Praeda&Co. formerly Boomerang

MuzikMekan

Initial Design Report

Eray Molla – e1448950
Gokhan Tuysuz – e1449172
Murat Ongan – e1449008
Cansin Yildiz – e1449271

Contents

1 Introduction

This report is intended to present initial design plan of the MuzikMekan Project which is conducted by Praeda&Co. In this report, general design architecture of the project will be enlightened and some implementation strategies will be explained.

1.1 Problem Definition

One of the biggest problems of online social networks is that it makes people more asocial. They discourage people to go outside and meet their friends. Therefore it is a good idea to combine the advantages of social networks with daily activities. Although there are some social networks that attempt to do this, most of them are not successful on mobile area.

As known, the musical taste of people is important to decide where to go for having fun. Moreover the style of the other people in a place is another important factor for this decision. Even though there are some web sites providing some services to inform people about the quality of an entertainment place in terms of musical style, none of them offers live services to follow which kind of songs are playing in a place and what kind of people are inside this place right now.

1.2 Motivation

"MuzikMekan" project aims to offer solutions for deficiencies stated at "Project Definition". Praeda&Co. is going to develop a social network which can be used on mobile devices. The main goal of this social network is to help people for deciding where to go for entertainment. Our system is going to provide people with live information about a place which makes our social network different than others. It will enable their users to meet new people that have similar tastes in terms of music and entertainment. Additionally the system is going to inform place owners about what kind of people is inside their place, so they can play songs according to visitors' tastes.

The system will provide users with capabilities like:

- Learning the musical history of a venue and following what kind of music is being played in a venue right now.
- Tracking what kind of people inside a venue.
- Getting suggestions for new places according to musical taste of the user.
- Synchronizing their profile with Facebook, Last.Fm and Twitter and enabling automatic status updates according to their locations.
- Getting more realistic information when selecting a venue.
- Encouraging people to be more social, unlike other online social networks.

And also the venue owners will be able to take advantage of:

- Learning what kind of people inside their venue and adjust their services according to this information.
- Presenting their venue to customers efficiently

2 Design Strategy and Development Methodology

Since Praeda&Co. team is not experienced in mobile application development, the group may make some mistake during development process and the details of the mobile application development will be learned by the team when developing the project. Therefore, to use a structural development model like “Waterfall” seems unreasonable because risk of making an important design mistake at the very beginning of the project is really high by an inexperienced team. For this reason, the team needs a more flexible development methodology like “Spiral”. It lets the development team learning by making mistakes and even an inexperienced team can form a solid structure after enough number of iterations.

Since we do not have enough time to make adequate number of iterations, taking advantage of “Agile Methodology” can also be really beneficial and can speed up the development process. Because, designing and implementing a non-working prototype of the system sometimes requires lots of effort and has the risk of unsuccessfulness. Therefore instead of complex designs, producing fully tested and working parts of the project can be more rapid and can produce more reliable outcome.

To sum up, Praeda&Co. will mainly follow ‘Spiral methodology’ during development process. However, for the parts that needs special attention ‘Agile methodology’ can be plausible.

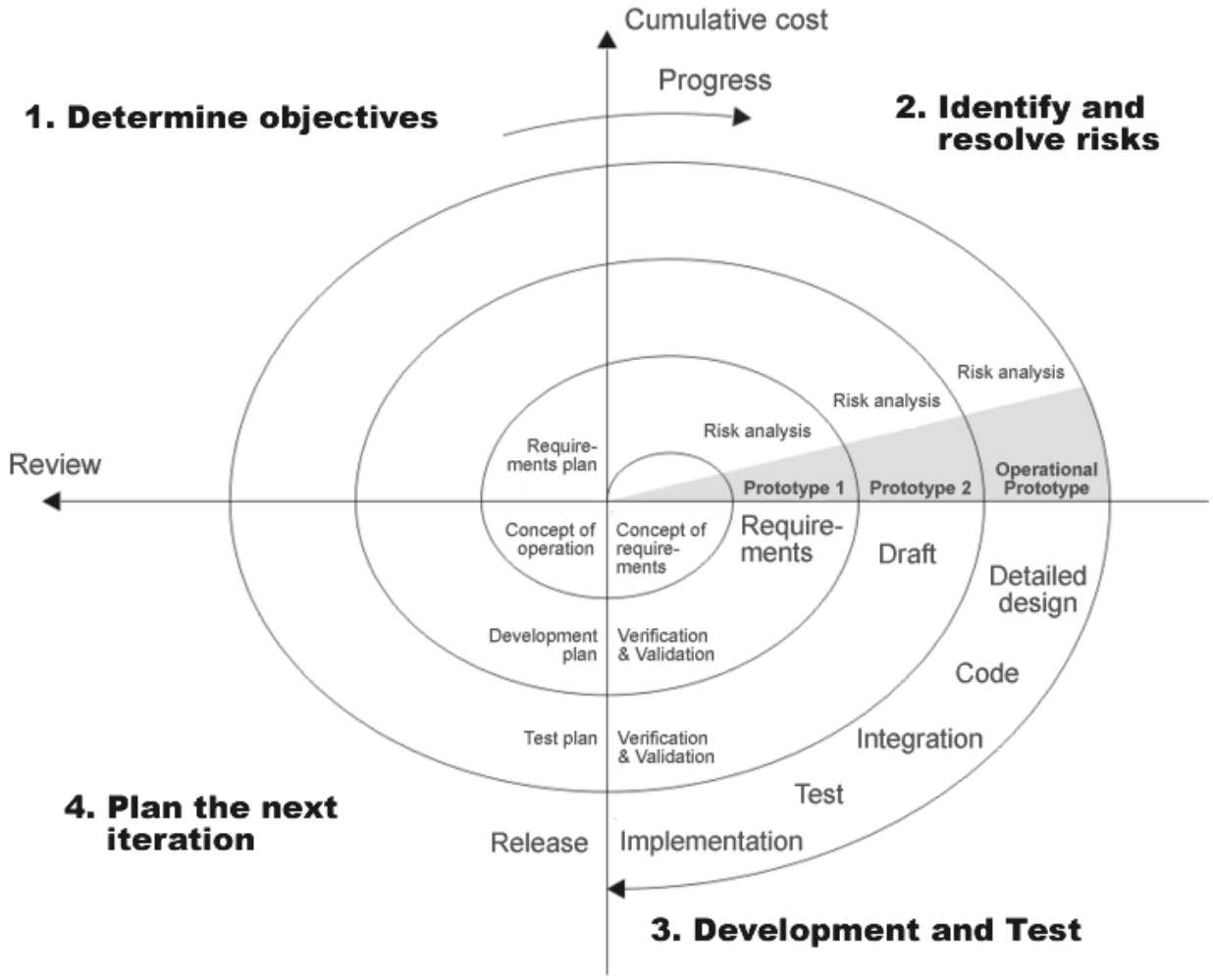


Figure 1 - Spiral model - http://en.wikipedia.org/wiki/File:Spiral_model_(Boehm,_1988).png

3 Data Design

3.1 ER Diagram

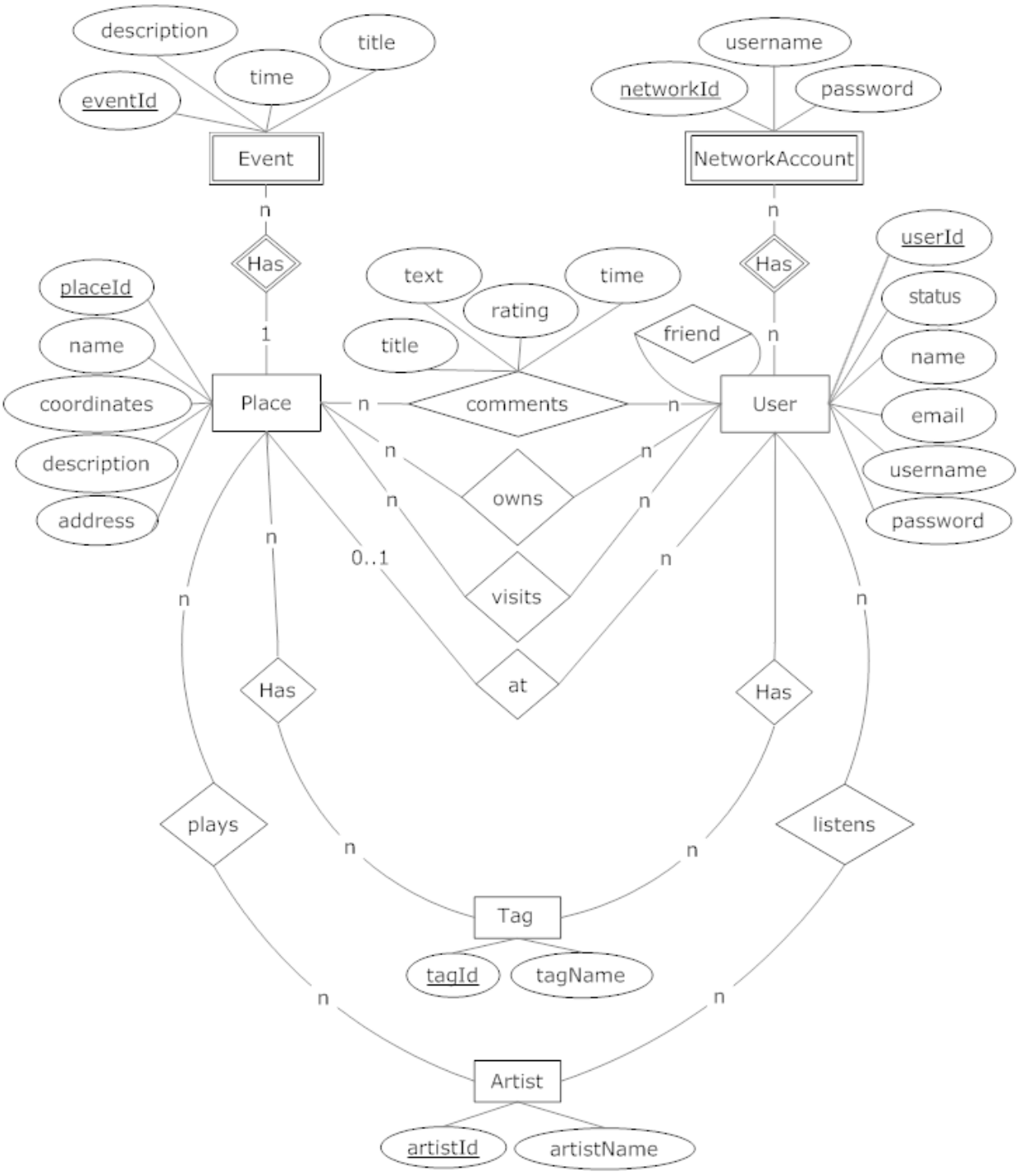


Figure 2 - ER Diagram of Database Design

3.2 Database Schemas

3.2.1 Place Table

Field	Type	Null	Foreign Key	References
placeId (P.K)	Integer	No	No	-
name	Varchar(30)	No	No	-
description	Varchar(500)	Yes	No	-
address	Varchar(200)	No	No	-
latitude	Float	No	No	-
longitude	Float	No	No	-
altitude	Float	No	No	-

This table holds information related to places that were registered to our system. It holds regular information like address and name of a place. Moreover it holds geographic coordinates that represents location of the place. Place owner will be able to choose the coordinates via Google Maps.

3.2.2 Event Table

Field	Type	Null	Foreign Key	References
eventId (P.K)	Integer	No	No	-
placeId	Integer	No	Yes	Place
title	Varchar(100)	No	No	-
description	Varchar(500)	Yes	No	-
time	Datetime	No	No	-

Event table is a table including information about special activities like concerts or parties. This table associates an event's id with a venue's id. It makes a connection between places and events.

3.2.3 User Table

Field	Type	Null	Foreign Key	References
userId (P.K)	Integer	No	No	-
name	Varchar(50)	No	No	-
email	Varchar(30)	No	No	-
username	Varchar(20)	No	No	-
password	Varchar(20)	No	No	-
status	Varchar(100)	Yes	No	-

This is a table that holds MuzikMekan's registered users' membership data.

3.2.4 Artist Table

Field	Type	Null	Foreign Key	References
artistId (P.K)	Integer	No	No	-
artistName	Varchar(50)	No	No	-

This table associates an artist with a unique id number. In order to avoid storage of same artist names that are entered by different users, it is more compact to give unique id numbers to artist names and to use this id if an artist name was entered before instead of storing same names repeatedly.

We will store the artist names in our database instead of Last.fm because we will enable members without a Last.fm account to enter artist names manually. However we will check the validation of the entry by using Last.fm API's "search" functionality because we will use Last.fm to get detailed information about this entry later.

3.2.5 Genre Table

Field	Type	Null	Foreign Key	References
genreId (P.K)	Integer	No	No	-
genreName	Varchar(50)	No	No	-

MuzikMekan will use musical taste of the members to identify their venue preferences as we indicated. Hence to hold genre information in a separate table is necessary. This table associates a place's or a user's genre with a unique id. Because different members can enter the same genres

repeatedly, we thought to give an id to genre names and to use this id instead of long genre names will save storage space. We will store genres in our database because of the same reason with artist table.

3.2.6 NetworkAccount Table

Field	Type	Null	Foreign Key	References
networkId(P.K)	Integer	No	No	-
userId (P.K)	Integer	No	Yes	User
password	Varchar(20)	No	No	-
username	Varchar(20)	No	No	-

Since MuzikMekan will be able to share information with other social networks accounts of the users, this table holds log-in data of the social network accounts belonging to them. Since a user cannot have more than one account on same social network, networkId and userId is primary key together.

3.2.7 Tag Table

Field	Type	Null	Foreign Key	References
tagId (P.K)	Integer	No	No	-
tagName	Varchar(50)	No	No	-

Tags are important data that identify people's and venue's characteristics so our system will enable its members to use tags. This table assigns a unique id to each tag. Giving ids to tags will save storage.

3.2.8 Comment Table

Field	Type	Null	Foreign Key	References
userId	Integer	No	Yes	User
placeId	Integer	No	Yes	Place
time	Datetime	No	No	-
title	Varchar(100)	No	No	-
text	Varchar(500)	No	No	-
rating	Float	Yes	No	-

Since MuzikMekan will enable the users to enter their personal comments about a place, this table will hold the comments of the users. Besides their comments they will be able to assign points to venues as rating.

3.2.9 Owner Table

Field	Type	Null	Foreign Key	References
userId (P.K)	Integer	No	Yes	User
placeId (P.K)	Integer	No	Yes	Place

This table specifies the owners of the places. It will be used for queries to identify owners of a place or places of an owner.

3.2.10 Visits Table

Field	Type	Null	Foreign Key	References
userId	Integer	No	Yes	User
placeId	Integer	No	Yes	Place
time	Datetime	No	No	-

This table holds data about when a user visits a place. Since our system will show location information of a user to his/her friends, we will need to record this information.

3.2.11 At Table

Field	Type	Null	Foreign Key	References
userId (P.K)	Integer	No	Yes	User
placeId	Integer	No	Yes	Place

This table includes the instant location information of the users. It associates a user id and a place id according to which venue a user is inside currently. The difference between visits table and at table is that while visits table holds history of the members, at table holds only current temporary location information.

