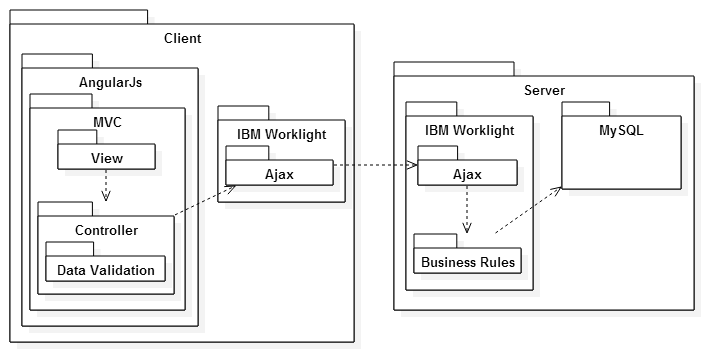
The recycling subsystem is related to recycling actions, items and materials administration. It allows registered users and administrators to recycle items in the system with consistent information. The data related to item and material is verified on the database to provide a reliable system.

The contest subsystem allows group owners and authorized users to launch contests and keep track of contest activities. The information of the contest is stored in the database and verified is consistent with user expectation.

The dashboard subsystem allows the user to access the information about site usage providing an intuitive interface that connects different components of the system. Groups, users, items information is accessed through the dashboard subsystem and customized according to user’s needs.

## 5.3 Hardware and Software Mapping

The system was implemented using AngularJS on the client side implementation for the MVC pattern. Javascript libraries using AngularJS directives were used to ensure consistency and supportability code. The IBM Worklight Adapter provides a layer of security allowing to implement server side security requirements on that component and allowing communication with the MySQL database.

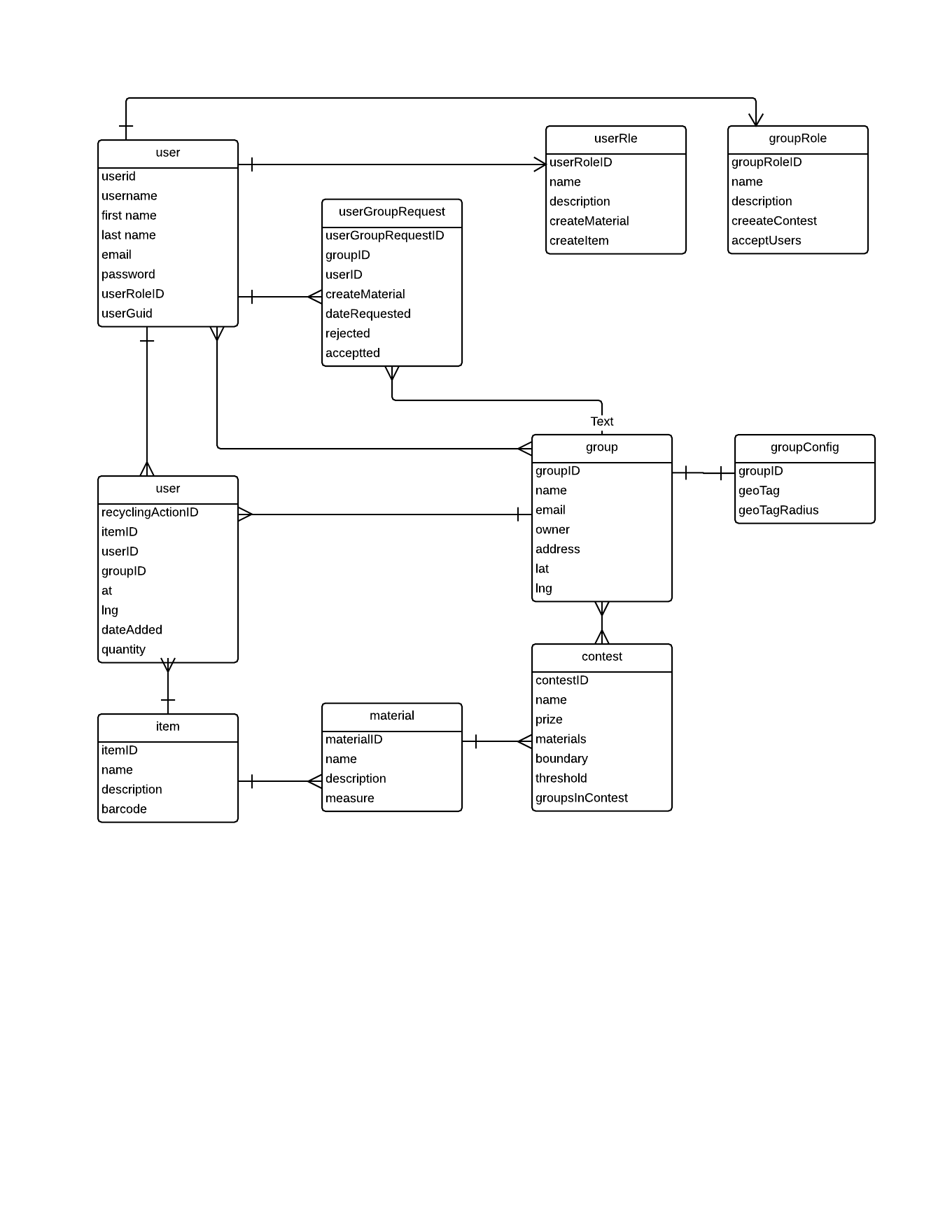


System Architecture and Implementation

## 5.4 Persistent Data Management

Data is being stored in a MySQL database. It is necessary to store all persistent data in tables to hold information such as: user profiles, groups, items, recycling actions, contests, forms static information, etc.

The design of the database can be visualized in the ER diagram below:



## 5.5 Security/Privacy

**Security Features**

* User password will be required to log in.

Upon registration, the passwords will be stored in the database. At this point the password will not be encrypted. .

* IBM Worklight Adapter access control rules

The Adapter framework provides access control with respect to controllers used. This access control will make sure that not logged users performing certain actions. For example, users that are not logged in will not have access to edit profile page.

**Privacy**

The only private information is the profile information respective to each user. To modify this information, the user needs to be authenticated in the system. However, This information is available to the users on the site taking in some pages.

# Detailed Design

This chapter will explain in depth the object design of the system. This chapter will provide class diagrams, along with a description and the purpose of each subsystem used for the subsystems described on the previous chapter. Sequence diagrams will be used to show object interactions for the corresponding use cases presented on this document.

## 6.1 Overview

The base structure of our application applies to all subsystems. On the client side application, AngularJS is used to ensure proper MVC implementation. Views are updated from the Model and controller retrieves information from the services which communicates with the IBM Worklight Adapter. The IBM Worklight Adapter ensures the required services are allowed by the system and retrieve, update or add information in the database and returns success or failure messages to the services. The IBM Worklight Adapter uses Javascript as base language.

The group subsystem allows functions such as creating groups, edit group profile, setting group permissions, material and item creation. Any user could create groups, but membership to groups must be requested and approved or rejected by group owners.

The user subsystem allows basic functions such as logging in, logging out and editing group profile.

The recycling subsystem is related to recycling actions and recyclable items. It also allows to search recycling locations which is controlled by the GeoMap IBM Worklight Adapter.

The contest subsystem requires special permission from group owners and is in charge of displaying and updating information related to contests to the users.

The dashboard subsystem displayed relevant information related to site groups, users and items. It allows users to filter information by group and user. It provides search functionalities to the users, so they can visualize information in the system.

