**SWE 574 Software Development As A Team, Fall 2015**

**Instructor: Suzan Üsküdarlı**

**Project BUCOMP**

**Design Specifications Document**

**02.11.2015**

**Revision 1.0**

**By Group 1**

# Revision History

|  |  |  |
| --- | --- | --- |
| **Revision** | **Date** | **Explanation** |
| 1.0 | 02.11.2015 | Initial design |
|  |  |  |

# Introduction

The purpose of this software project is to develop the BUCOMP, **B**oğaziçi **U**niversity **CO**mmunity **M**anagement **P**latform, in Java, Linux/Windows and MySQL environment. BUCOMP is a web/mobile application which allows its users to manage communities and communications among community members by providing various functionalities.

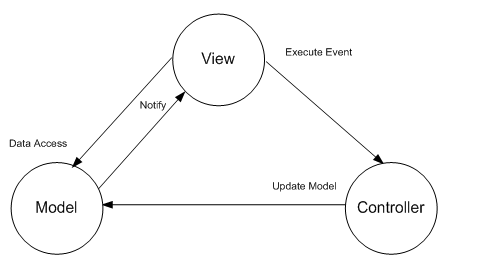
The design is based on BUCOMP Requirements Specification Document, final revision. The notation used in this document to describe the design of the application is mainly UML and conforms to organizational specifications given in [1]. The software architecture, overall high-level structure, components in terms of packages and classes and design details of all application functions and the user interface are given in later sections of this document.

# System Design

To be filled

## Software Architecture

MVC is used in BUCOMP project as the software architectural pattern. The model is MSSQL; View is an interface that is created with HTML, CSS and JQuery classes. Controller is the Java based web service and the communication between controller and the view is done by Ajax methods. See the figure below for MVC pattern.



## Software System Components

The BUCOMP solution will comprise of following system components.

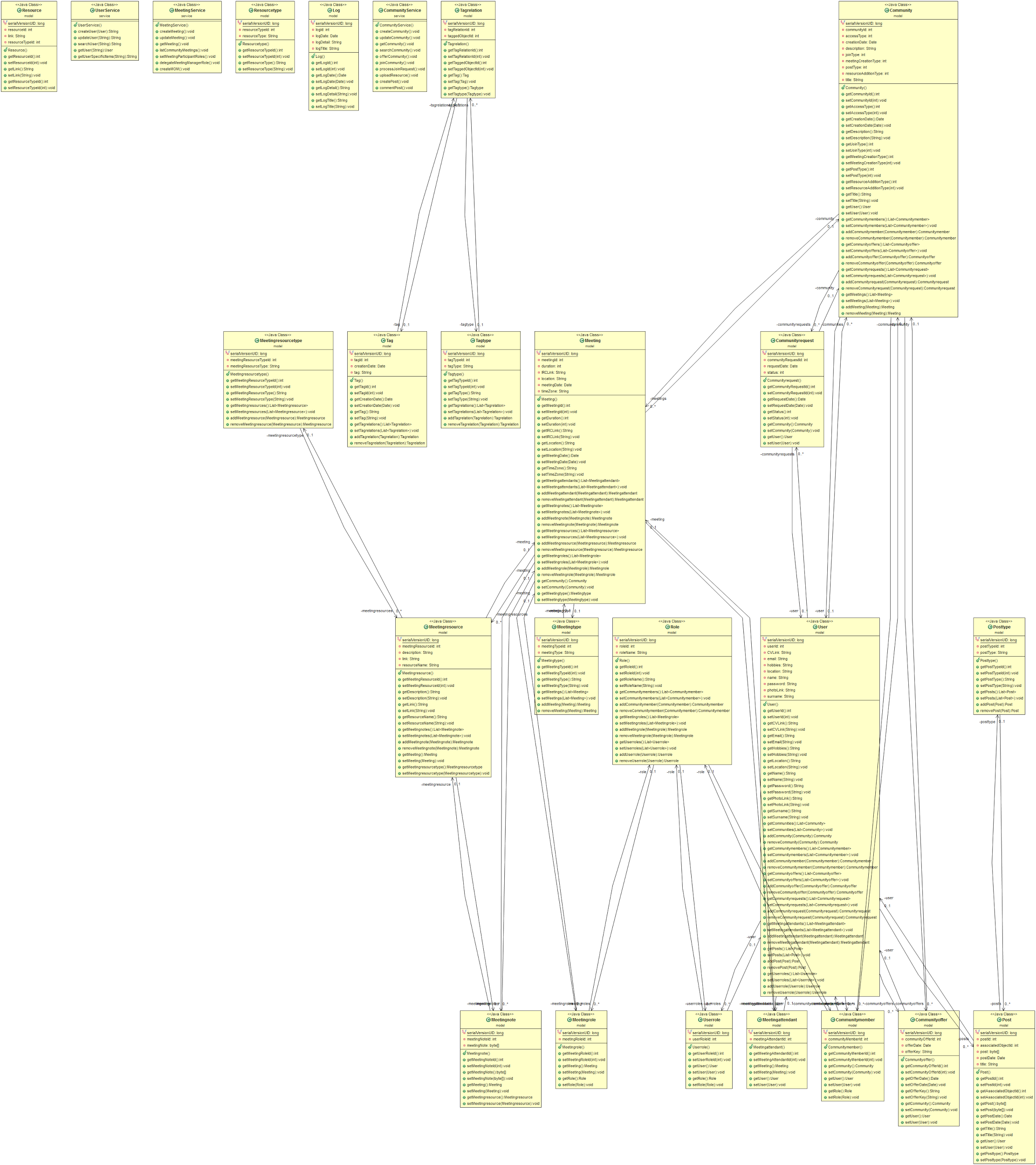
1. Database management system
2. Backend application
3. Web application
4. Mobile application

Backend application constitute the application’s backbone as it will provide access to database, and both mobile and web application will consume its RESTful web services. This part of the application is mainly responsible for data, thus a UML class diagram to show all the classes and data attribute is provided.