3.2.12 User Tags Table

Field	Type	Null	Foreign Key	References
tagId (P.K)	Integer	No	Yes	Tag
userId (P.K)	Integer	No	Yes	User

It associates a tag id and a user id hence the system knows which tags a user have. Both tagId and userId is primary key in order to avoid giving same tags to a user.

3.2.13 Place Tags Table

Field	Type	Null	Foreign Key	References
tagId (P.K)	Integer	No	No	Tag
placeId (P.K)	Integer	No	Yes	Place

According to tags the venues have this table associates tag ids and place ids. Because of the same reason with User Tags table, tagId and placeId is primary key together.

3.2.14 Place Genres Table

Field	Type	Null	Foreign Key	References
placeId (P.K)	Integer	No	Yes	Place
genreId (P.K)	Integer	No	Yes	Genre

This table holds information about genre of a place. It associates a place id and a genre id if this venue has characteristics compatible with this genre.

3.2.15 User Genres Table

Field	Type	Null	Foreign Key	References
userId (P.K)	Integer	No	Yes	User
genreId (P.K)	Integer	No	Yes	Genre

It associates a user id and a genre id according to taste of a user.

3.2.16 Place Artists Table

Field	Type	Null	Foreign Key	References
placeId (P.K)	Integer	No	Yes	Place
artistId (P.K)	Integer	No	Yes	Artist

This table associates a place id and an artist id if this place usually plays the songs of this artist.

3.2.17 User Artists Table

Field	Type	Null	Foreign Key	References
userId (P.K)	Integer	No	Yes	User
artistId (P.K)	Integer	No	Yes	Artist

It holds information about the favorite artists of the members and associates a user id and an artist id according to it.

3.2.18 Friend Table

Field	Type	Null	Foreign Key	References
user1Id (P.K)	Integer	No	Yes	User
user2Id (P.K)	Integer	No	Yes	User

This table holds friendship status of the users. An entry means users with id “user1Id” and “user2Id” are friends.

4 Architectural Design

4.1 Design Overview

As it is explained throughout the report, MuzikMekan system needs a complex architecture because lots of modules will work cooperatively. Furthermore, together with the mobile and internet technology also social networks and user interests are changing rapidly. Therefore, our architecture should be easily modifiable according to these changes and it should allow developers for developing new modules. Moreover, it should make this complex system's development phase less difficult with good separation of layers.

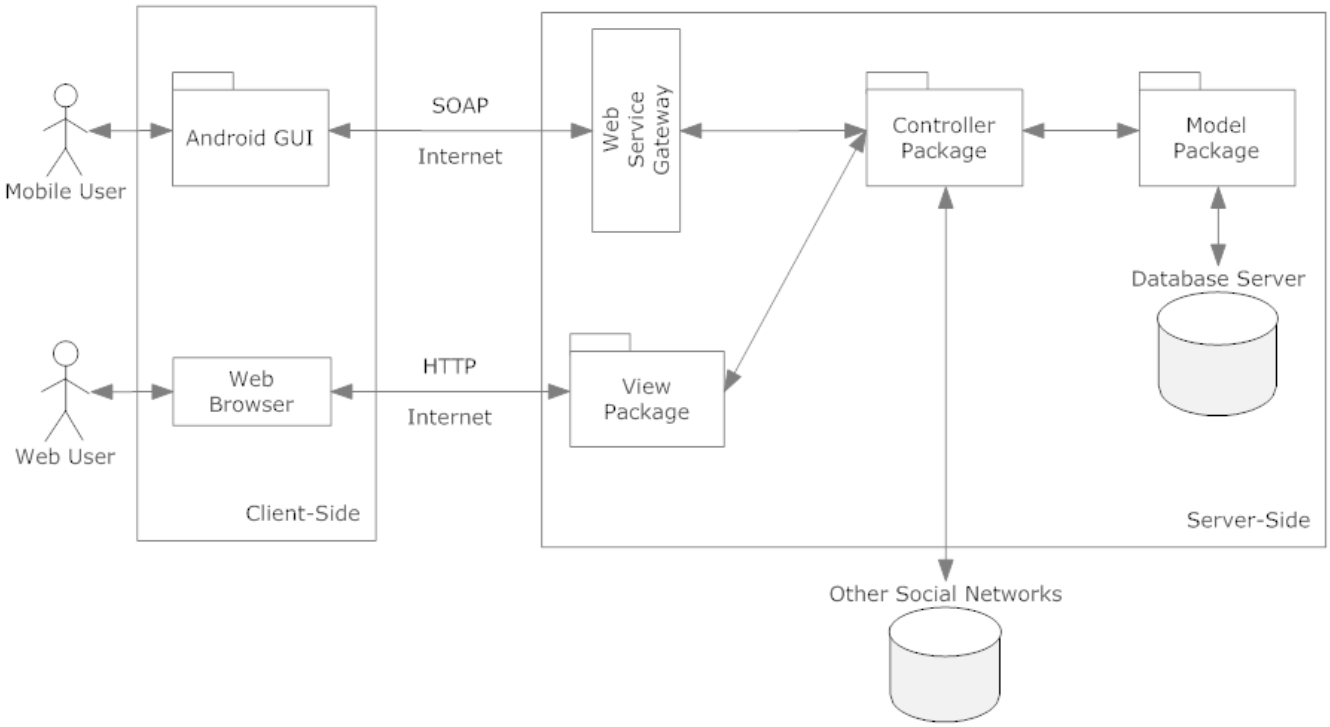


Figure 3 - Design Overview

The team decided to pursue a Model View Controller (MVC) design pattern. It enables separating the view and the business logic entirely so in the future to support other client devices will not cause an extra effort except for writing a new user interface. For now, we will develop user interfaces for Android

and for the web in order to show the capabilities of the system but in the future giving support for another device will be nothing but implementing a simple user interface.

As shown in Figure 3, when the user interacts with the web browser, the data will be transported to view classes via HTTP. After data is taken by the view classes they will do nothing but presenting the data to controller classes in a convenient type. For now, we intend to use Java Server Faces (JSF)[1] technology for the view tier of our MVC implementation. Because it automatically adjusts view code format according to device interacting with it, even the client is a mobile one, but the most important reason is it enables the implementation of the view classes with minimum interaction with controller classes. It is very useful for MVC design pattern.

If the client has a mobile device like an Android one, its view part will be handled by a client application rather than web browser. But please note that since our "web clients" will be handled by JSF if the client uses the web browser of its mobile device it can also use the web interface without a problem. When the client submits data using mobile application, it will interact with the main system using web-services via SOAP messages. When a message reaches to web-services, it will just convey the data to controller classes. Since Apache Axis generates very clear client and server code, we'll implement our web services with Apache Axis technology.

After data reached to controller package it will be interpreted by the relevant class. According to content of the data, controller class will either call the relevant API function of the other social networks like Last.Fm and Facebook or interpret the data with its own methods. If the operation requires a database connection or an alteration in the model part it will notify the model package and wait for the response. If the operation does not require editing of permanent data, as an operation that only calls another social network API function and show the results of the operation to the client, the controller classes will obtain the result of the operation and convey it to the view classes for showing to the end user without interacting with the model part. We will use Spring[2] Framework because it perfectly integrates with JSF. We can make JSF using Spring beans instead of its own beans by only changing one line of configuration file. And it has lots of beneficial packages like security package and its good Hibernate[3] support is make it very convenient for Create - Read - Update - Delete (CRUD) applications like our system.

Model classes form the skeleton of our system. They are responsible for holding permanent data of a user session and database operations. As it is mentioned, we'll use an object relational mapping tool which is very handy for CRUD applications named Hibernate.

[1] http://java.sun.com/javaee/javaserverfaces/
[2] http://www.springframework.org/
[3] http://www.hibernate.org/

4.2 Data Flow Diagrams

The data flow within the system and the external interfaces are illustrated in the data flow diagrams below. As it can be seen from the diagrams, the application is mainly interacts with user as a mobile application. The system also gets location information from mobile devices. This data can be gathered by GPS or other location services provided by mobile phone operators. The Android API will be used to get this information from mobile device. Other system services like, Internet communication and map services are also provided by Android API.

The system also communicates with other social network applications with their web services. This communication will be handled by the server part of MuzikMekan system using REST and SOA interfaces of Last.fm, Facebook, Twitter and other related social networks.