## 6.2 Static model

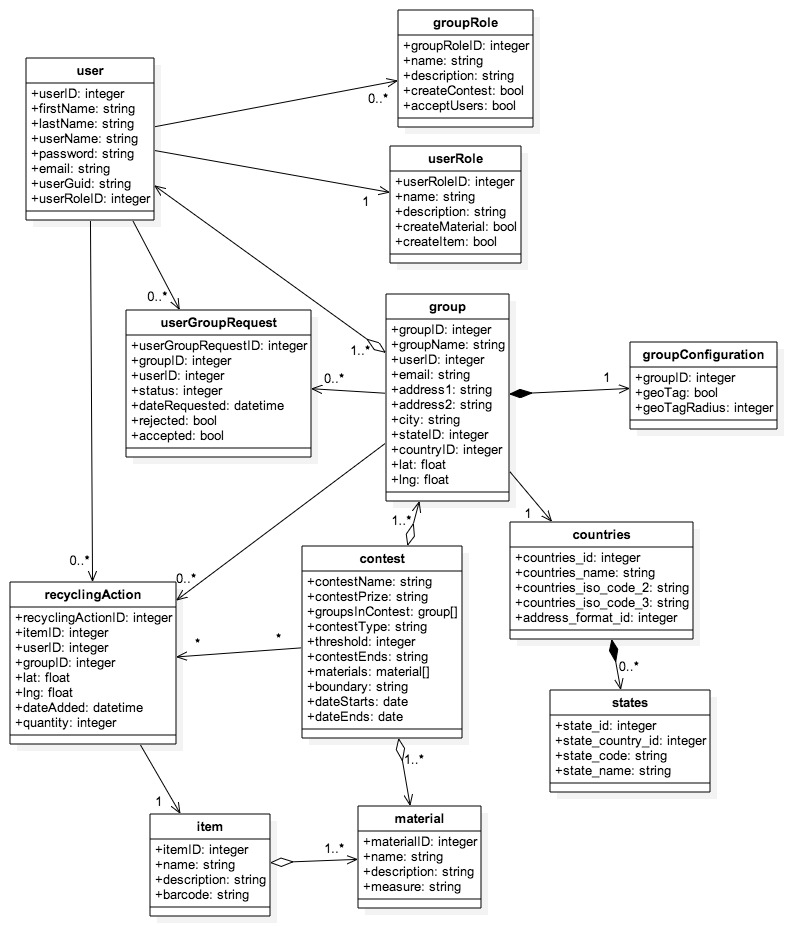


Figure 3.2.1- Class Diagram

## 6.4 Dynamic model

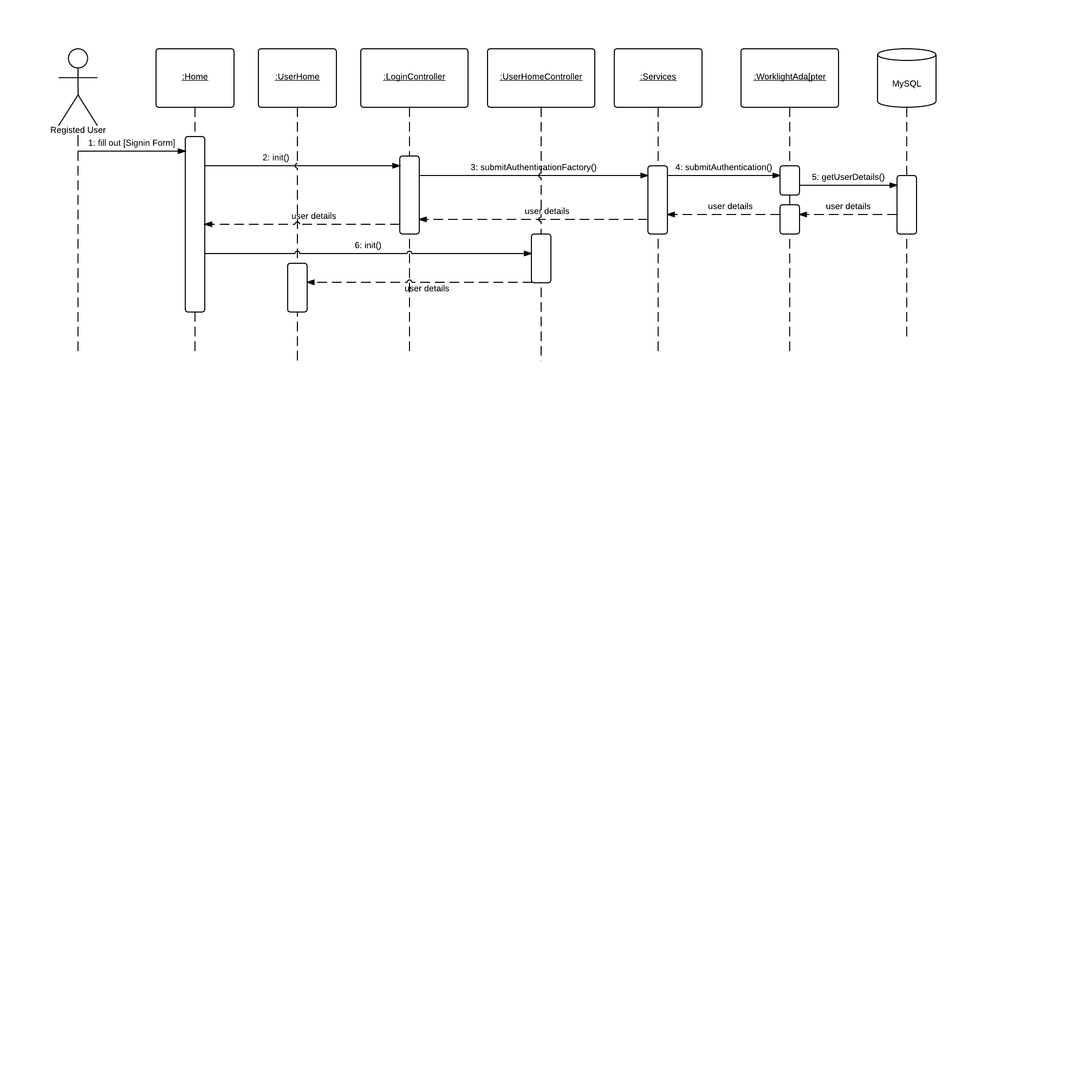


Figure 6.4.1- Log In

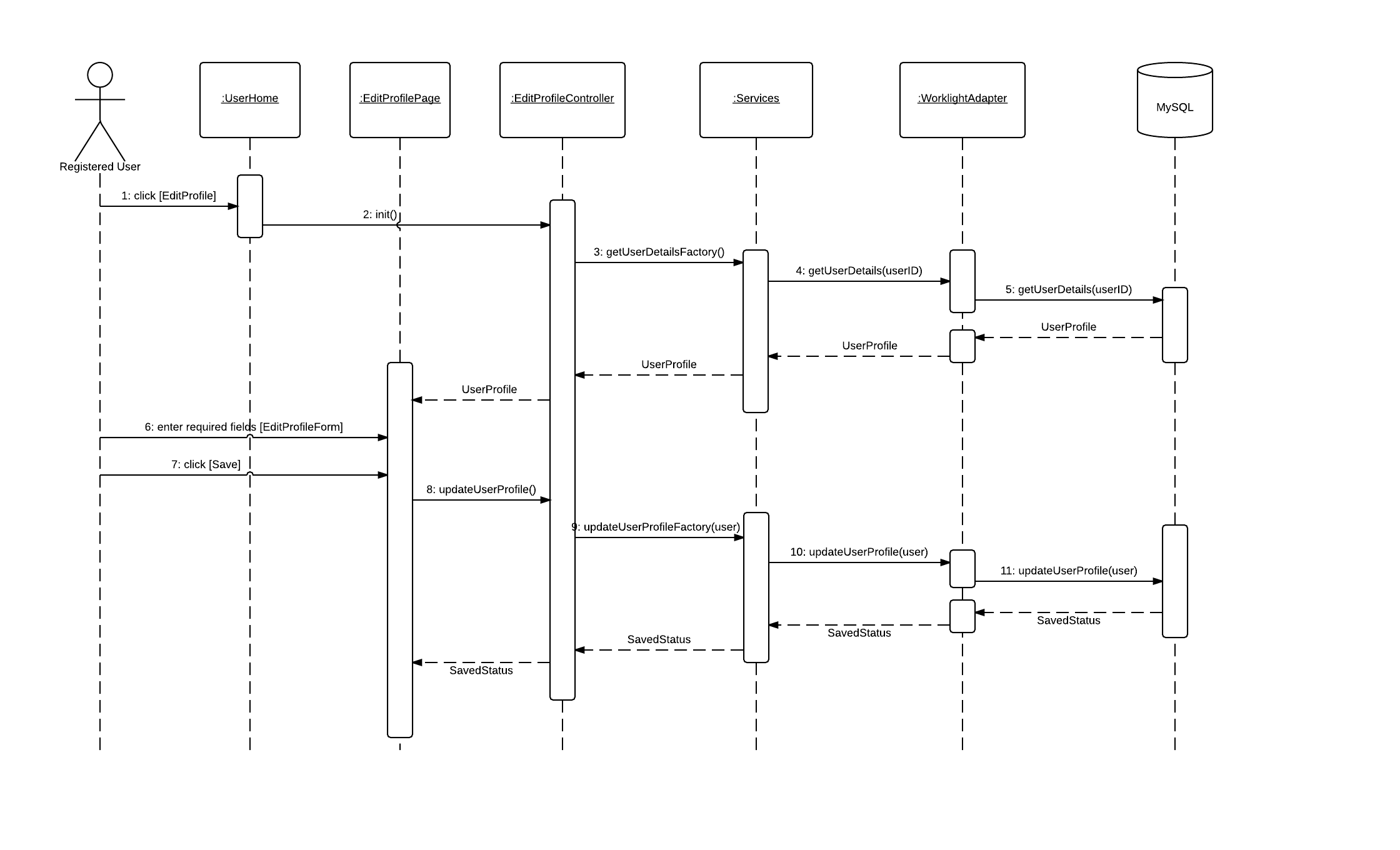


Figure 6.4.2- Edit Profile

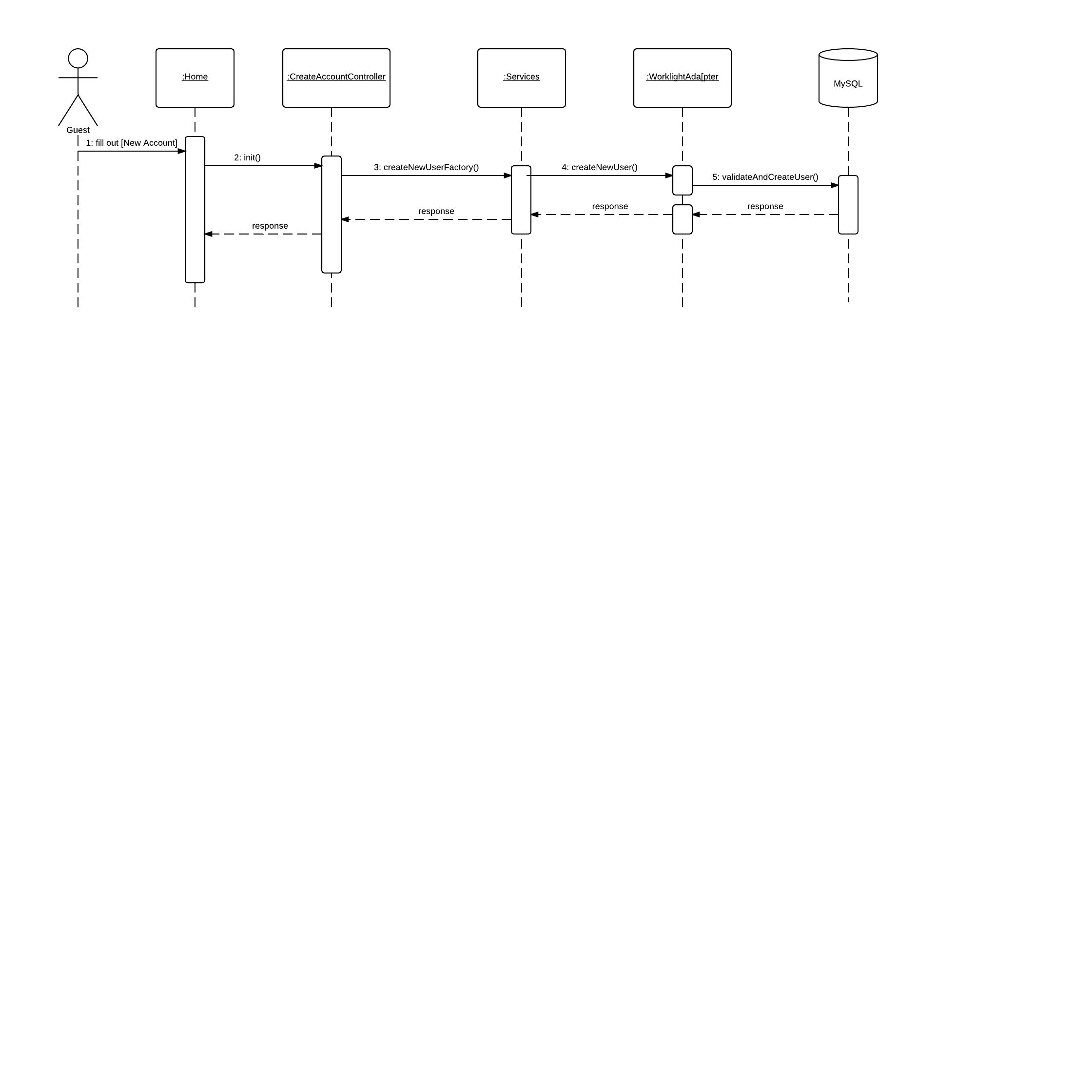


Figure 6.4.3 Create Account

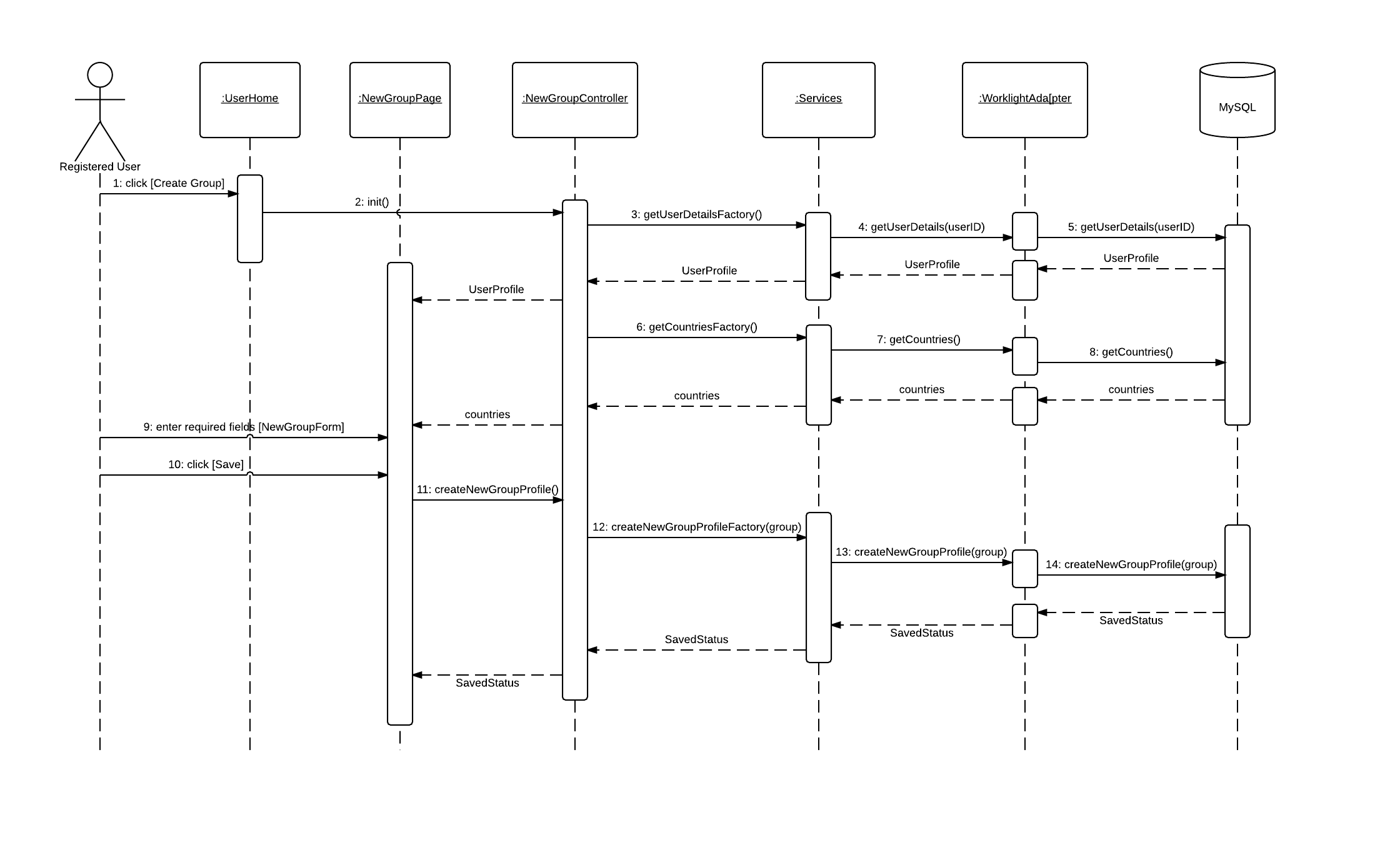


Figure 6.4.4- Create Group

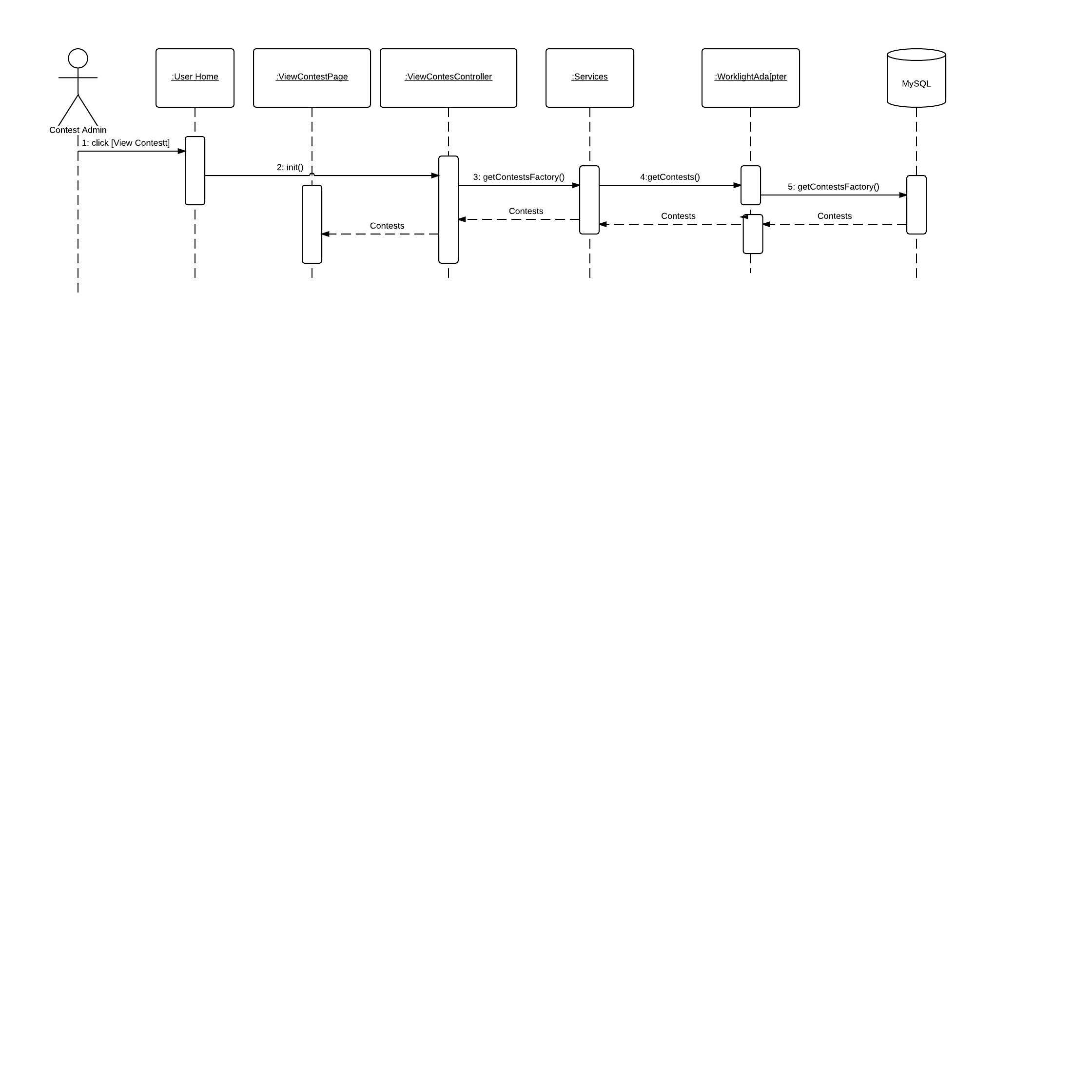


Figure 6.4.5 Create Contest

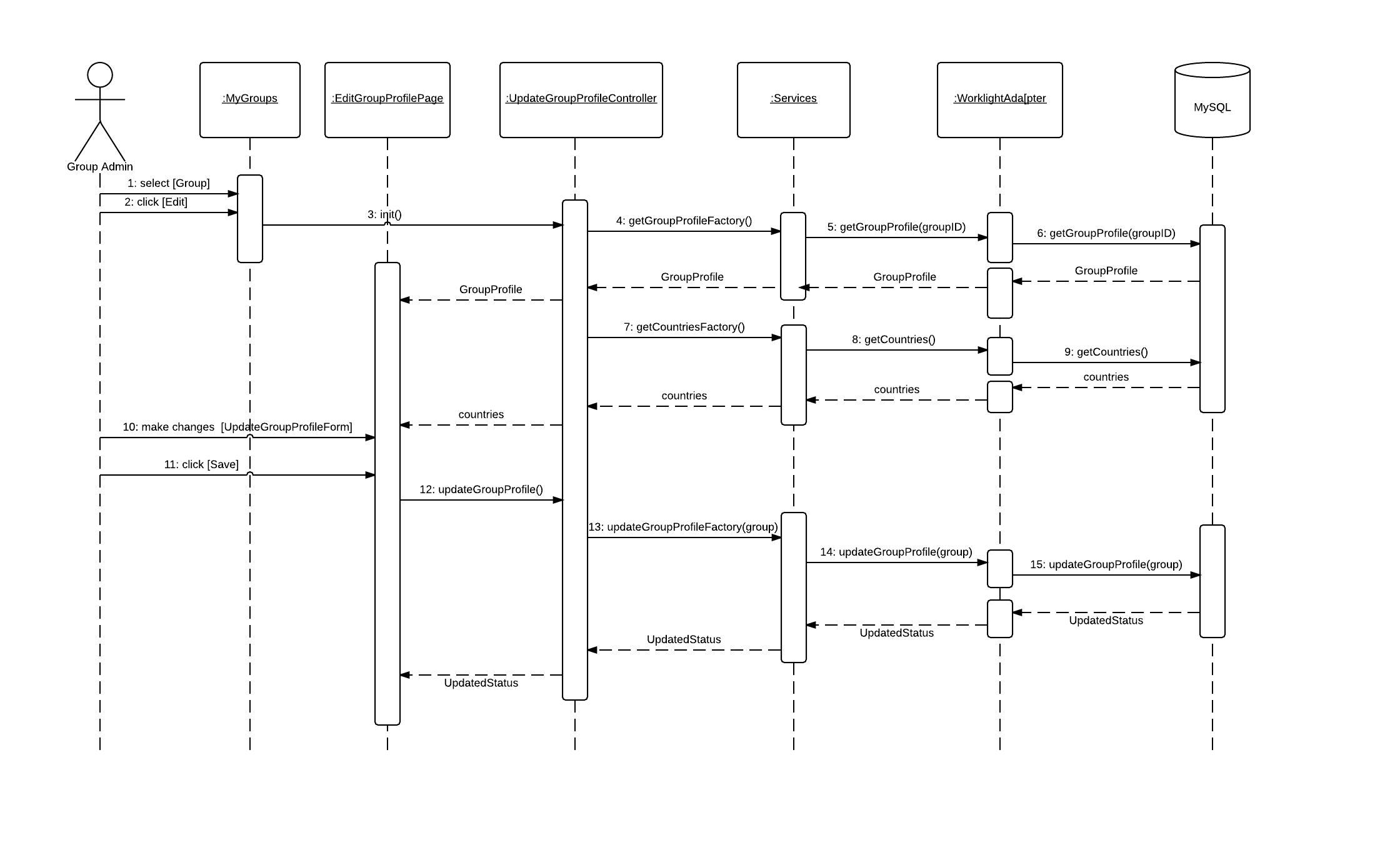


Figure 6.4.6 – Edit Group Profile

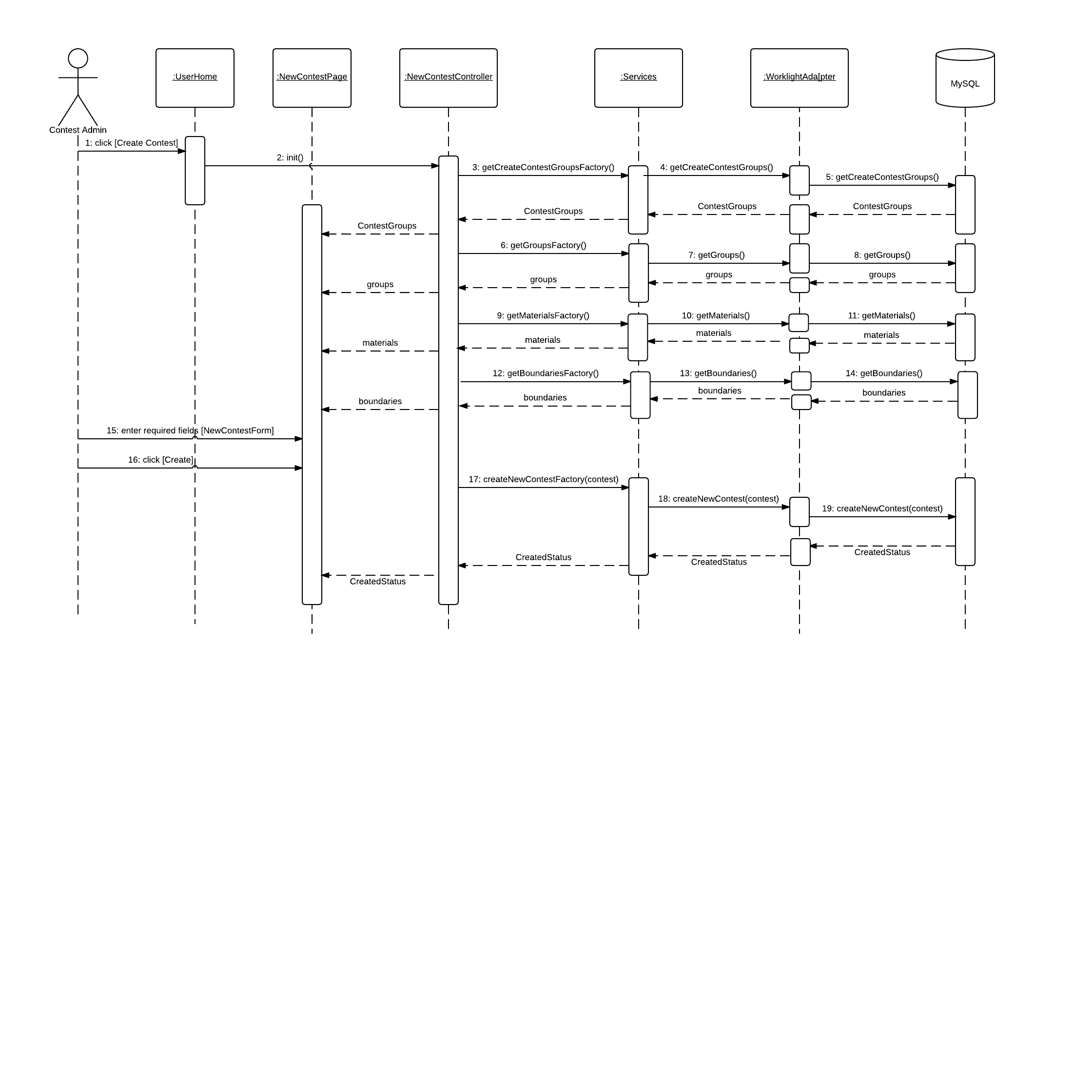


Figure 6.4.7 View Contest

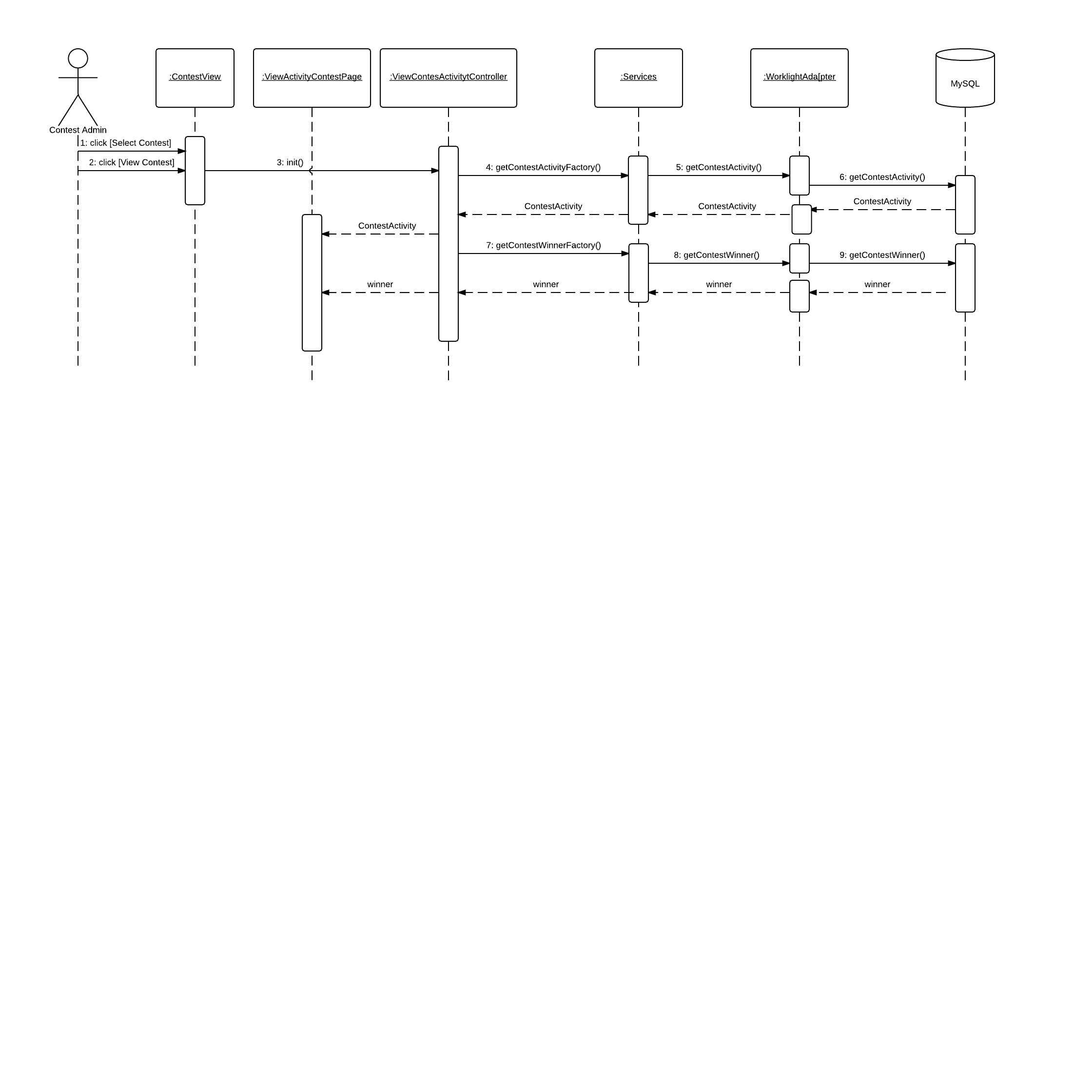


Figure 6.4.8- View Contest Activity

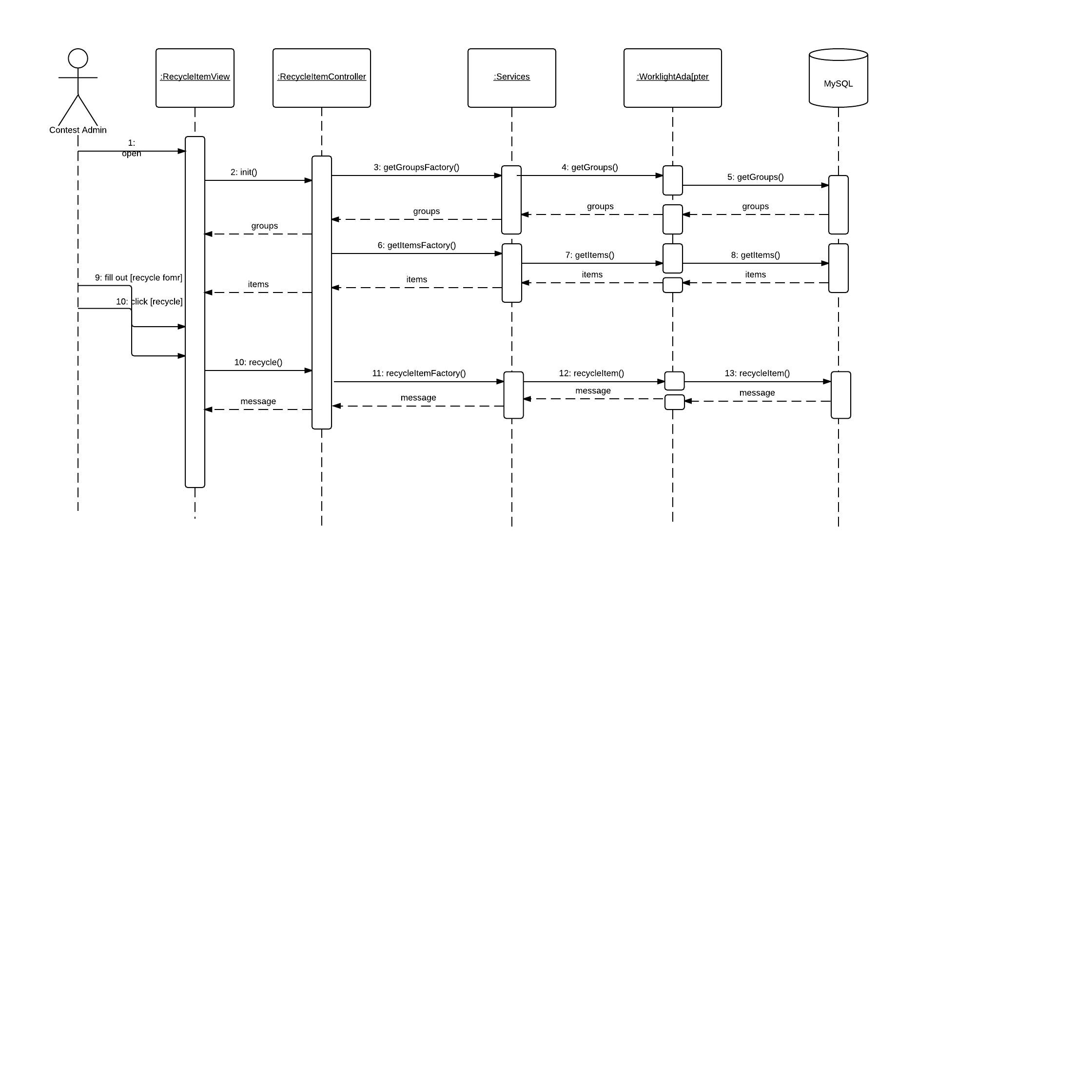


Figure 6.4.9- Recycle Item

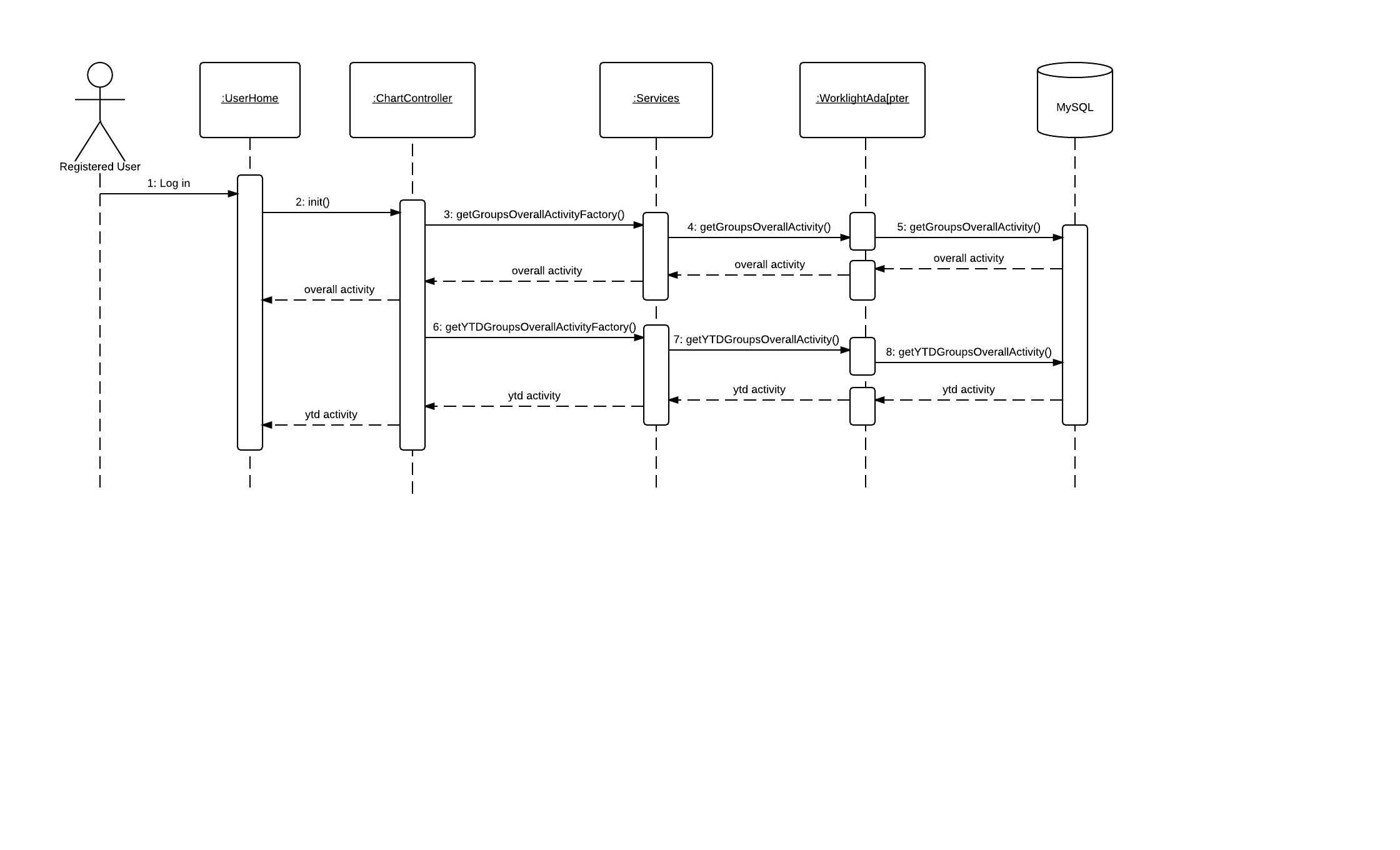


Figure 6.4.10 – Dashboard Charts

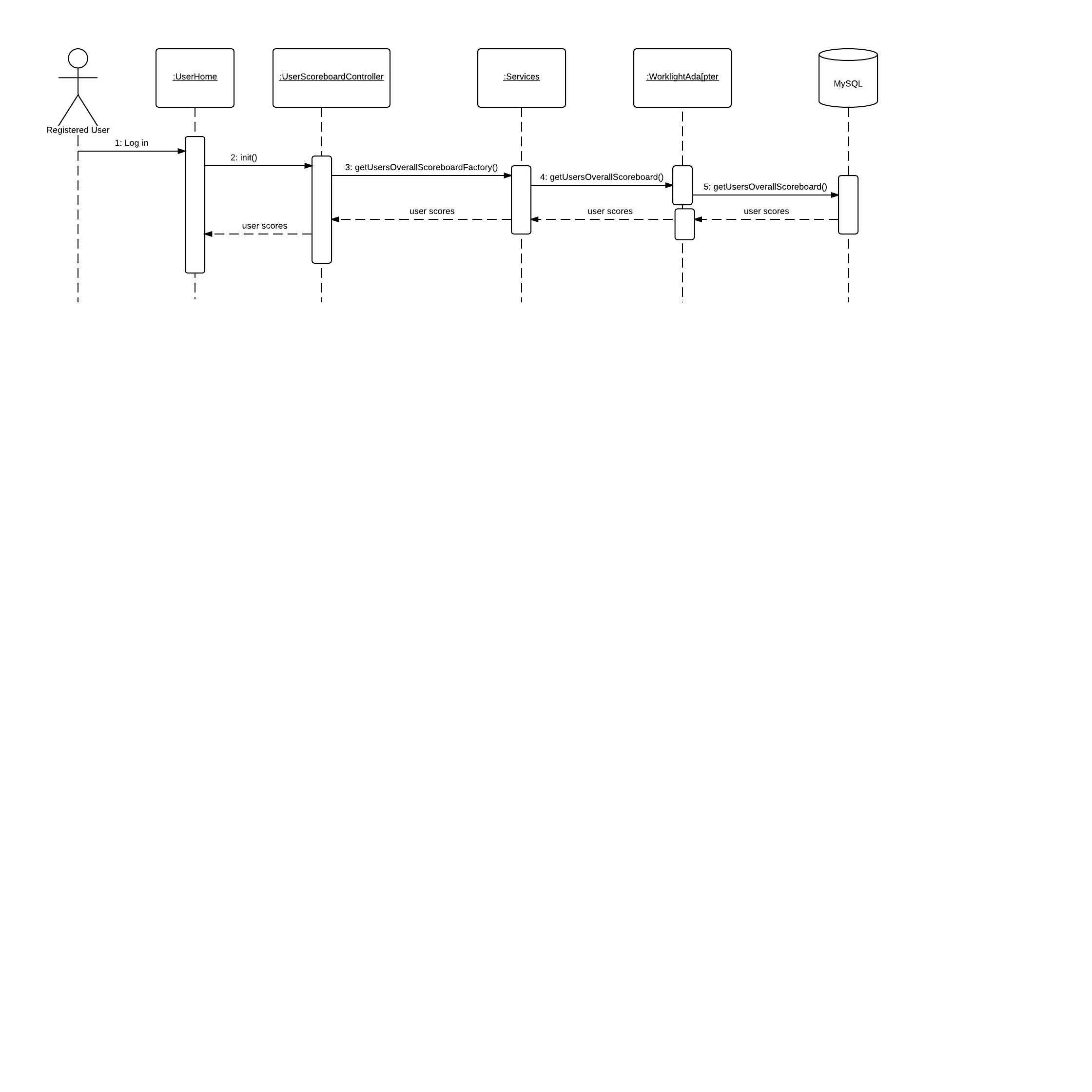


Figure 6.4.11- User Scoreboard

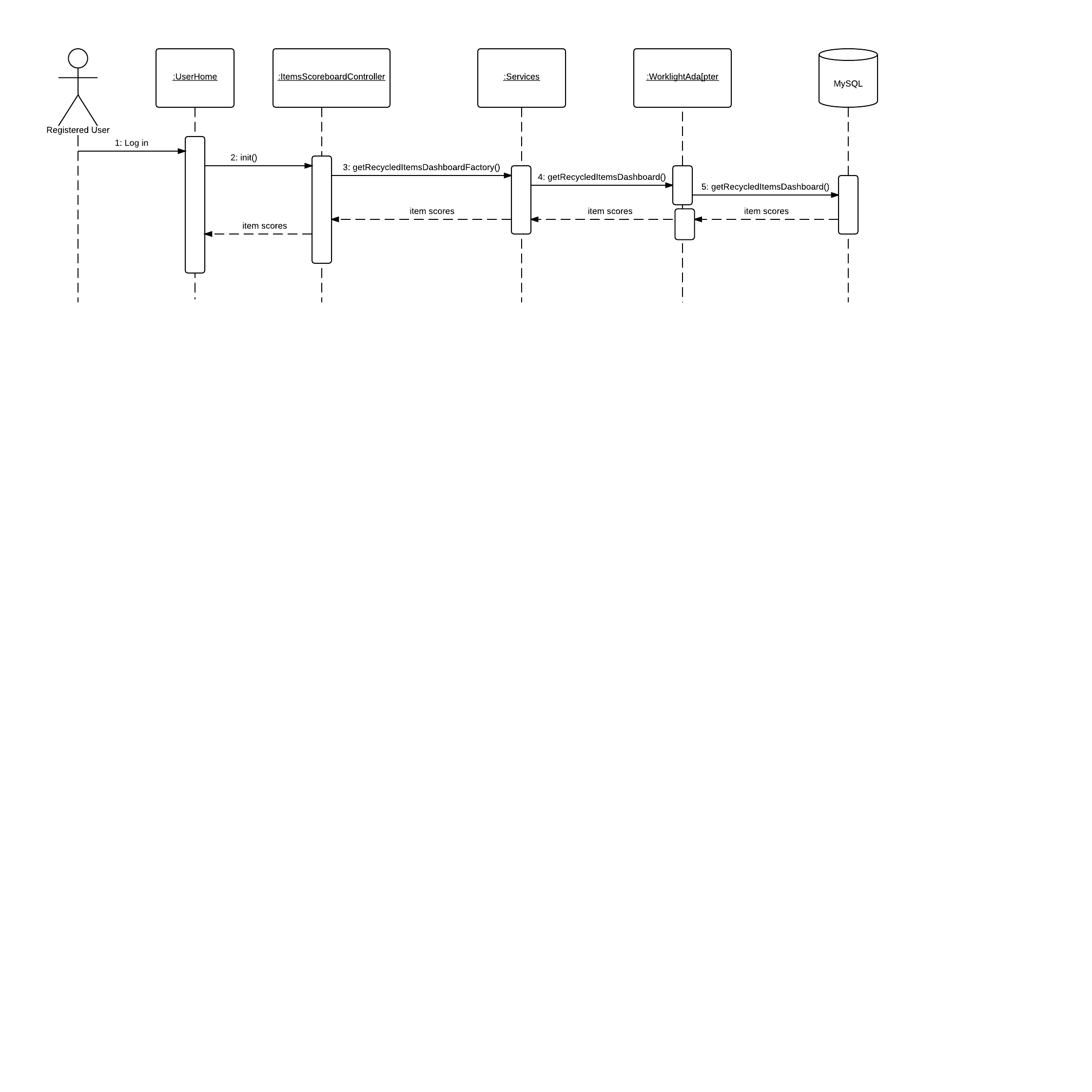


Figure 6.4.12- Item Scoreboard



Figure 6.4.13- Group Scoreboard

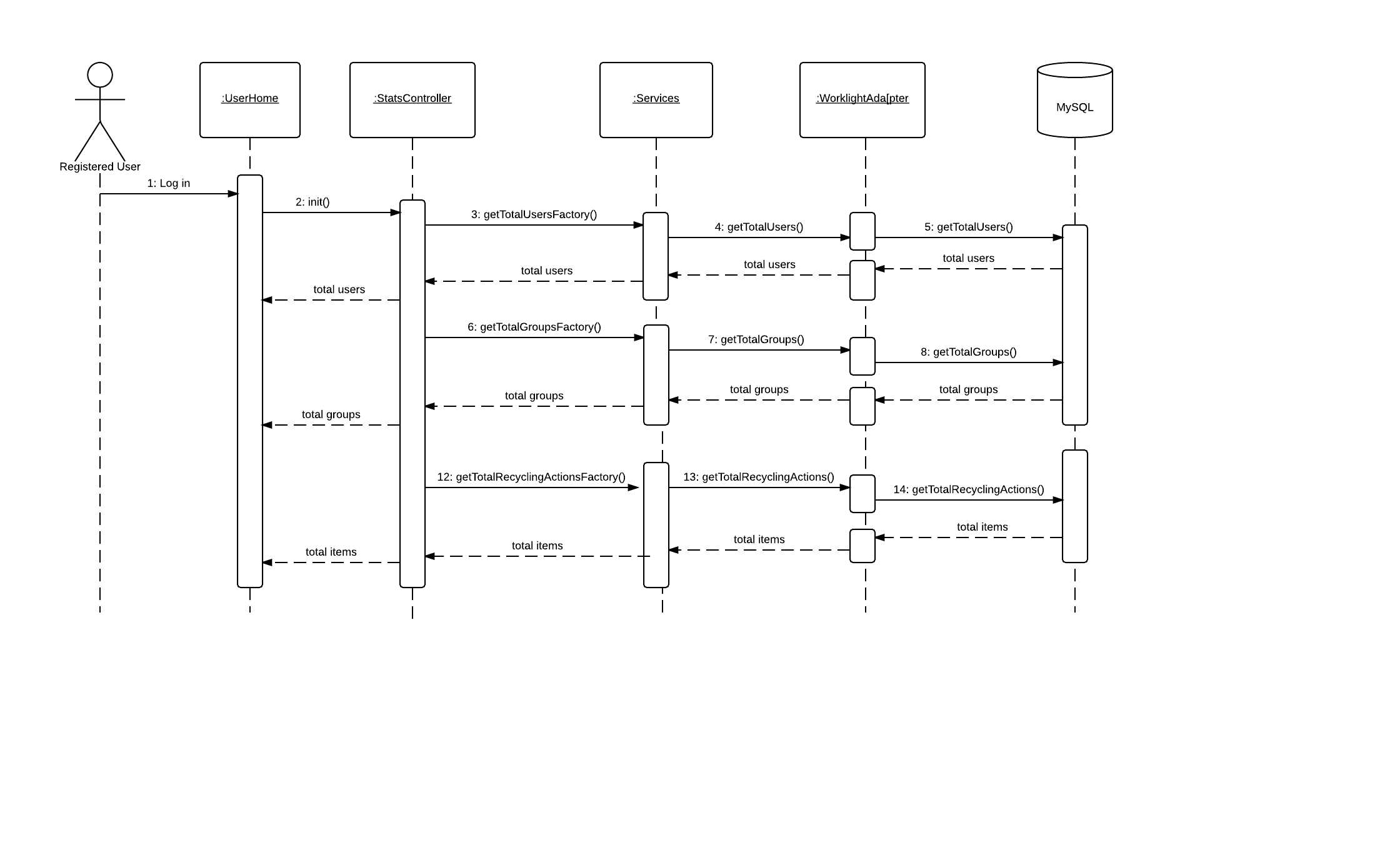


Figure 6.4.14- Site Usage widgets



Figure 6.4.15- Filter Dashboard

## 6.5 Code Specification

The application functions are implemented using AngularJS scope model and accessed by the controllers that serve specific purposes. The controllers retrieve and update information to the database using service calls that communicate with the IBM Worklight Adapter. The information handled by the application is basically group information, user information and recycling activity information.

# System Validation

Section 7 is committed to the testing process that was used in order to verify that the system is working properly. In appendix G, code for the test drivers and stubs has been included

## 7.1 Subsystem Tests

In order to test the system as a whole and to verify that the user interface is working correctly we used system testing. In addition, unit testing was used to test controllers separately by writing test cases and then executed the system against the test classes that were selected. When all test cases pass,then the results are collected. The chapter ends summarizing the results of the test cases which then completes the testing cycle.

## 7.2 System Tests

|  |  |
| --- | --- |
| Test-ID | TC-01 |
| Purpose | Create Account |
| Setup | User is in home page |
| Inputs | The user enters credentials and create the account |
| Expected Outputs | The user will be registered into the system. |

|  |  |
| --- | --- |
| Test-ID | TC-02 |
| Purpose | Create Item |
| Setup | User is in new item page |
| Inputs | The user enters required fields and create the item |
| Expected Outputs | The new item is created in the system. |

|  |  |
| --- | --- |
| Test-ID | TC-03 |
| Purpose | Create Material |
| Setup | User is in new material page |
| Inputs | The user enters required fields and create the material |