The classes in this diagram can be divided in 2 groups. First group is entity group, which coresponds database tables and largely for data representation. It conforms the “data transfer object” paradigm.

The second part is related to service classes, mainly provide web service methods which carry business logic, to manipulate data inside entity classes.

Following is the UML class diagram for BuComp back-end applcation.



[Full View of the Class Diagram](https://drive.google.com/file/d/0B5W-_hH437BiNDVxMzgzdDc5NTQ/view?usp=sharing)

## Environment

BUCOMP application development & runtime environment tools are listed in below:

|  |  |  |
| --- | --- | --- |
| Name | Type | Version |
| OS | Windows or Linux |  |
| Java | Programming Language | JDK 1.8 |
| JQUERY | Javascript Library | 1.11.x |
| Eclipse | IDE | Mars 4.5 |
| Maven | Build & Dependency Management | 3.3.3 |
| IntelliJ | IDE | 2015 |
| MySQL Server | Database Server | 5.7.9 |
| MySQL Workbench | Database Design & Modeling | 6.3.5 |
| Postman | Build, test and documentation tool for APIs. |  |

# Detailed Design

To be filled.

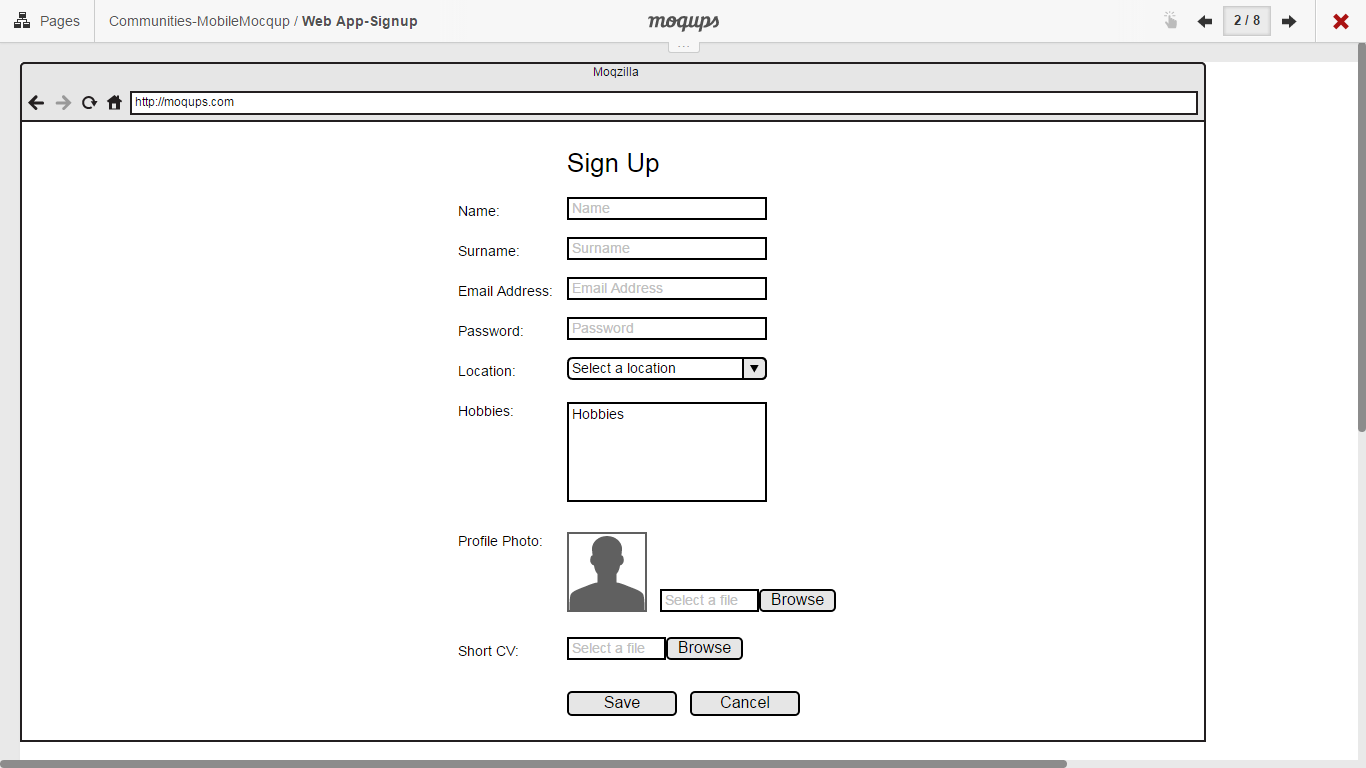
## Frontend Design

The BUCOMP system is divided into three components for web application. The view component will use;

* HTML pages
* CSS scripts for designing HTML pages
* Ajax call for controller methods
* JSON data that will be provided by Ajax calls