4.2.1 Level 0

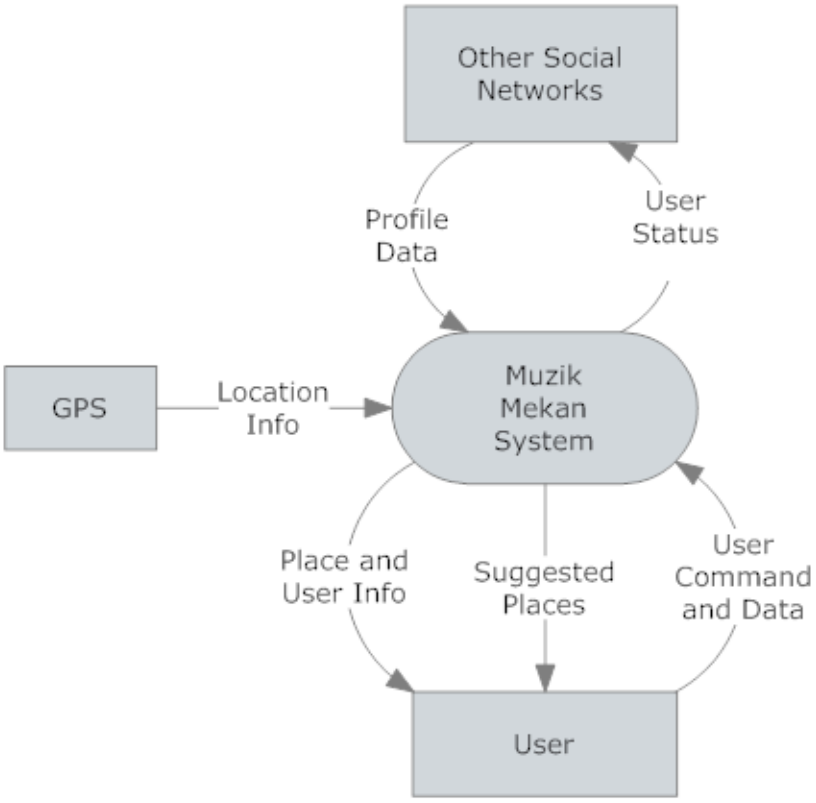


Figure 4 - Context Level DFD

4.2.2 Level 1: MuzikMekan System

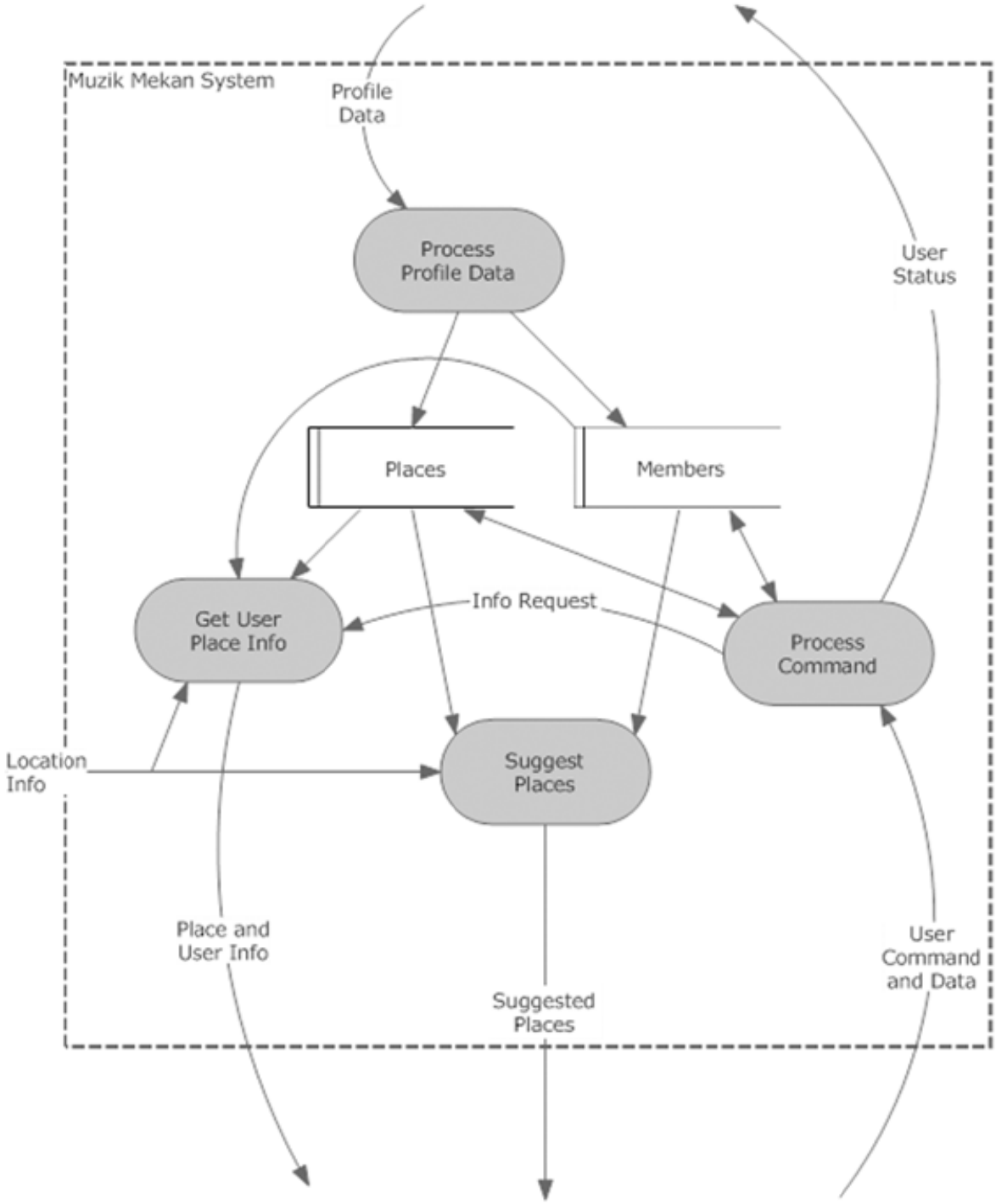


Figure 5 - MuzikMekan System Level 1 DFD

4.2.3 Level 2: Process Command

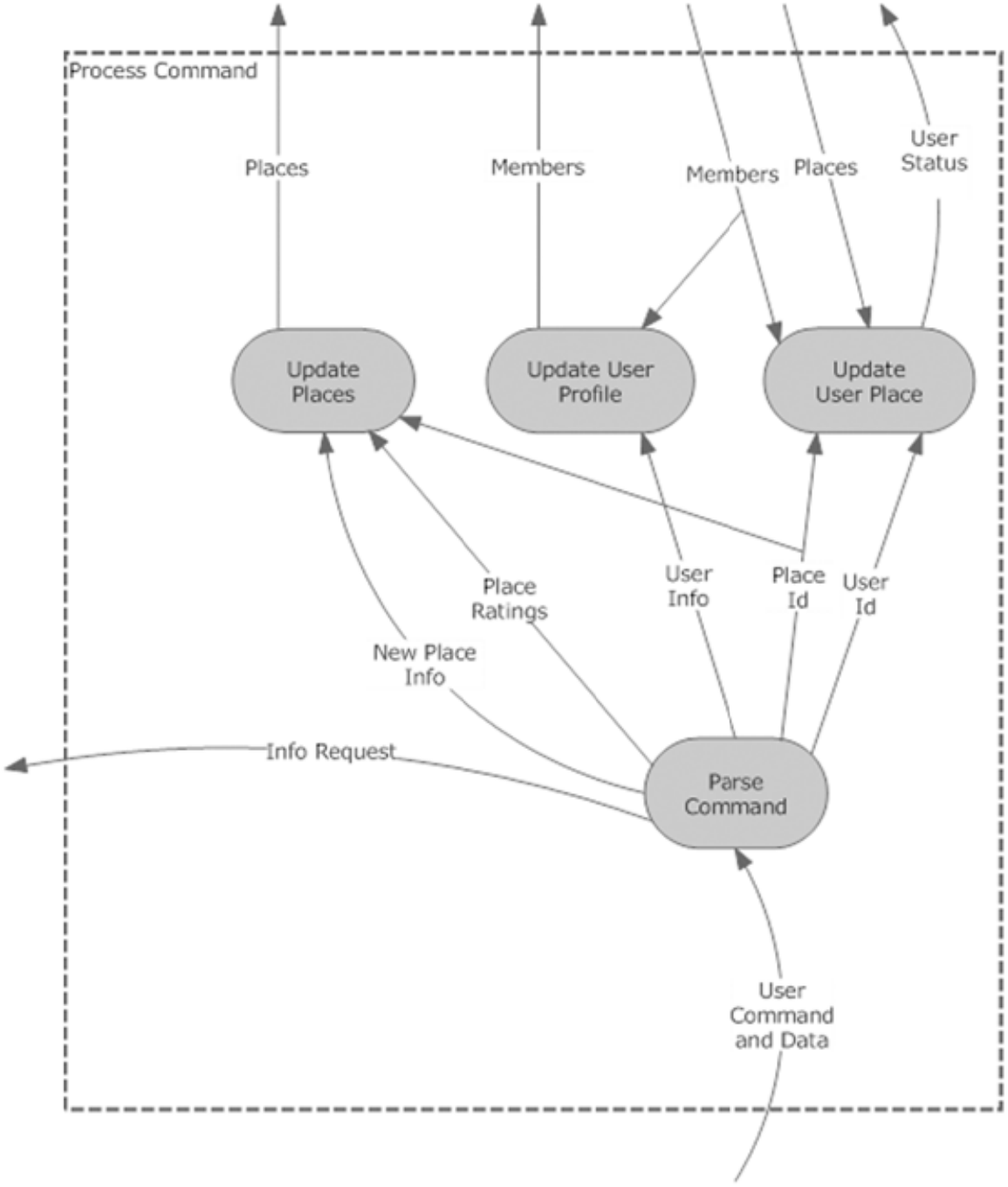


Figure 6 - Process Command Level 2 DFD

4.2.4 Level 2: Get User Place Info

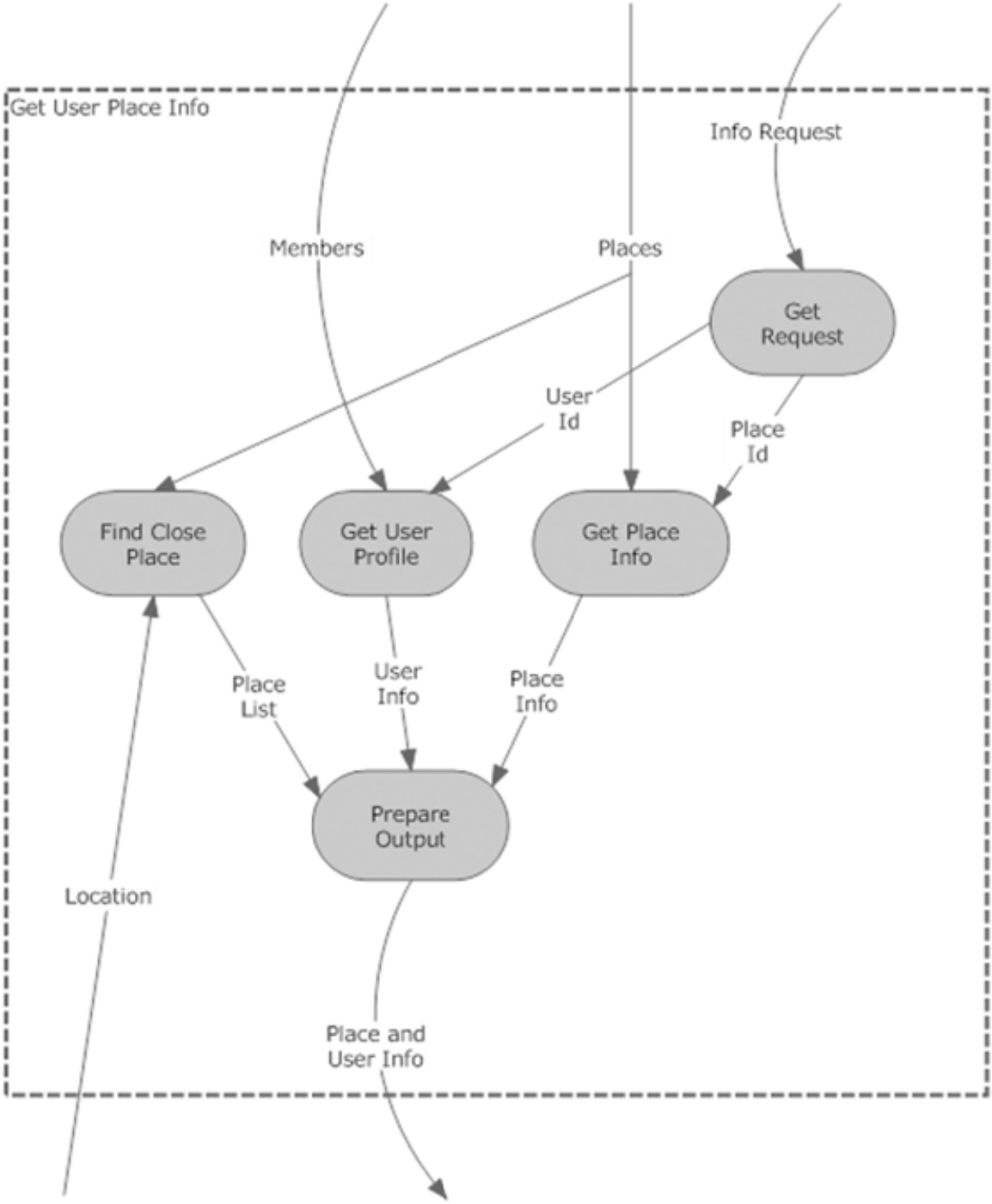


Figure 7 - Get User Place Info Level 2 DFD

4.3 Class Diagrams

The class hierarchy of the MuzikMekan application is illustrated as class diagrams in each package. The detailed explanations about these classes and their functions will be specified in the detailed design report.

4.3.1 View Package

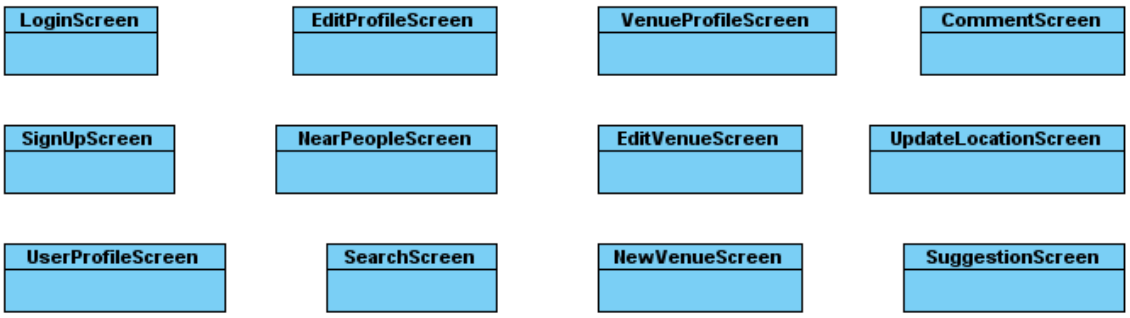


Figure 8 - View Package

View package holds the classes which implement user interfaces. Each screen class processes user request and calls related controller functions with appropriate arguments. These classes also renders user interface using outputs retrieved from controller functions.

4.3.2 Controller Package

The classes which handle the business logic of application reside in the controller package. Controller functions have well-defined API's so that, both view classes and web service requests can easily use them. The controller package is separated into three sub packages according to the usage areas.

4.3.2.A User Sub-Package

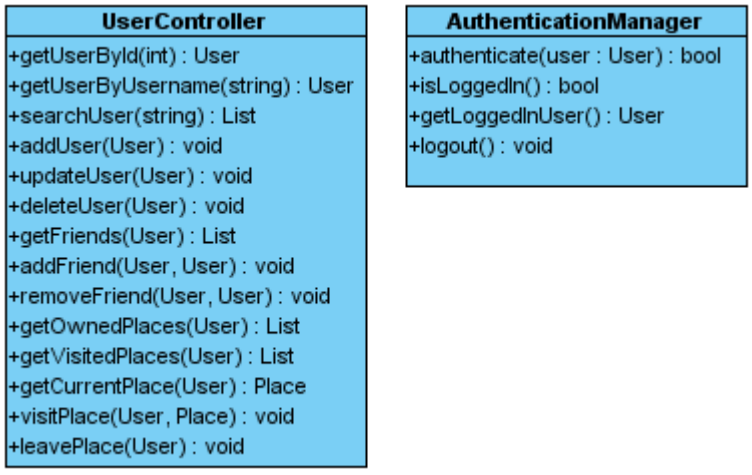


Figure 9 - User Sub-Package

This package contains two classes which are related to user operations, as its name implies. UserController class is used to get information about a specific user and applies the necessary changes required by user actions. AuthenticationManager class is responsible for user authentication issues like login and logout. The user package is in a close relationship with User class in model package. It uses Place class as well.

4.3.2.B Place Sub-Package

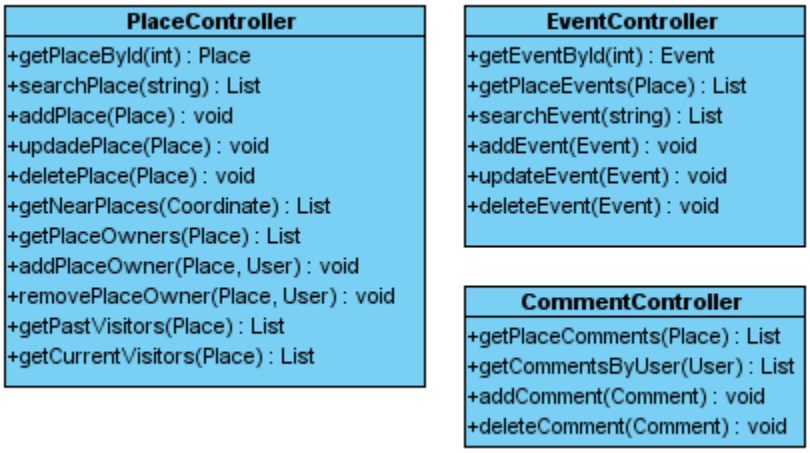


Figure 10 - Place Sub-Package

Place sub-package controls place-based events like venue locations, place comments and events. PlaceController class is used primarily for operations on the Place model class. EventController and CommentController classes on the other hand, handle Event and Comment entities. They simply contains add, modify and delete functions and some other methods for handling user-place relations and searching.

4.3.2.C Suggestion Sub-Package

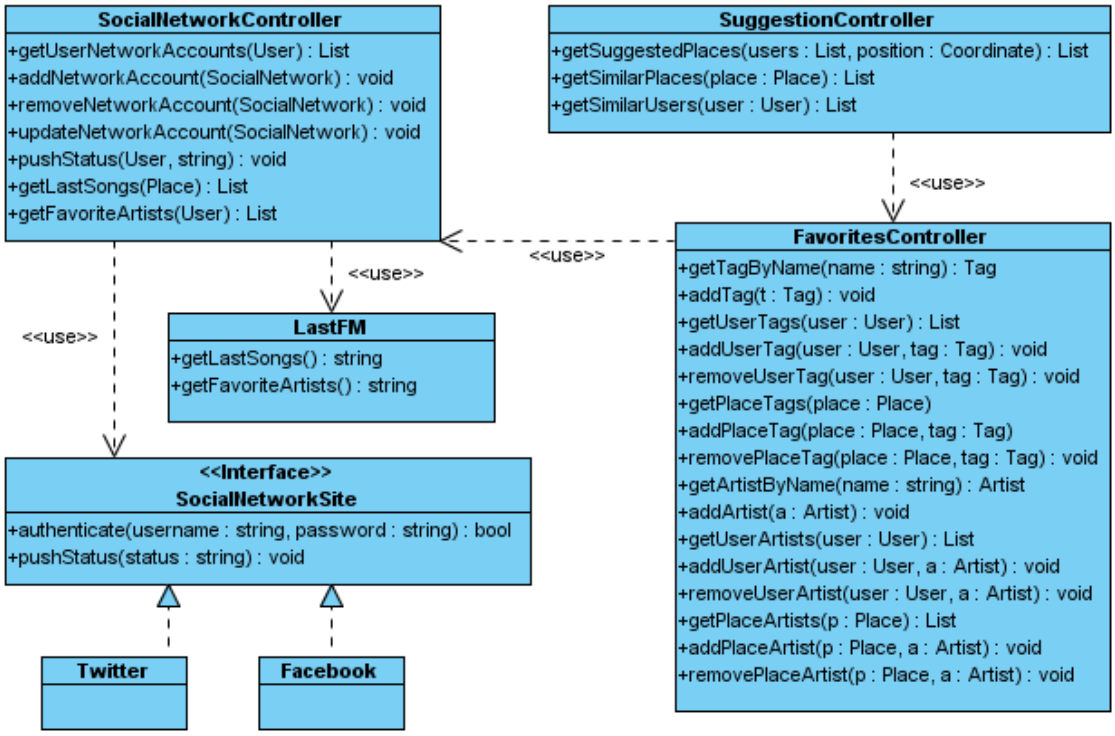


Figure 11 - Suggestion Sub-Package

This package includes controller classes for tags, artists and social networks. It also manages venue suggestion and other features which require intelligent computations. FavoriteController class in this package is used to add, remove and get favorite artists and tags for users and places. SocialNetworkController class, on the other hand, is used to add and remove social network accounts for users. It also satisfies synchronization with users other social networking accounts. It simply uses a generic interface which includes authentication and status push features. New social networking sites will be appended into the project by easily implementing this interface with new classes. Last.fm site is also used by Last.Fm class in the package. This class is used to retrieve data about music taste of users and recently played tracks of places from Last.fm. Last.Fm class does not implement the general SocialNetworkSite interface because Last.fm API does not require a user to be authenticated to get its data. Moreover, the API does not have a user status feature like Facebook and Twitter.