| Expected Outputs | The new material is created in the system. |

|  |  |
| --- | --- |
| Test-ID | TC-04 |
| Purpose | Create Group |
| Setup | User is in new group page |
| Inputs | The user enters required fields and create the group |
| Expected Outputs | The new group is created in the system |

|  |  |
| --- | --- |
| Test-ID | TC-05 |
| Purpose | Create Contest |
| Setup | User is in new contest page |
| Inputs | The user enters the required fields and create the contest |
| Expected Outputs | The new contest is created in the system |

|  |  |
| --- | --- |
| Test-ID | TC-06 |
| Purpose | Log In |
| Setup | User is in home page and already in the system |
| Inputs | The user enters credentials and click Log in |
| Expected Outputs | User access the user home page |

|  |  |
| --- | --- |
| Test-ID | TC-07 |
| Purpose | Edit Profile |
| Setup | User has an account in the system and accesses edit profile page |
| Inputs | The user changes the required fields and edit the profile |
| Expected Outputs | The new information is updated in the system |

|  |  |
| --- | --- |
| Test-ID | TC-08 |
| Purpose | Send group request |
| Setup | User is in search group page |
| Inputs | The user selects a group and send the request |
| Expected Outputs | The request is stored in the system |

|  |  |
| --- | --- |
| Test-ID | TC-09 |
| Purpose | Accept group request |
| Setup | User is in requests page |
| Inputs | The user select a request and clicks accept |
| Expected Outputs | The new user is part of the group |

|  |  |
| --- | --- |
| Test-ID | TC-10 |
| Purpose | Decline group request |
| Setup | User is in requests page |
| Inputs | The user select a request and clicks decline |
| Expected Outputs | The user is not part of the group |

|  |  |
| --- | --- |
| Test-ID | TC-11 |
| Purpose | Filter Dashboard |
| Setup | User is in home page |
| Inputs | The user clicks filter and select a group or user to filter the dashboard |
| Expected Outputs | The dashboard is updated with the selected information |

## 7.3 Evaluation of Tests

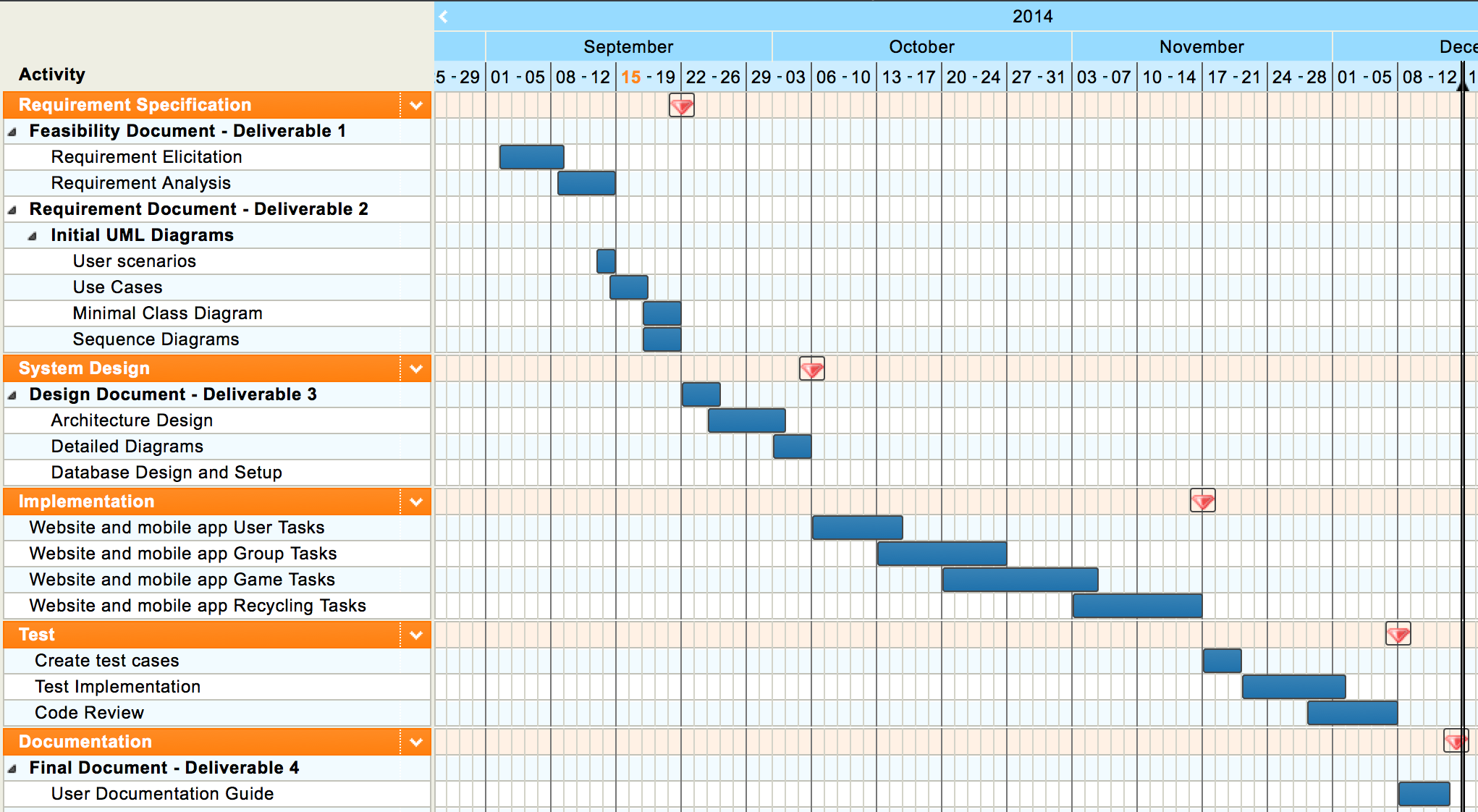
|  |  |
| --- | --- |
| TC-01 | PASS |
| TC-02 | PASS |
| TC-03 | PASS |
| TC-04 | PASS |
| TC-05 | PASS |
| TC-06 | PASS |
| TC-07 | PASS |
| TC-08 | PASS |
| TC-09 | PASS |
| TC-10 |  |
| TC-11 | PASS |

# Glossary

|  |  |
| --- | --- |
| **Term** | **Definition** |
| Guest User | A user that has not created an account in the system |
| Registered User | A user that has created an account in the system |
| Group Owner | A user that created a group is automatically that group’s owner |
| Group Member | A registered user that joined a group. |
| UML | Unified Modeling Language |
| Class diagram | Describes the system in terms of classes, attributes, and their relationships. |
| Server-side architecture | An architectural pattern that consist of a server handling requests from clients . |