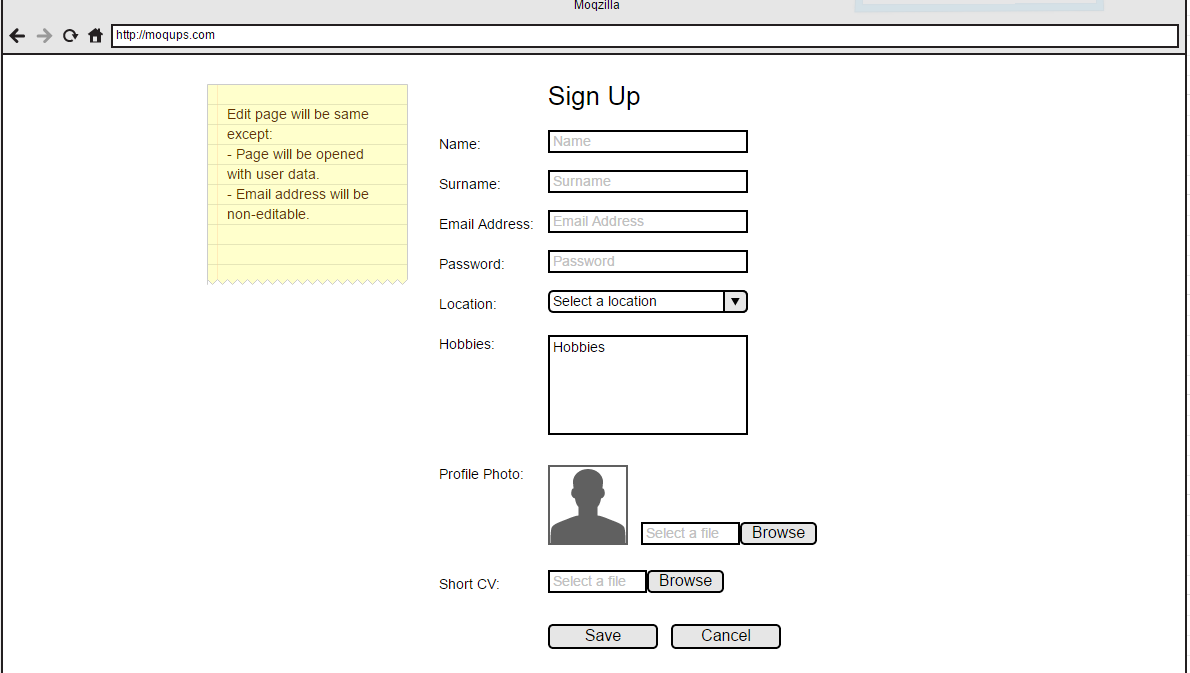
## Create User

Visitors should be able to register to the system by creating their profile on the system via “register.html” page which is shown in mockup below.



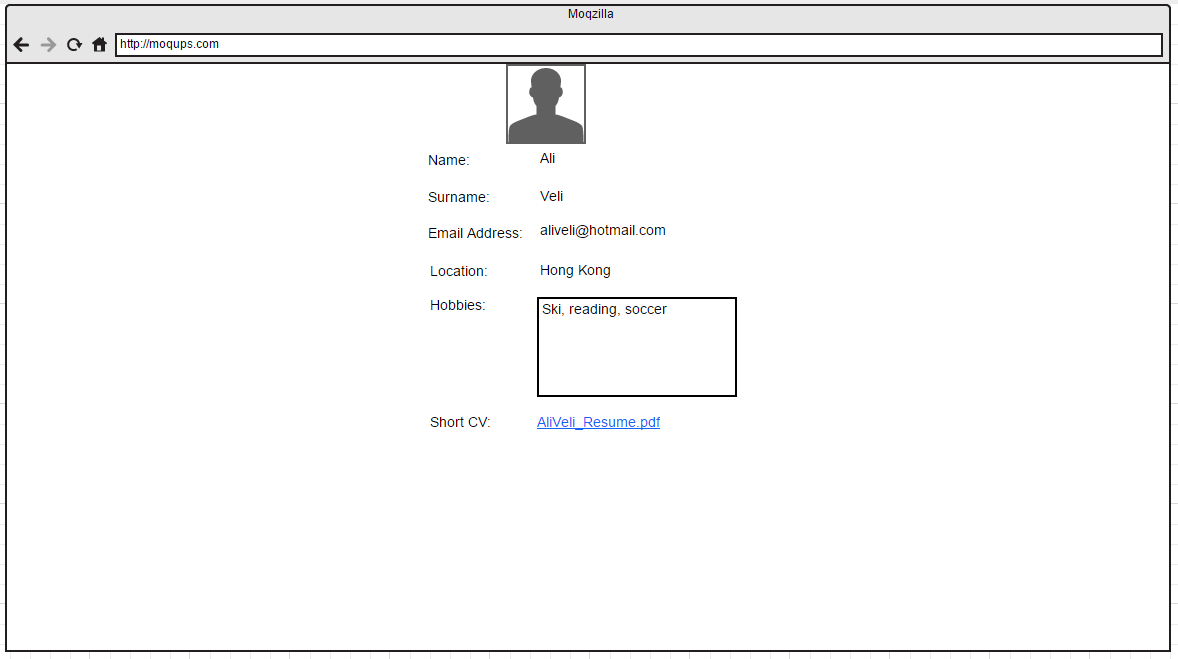
## Update User Information

Registered users should be able to update their profile information. on the system via “profile.html” page which is shown in mockup below.



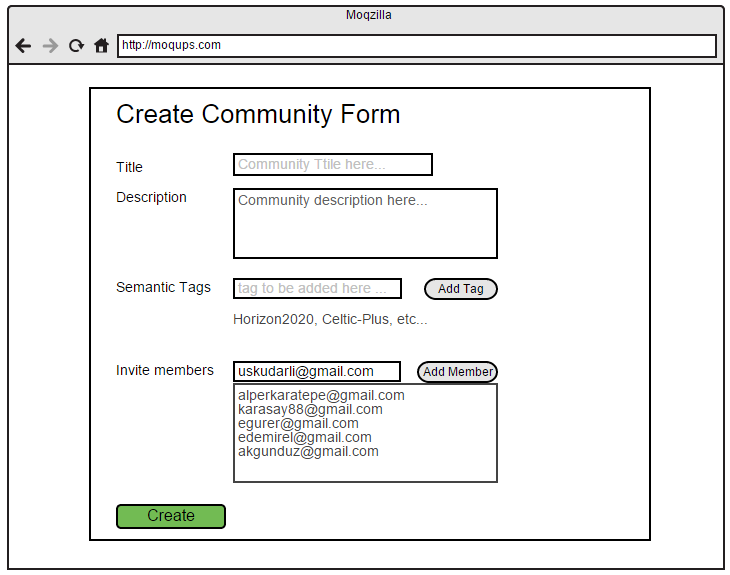
## Show/update User Details

## All registered users should be listed and user details should be displayed when clicked on the name via “UserProfile.html” page which is shown in mockup below.