4.3.3 Model Package

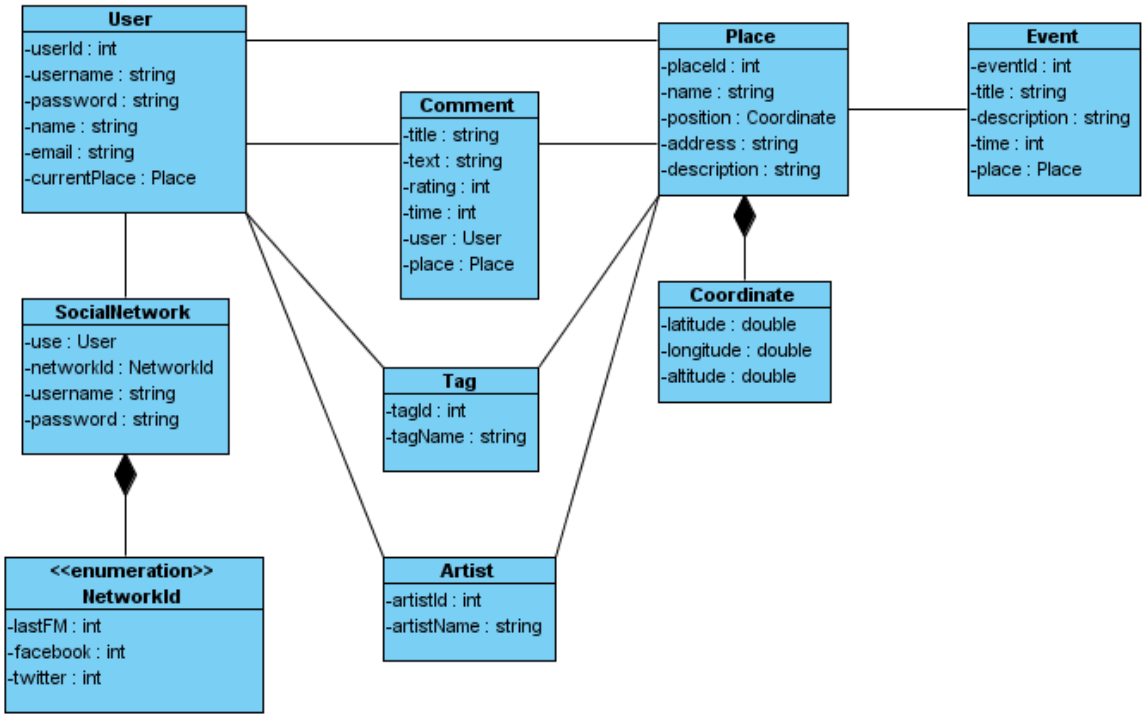


Figure 12 - Model Package

This package simply includes data model entities corresponding to database tables of the project. Refer the explanations of database tables in order to get details. Note that, those tables, which are only used to express relations, do not have class equivalents because they are not actual entities.

4.4 Activity Diagrams

The most important and most complex features of MuzikMekan system are explained as activity diagrams below. Detailed information and other features will be appended into document at 'Detailed Design Phase'.

4.4.1 Login Diagram

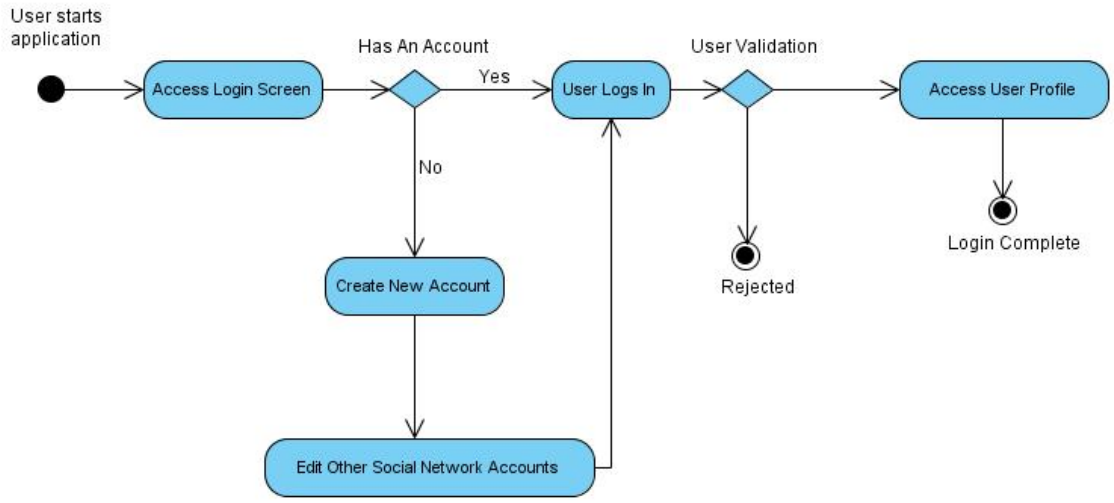


Figure 13 - Login Diagram

4.4.2 Suggest Venue Diagram

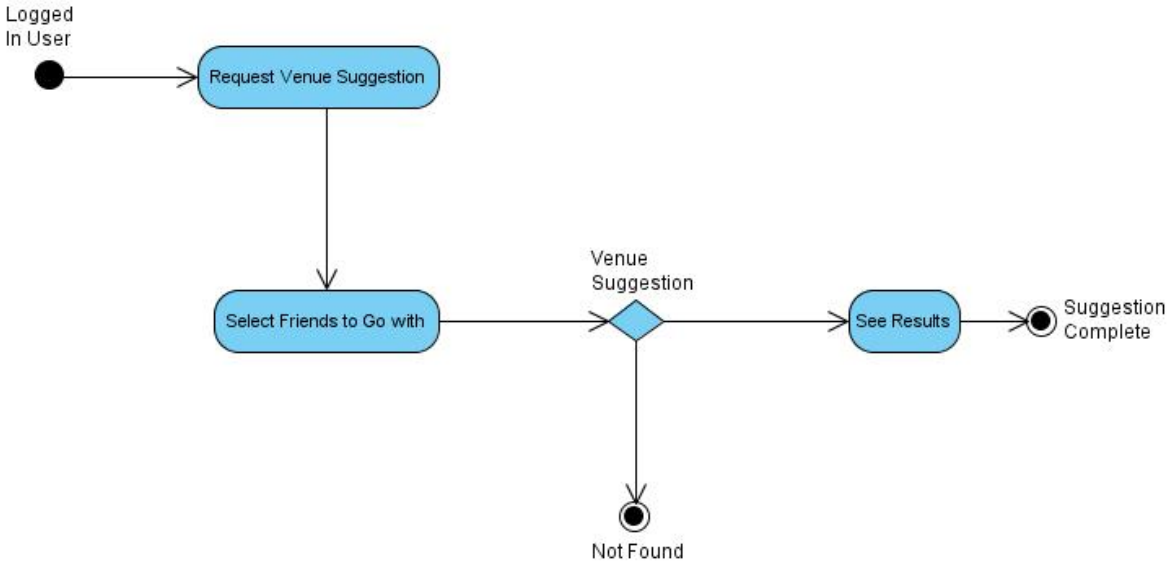


Figure 14 - Suggest Venue Diagram

4.4.3 Update Location Diagram

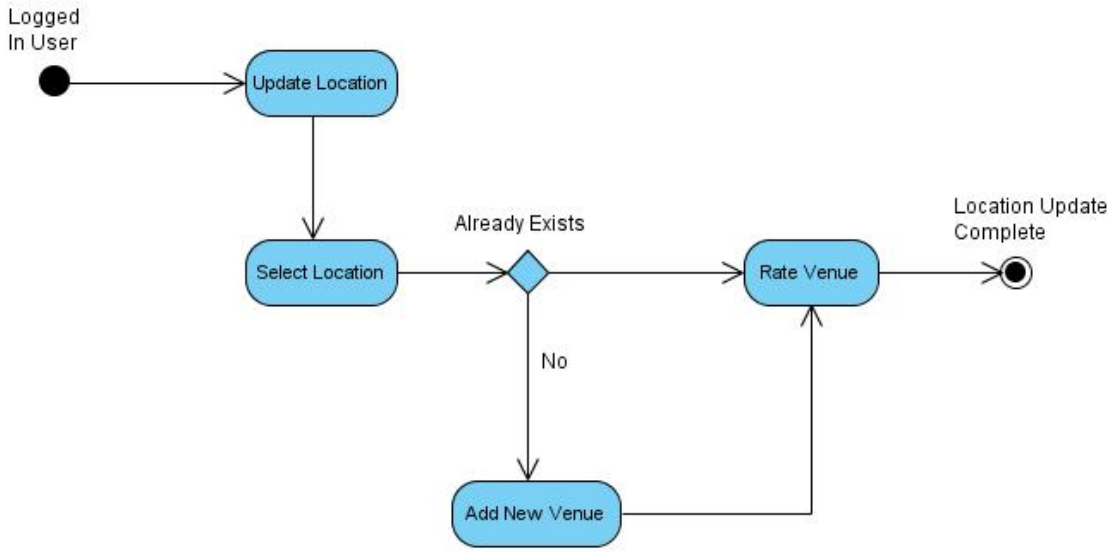


Figure 15 - Update Location Diagram

4.5 Sequence Diagrams

In this part of the report, the logic and flow of the operations will be explained using the class methods. Since we already showed the basics of the operation flows at Activity Diagrams, we planned to append detailed Sequence Diagrams at 'Detailed Design Phase'.

4.5.1 Login Diagram

4.5.2 Suggest Venue Diagram

4.5.3 Update Location Diagram

4.5.4 View User/Place Profile Diagram

4.5.5 Edit User/Place Profile Diagram

4.5.6 People Near You Diagram

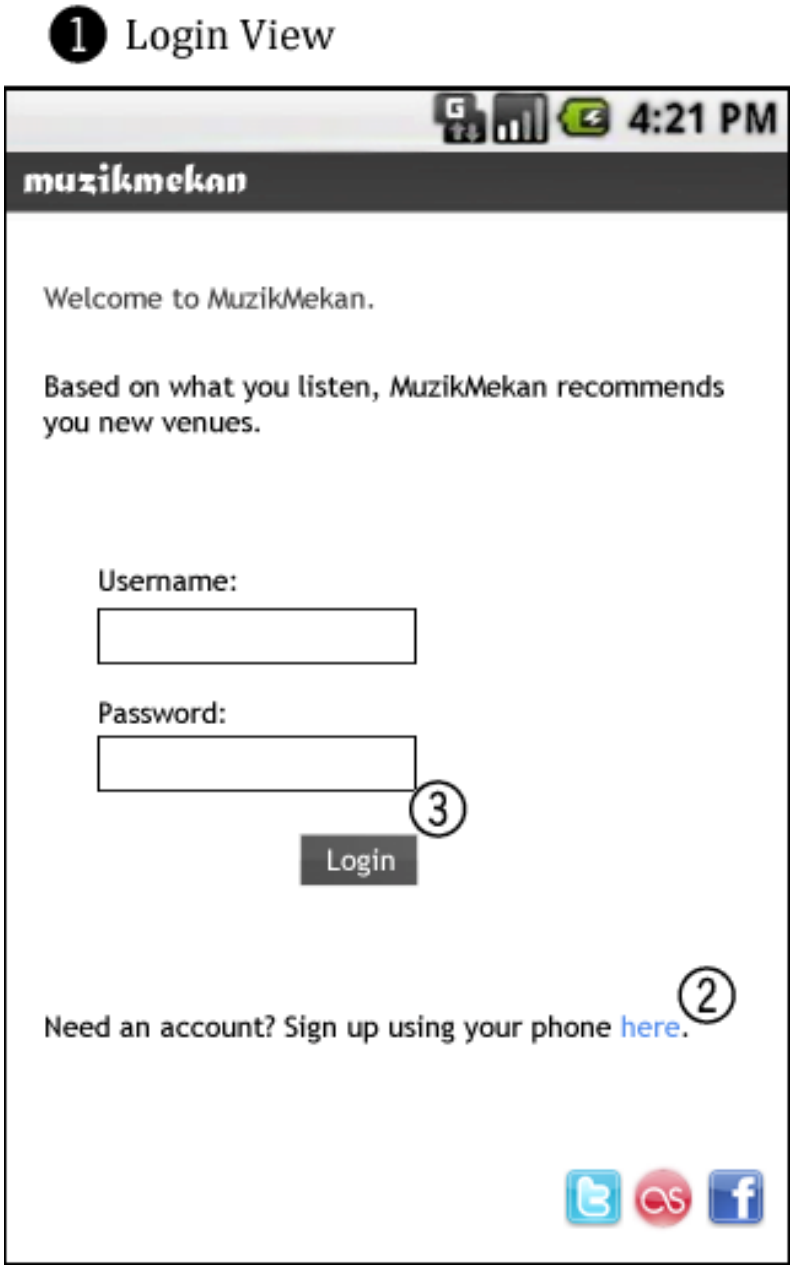
4.5.7 Get Dominant Music Taste Diagram

5 Interface Design

5.1 User Interfaces

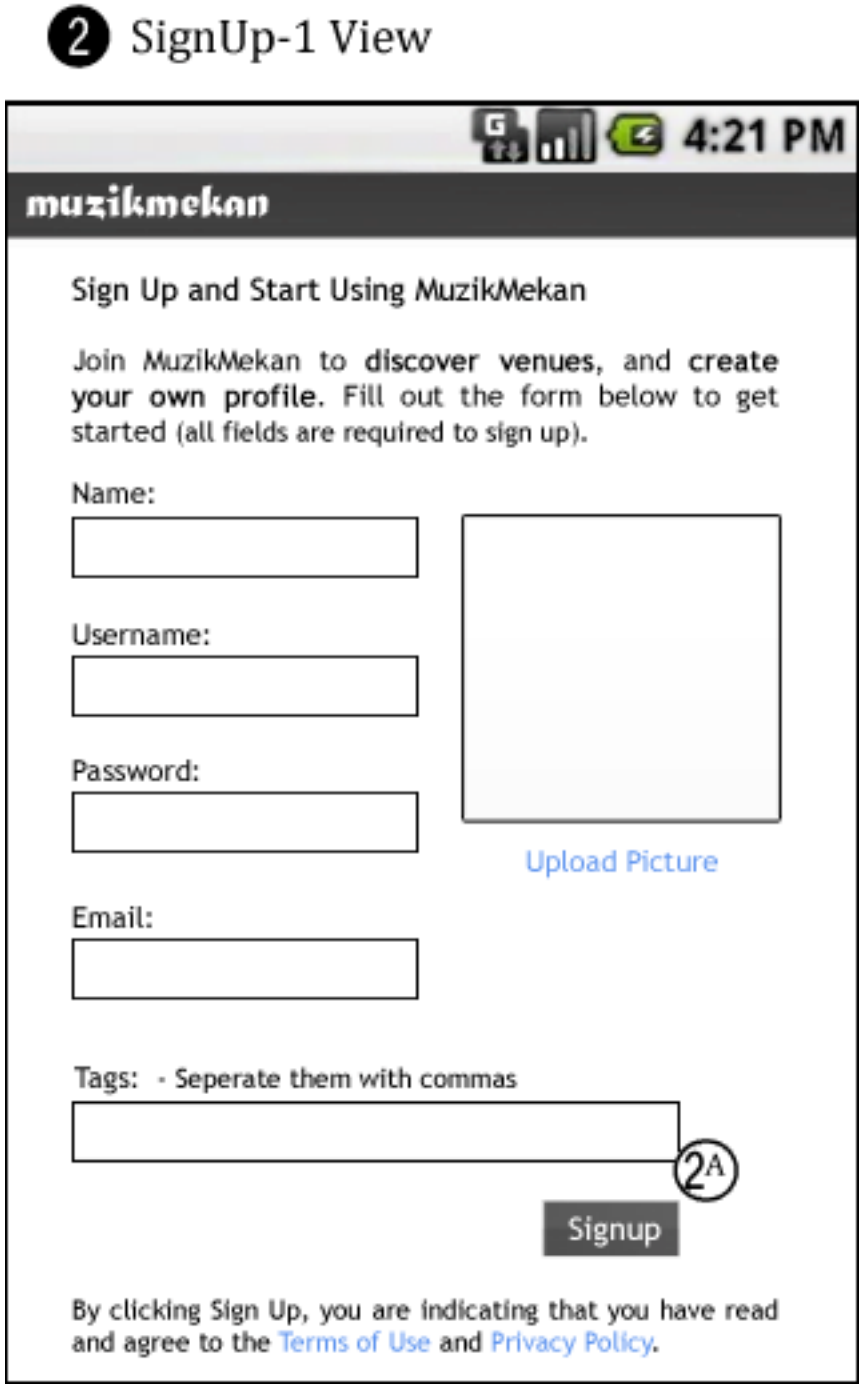
The user interface mockups are illustrated below.