# Appendix

## 9.1 Appendix A - Project schedule



## 9.2 Appendix B – Use Cases

|  |  |
| --- | --- |
| Use Case ID | US1001 |
| Description | Allows a Guest to create an account and become a registered user. |
| Actor | Guest |
| Pre-conditions | 1. The Guest accesses the home page of the application. |
| Steps | 1. The Guest fills out the required user account information. 2. The Guest clicks on “Create Account” button. 3. The user account information is validated by the System. 4. The user account information is created by the System. 5. The Guest is notified that the account has been created. |
| Post-Conditions | 1. The Guest is now a registered user in the system. |
| Exceptions | Exception: User account information is invalid  Raised when any of the user information fields contains invalid information. |

|  |  |
| --- | --- |
| Use Case ID | US1002 |
| Description | A Registered User should be able to login into the system. |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User access the home page of the application. |
| Steps | 1. The Registered User fills out his credentials. 2. The Register User clicks on “Sign in”. 3. The system validates the credentials. 4. The system logs in the Registered User. 5. The system redirects the user to the User Home page. |
| Post-Conditions | 1. The user has access to site data |
| Exceptions | Exception: User/Password incorrect  Raised when the entered credentials do not match anything in the system. |

|  |  |
| --- | --- |
| Use Case ID | US1003 |
| Description | A Registered User should be able to edit their user profile in the system to keep it up to date. |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. 3. The Registered User clicks the “User” menu item. |
| Steps | 1. The Registered User clicks the “Edit Profile” menu item. 2. The Registered User changes the user account information as need be. 3. The Registered User clicks on “Save” button. 4. The System validates the user account information. 5. The System updates the user account information. 6. The Registered User receives notification the user account information was updated. |
| Post-Conditions | 1. The new information is stored in the database |
| Exceptions | Exception: User account information is invalid  Raised when any of the user account information fields contains invalid information. |

|  |  |
| --- | --- |
| Use Case ID | US1004 |
| Description | A Registered User should be able to view the overall group’s activity of the site. |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User logs into the application. 2. The Registered User accesses the user home page of the application. |
| Steps | 1. The System shall retrieve groups, recycled and users activity 2. The system shall display the information on the dashboard |
| Post-Conditions | 1. The user can see the group, users and items activity |
| Exceptions | Exception: System is not able to retrieve data |

|  |  |
| --- | --- |
| Use Case ID | US1005 |
| Description | A Registered User should be able to send a request to a group they are not a member of, to join the group. |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. 3. The Registered User clicks the “Groups” menu item. 4. The Registered User clicks the “Search Group” menu item. |
| Steps | 1. The Registered User searches and clicks on the Group they wish to join. 2. The Registered User clicks on “Join” button. 3. The Registered User receives notification the request was sent. |
| Post-Conditions | 8. The notification is added to the queue of that group’s administrator |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1008 |
| Description | A Registered User should be able to create a new group. |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. 3. The Registered User clicks the “Groups” menu item. |
| Steps | 1. The Registered User clicks the “Create Group” menu item. 2. The Registered User fills out the group account information. 3. The Registered User clicks on the “Save” button. 4. The System validates the group account information. 5. The System group account information is created by the system. 6. The Registered User receives notification the group has been created. |
| Post-Conditions | The Registered User is assigned as the Group Owner of the created group. |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1009 |
| Description | A Registered User can log recycling actions. |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. |
| Steps | 1. The Registered User clicks the “Recycle” menu item. 2. The Registered User selects the appropriate information for the recycling action. 3. The Registered User clicks on the “Create Recycling Action” button. 4. The system will add the new recycling action. 5. The system shall verify if the recycled item is part of a contest. 6. The Registered User receives notification that the recycling action has been created. |
| Post-Conditions | The system displays success/failure message |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1011 |
| Description | A Registered User can see recycling locations near by. |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. |
| Steps | 1. The Registered User clicks the “Recycle Locations” menu item. |
| Post-Conditions |  |
| Exceptions | The system is not able to retrieve location. |

|  |  |
| --- | --- |
| Use Case ID | US1012 |
| Description | A Registered User can review all materials created in the system. |
| Actor | Registered Users |
| Pre-conditions | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. |
| Steps | 1. The Registered User clicks the “Materials” menu item. 2. The system shall retrieve the materials information from the database. |
| Post-Conditions | The system displays the list of materials |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1013 |
| Description | A Material Administrator wants to create a new Material. |
| Actor | Material Administrator |
| Pre-conditions | 1. The Material Administrator accesses the home page of the application. 2. The Material Administrator logs into the application. |
| Steps | 1. The Material Administrator clicks the “Item” menu item. 2. The Material Administrator clicks the “New Material” menu item. 3. The Material Administrator fills out the new material information. 4. The Material Administrator clicks on “Create Material” button. 5. The System validates the new material information. 6. The System creates the new material. 7. The Material Administrator is notified that the material was created. |
| Post-Conditions |  |
| Exceptions | Exception: New material information is invalid  Raised when any of the new material information fields contains invalid information. |

|  |  |
| --- | --- |
| Use Case ID | US1014 |
| Description | A Registered User should be able to review Items in the System. |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. |
| Steps | 1. The Registered User clicks the “Item” menu item. 2. The Registered User clicks the “Items” menu item. |
| Post-Conditions |  |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1015 |
| Description | An Item Administrator wants to create a new Item. |
| Actor | Item Administrator |
| Pre-conditions | 1. The Item Administrator accesses the home page of the application.   2. The Item Administrator logs into the application. |
| Steps | 1. The Item Administrator clicks the “Item” menu item. 2. The Item Administrator clicks the “New Item” menu item. 3. The Item Administrator types in some information on the new item and clicks on “Next” button. 4. The Item Administrator confirms this item is not already in the System. 5. The Item Administrator enters the new item information. 6. The Item Administrator clicks on “Create Item” button. 7. The System validates the new item information. 8. The System creates the new item. |
| Post-Conditions | The Item Administrator is notified that the material was created. |
| Exceptions | Exception: New item information is invalid  Raised when any of the new item information fields contains invalid information. |

|  |  |
| --- | --- |
| Use Case ID | US1016 |
| Description | A Registered User can view groups they are members of. |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. |
| Steps | 1. The Registered User clicks the “Groups” menu item. 2. The Registered User clicks the “Joined Groups” menu item. 3. The system shall list the groups the user has joined |
| Post-Conditions |  |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1018 |
| Description | A Registered User can view the members of a group they have joined. |
| Actor | Registered User |
| Pre-conditions |  |
| Steps | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. 3. The Registered User clicks the “Groups” menu item. 4. The Registered User clicks the “Joined Groups” menu item. 5. The Registered User clicks on a Group. 6. The Registered User clicks the “members” button. 7. The Registered User is shown all members of that Group. |
| Post-Conditions |  |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1020 |
| Description | A registered user can log out |
| Actor | Registered User |
| Pre-conditions | 1. The Registered User accesses the home page of the application. 2. The Registered User logs into the application. |
| Steps | 1. The Registered User accesses User drop down menu 2. The registered user click log out 3. The system shall log out the user from the application |
| Post-Conditions | 6. The user cannot access any functionality of the application without logging in again |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1021 |
| Description | A Group Administrator can edit the information of the group they administrate. |
| Actor | Group Administrator |
| Pre-conditions | 1. The Group Administrator accesses the home page of the application. 2. The Group Administrator logs into the application. 3. The Group Administrator clicks the “Groups” menu item. 4. The Group Administrator clicks the “My Groups” menu item. |
| Steps | 1. The Group Administrator clicks on a Group. 2. The Group Administrator clicks the “edit” button. 3. The Group Administrator updates the Group information that needs update. 4. The Group Administrator clicks on the “save” button. 5. The System validates the information. 6. The System updates the new information for the group. |
| Post-Conditions |  |
| Exceptions | Exception: Group information is invalid  Raised when any of the group information fields contains invalid information. |

|  |  |
| --- | --- |
| Use Case ID | US1022 |
| Description | A Group Administrator can accept or decline users from joining the groups they own. |
| Actor | Group Administrator |
| Pre-conditions | 1. The Group Administrator accesses the home page of the application. 2. The Group Administrator logs into the application. |
| Steps | 1. The Group Administrator clicks the “Groups” menu item. 2. The Group Administrator clicks the “Accept Users” menu item. 3. The Group Administrator clicks on a Group. 4. The Group Administrator clicks the “Requests” button. 5. The Group Administrator clicks on a user and clicks “Accept” or “Reject”. |
| Post-Conditions | The user is deleted from the group owner’s request queue |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1023 |
| Description | A Group Administrator or authorized user can create a contest |
| Actor | Group Administrator/Authorized user |
| Pre-conditions | 1. The actor accesses the home page of the application. 2. The actor logs into the application. |
| Steps | 1. The Actor clicks on Contest menu item. 2. The Actor clicks the “New Contest” menu item. 3. The Actor fills out the required information 4. The actor clicks create contest |
| Post-Conditions | The contest is stored in the database and available |
| Exceptions | The actor entered incorrect contest information and the system cannot create the contest |

|  |  |
| --- | --- |
| Use Case ID | US1024 |
| Description | A User should be able to view contests |
| Actor | Registered User |
| Pre-conditions | 1. The Actor accesses the home page of the application. 2. The Actor logs into the application. |
| Steps | 1. The Actor clicks the “Contest” menu item. 2. The Actor clicks the “View Contest” menu item. 3. The System shall get contest information from the database. 4. The system displays the list of groups |
| Post-Conditions | The user is able to select a contest and perform actions on them |
| Exceptions |  |

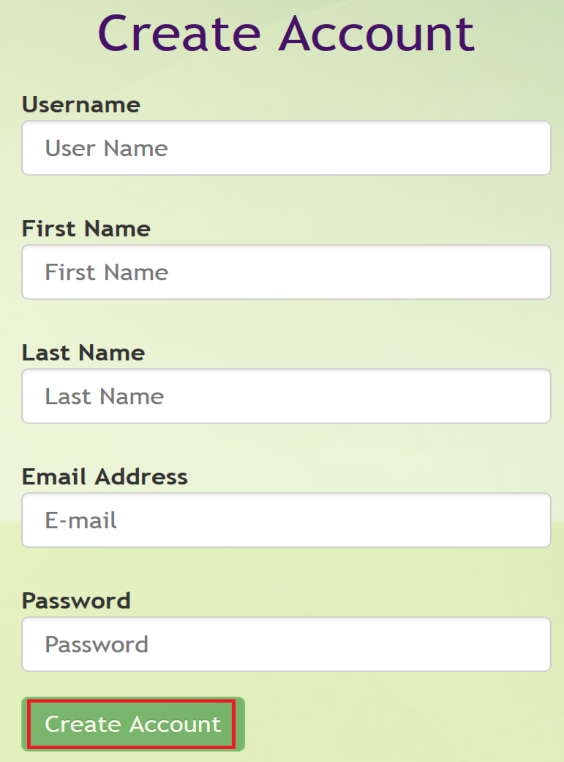
|  |  |
| --- | --- |
| Use Case ID | US1025 |
| Description | A user should be able to view contest activity |
| Actor | Registered User |
| Pre-conditions | 1. The Actor accesses the home page of the application.   2. The Actor logs into the application. |
| Steps | 1. The Actor clicks the “Contest” menu item. 2. The Actor clicks the “View Contest” menu item. 3. The actor selects a contest 4. The actor clicks view activity 5. The System shall get contest activity from the database. |
| Post-Conditions | The shall display the status of the contest |
| Exceptions |  |

|  |  |
| --- | --- |
| Use Case ID | US1026 |
| Description | A user should be able to see to search overall activity of a group. |
| Actor | Registered User |
| Pre-conditions | 1. The Actor accesses the home page of the application.   2. The Actor logs into the application. |
| Steps | 1. The system shall display groups activity in the dashboard 2. The user types a name on the group search box 3. The system shall retrieve the data associated with that group |
| Post-Conditions | The user gets information filtered by group name |
| Exceptions | The group name that the user entered does not exist |

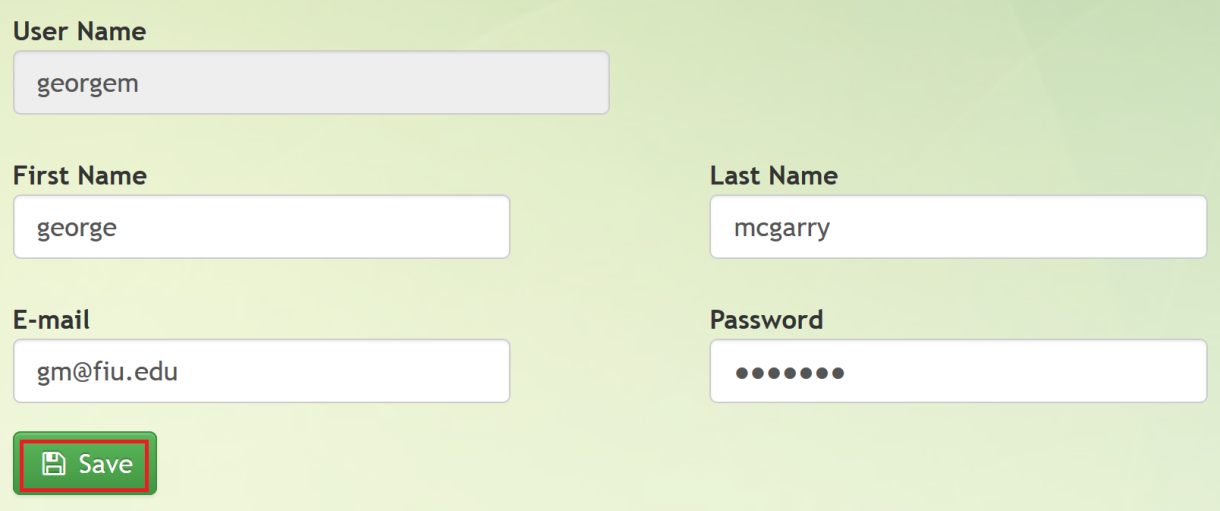
|  |  |
| --- | --- |
| Use Case ID | US1027 |
| Description | A user should be able to see to search overall activity of a user. |
| Actor | Registered User |
| Pre-conditions | 1. The Actor accesses the home page of the application.   2. The Actor logs into the application. |
| Steps | 1. The system shall display users activity in the dashboard 2. The user types a name on the user search box 3. The system shall retrieve the data associated with that user |
| Post-Conditions | The user gets information filtered by user name |
| Exceptions | The user name that the user entered does not exist |

|  |  |
| --- | --- |
| Use Case ID | US1028 |
| Description | A user should be able to search recycled items |
| Actor | Registered User |
| Pre-conditions | 1. The Actor accesses the home page of the application.   2. The Actor logs into the application. |
| Steps | 1. The system shall display items activity in the dashboard 2. The user types a name on the items search box 3. The system shall retrieve the data associated with that item |
| Post-Conditions | The user gets information filtered by item name |
| Exceptions | The items name that the user entered does not exist |

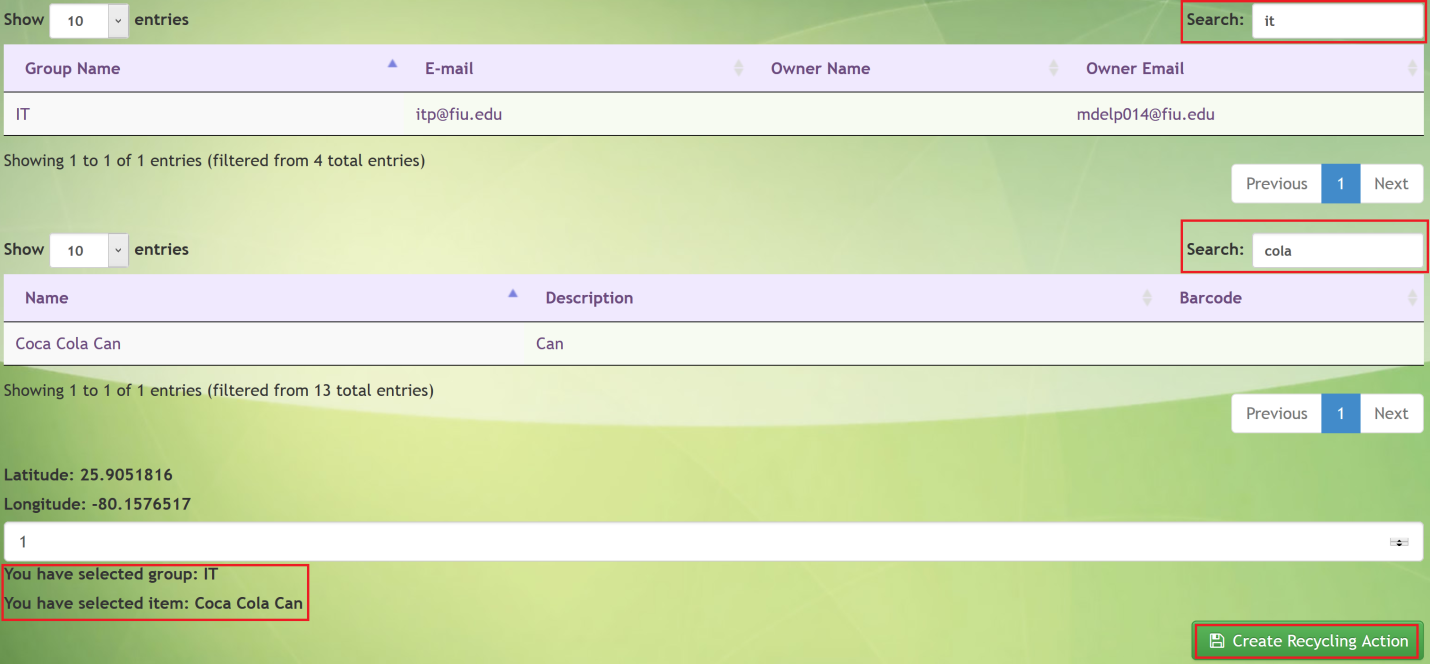
* 1. Appendix C – User Interface designs.