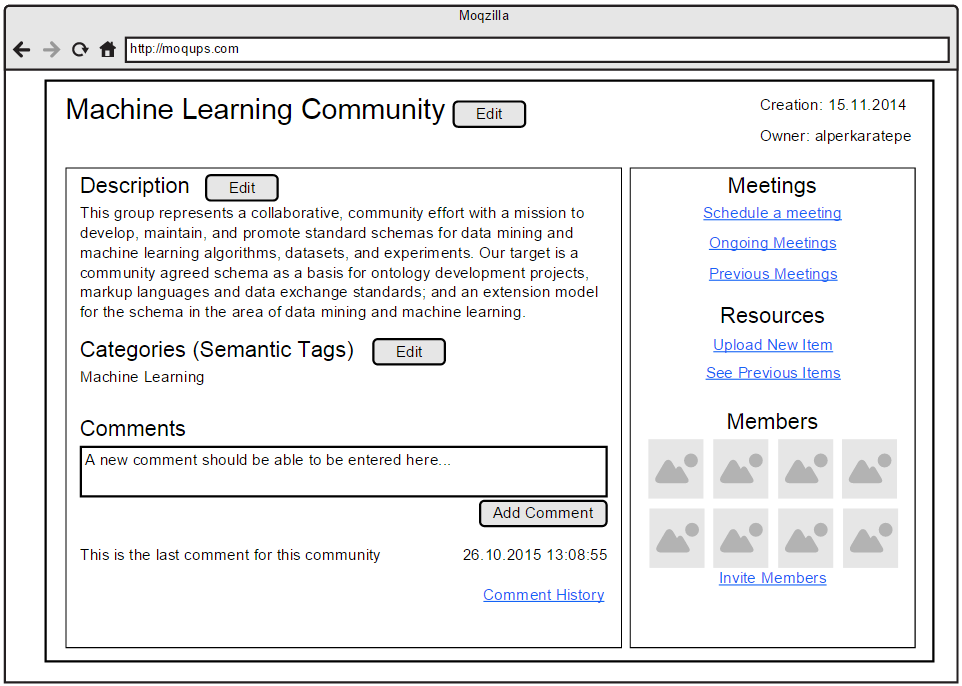
## Create Community

## Registered users should be able to create communities via “CreateCommunity.html” page which is shown in mockup below. The same page will be used for updating related community information.



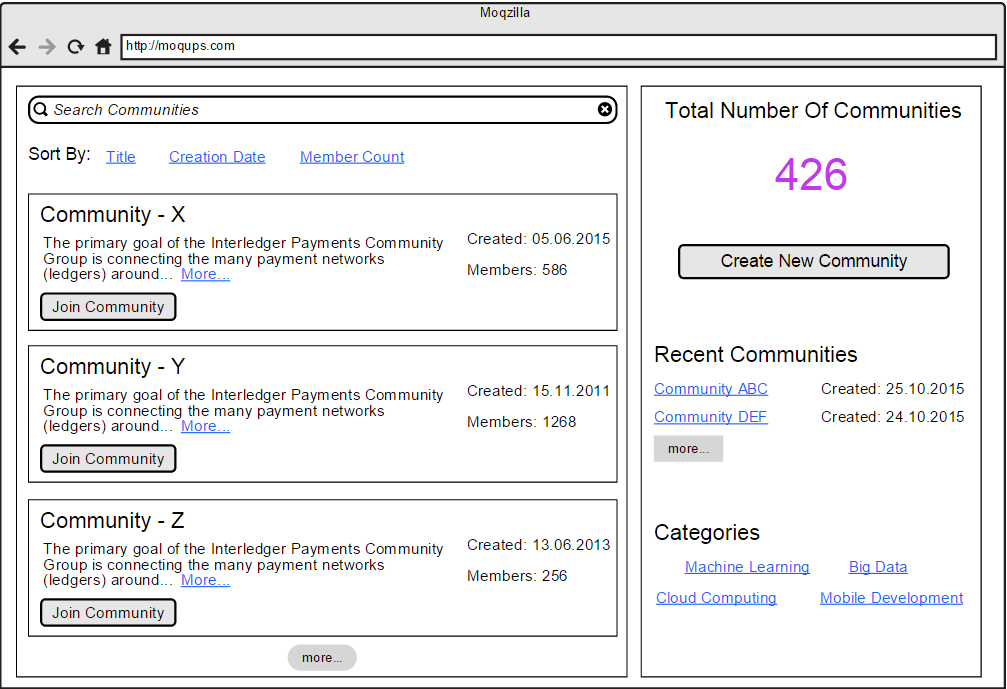
## Show Community Details

## When clicked on the Community name, community details should be able to be viewed in a separate page via “CommunityDetails.html” page which is shown in mockup below.



## Search And List Community

## All users should be able to search communities and list the result of any search via “CommunitySearch.html” page which is shown in mockup below.