5.1.1 Login View



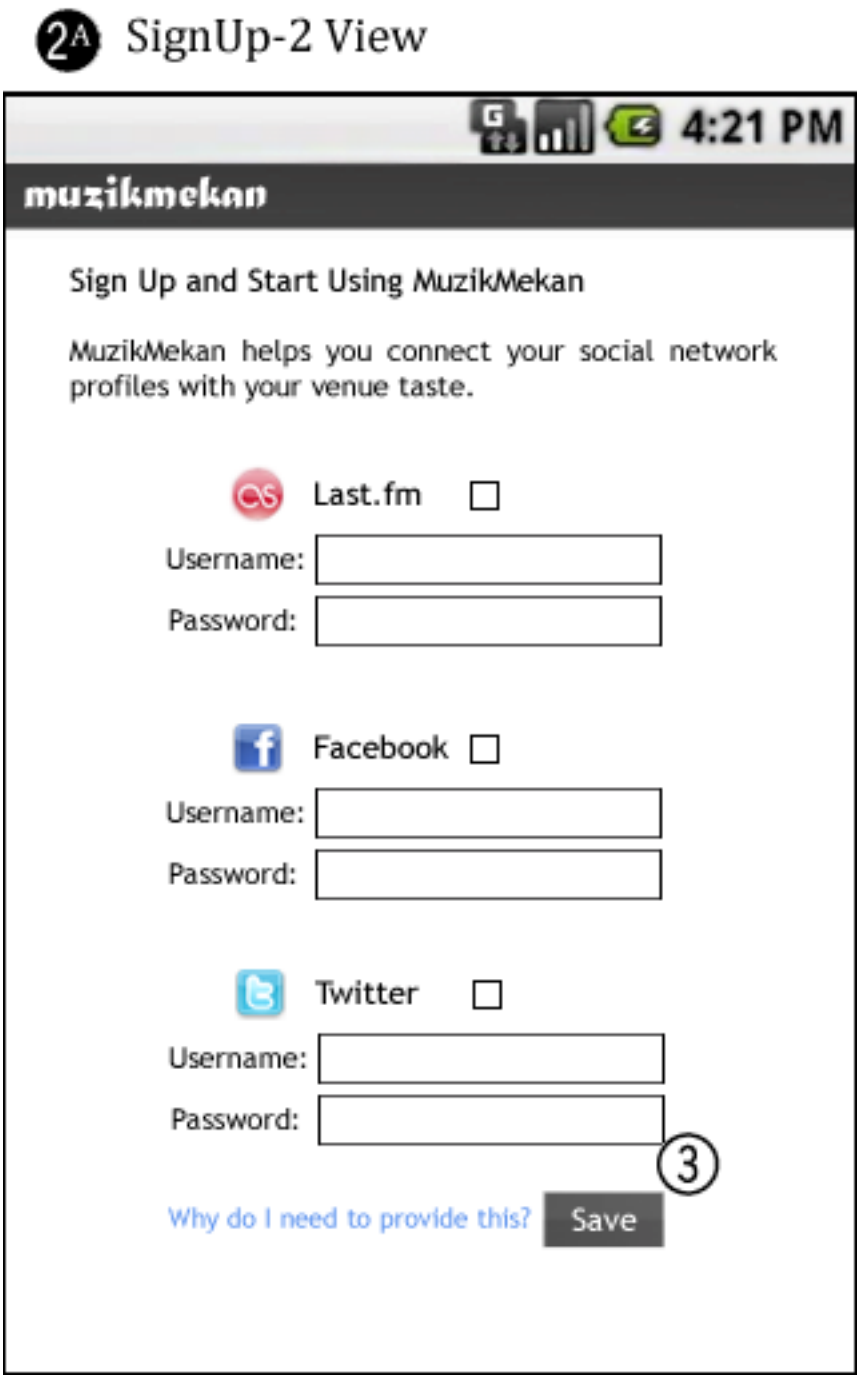
MuzikMekan system provides a simple user interface to the user to login the system. Users are able to reach their **'3 User Profile View'** after they fill out the username and password fields correctly and click 'Login' button. Also this view provides a link to **'2 Sign-Up View'** at the bottom of the view.

5.1.2 Sign-Up 1 View



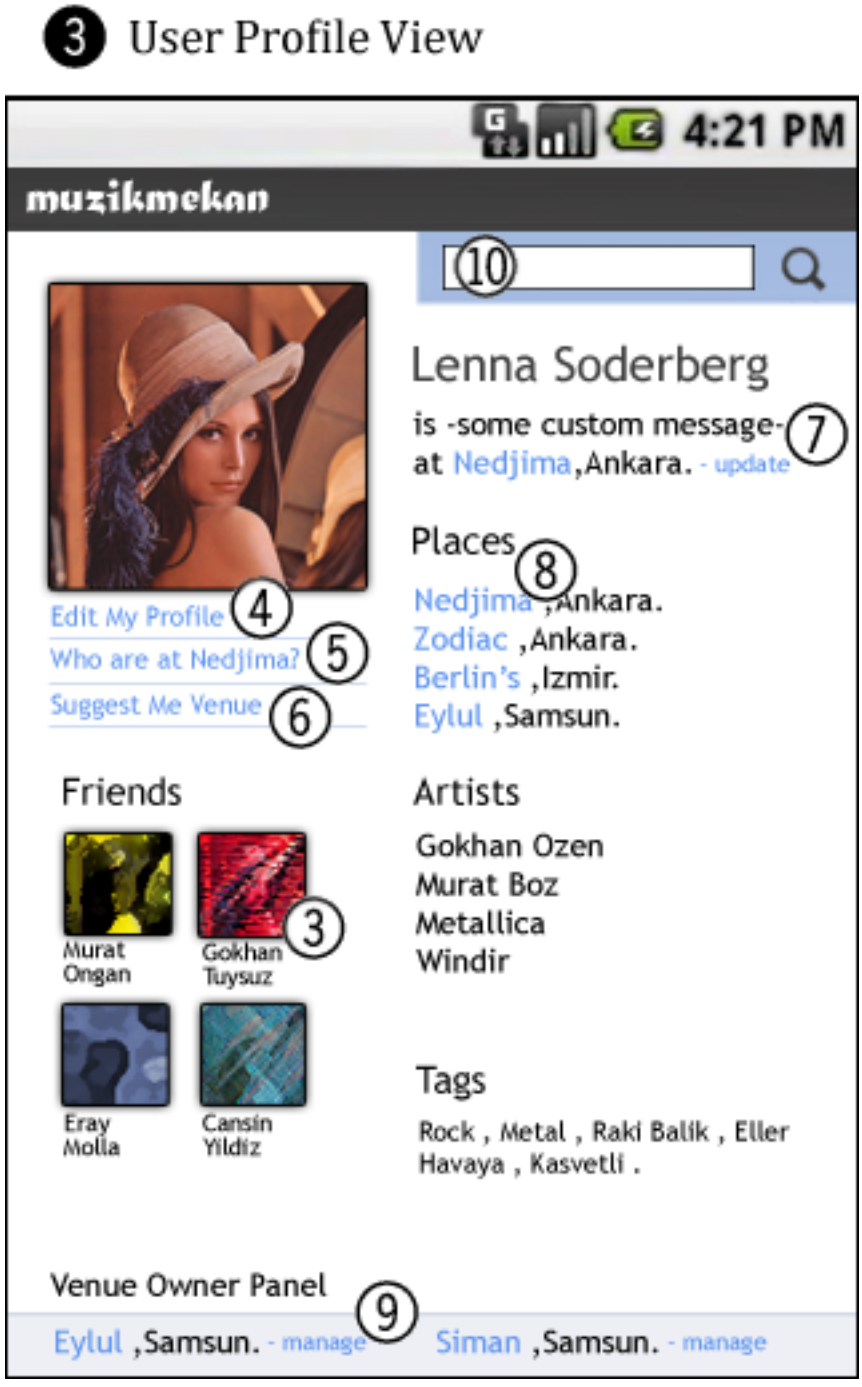
User must have an account to use the specific features of the application. Hence users are able to create an account by this view. Users must supply unique username and e-mail address to register the system. Optionally users may upload picture. In tags field, user should enter the tags that reflect their music and venue tastes. These tags are necessary to get a meaningful suggestion from the system. When user clicks 'Signup' button, he will directed to **'2A Sign-Up View'** to configure network accounts.

5.1.2.A Sign-Up 2 View



System needs the music and venue taste of the user to suggest a place. Users can supply tags to the system in the first view of create an account page. Yet, the system is able to synchronize their 'Last.Fm' account to their 'MuzikMekan' account and take the necessary information (Favorite Artists, Favorite Tracks, etc.). Moreover, users can use the system to publish their status in their 'Facebook' and 'Twitter' account. When saved, user will be directed to **'3 User Profile View'**.

5.1.3 User Profile View



This is the main view of the application. Users are able to reach almost all functionality of the application from this view. Users may reach the **'4 Edit User Profile-1 View'** from the link below the profile picture named 'Edit My Profile'. If user had marked himself inside a place, there will exist a link below the profile picture 'Who are at <a Venue>' and when user click this link, system will direct the user to **'5 People Near You View'**.

'Suggest Me Venue' link will direct the user to **'6 Suggestion-1 View'** where user can get a suggestion according to his profile information.

Also when user mark himself inside a place, system will automatically show 'is <Custom Message> at <a Venue>, <the City>' below the username and there will be a link 'Update', which redirects to **'7 Update Location View'**. At **'7 Update Location View',** user can update in which venue he is. Moreover, users may enter a custom message to the near of 'is', by simply clicking the field.

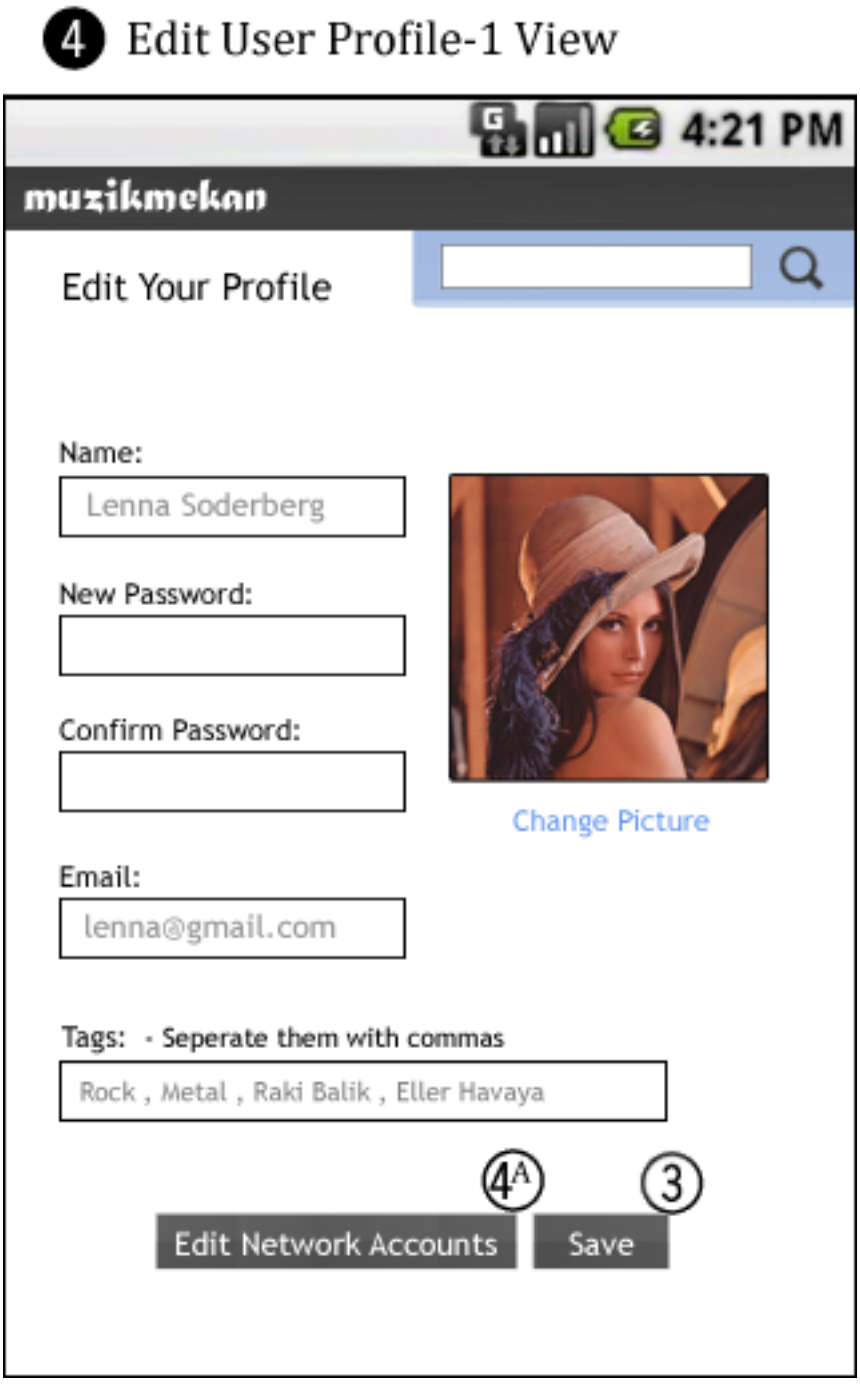
User may reach the profile of their friends (**'3 User Profile View'**), by clicking the picture or name of them, and he can reach the profile of the venues (**'8 Venue Profile View'**) by clicking on the venue name as well. It must be noted that 'Edit My Profile', 'Suggest Me Venue' like links will not be visible to other users, i.e. these links are only visible to user's himself at his own profile view.