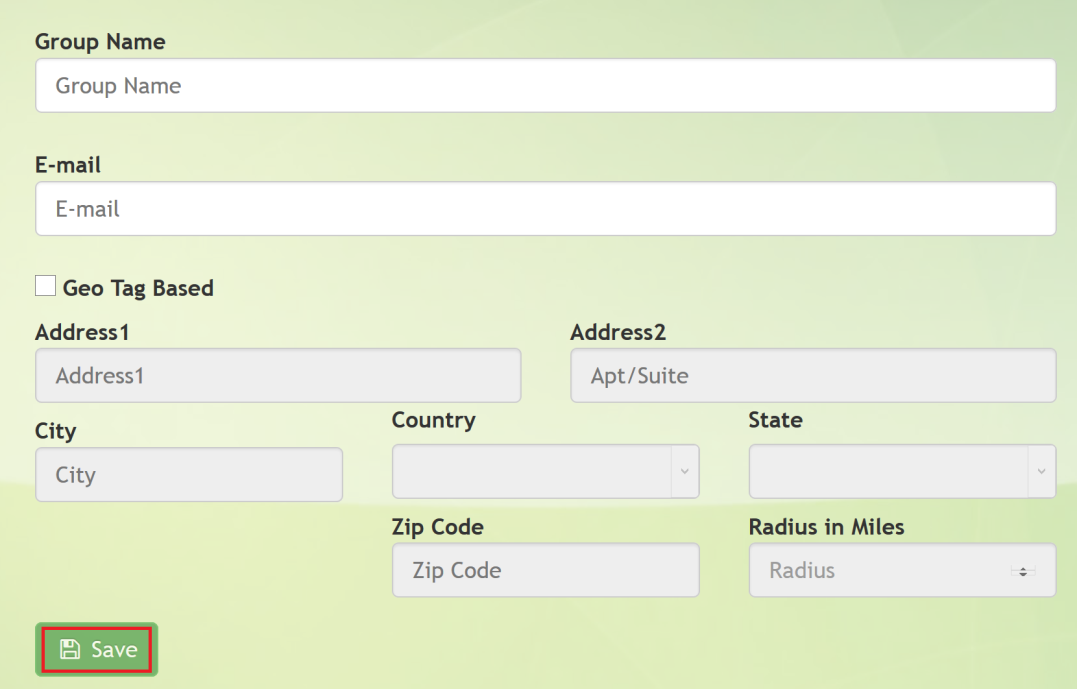
Create Account



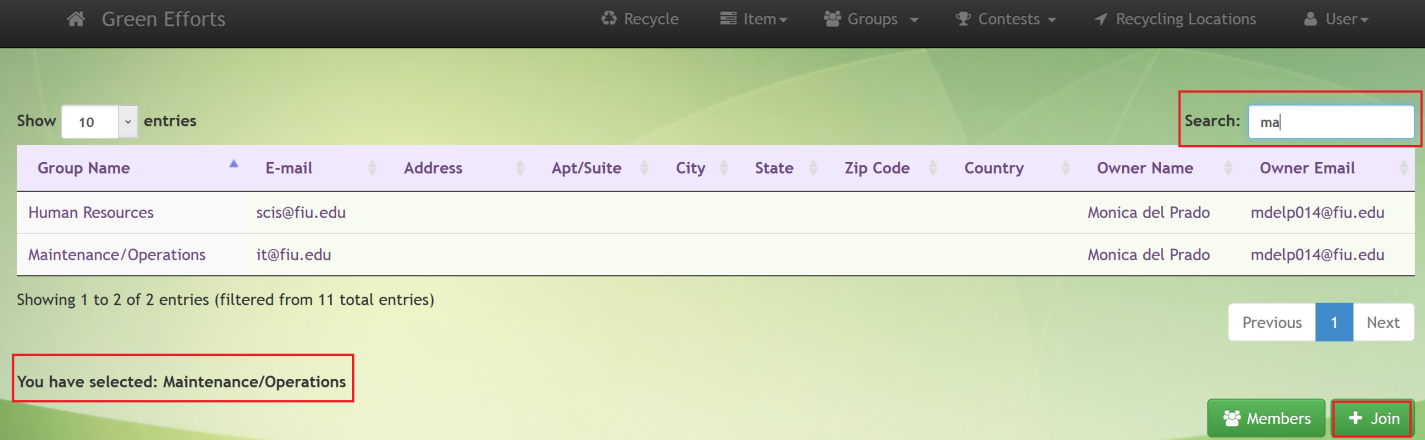
Edit Profile



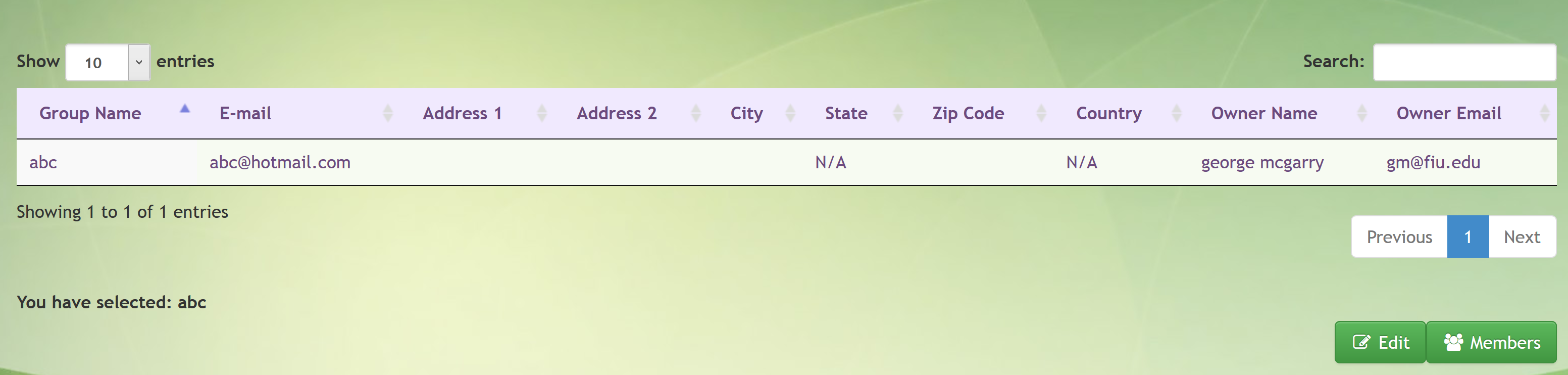
Recycle



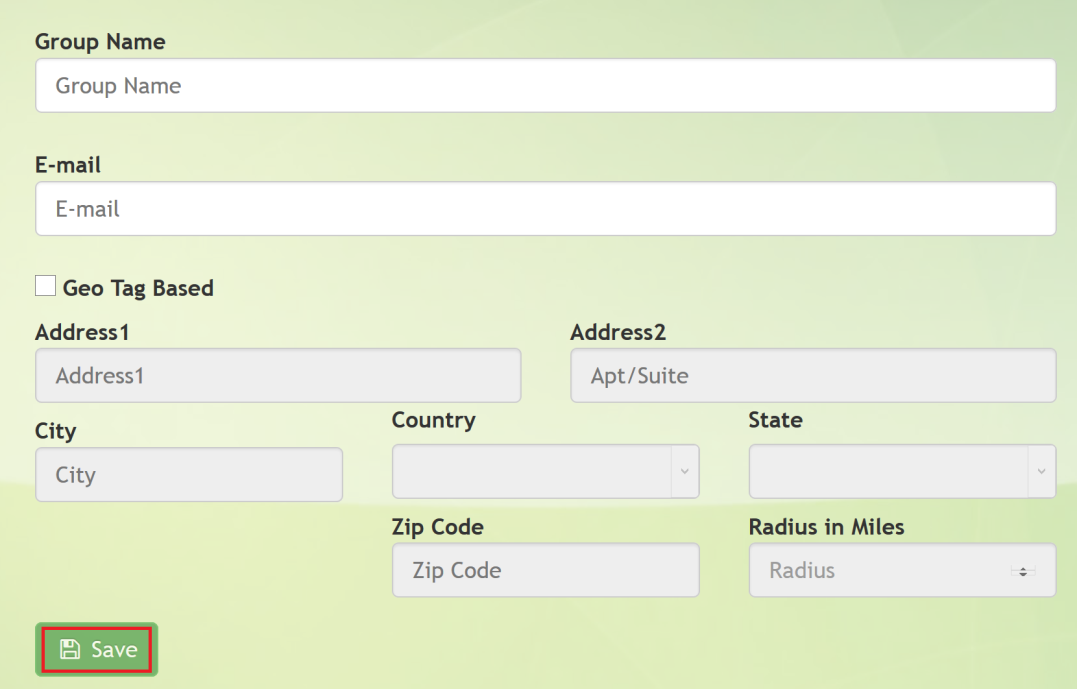
Create Group



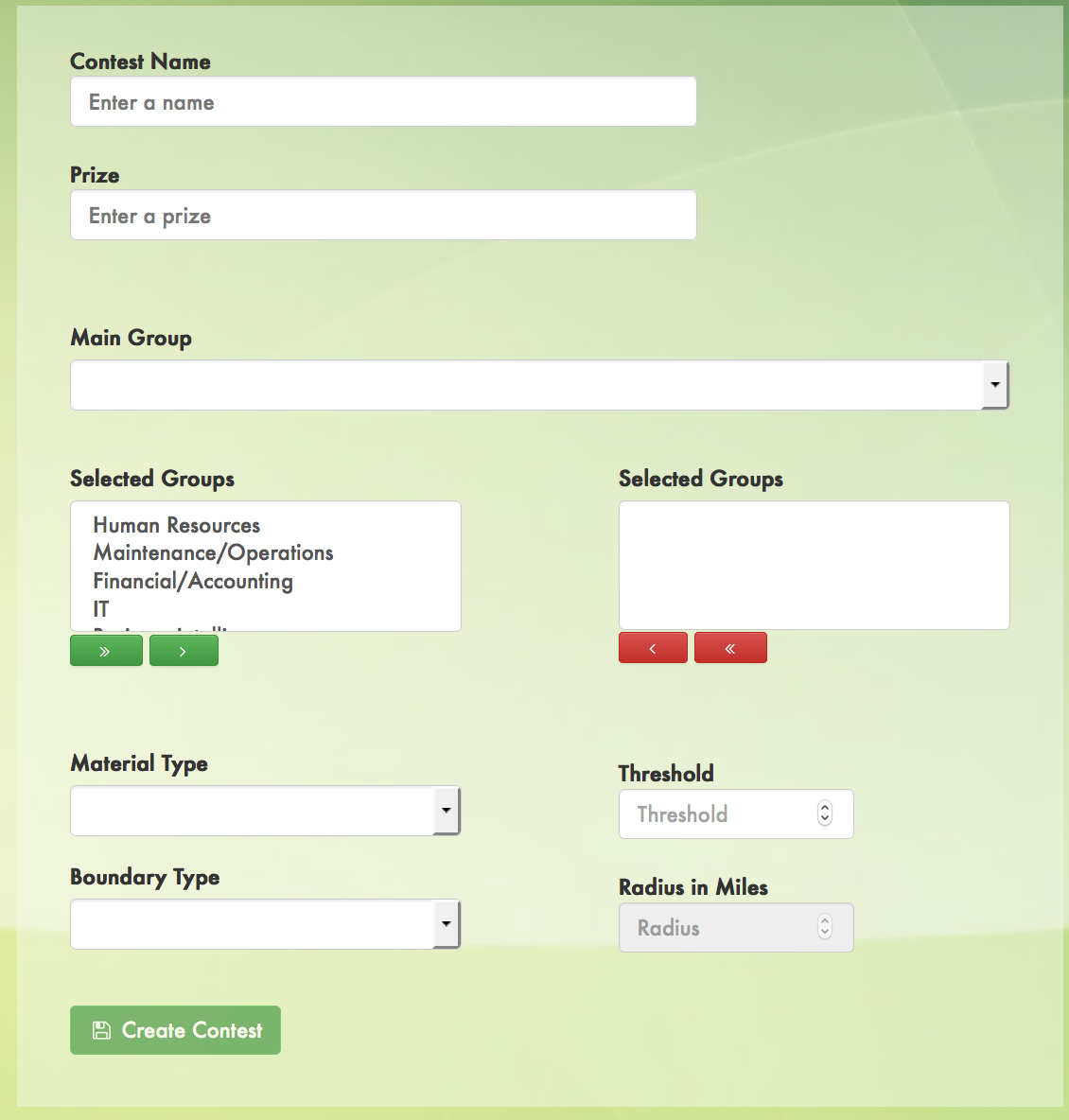
Join Group



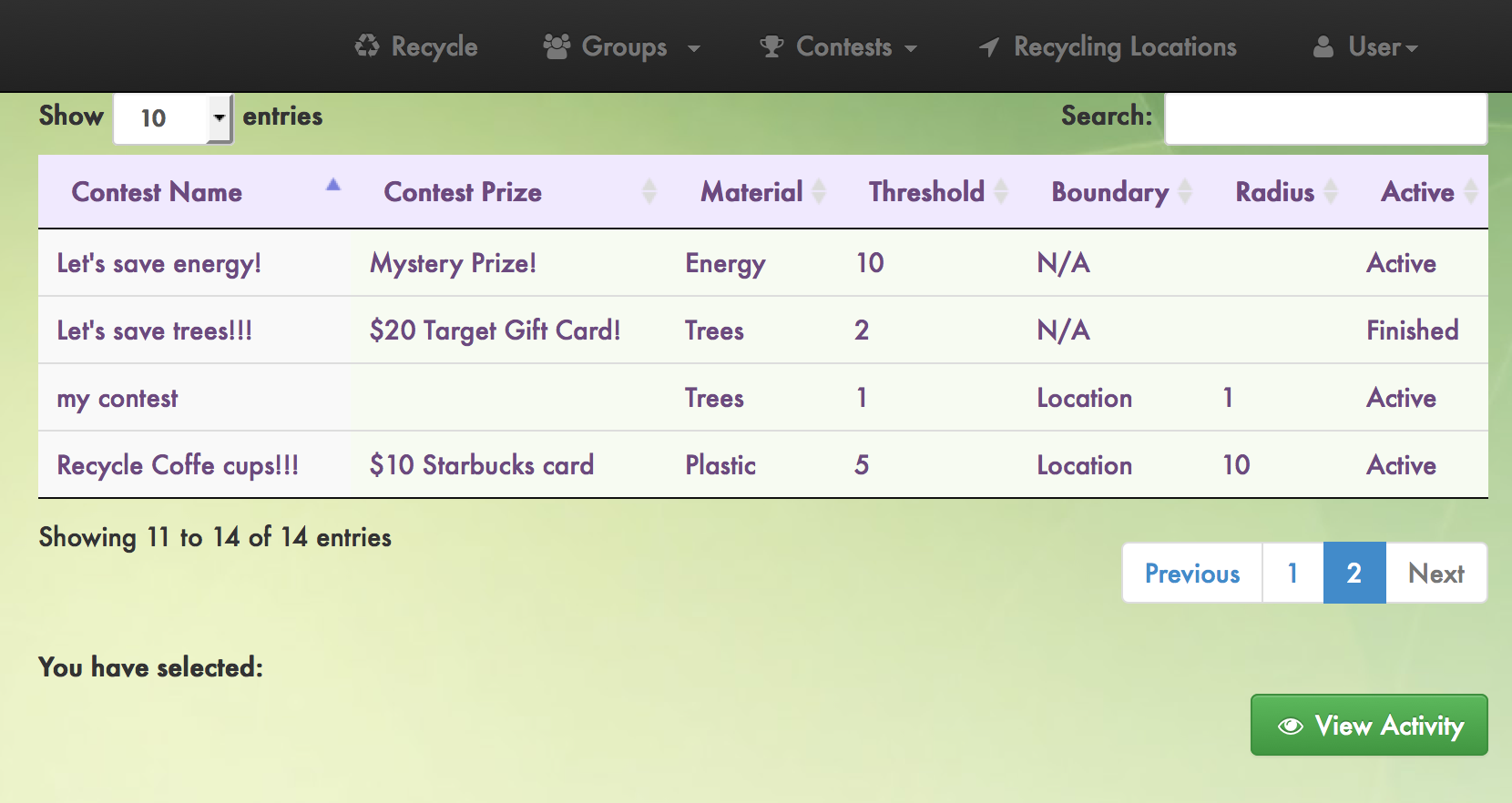
My Groups Selection Screen



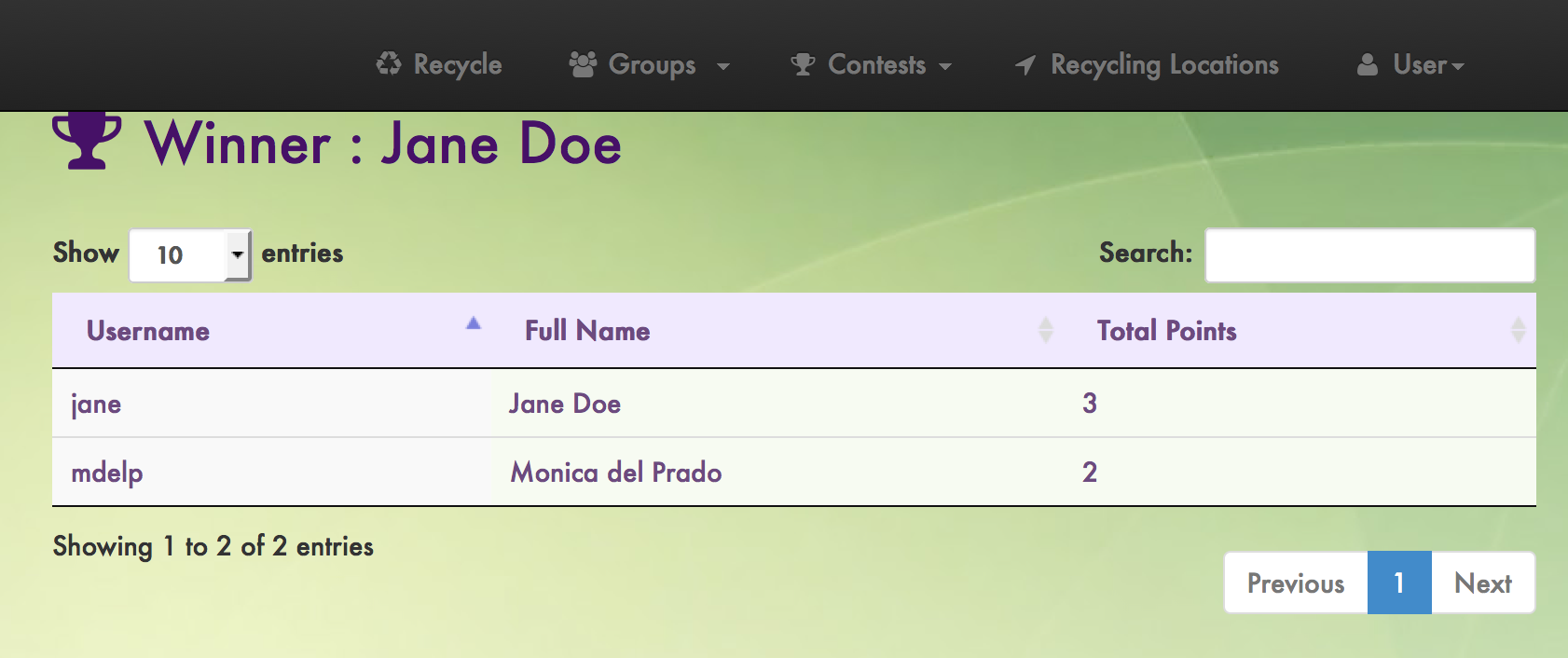
My Groups Edit Profile



New Contest Form



View Contest

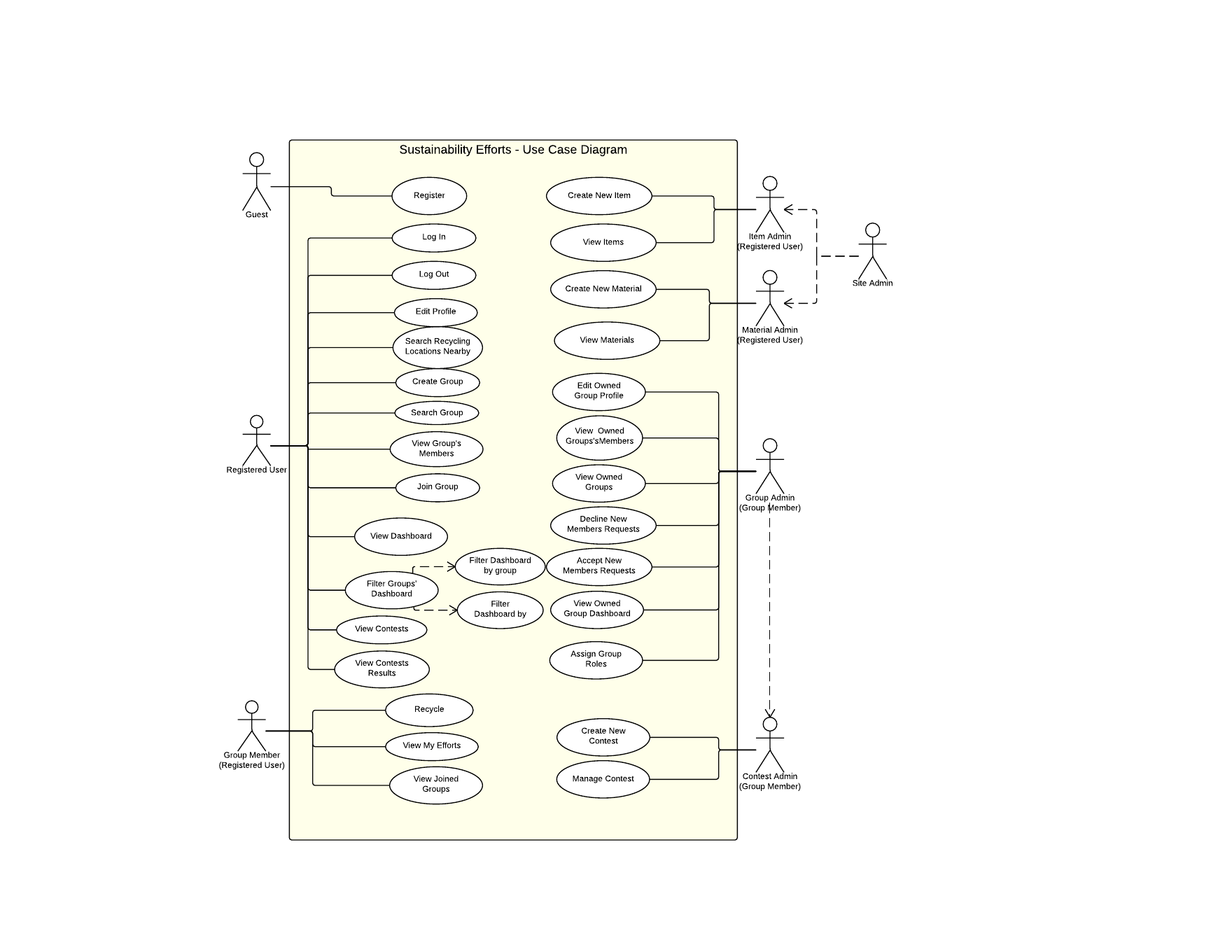


Contest Activity



Main dashboard

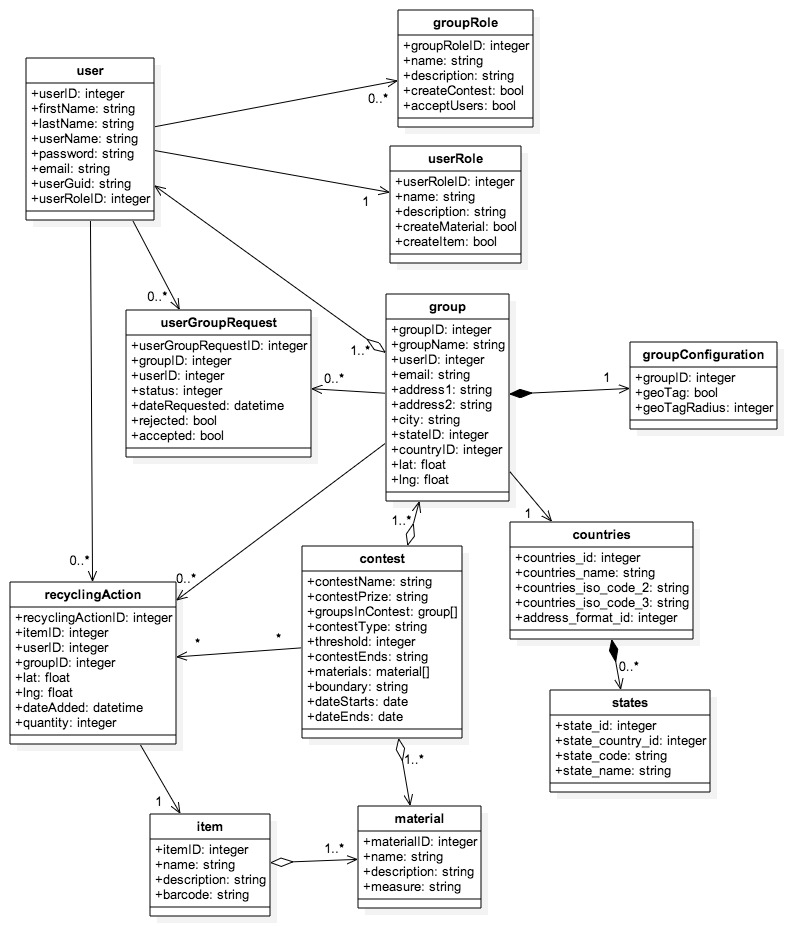
## 9.4 Appendix D – Analysis models



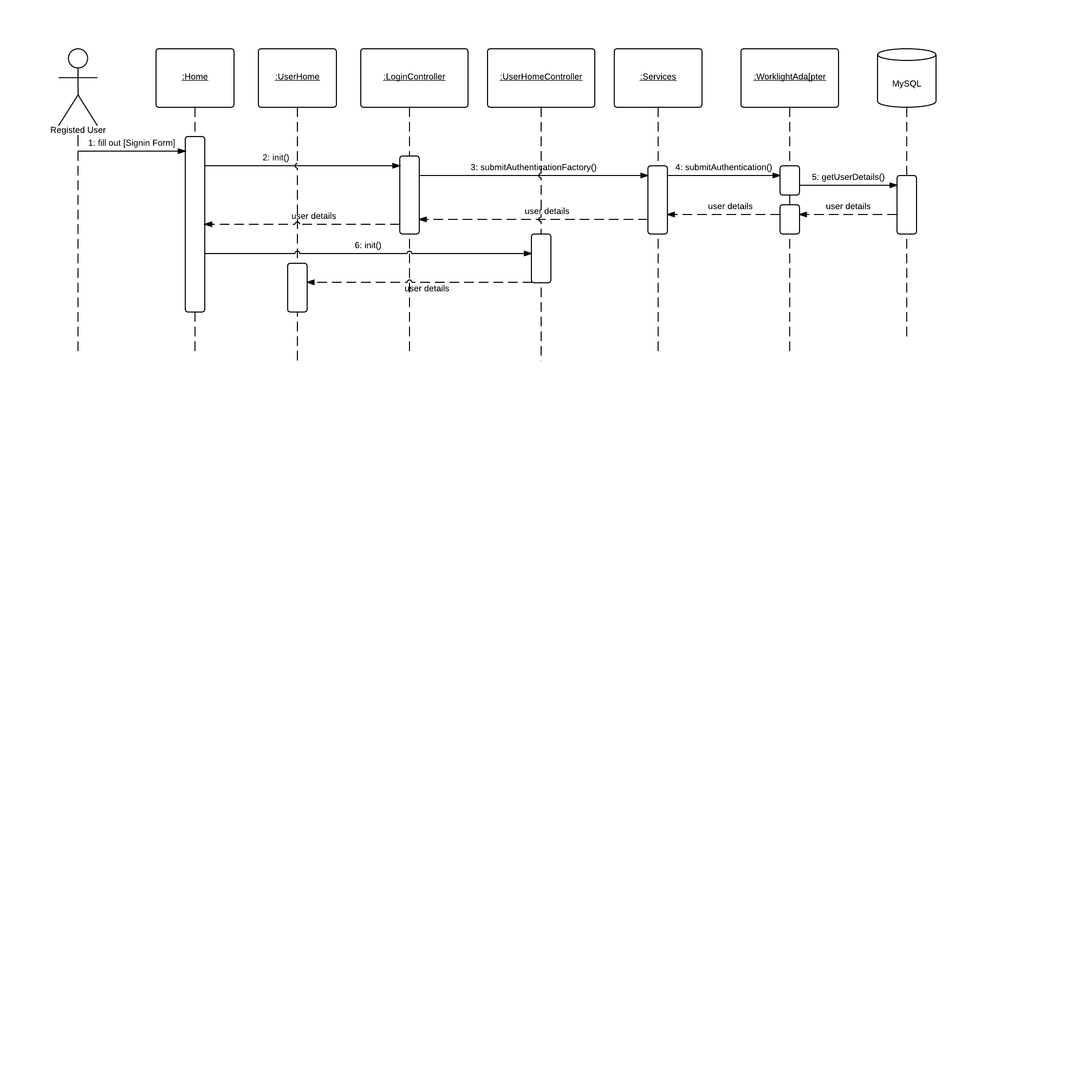
Use Case Diagram

## 9.5 Appendix E – Design Models

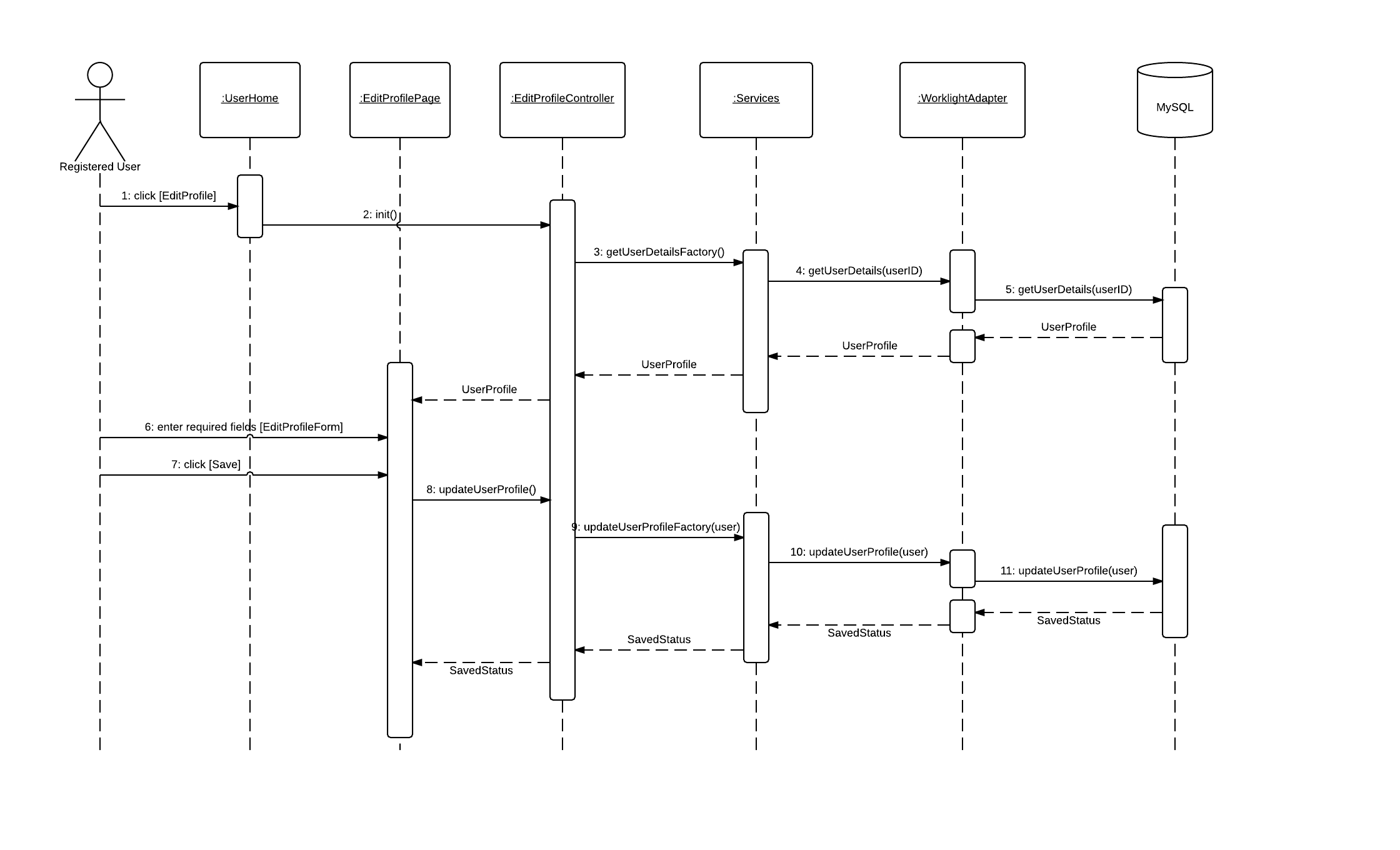
Class Diagram



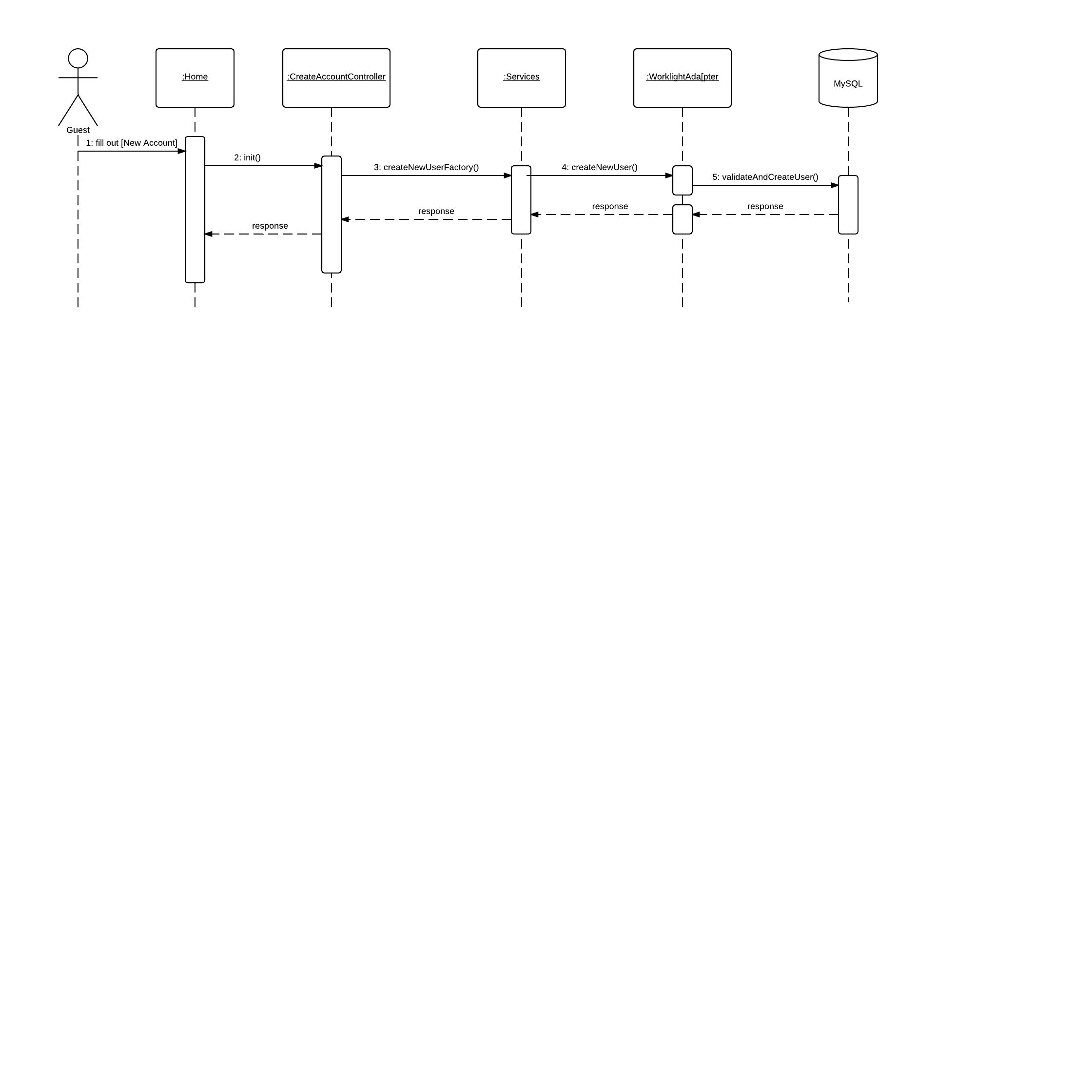
Sequence Diagrams



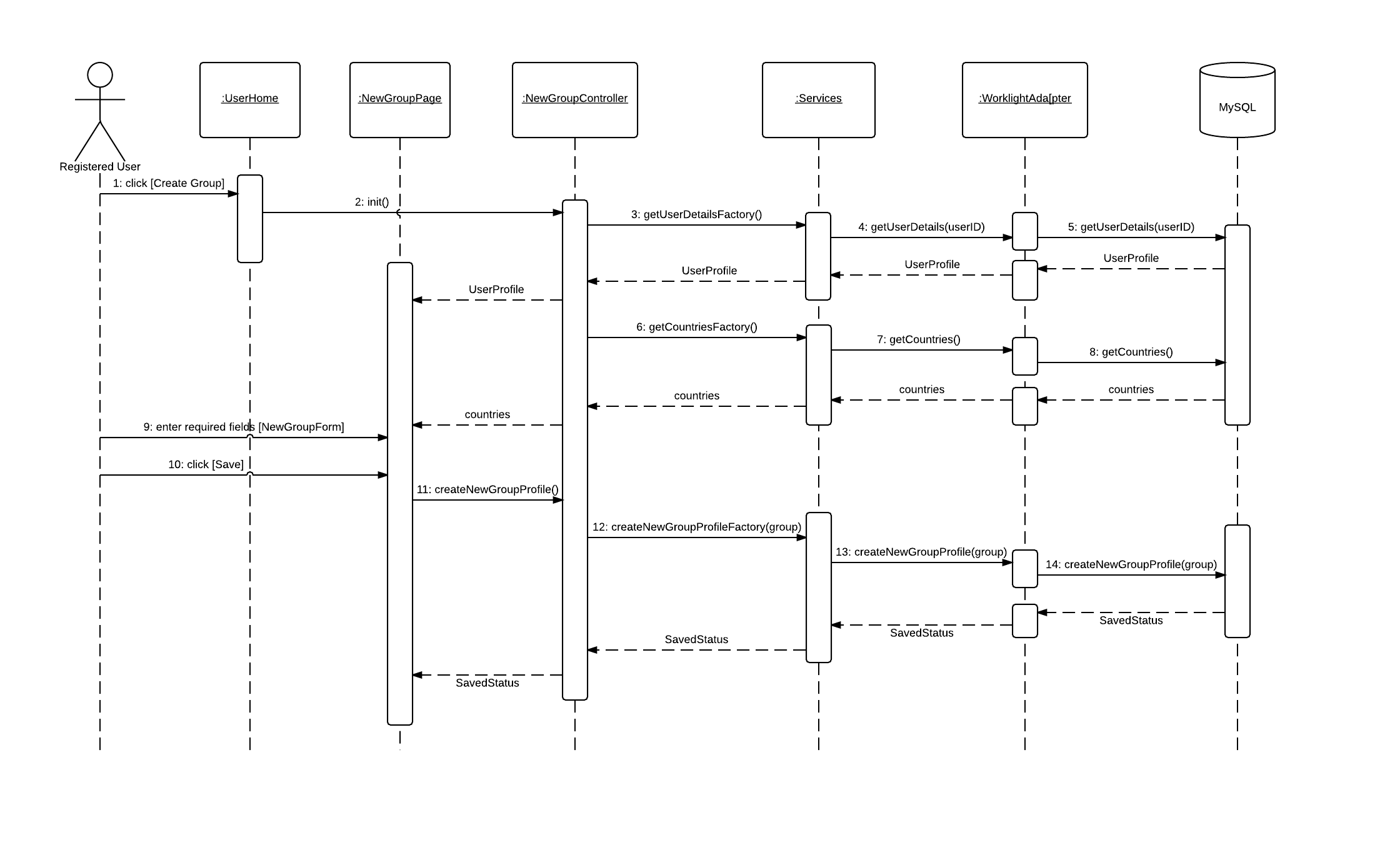
Log In



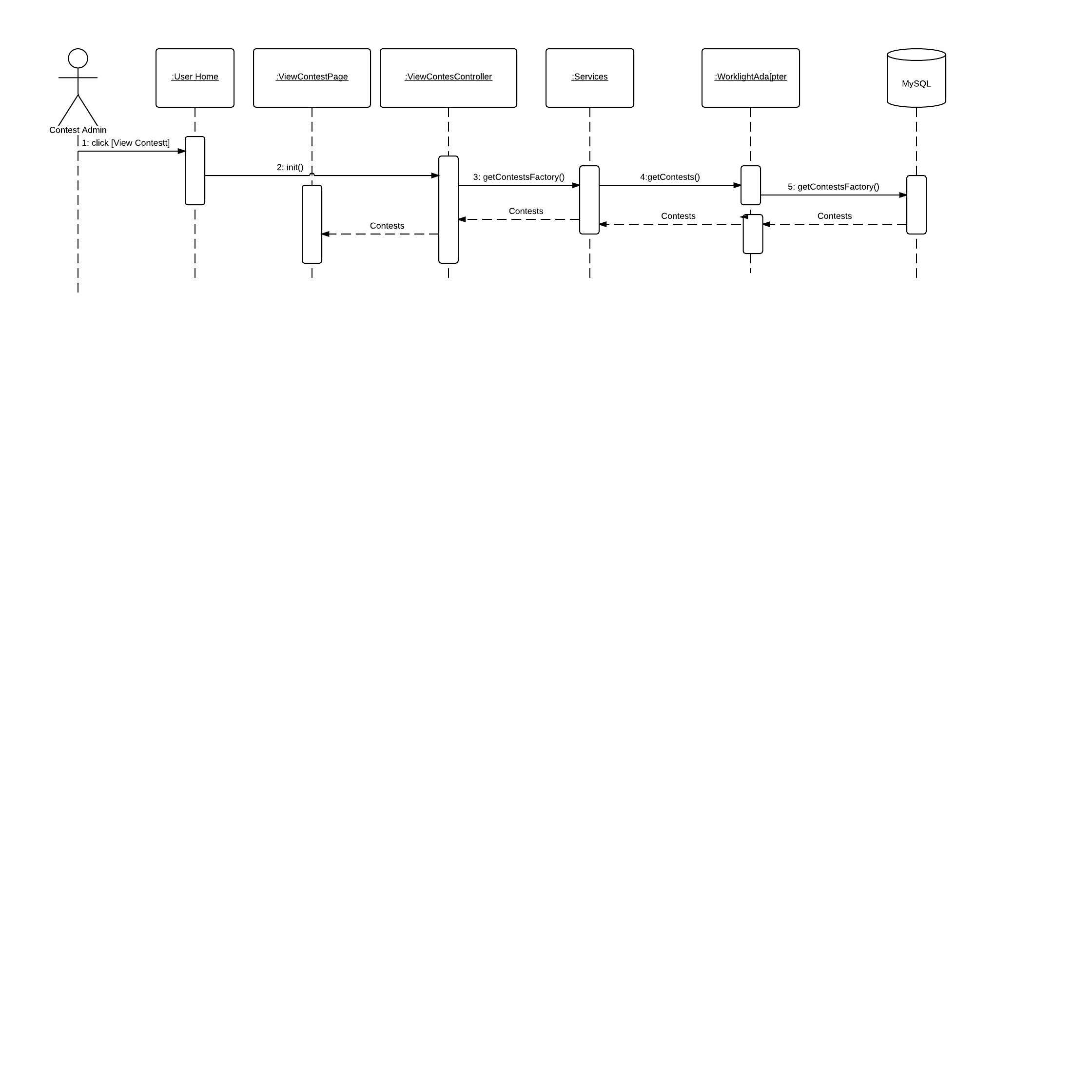
Edit Profile



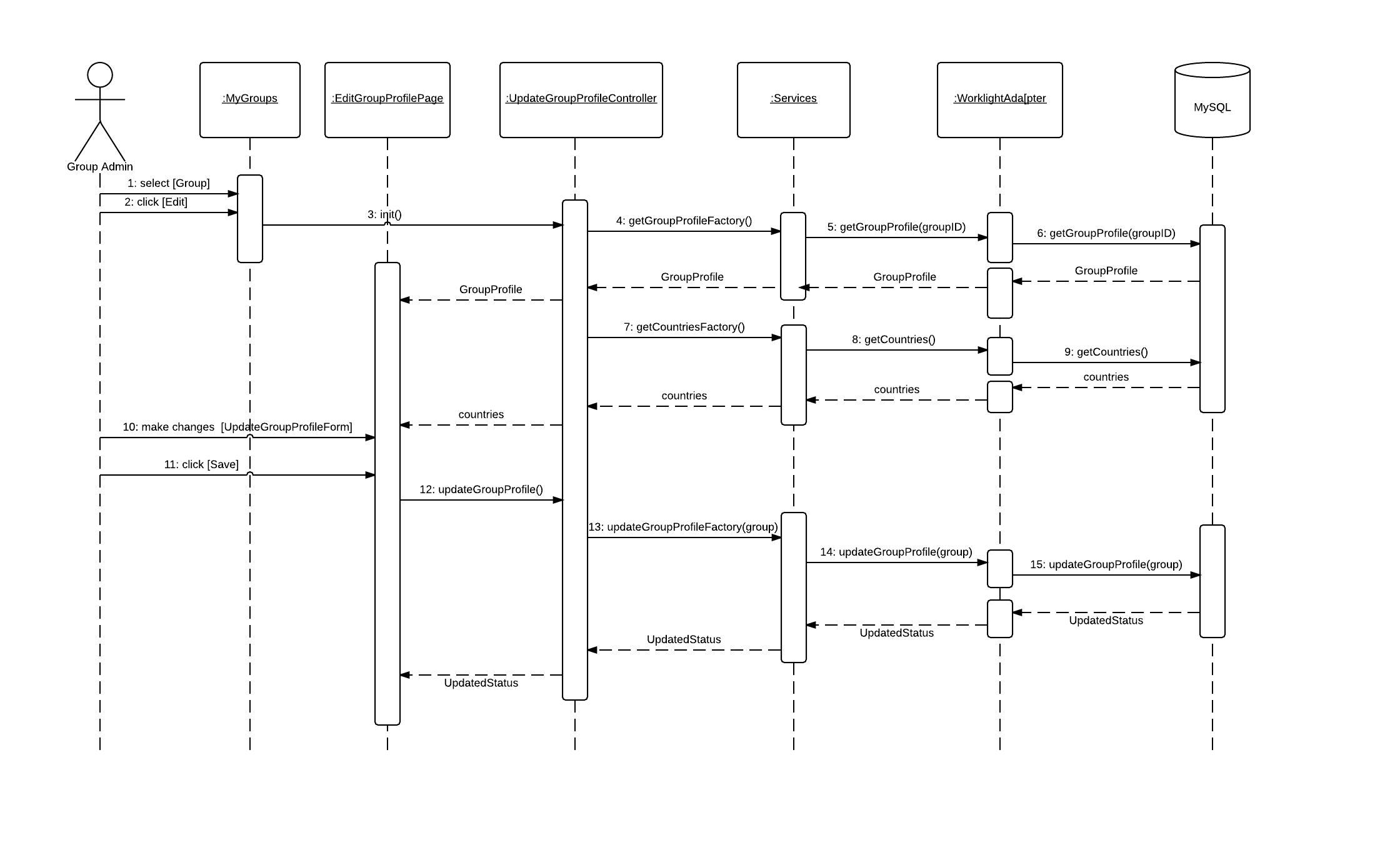
Create Account



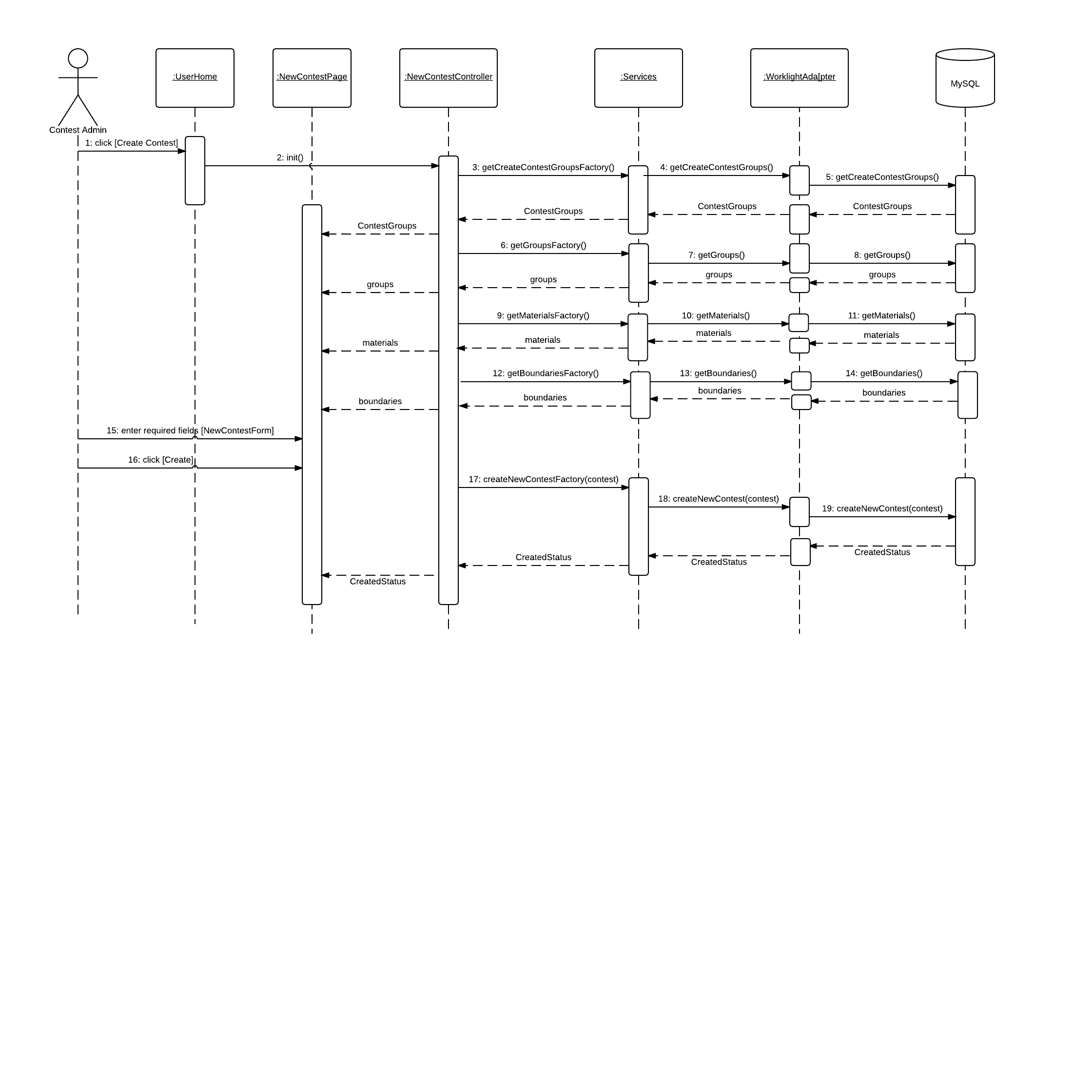
Create Group



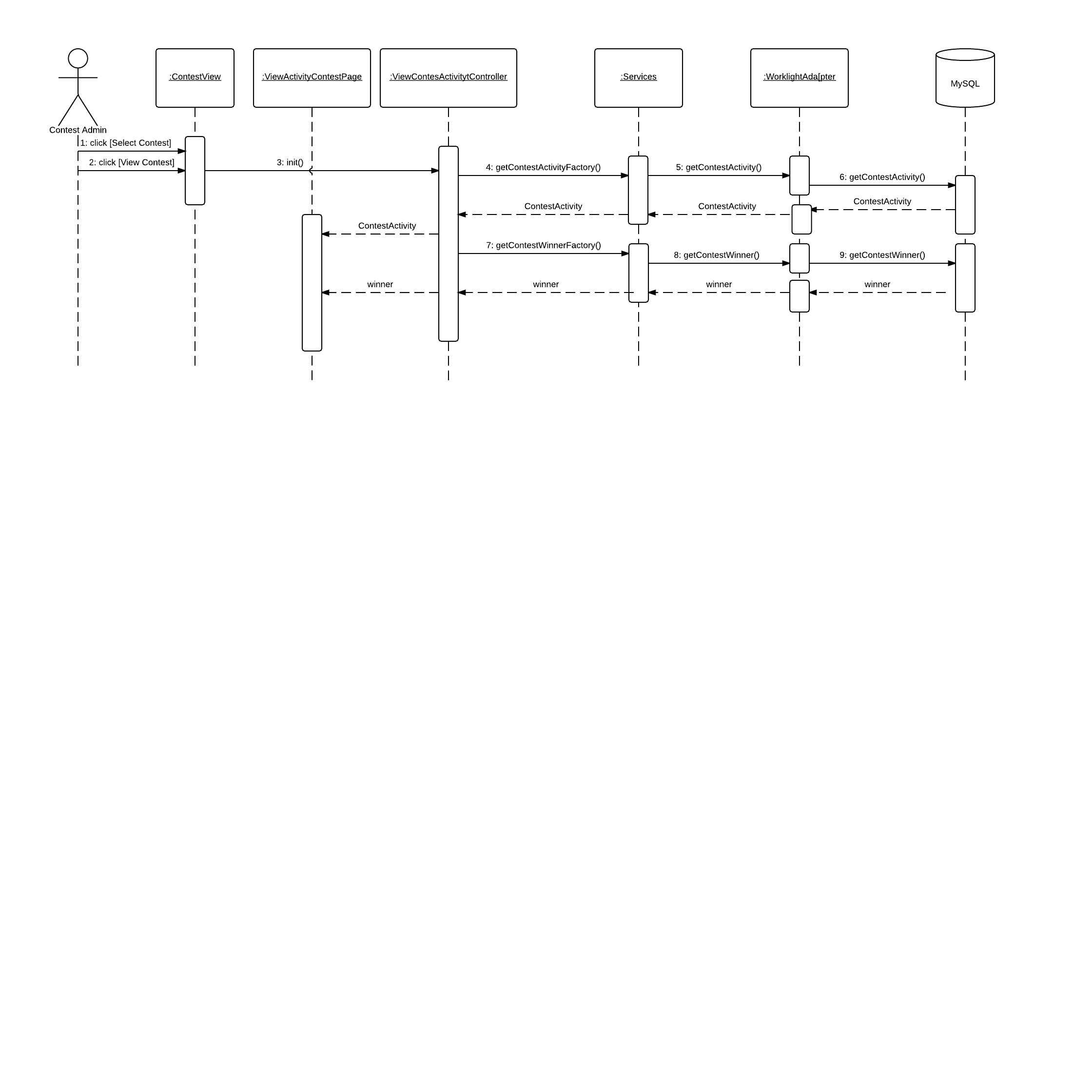
Create Contest



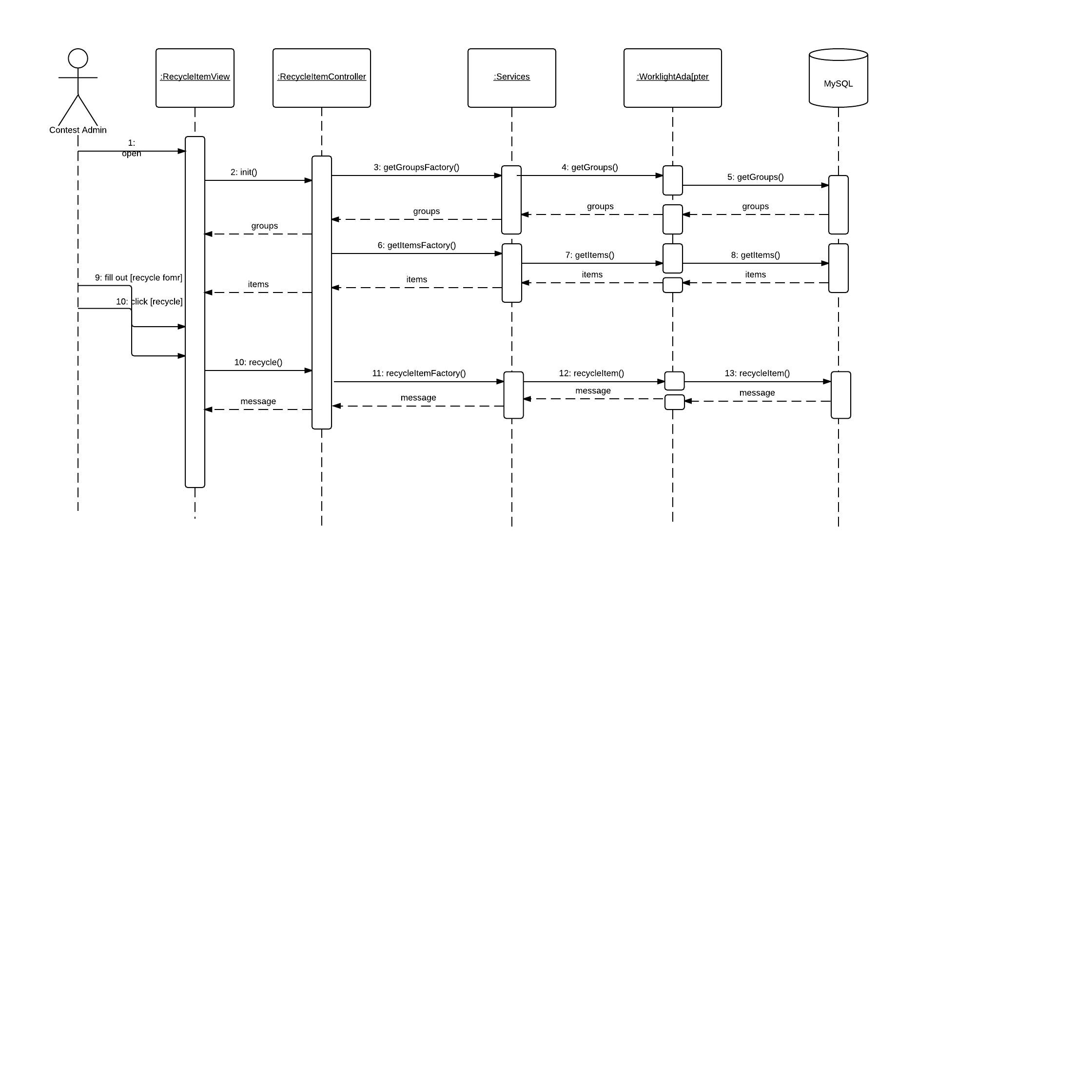
Edit Group Profile



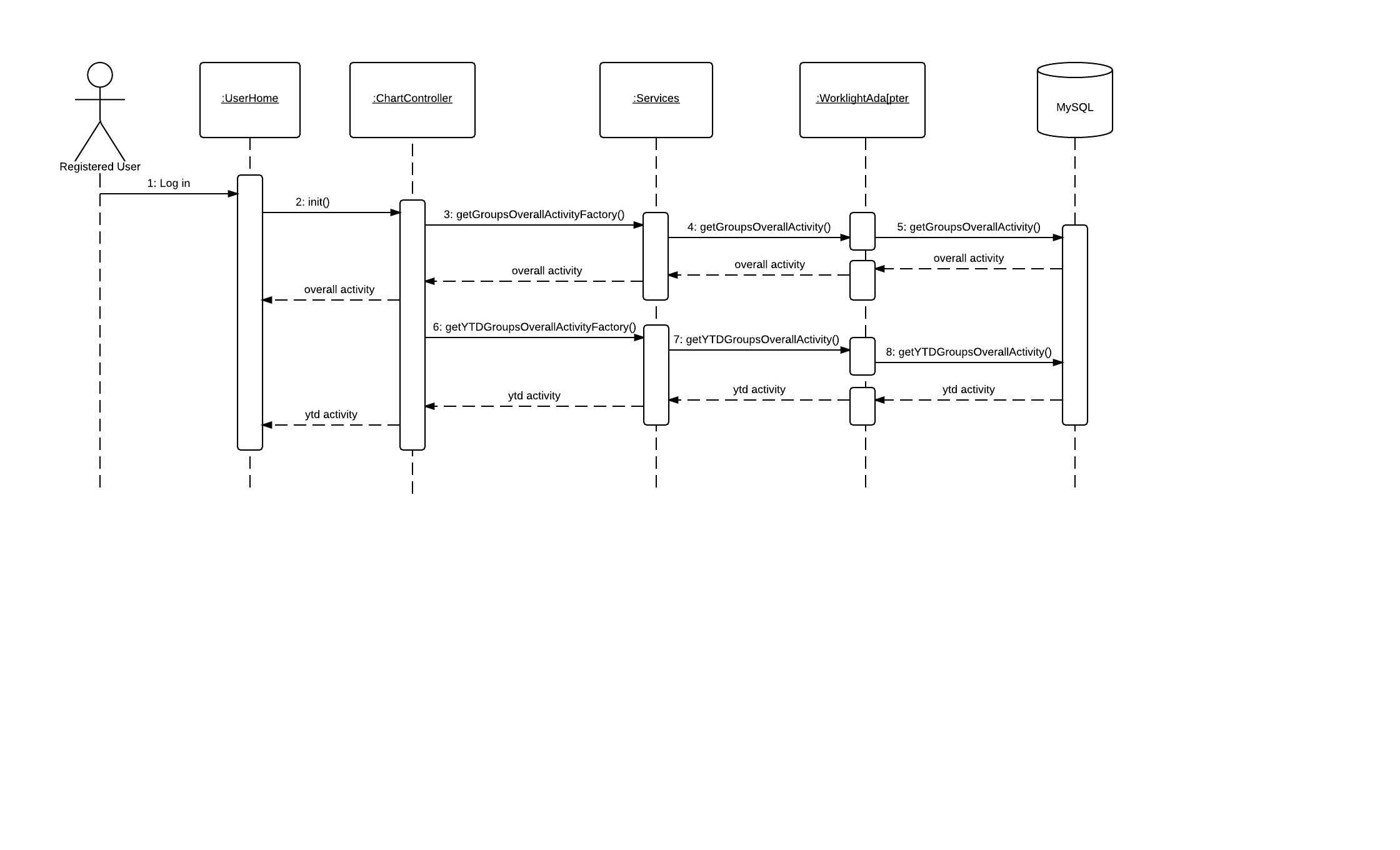
View Contest



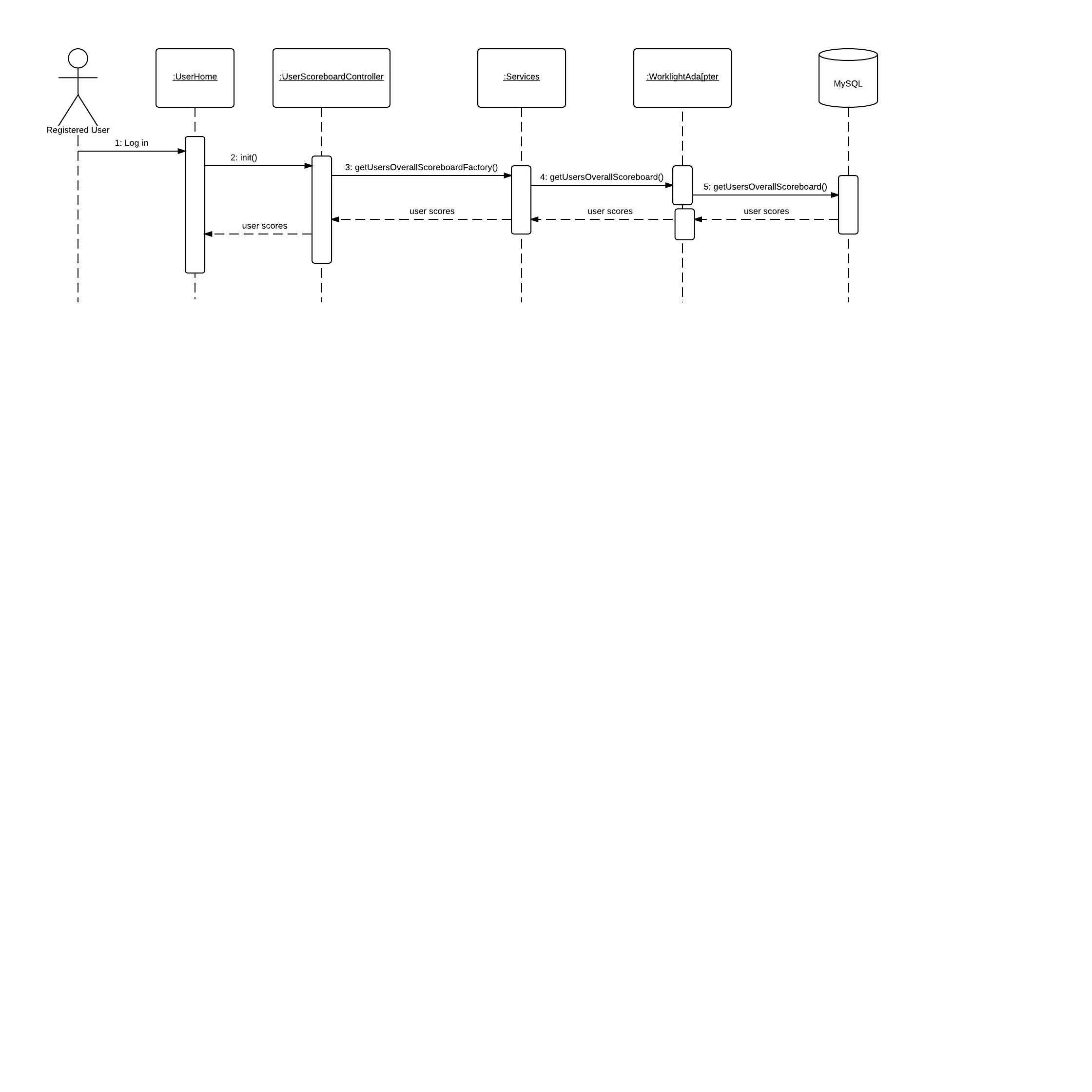
View Contest Activity



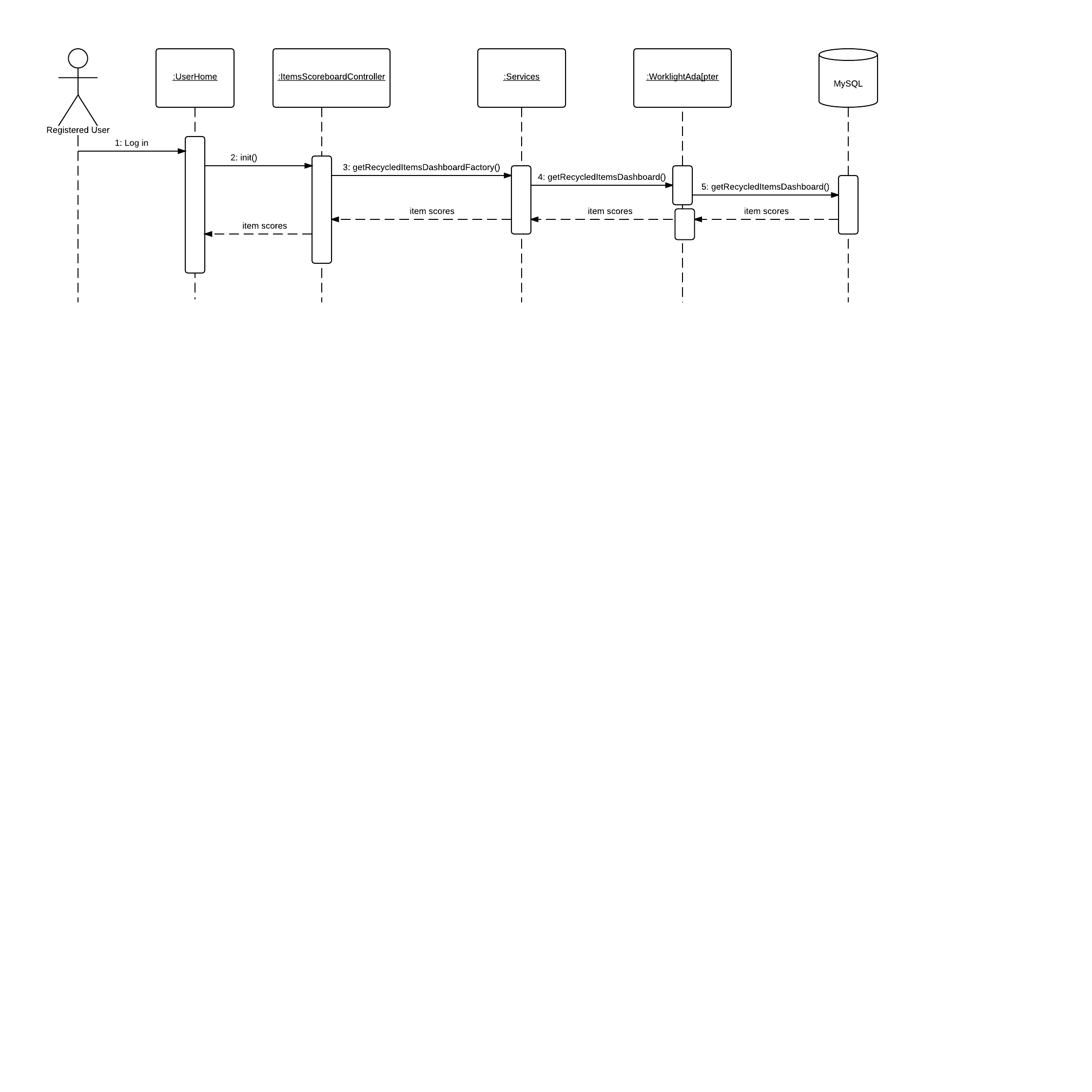
Recycle Item



Dashboard Charts



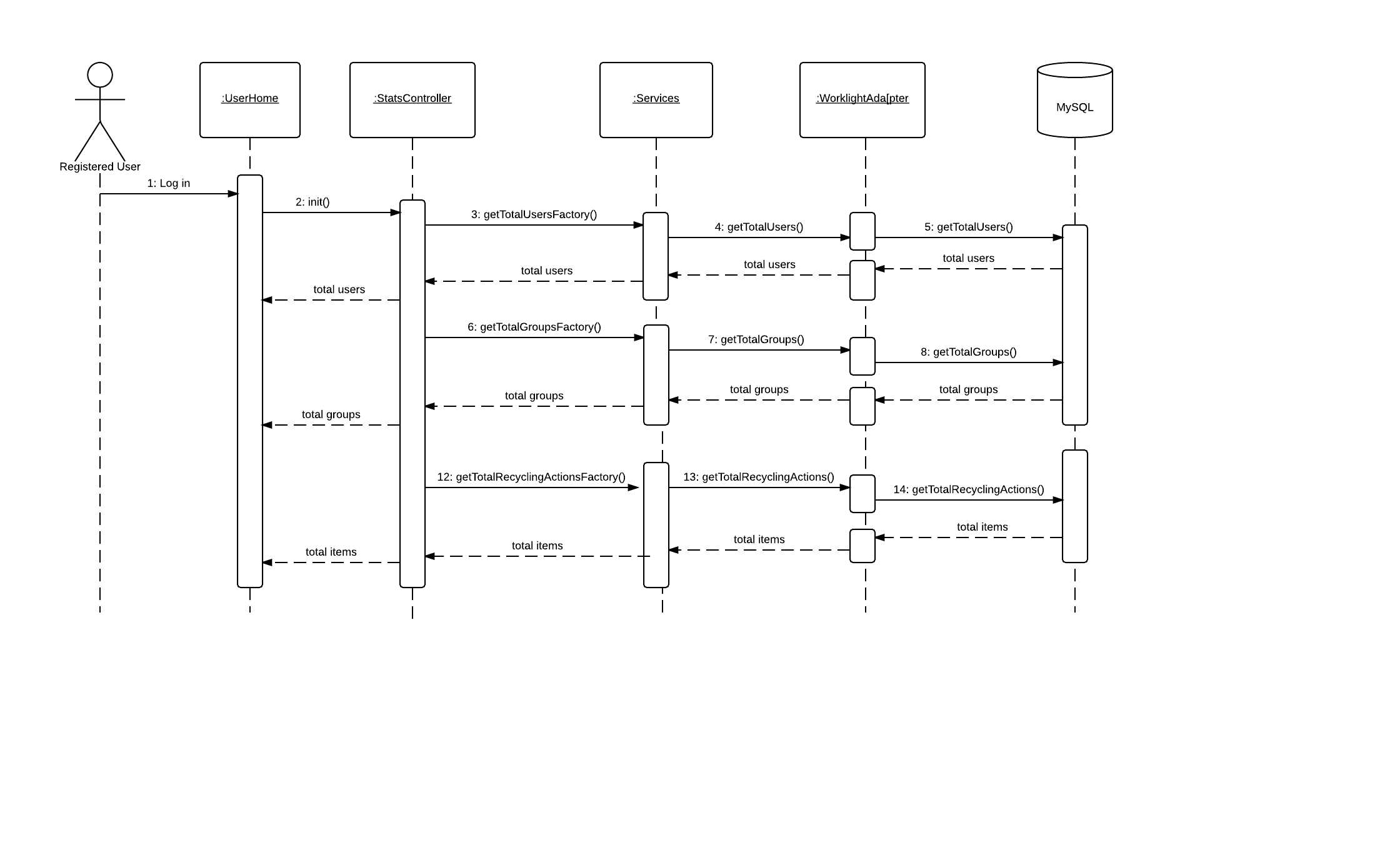
User Scoreboard



Item Scoreboard



Group Scoreboard



Site Usage widgets



Filter Dashboard

## 9.6 Appendix F – Documented Class interfaces

// User related functions

$scope.userID: userid, irstname, lastname, username, password, email //JSON object

function subithAuthentication(userID){

}

function createUser(firstname, lastname, username, password, email){

}

function getUserDetails(userID){

}

function updateUserProfile(userid, username, fname, lname, email, password) {

}

// Group related functions

$scope.group: groupName, email, address1, address2, city, stateID, countryID, zip, geoTag, radius, lat, lng//JSON object

function createNewGroupProfile(groupName, email, address1, address2, city, stateID, countryID, zip, geoTag, radius, lat, lng) {

}

function updateGroupProfile(groupid, groupname, email, address1, address2, city, stateID, countryID, zip, geoTag, radius, lat, lng) {

}

function getGroupProfile(groupID) {

}

function getGroups() {

}

function getGroupsMemberOf(){

}

function getGroupsNotMemberOf(){

}

function getGroupsOwnerOf(){

}

function getGroupUsers(groupID) {

}

function getJoinedGroupUsersView(groupID) {

}

function getStateList(countryID) {

}

function getStateByID(stateID) {

}

function getCountryList() {

}

function getCountryByID(countryID) {

}

//Material related functions

$scope.material: materialID, name, description, measure //JSON object

function createMaterial(name, description, measure){

}

function getMaterials() {

}

//Item related functions

$scope.item: itemID, name, description, barcode, materials[] //JSON object

function getItemDuplicates(name, description) {

}

function getItems() {

}

function createItem(name, description, barcode, materials) {

}

// Dashboard

$scope.charts: groups, users //JSONs objects

function getGroupsOverallActivity(){

}

function getYTDGroupsOverallActivity(){

}

function getUsersOverallScoreboard(){

}

function getGroupsOverallScoreboard(){

}

function getTotalUsers(){

}

function getTotalGroups(){

}

function getTotalRecyclingActions(){

}

function getRecycledItemsDashboard(){

}

//Contest

$scope.contet: contestName, prize, groups, threshold, ends, material, boundary, radius, dateFrom, dateTo //JSON object

function getCreateContestGroups(){

}

function createNewContest(contestName, prize, groups, threshold, ends, material, boundary, radius, dateFrom, dateTo){

}

function getBoundaries(){

}

function getContestEnds(){

}

function getContests(){

}

function getContestActivity(contestID){

}

function getContestWinner(contestID){

}

// Recycling Action related functions

$scope.recyclingAction: itemID, groupID, lat, lng, quantity//JSON object

function createRecyclingAction(itemID, groupID, lat, lng, quantity) {

}

// user\_role related

function getUserRole(){

}

// user\_group\_request related functions

function createUserGroupRequest(groupID){

}

function acceptUserGroupRequest(groupID, userID){

}

function rejectUserGroupRequest(groupID, userID){

// accept\_user\_groups\_view related functions

function getAcceptUserGroupsView(){

}

//accept\_user\_users\_view related functions

function getAcceptUserUsersView(groupID){

}

// group\_role related

function getGroupRoles(){

}

// user\_to\_group related functions

function updateUserToGroup(groupid, userid, userRoleID){

}

//Groups Filter Dashboard related funcitons

function getGroupstoFilterDashboard(){

}

function userScoreboardGetSelectedGroup(groupID){

return WL.Server.invokeSQLStoredProcedure({

}

function groupScoreboardGetSelectedGroup(groupID){

}

function itemsBoardGetSelectedGroup(groupID){

}

function getTotalUsersSelectedGroup(groupID){

}

function getTotalRecyclingActionsSelectedGroup(groupID){

}

function getYTDSelectedGroupOverallActivity(groupID){

}

function getUsersActivitySelectedGroup(groupID){

//Users Filter Dashboard functions