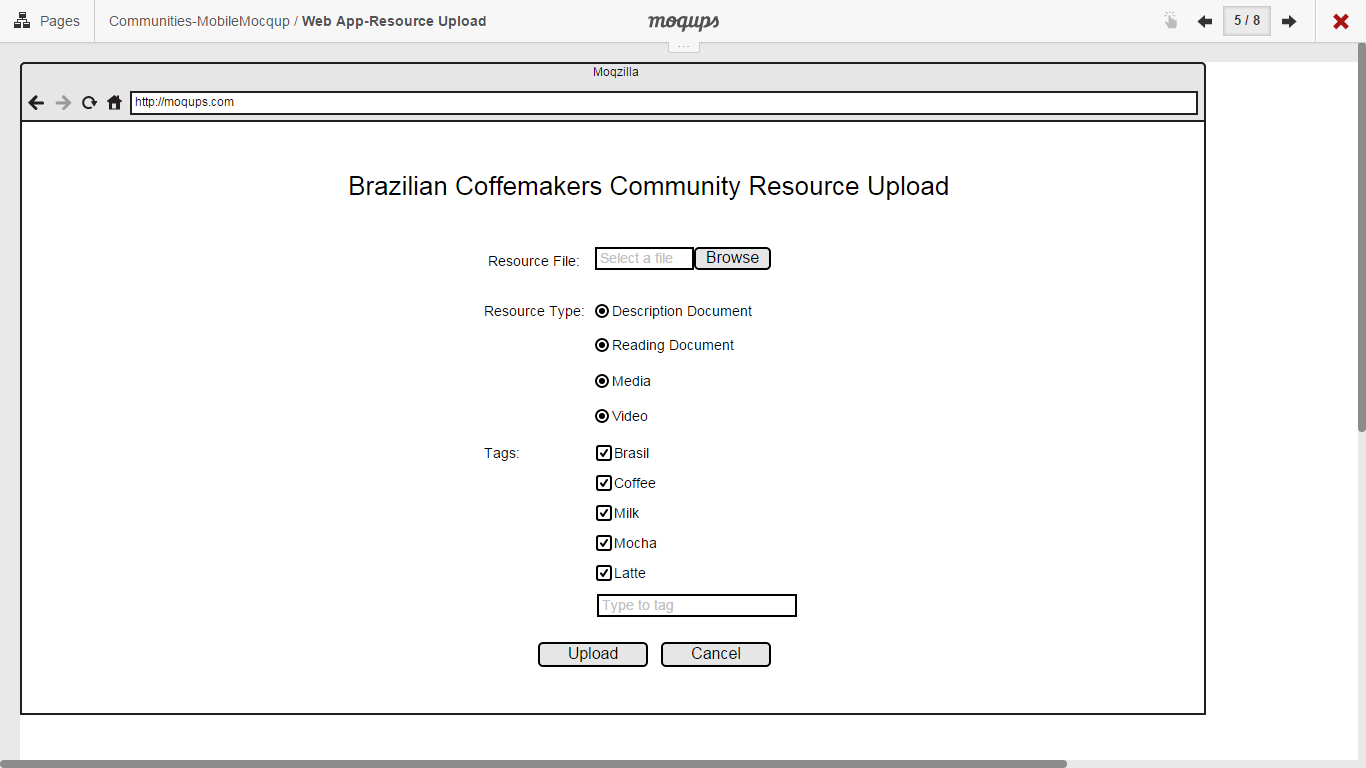
## List and Proceed Community Join Requests

## Users should be able to send request to join any community.Community admins should be able to see the requests coming from users for joining the community via “CommunityRequests.html” page which is shown in mockup below.



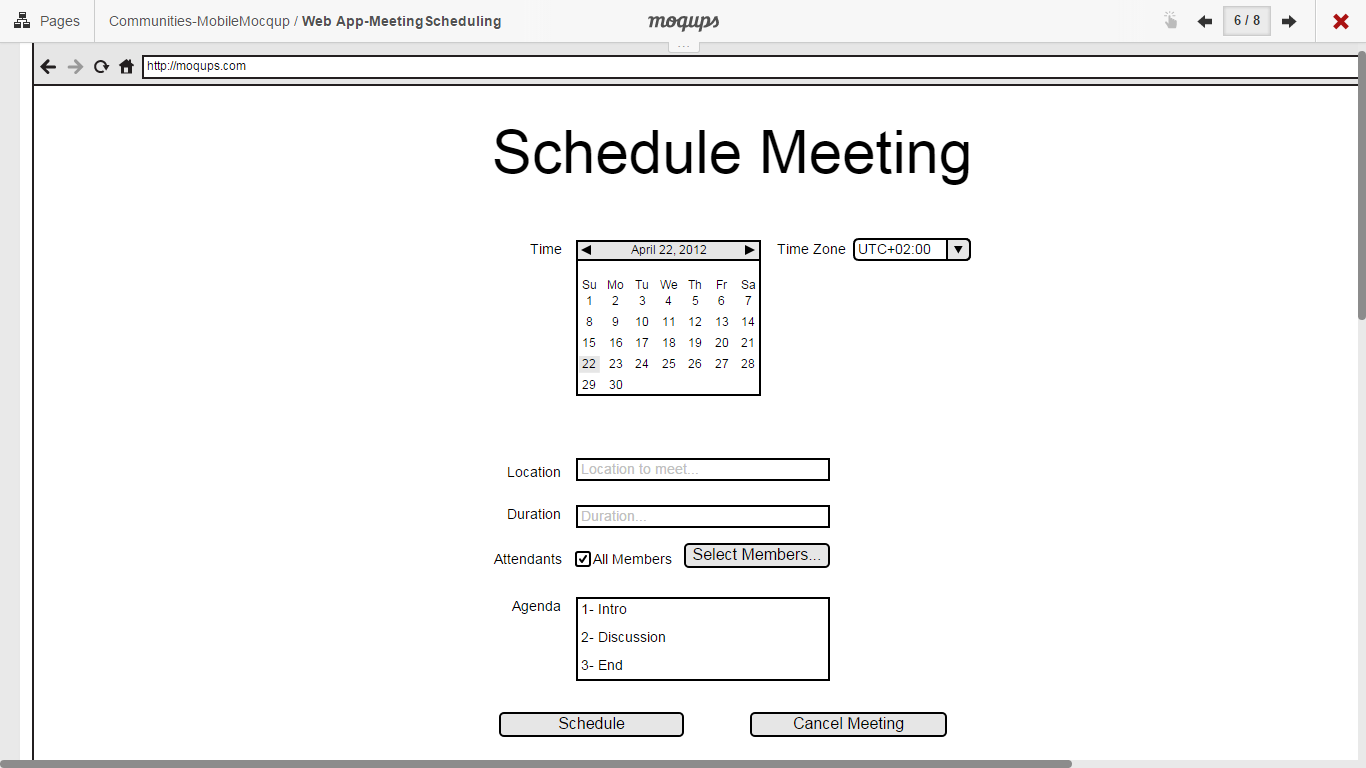
## Upload Resources to Community

## Community admins should be able to upload resources via “CommunityResources.html” page which is shown in mockup below.