Furthermore, this view will show the information about the artists and tags of the user.

At the very below of the page, there exists a 'Venue Owner Panel'. This panel is only visible to users who own a venue, where they can manage their venues view by clicking the 'manage' link (which redirects to **'9 Edit Venue Profile View'**).

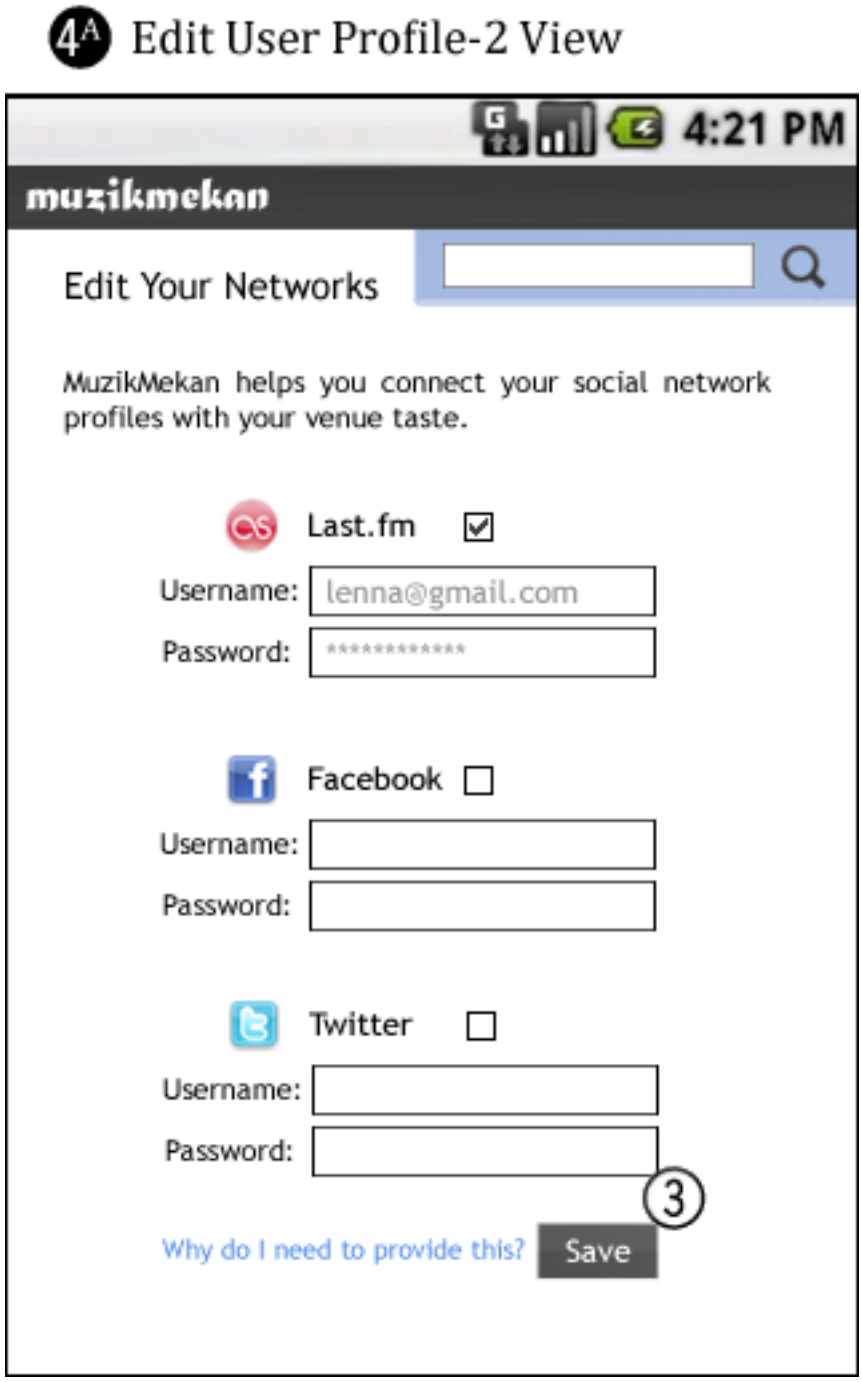
At the very above of the page, there exists a search panel where user can search general keywords among venues and users. After search done, **'10 Search View'** will be shown.

5.1.4 Edit User Profile-1 View



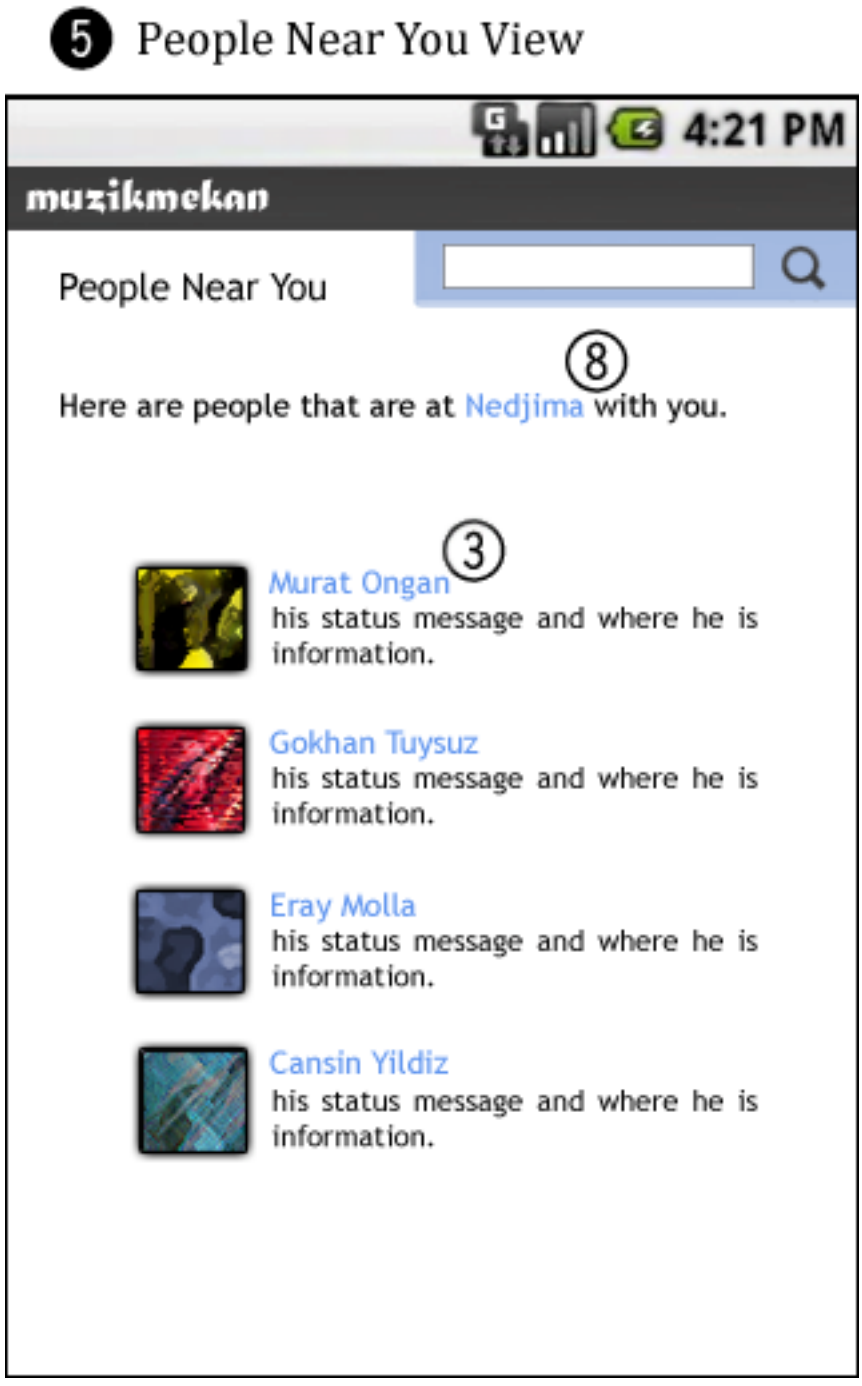
Users are able to edit their profile using this edit view. User should fill out the field that they want to update and leave other fields as it is. Moreover, at this view user can able to reach the **'4A Edit User Profile-2 View'** by clicking the button 'Edit Network Accounts'. When user fills out the fields that he wants to update and clicks the 'Save' button, system will update the account of user and will direct the user to **'3 User Profile View'**.

5.1.4.A Edit User Profile-2 View



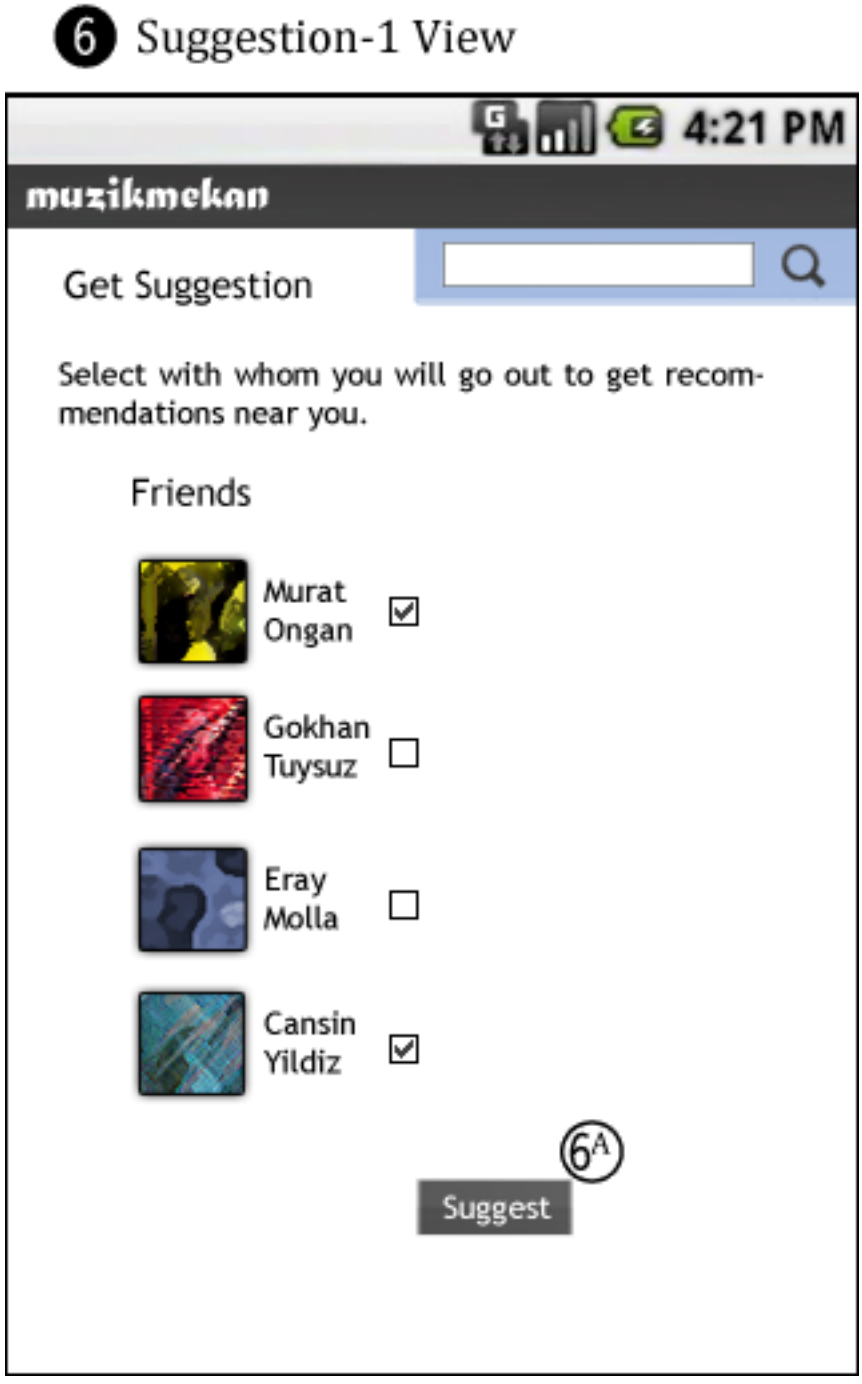
System provides an edit view to update the social network accounts of user. User simply checks the box near network name and fills out the required information. After 'Save' button is clicked, system updates the account information of the user and starts to synchronize the accounts. Then, user will be directed to **'3 User Profile View'**.

5.1.5 People Near You View



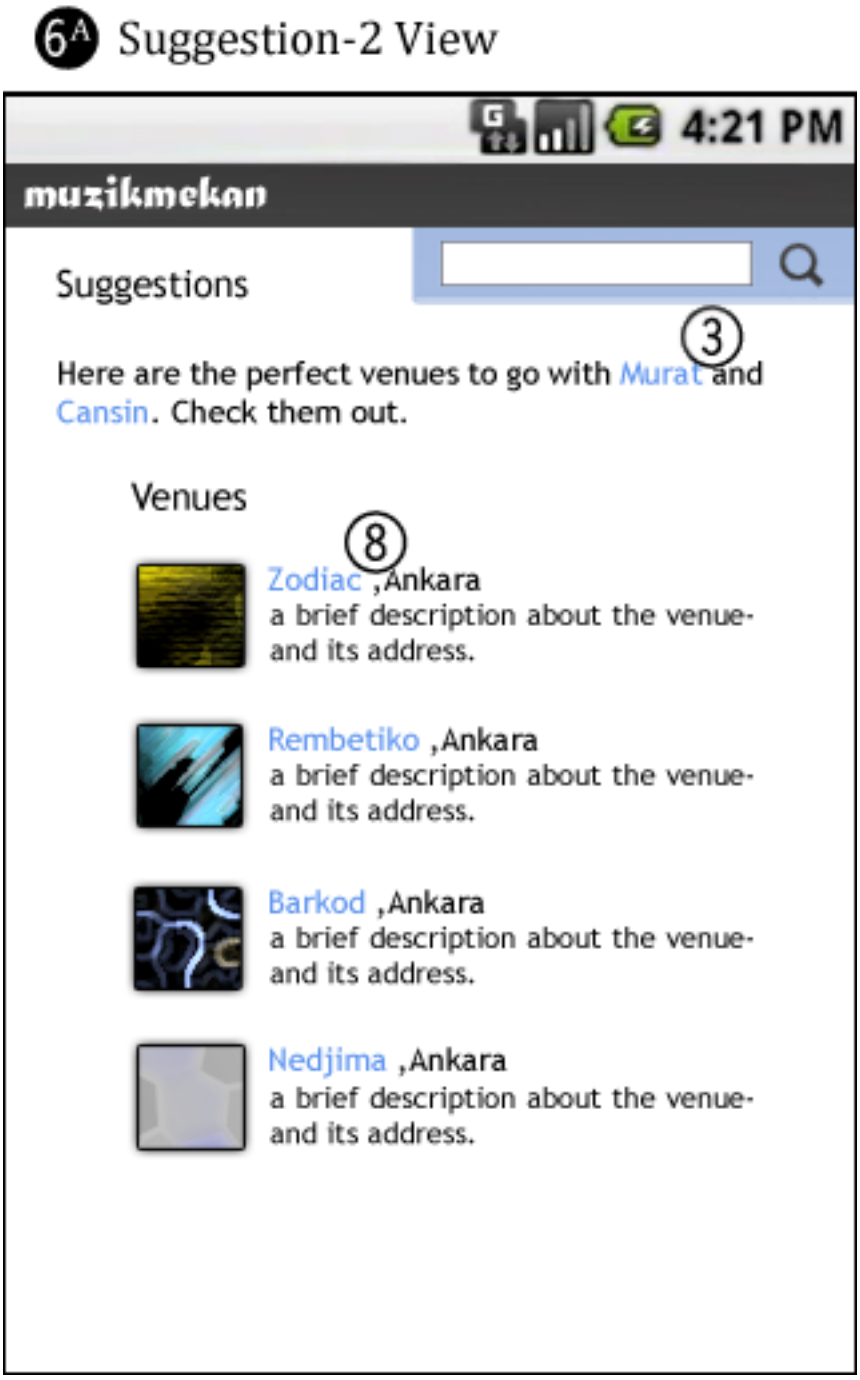
User may browse the people who are in the same venue with him. User will be directed to this view by clicking the 'Who are at <a Venue>' link if he had already marked that he is inside some venue. User may reach **'8 Venue Profile View'** by clicking the name of the place in this view. Moreover, user may reach the profiles of the users (**'3 User Profile View'**) by clicking the pictures or names of the users.