function getUserstoFilterDashboard(){

}

function getTotalGroupsSelectedUser(userID){

}

function getTotalRecyclingActionsSelectedUser(userID){

}

function userScoreboardGetSelectedUser(userID){

}

function groupScoreboardGetSelectedUser(userID){

}

function itemsBoardGetSelectedUser(userID){

}

function getYTDSelectedUserOverallActivity(userID){

}

function getGroupsActivitySelectedUser(userID){

}

## 9.7 Appendix G – Documented code for test drivers and stubs

var app = angular.module('app', []);

app.factory('restService', function() {

return {

getAll: function() {

// We do a $http call to retrieve the stuff

},

create: function(itemName) {

// We do a $http post to send the new one

}

}

});

app.controller('TestCtrl', function($scope, $location, restService) {

restService.getAll().then(function(items) {

$scope.libraries = items;

});

$scope.create = function() {

restService.create($scope.newItemName).then(function(item) {

$scope.libraries.push(item);

});

};

$scope.goToDetails = function(library) {

$location.path('/libraries/' + library.id + '/details');

};

});

// Tests here

describe('Controller: dashboard', function() {

var scope, restService, $location;

beforeEach(function() {

var mockRestService = {};

module('app', function($provide) {

$provide.value('restService', mockRestService);

});

inject(function($q) {

mockRestService.data = [

{

id: 0,

name: 'Angular'

},

{

id: 1,

name: 'Ember'

},

{

id: 2,

name: 'Backbone'

},

{

id: 3,

name: 'React'

}

];

mockRestService.getAll = function() {

var defer = $q.defer();

defer.resolve(this.data);

return defer.promise;

};

mockRestService.create = function(name) {

var defer = $q.defer();

var id = this.data.length;

var item = {

id: id,

name: name

};

this.data.push(item);

defer.resolve(item);

return defer.promise;

};

});

});

beforeEach(inject(function($controller, $rootScope, \_$location\_, \_restService\_) {

scope = $rootScope.$new();

$location = \_$location\_;

restService = \_restService\_;

$controller('TestCtrl', {$scope: scope, $location: $location, restService: restService });

scope.$digest();

}));

it('should contain all groups activity at startup', function() {

expect(scope.libraries).toEqual([

{

id: 0,

name: 'Angular'

},

{

id: 1,

name: 'Ember'

},

{

id: 2,

name: 'Backbone'

},

{

id: 3,

name: 'React'

}

]);

});

it('should contain ytd groups activity at startup', function() {

expect(scope.libraries).toEqual([

{

id: 0,

name: 'Angular'

},

{

id: 1,

name: 'Ember'

},

{

id: 2,

name: 'Backbone'

},

{

id: 3,

name: 'React'

}

]);

});

it('should contain widgets info at startup', function() {

expect(scope.libraries).toEqual([

{

id: 0,

name: 'Angular'

},

{

id: 1,

name: 'Ember'

},

{

id: 2,

name: 'Backbone'

},

{

id: 3,

name: 'React'

}

]);

});

it('should filter by user', function() {

expect(scope.libraries).toEqual([

{

id: 0,

name: 'Angular'

},

{

id: 1,

name: 'Ember'

},

{

id: 2,

name: 'Backbone'

},

{

id: 3,

name: 'React'

}

]);

});

it('should filter by group', function() {

expect(scope.libraries).toEqual([

{

id: 0,

name: 'Angular'

},

{

id: 1,

name: 'Ember'

},

{

id: 2,

name: 'Backbone'

},

{

id: 3,

name: 'React'

}

]);

});

});

## 9.8 Appendix H – Diary of meeting and tasks

**Meeting 09/08**  
  
Attendants: Monica del Prado, Jorge McGarry  
  
-Created website template  
  
-Agreed on initial main features:  
User administration: create, delete, change account  
User authentication  
Framework for reporting dashboard  
Group Administration by group's owners  
Group administrator's approval for new user request  
Recyclable items' administration  
Group promotion administration  
Savings ticker  
Site administration: eco saver ticker, web admin, items admin, user admin, group admin  
Detailed data page for recycled items  
Assign redemption value to recycled items  
Scan items to validate recycling action  
GPS Location with every scan  
Recycling locations' administration

Work assigned:  
Monica will work on user scenarios  
Jorge will work on website

**Meeting Minutes 9/14:**  
Attendants: Monica del Prado, Jorge McGarry  
  
Refined Website Mock up and overall design of system.  
Added a scenario to Create Recyclable Item.  
Discussed what a Recyclable Item should include. It was decided Recyclable items should be only be items that have barcodes. It was decided that the Recyclable item was suggested by users and accepted by a Site Administrator.  
Discuss adding a team aspect to the system.  
Discussed how My Groups versus Joined Groups worked.  
Discussed what Recycling locations was going to show. It was decided it would show Recycling Centers, and Location Recyclable Groups.  
Decided that a Recycled item can only be associate to one location at a time.  
Work assigned:  
Monica will finish feasibility and start working on requirement document  
Jorge will finish mobile app mockup and set up test environment

**Meeting minutes 9/15**

Attendants: Monica del Prado, Jorge McGarry

We ordered our Trello cards appropriately. They were a bit randomly assigned up to this point, so we finalized a format.

We started cross training and learning IBM worklight, working on a Hello World Project tutorial from the IBM site.