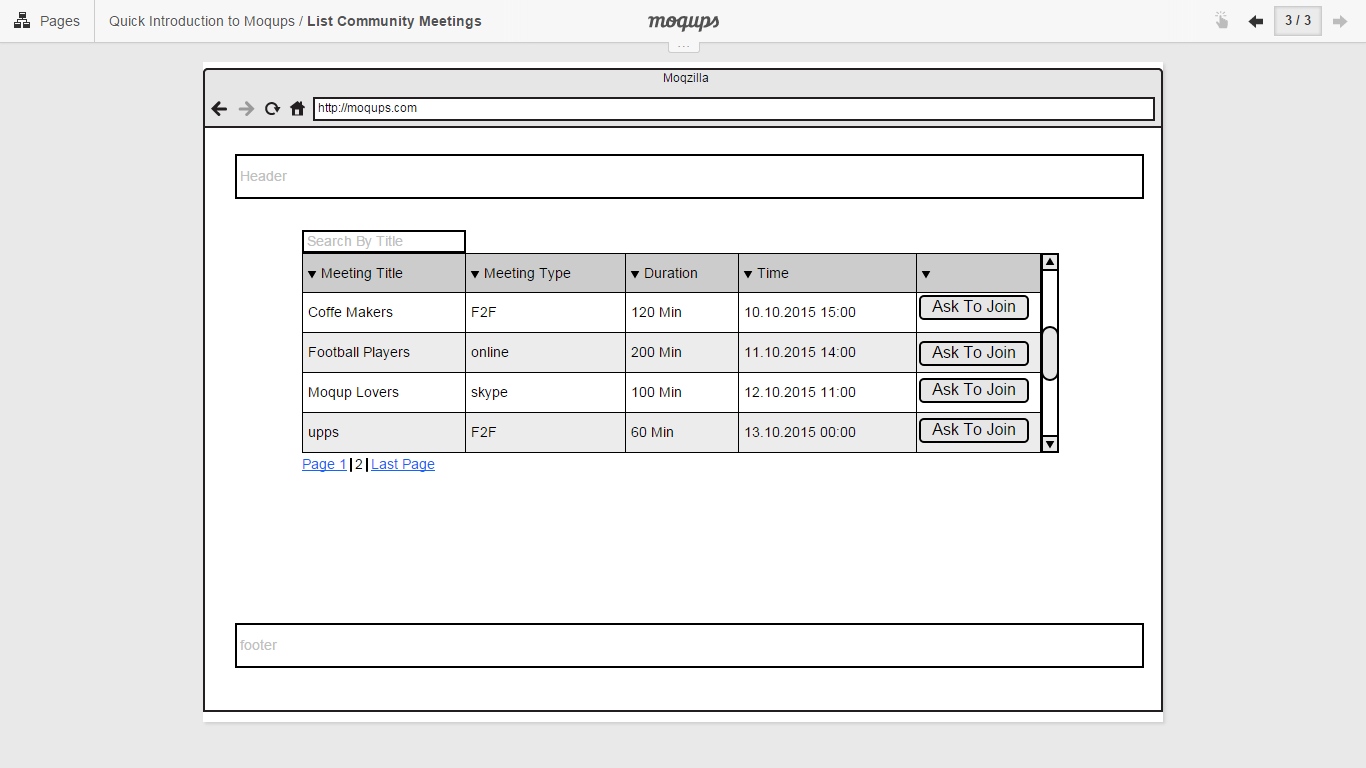
## Create Meeting

## Community members will be able to create meetings via “CreateMeeting.html” page which is shown in mockup below. Created Meeting will be updatable on the same page.



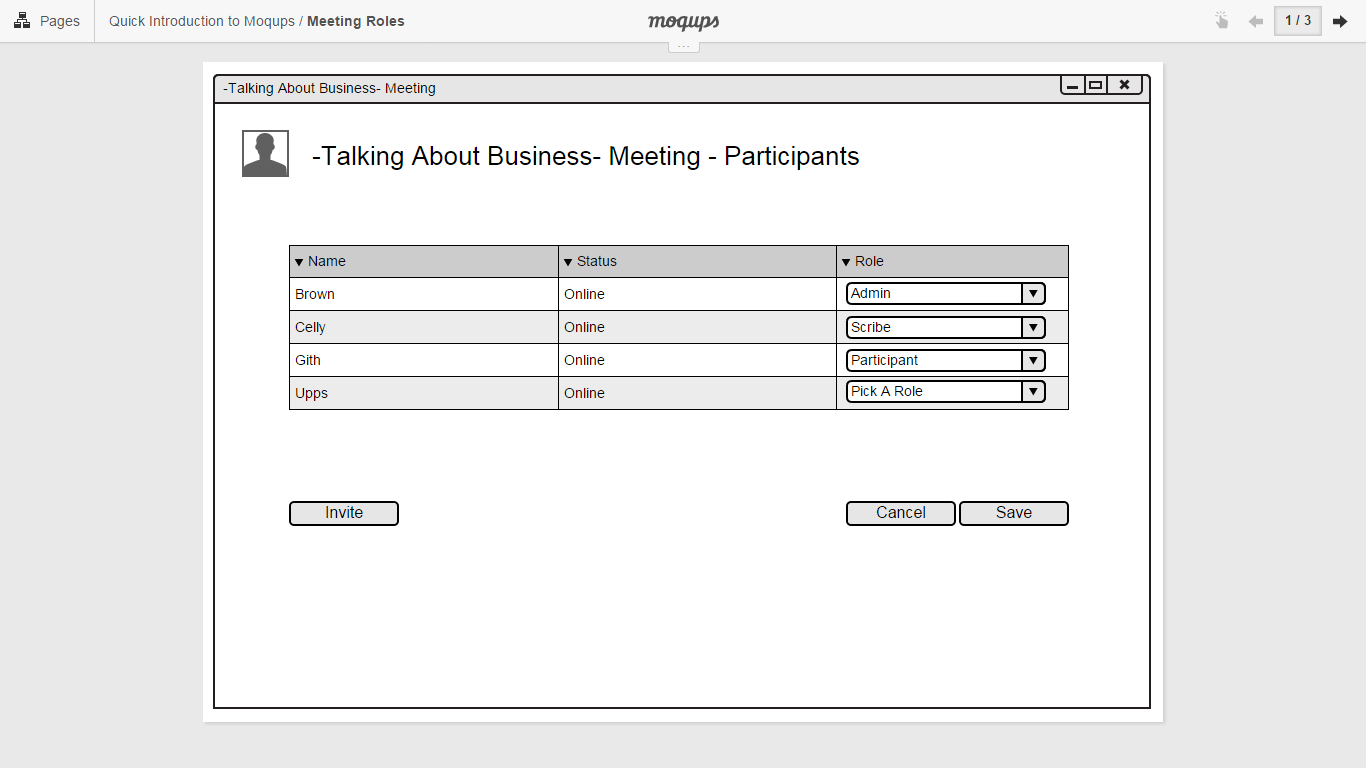
## List Community Meetings

## Community members should be able to see the list of meetings organized in that community via “CommunityMeeting.html” page which is shown in mockup below.