5.1.6 Suggestion-1 View



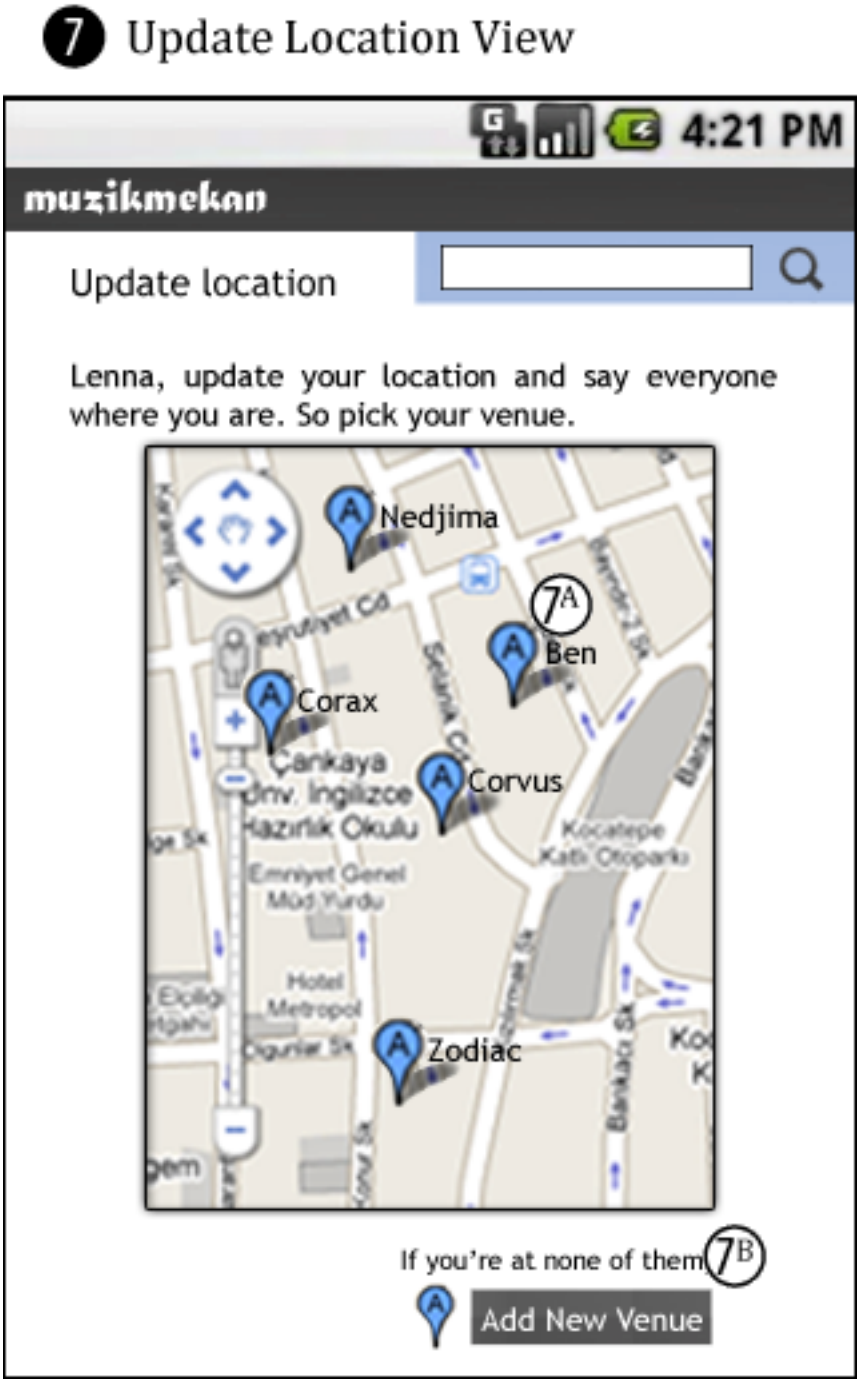
Users are able to get a venue suggestion in this view. The friend list will be shown and user can choose the friends that he wants to go with by simply checking the box near of their names. When 'Suggest' button is clicked, the system will return the list of the places where are near to them and are more compatible for all. Hence, the system will direct the user to **'6A Suggestion-2 View'**.

5.1.6.A Suggestion-2 View



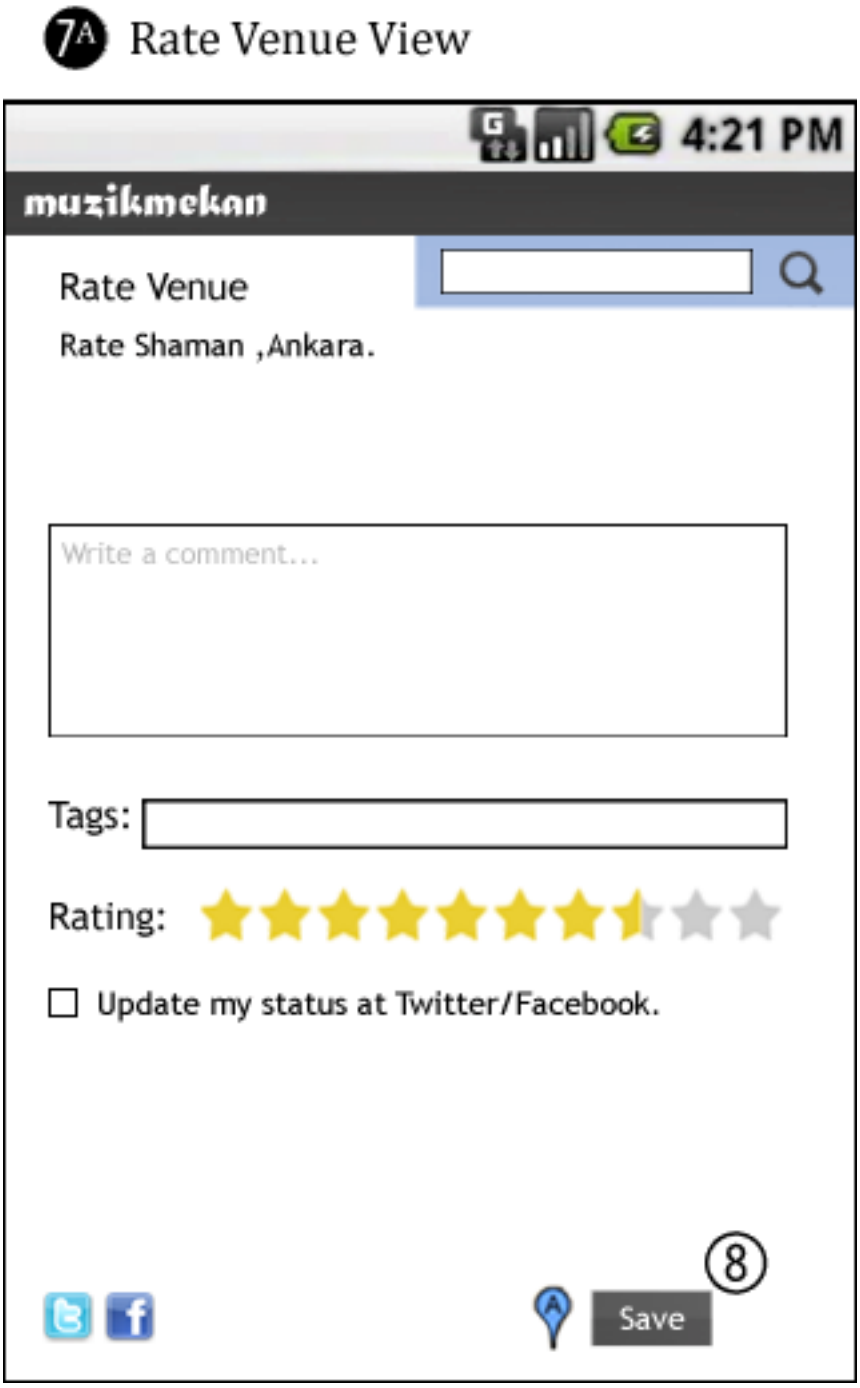
This view lists the suggested venues from best alternative to the worst alternative. This view also shows the name of the friends that the user chooses to go with and user can reach profile information of his friends (**'3 User Profile View'**) by clicking their name. Moreover, User can reach the venue profile (**'8 Venue Profile View'**) by clicking the picture or the name of the places that are suggested.

5.1.7 Update Location View



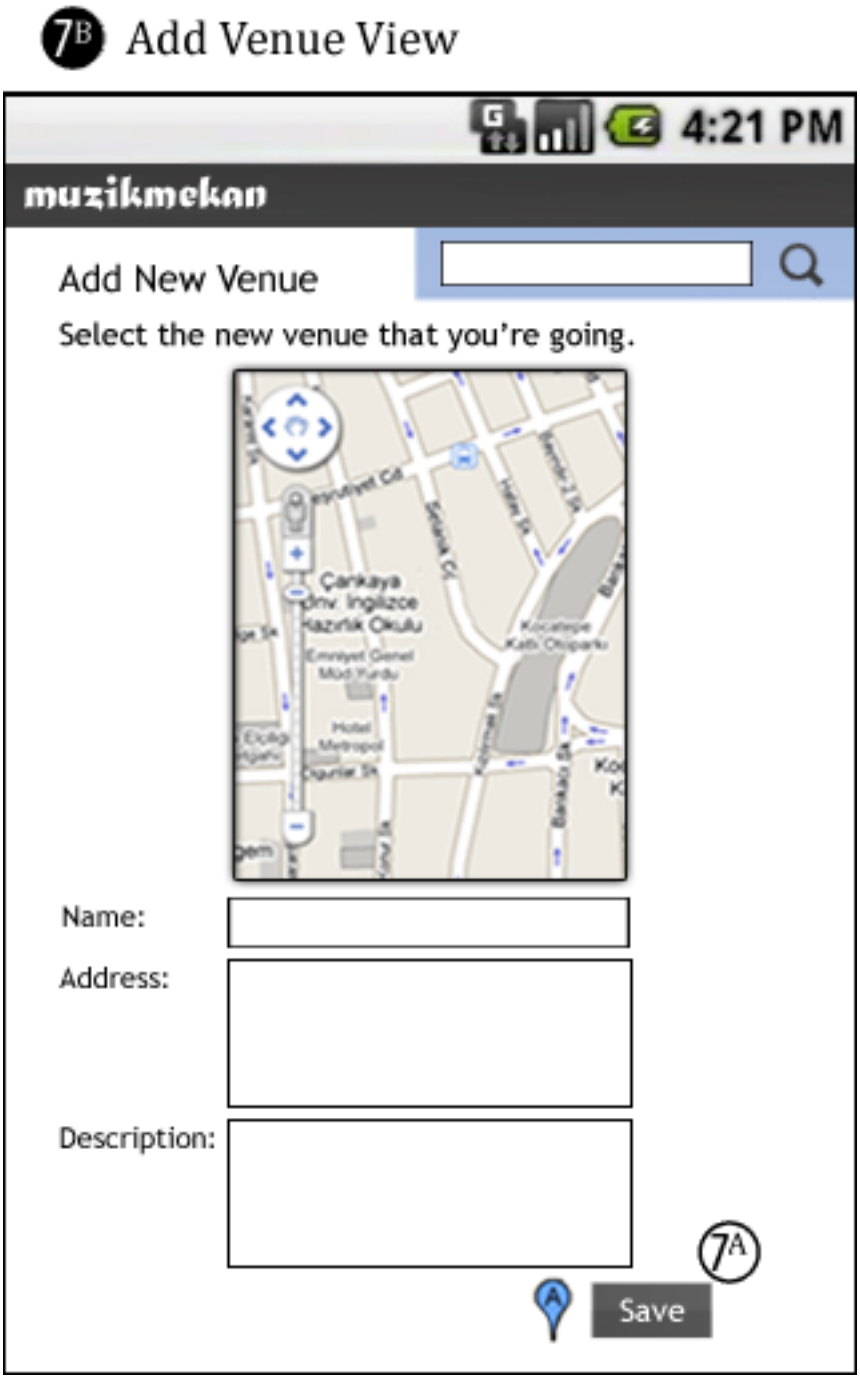
User is able to mark the venue that he is currently inside. MuzikMekan system will provide a map that is showing the venues that are registered to the system and are near to the user. When user clicks the venue, he will be directed to **'7A Rate Venue View'**. If the venue is not registered to the system, user can add new venue to the database. 'Add New Venue' button will direct to the user **'7B Add Venue View'**.

5.1.7.A Rate Venue View



User is able to rate the venue he is currently inside. User will be directed to this page when he marked the location in **'7 Update Location View'**. When user fills out the field that he wants to add to the system and clicks the save button, system will update the database of filled fields and user will be directed to **'3 User Profile View'**. If user checks the box 'Update my status at Twitter/Facebook', MuzikMekan system will push the status of the user to these networks.

5.1.7.B Add Venue View



In this view user will mark the location of the place on the map and fill out the required fields to add a new place to the system and when 'Save' button is clicked, MuzikMekan system will add this place to the database and will direct to the user **'7A Rate Venue View'**.

5.1.8 Venue Profile View

This view will show address, tags, recently played tracks information of the place. Inside part will list the names of users that marked they are currently inside. Also, user can reach the **'8B Comments View'** by clicking the 'Read Comments' link and **'8A Events View'** by clicking 'See Events' link. System will provide the location of the place on the map. Owners can edit their venue profile account by clicking the 'Manage' link and will be directed to **'9 Edit Venue Profile View'**.