We spoke to Mr. Caraballo about our steps going forward. He recommended we do a story board with the mock ups and scenarios we have on our Feasbility document.

Work assigned,

Monica will work on the first Half the user scenario's for the Story Board.

Monica will work on completing the Hello World Project tutorial for the IBM site.

Monica will start creating the Web Page templates.

**Meeting Minutes 09/17**  
  
Attendants: Monica del Prado, Jorge McGarry, Juan Caraballo  
  
New approaches for the application:  
-More flexibility on validation aspect of the application (accepting to log items without barcodes)  
-Taking a picture of an item without could get you points as well  
-Educate people about what you recycled  
-Encourage people by providing information such as "on this program today, 5 other people did that" or "last week 100 other people did that"  
-Analysis of recycled items in terms of resources they consume or save (today because you throw this away you save this much energy and its 3 less power plants they want to build)  
  
Storyboard:  
-Elaborate more on storyboards so we can show the general flow of the application  
  
Next steps:  
-Try to schedule meeting to discuss feedback  
-Monday meeting at FIU with IBM team  
-FPL meeting will be schedule for next week (probably Wednesday)

**Meeting Minutes 9/18**

Attendants: Monica del Prado, Jorge McGarry,

We meet at starbucks.

We got Eclipse and IBM worklight installed in both are systems.

Going through the Hello World project on the ibm worklight tutorial site, we couldn't get certain things to work. The Worklight development server was causing us huge headaches as on Monica's system it failed to start and on mine as well. We finally got it working perfectly and completed the hello world tutorial (among other tutorials we have been going through).

We then installed MySQL and got the database connectivity between ibm worklight and MySQL working.

Work assigned:

Monica will work on doing more tutorials from the ibm worklight website

**Meeting Minutes 9/29**

Meeting:

Team Meeting

Date:

9/29/2014 5:00 pm to 7:45 pm

Location:

FIU Library

Attending:

Jorge McGarry, Student.

Monica Del Prado, Student.

Taking Minutes:

Jorge McGarry

Purpose:

This is a regular scheduled meeting.

Meeting Points:

* Discussed the “Stack” we would be using for the website. We agreed upon the following,
  + JQuery
  + AngularJs
  + Dojo or D3
  + MySQL
* Discussed the contests mock up we would present to the client
  + We came up with some base contests that would be created by the group owners and identified how the database structure would look
* Discussed the item creation mock up we would present to the client
  + We came up with the base elements that are required for items
    - Items should have some setup that ties into the “Green Message” ticker we plan to create
    - Items should be created and approved by the community, not some “Admin”
    - Created or Edited items need to have some sort of approval process by the “Community”
* Drafted rough copy of a class diagram. When we say rough, we mean certain things are missing but a good beginning is there.
* Refined the backlog by assigning tasks to team members responsible for completing said tasks.
* Discussed what the “Meat” of the project would be for each individual and assigned work in that fashion.
* Reviewed MyFitnessPal’s item creation process.
* Reviewed VirginPulses contest creation process.
* Put cards in development relevant to what we would like to have done by our next presentation.
* We agreed that the person who codes a particular scenario/use case will develop the use case and/or sequence diagram ahead of implementing the solution.

**Meeting Minute 10/3**

Meeting:  
Team Meeting  
Date:  
10/3/2014 4:00 pm to 5 pm  
Location:  
Conference Call  
Attending:

Juan Caraballo  
Jorge McGarry  
Pradeep P Mansey  
Roland Barcia  
Sean Sundberg  
Dennis Bly   
Taking Minutes:  
Jorge McGarry  
Purpose:  
Technical Meeting   
Meeting Points:

We were still having issues getting a production version of the IBM Worklight Server to install on our VM. Dennis was on the phone call and directed us to the area on the IBM website where we should be going, https://www-01.ibm.com/isc/esd/dswdown/dswdown.wss/web/home/home.ftl?ticket=Xa.2/Xb.XJi39I33KMb2eGC74hfJktx4f3TI2ckKLkRvOTXv\_A/Xc.2000001GKJ/Xd./Xf./Xg.7746842/Xi.2000001GKJ/XY.scholars/XZ.QlEJt3b3ViFLLOFKoUxRJqri2sI&link\_id=SDHome.

I asked some questions in regards to authentication and how we should be going about it with IBM Worklight. Rolando suggest that having a simple username / password lookup that we are currently be using may not be the best route to go. He suggested we tap into LDAP functionality of the WebSphere liberty application, which is packaged with IBM Worklight Server.

I asked some question in regards to our authentication scheme. Rolando and Sean suggested that we needn't worry about protecting pages which don't have sensitive data, as those pages can easily be reviewed by anyone anyway. Any sensitive data needs to be passed through an adapter thats automatically authenticated by worklight.