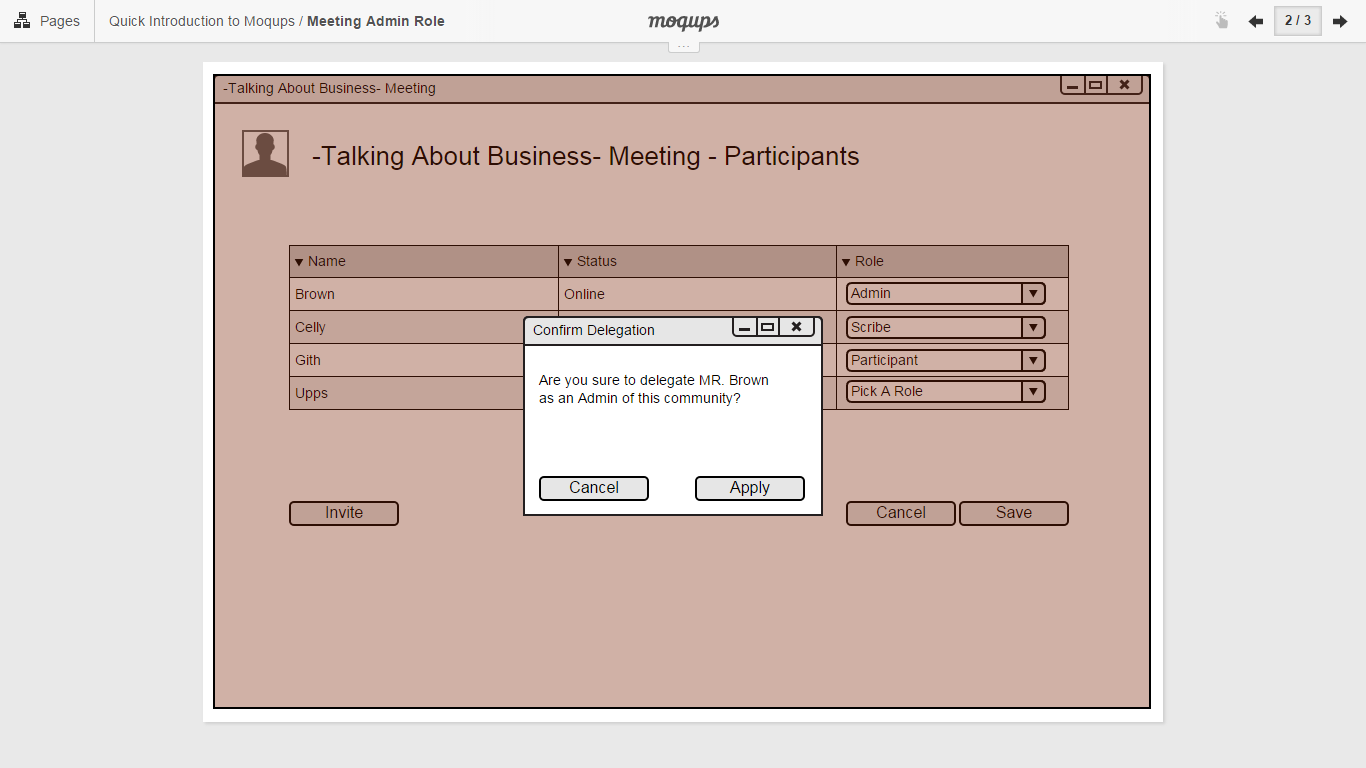
## Set Participants’ Role on Meeting

## The manager of the meeting should be able to determine the roles of the Meeting participants via “RoleMeeting.html” page which is shown in mockup below.



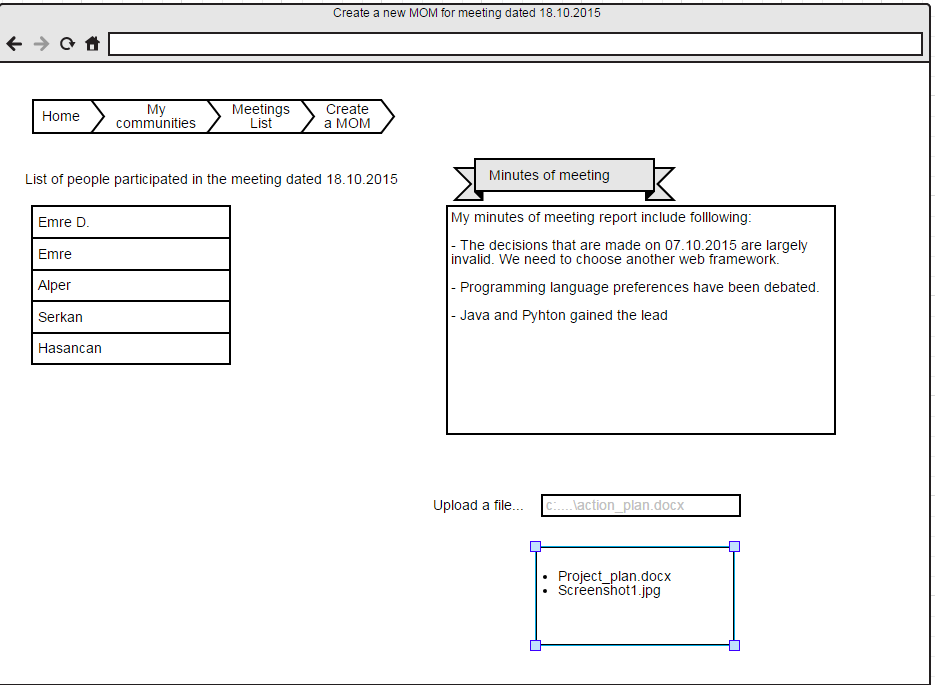
## Delegate Managers’ Role on Meeting

## Meeting manager will be able to delegate managerial role to other members of the community via “RoleMeeting.html” page which is shown in mockup below.