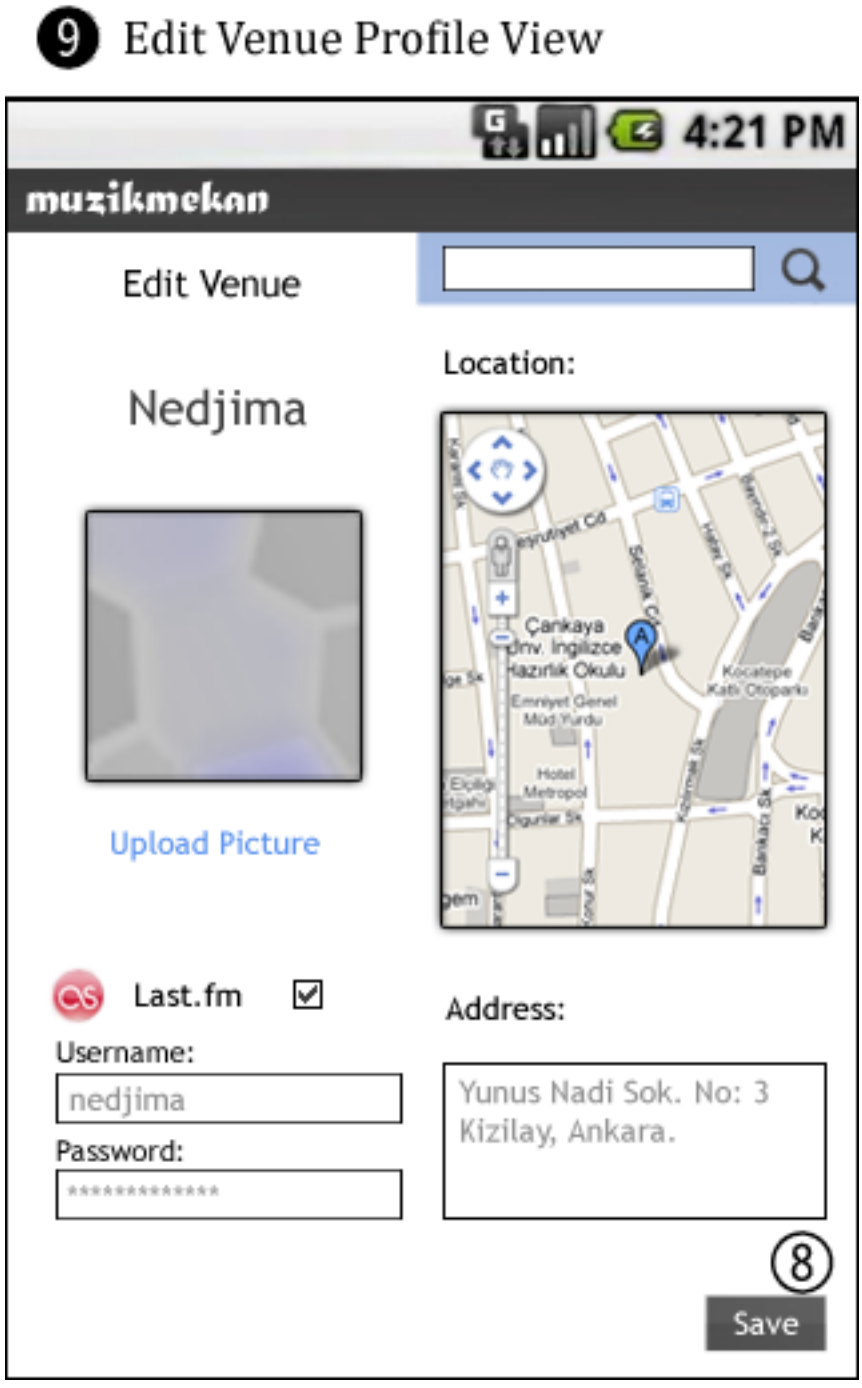
5.1.8.A Events View

This view will show the list of the upcoming events that will be arranged in the venue.

5.1.8.B Comments View

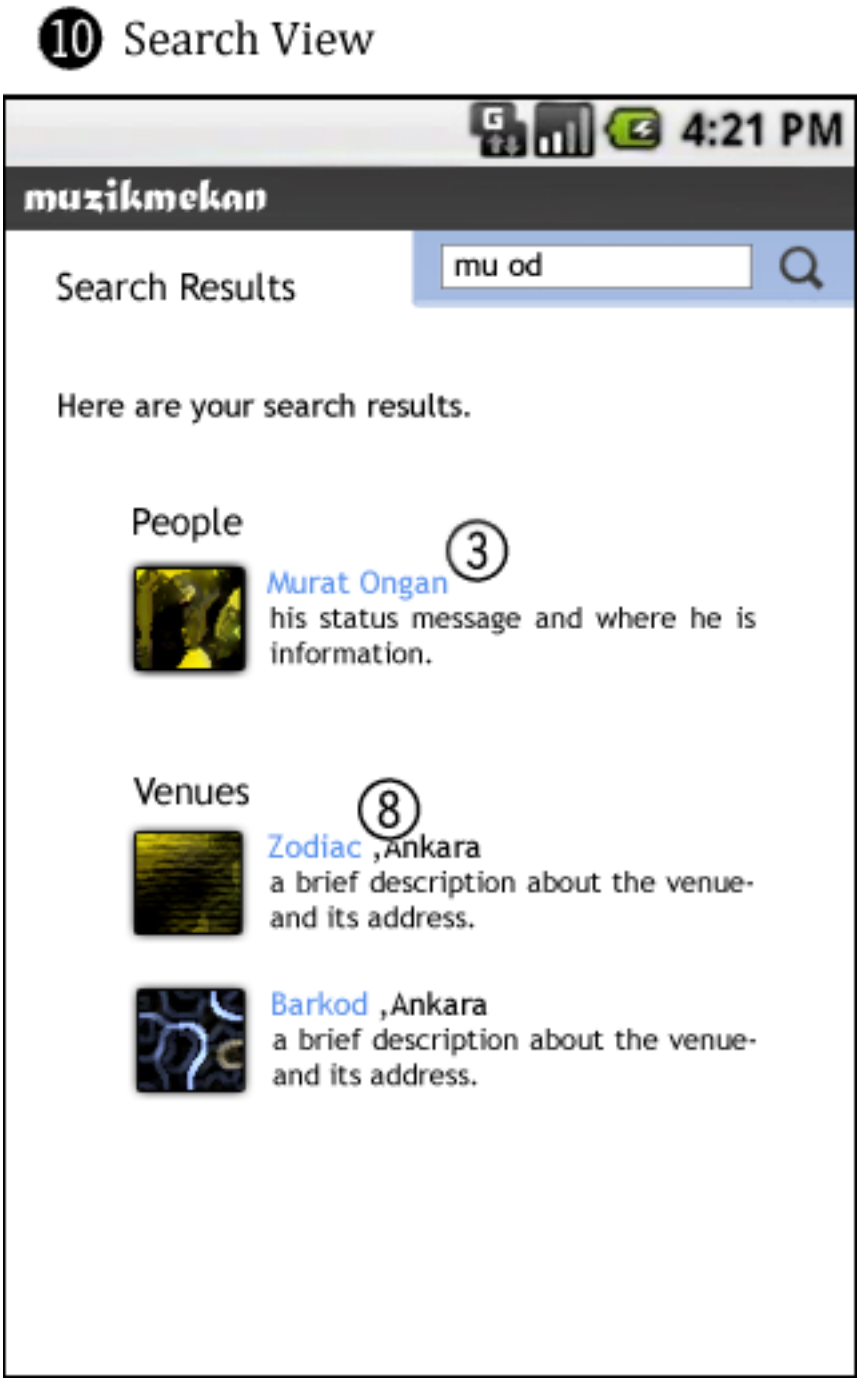
User is able to make comments about a place by using this view. After user enters the comments about place and clicks 'Send' button, system will add new comment to the database. Also, this view shows the address of the venue and the comments that is made before.

5.1.9 Edit Venue Profile View



Venue owner is able to edit the venue profile in this view. This view has the functionality of uploading picture, changing address and location and adding or changing the Last.Fm account of this venue. To be able to add Last.Fm account, box must be checked near the Last.Fm name. When owner clicks the 'Save' button, he will be directed to **'8 Venue Profile View'**.

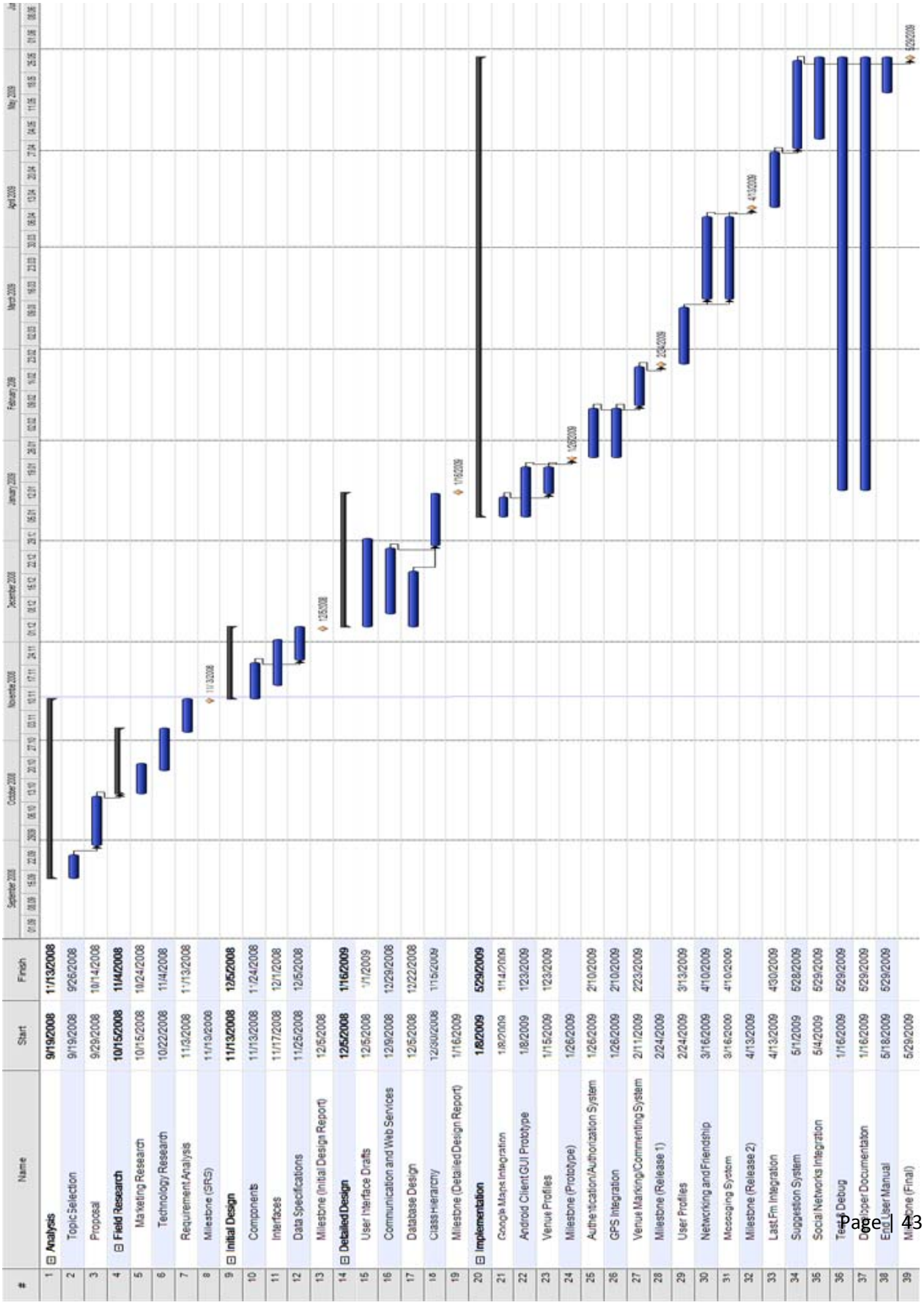
5.1.10 Search View



User is able to search the keyword, which they want. When user uses the search field at the very top of the page, he will be directed to this view. Hence, user may reach the profile view of any result.

6 Project Schedule

The roadmap of the MuzikMekan project is illustrated as a Gantt chart below.



#	Name	Start	Finish
1	**Analysis**	**9/19/2008**	**11/13/2008**
2	Topic Selection	9/19/2008	9/26/2008
3	Proposal	9/29/2008	10/14/2008
4	**Field Research**	**10/15/2008**	**11/4/2008**
5	Marketing Research	10/15/2008	10/24/2008
6	Technology Research	10/22/2008	11/4/2008
7	Requirement Analysis	11/3/2008	11/13/2008
8	Milestone (SRS)	11/13/2008	
9	**Initial Design**	**11/13/2008**	**12/5/2008**
10	Components	11/13/2008	11/24/2008
11	Interfaces	11/17/2008	12/1/2008
12	Data Specifications	11/25/2008	12/5/2008
13	Milestone (Initial Design Report)	12/5/2008	
14	**Detailed Design**	**12/5/2008**	**1/16/2009**
15	User Interface Drafts	12/5/2008	1/1/2009
16	Communication and Web Services	12/9/2008	12/29/2008
17	Database Design	12/5/2008	12/22/2008
18	Class Hierarchy	12/30/2008	1/15/2009
19	Milestone (Detailed Design Report)	1/16/2009	
20	**Implementation**	**1/8/2009**	**5/29/2009**
21	Google Maps Integration	1/8/2009	1/14/2009
22	Android Client GUI Prototype	1/8/2009	1/23/2009
23	Venue Profiles	1/15/2009	1/23/2009
24	Milestone (Prototype)	1/26/2009	
25	Authentication/Authorization System	1/26/2009	2/10/2009
26	GPS Integration	1/26/2009	2/10/2009
27	Venue Marking/Commenting System	2/11/2009	2/23/2009
28	Milestone (Release 1)	2/24/2009	
29	User Profiles	2/24/2009	3/13/2009
30	Networking and Friendship	3/16/2009	4/10/2009
31	Messaging System	3/16/2009	4/10/2009
32	Milestone (Release 2)	4/13/2009	
33	Last Fm Integration	4/13/2009	4/30/2009
34	Suggestion System	5/1/2009	5/28/2009
35	Social Networks Integration	5/4/2009	5/29/2009
36	Test & Debug	1/16/2009	5/29/2009
37	Developer Documentation	1/16/2009	5/29/2009
38	End User Manual	5/18/2009	5/29/2009
39	Milestone (Final)	5/29/2009	

7 References

- Software Engineering A Practitioner's Approach, 5th Edition , Roger S. Pressman
- Component Oriented Software Engineering, Ali H. Doğru
- http://www.databaseanswers.org/data_models/index.htm
- http://www.agilemodeling.com/artifacts/classDiagram.htm
- http://www.lastfm.com.tr/api
- http://www.w3.org/TR/ws-arch/
- http://www.javaworld.com/javaworld/jw-07-2004/jw-0719-jsf.html
- http://java.sun.com/blueprints/guidelines/designing_enterprise_applications_2e/web-tier/web-tier5.html
- http://code.google.com/intl/tr-TR/android/documentation.html
- Prentice Hall Core Java Server(TM) Faces 2nd edition, David Geary, Cay S. Horstmann