I asked a question about how to setup environment to use two seperate frameworks, boostrap for desktop browser and ionic for mobile. The purpose of the question was to see if there was a more efficient way of combining the two rather than re-coding all front end pages. Rolando and Sean chimed in and said there wasn't a "common" way to do this and write one html page, it would have to be seperated.

**Meeting Minute 10/5**

Meeting:  
Team Meeting  
Date:  
10/5/2014 2pm to 7 pm  
Location:  
Starbucks  
Attending:  
Monica del Prado  
Jorge McGarry  
  
Taking Minutes:  
Monica del Prado  
Purpose:  
Prepare next presentation  
Meeting Points:  
Complete merging Jorge's and Monica's code  
Debug errors on Monica's laptop  
Jorge finished implementation of New User and Login use cases  
Monica completed Edit Profile use case  
Jorge started installation of IBM Installation Manager  
Jorge installed ftp on the VM  
Got ready for Monday's presentation

**Meeting Minute 10/06**

Meeting:  
Team Meeting  
Date:  
10/6/2014 4:00 pm to 5 pm  
Location:  
FIU Library  
Attending:  
Jorge McGarry  
Monica Del Prado  
Taking Minutes:  
Jorge McGarry  
Purpose:  
Regular Weekly Meeting  
Meeting Points:  
Met to decide how we go forward based on this weeks presentation  
We need Setup meeting with Manual Bascuas to approve contest mock-up ideas  
We decided on what use cases each of us would start to work on.

**Meeting Minutes 10/17**

Meeting:  
Client Meeting

Team Meeting  
Date:  
  
10/20/2014 6 pm to 9 pm  
Location:  
Starbucks  
Attending:

Manuel Bascuas (IBM)

Nancy Mulshine (FPL)  
Jorge McGarry  
Monica Del Prado  
Taking Minutes:  
Jorge McGarry  
Purpose:

We wanted to finalize our contest requirements with Mr. Bascuas. Unfortunately Mr. Bascuas was not ready to discuss, nor was Nancy Mulshine, who also attended the call to get status on our progress. Mr. Bascuas suggested we should meet in person on Monday after class, to which we agreed.

We optimized the time by going to starbucks and deciding what to work on and what to have done before our presentation on Monday.

**Meeting Minutes 10/19**

Meeting:  
Client Meeting

Team Meeting  
Date:  
10/19/2014 5 pm to 10 pm  
Location:  
Starbucks  
Attending:

Jorge McGarry

Monica Del Prado  
Taking Minutes:  
Jorge McGarry  
Purpose:

We wanted to finalize our first official build which we were going to introduce in our presentation and to our clients.

We had a lot of code that required merging, since we had been working a bit seperately until this point.

I did work on pushing the application to the VM and testing it.

Monica did work on beautifying the website and testing it.

**Meeting Minutes 10/20**

Meeting:  
Team Meeting  
Date:  
10/23/2014 6 pm to 8 pm  
Location:  
Starbucks  
Attending:  
Jorge McGarry  
Monica Del Prado  
Taking Minutes:  
Monica del Prado  
Purpose:  
Discussion about what needed to be finished for Version 1 of the application and features for Version 2 of the application.

Discussion about new changes to current application to deliver Version 1.

**Meeting Minutes 10/28**

Team Meeting  
Date:  
10/28/2014 5 pm to 6 pm  
Location:  
FIU  
Attending:  
Jorge McGarry  
Monica Del Prado  
Taking Minutes:  
Jorge McGarry  
Purpose:

Met with Monica to discuss current sprint progress.

Monica has found a suitable dashboard framework she needs to use, chartjs. It seems to integrate really nicely with angularjs doing it the "Angular Way". She has begun testing with it and will report on progress and hopefully have something for production by next presentation.

I managed to get the VM up and running but there are still several bugs on the VM that do not appear on the local version of the application which i'm still troubleshooting. I'm hoping I can provide an android version and a desktop version by presentation time that I can [demo](https://moodle.cis.fiu.edu/v2.1/mod/assignment/view.php?id=22689).

We touched a bit on documentation and we know we are a bit behind and will try to get that caught up by presentation time as well.

We had a planned meeting with our clients for 5:30pm but it was moved to 5:30pm Tuesday rather than Monday, due to scheduling conflicts.

**Meeting Minutes 11/08**

Meeting:  
Date:  
11/8/2014 3 pm to 7 pm  
Location:  
Starbucks  
Attending:  
Jorge McGarry  
Monica Del Prado  
Taking Minutes:  
Monica del Prado

Jorge worked on:

Add recycling action

Explain Worklight Server on VM to Monica

Monica worked on:

Fix bugs in application

complete pending functionalities on existing pages

start contest creation

**Meeting Minutes**

Meeting:  
Date:  
11/9/2014 8 am to 6 pm  
Location:  
Starbucks  
Attending:  
Jorge McGarry  
Monica Del Prado  
Taking Minutes:  
Jorge McGarry

Purpose:

To get together and work on merging and discussing the agenda for next week.

We discussed that we would like to have all the main components of the application complete by Monday 11/10. Then, we would like to start working on documentation and getting the site approved by the next presentation.

We discussed that I would be working on the following remaining functionality,

Site Navbar work based on user role

Group owner assigning group roles to certain users

Monica would working on the following remaining functionality,

Creating the contest

Actually coding the contest algorithm so it picks winners, etc.

Monica will start coding the edit profile page.

# References

**None**