## Create MOM

## Note takers will be able to create and upload meeting decision documents via “CreateMOM.html” page which is shown in mockup below.



## Show Reports

## The system should be able to show required reports in a dashboard style page via “Reports.html” page.

## Create Post

## Community members should be able to upload media, document or write text for the posts section of the community via “postfiles.html” page.

## Comment Post

## Community members should be able to write comment for the posts via “postfiles.html” page.

## Comment Post

## The system should display the user-specific items on landing page after logged-in via “UserProfile.html” page.

## Backend Design

### Web Service API

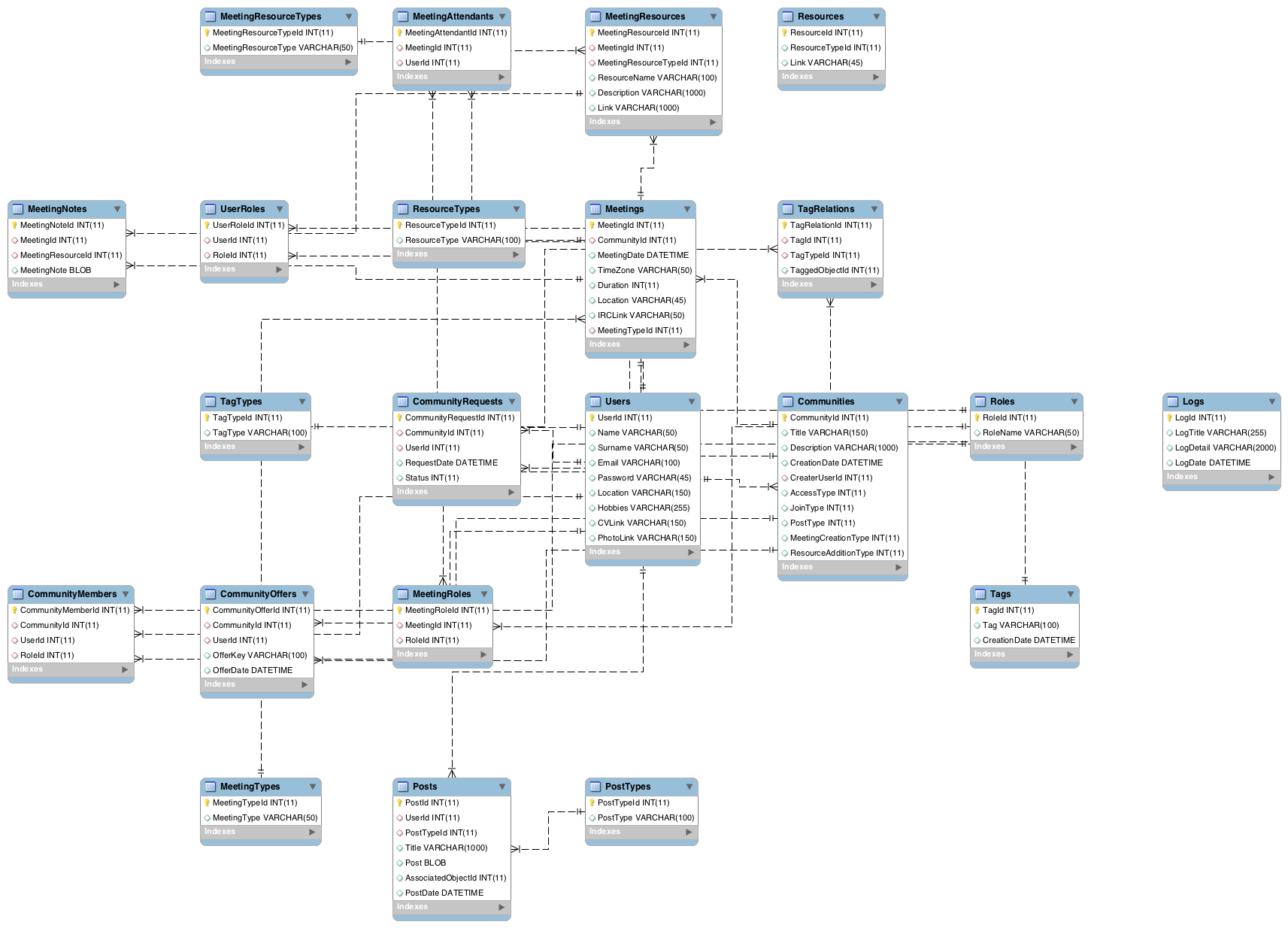
Backend side of the application will provide following web service api for frontend side.

* User Services
  + createUser
  + updateUser
  + searchUser
  + getUser
  + getUserSpecificItems
* Community Services
  + createCommunity
  + updateCommunity
  + getCommunity
  + searchCommunity
  + offerCommunity
  + joinCommunity
  + processJoinRequest
  + uploadResource
  + createPost
  + commentPost
* Meeting Services
  + createMeeting
  + updateMeeting
  + getMeeting
  + listCommunityMeetings
  + setMeetingParticipantRoles
  + delegateMeetingManagerRole
  + createMOM
* Report Services

## Authentication & Authorization

Spring-security framework will be used for authentication and authorization requirements.

# Database Design



# References

[1] UML2.0 Specifications, [http://www.uml.org/#UML2.0.](http://www.uml.org/#UML2.0)