Term Project for Advanced Software Paradigms (CSCI-210-10)

The Department of Computer Science

The George Washington University

# TODAY

# Requirements Specification

Version: 1.0

Oct. 10, 2010

Prepared for: Prof. Abdelghani Bellaachia

Prepared by: Maoxu Li & Zhong Xie

## Description

Developing an Android application named Today to facilitate recording and tracking on moods, sharing moods with friends and making new friends via moods matching.

## Team Members

| Maoxu Li | Zhong Xie |
|---|---|
| lim@gwmail.gwu.edu | xz1989@gwmail.gwu.edu |

# Table of Contents

# Today – A mobile tool of moods management

Maoxu Li, Zhong Xie

*The department of Computer Science, The George Washington University*

*lim@gwmail.gwu.edu, xz1989@gwmail.gwu.edu*

## Requirements Analysis

## I. Introduction

This project will design and implement a local application running on Android platform with the graphic user interface and the interactions with the web services running on central server (The development of the server side, including database and web services, will be completed as anther class project).

The application, named Today, is used to record users' mood every day, even any time. Based on the mood data recorded in the database, users may track on their moods changing during the past days. Users may share moods with their friends. The application also allows the user to find other users with the same mood and then make friends with them.

## II. Functions and User Interface

### 1. Overview

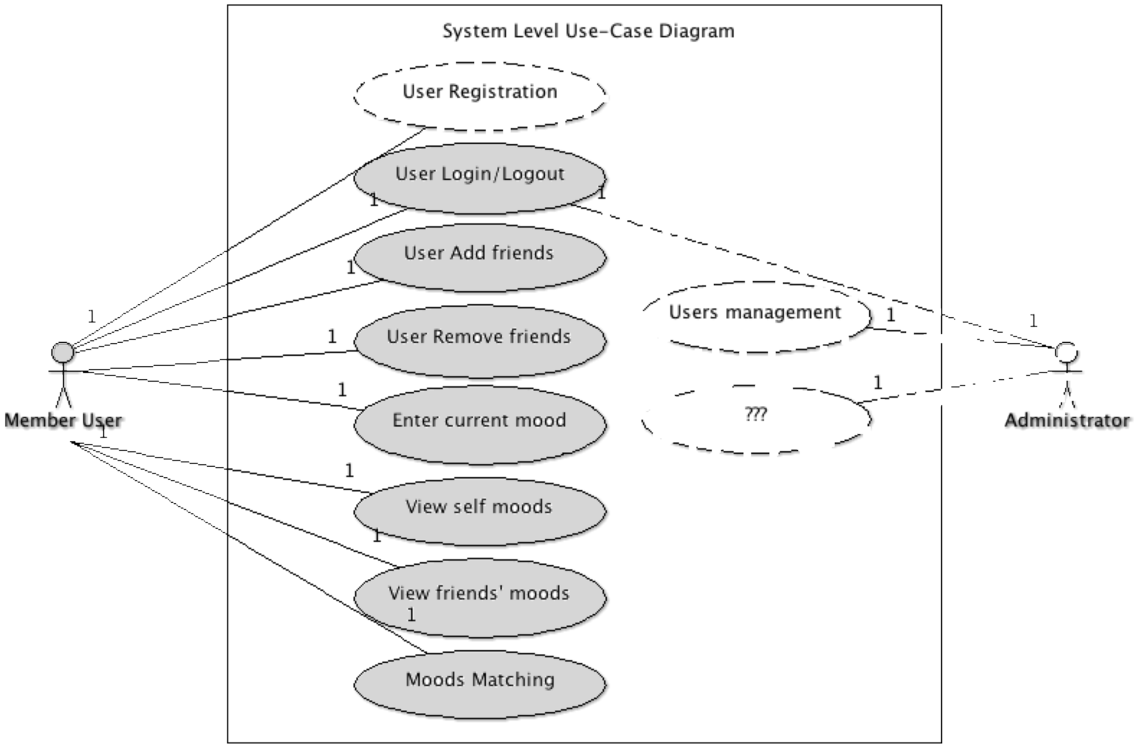
The system records users' moods and provides some analysis functions. From the viewpoint of application running on Android platform, Today displays a list of icons corresponding to various feelings. You may click one of the icons to record your current mood everywhere and anytime. The moods are permanently stored in a central database. Today also displays an icon to show your current mood and a curve to denote how your moods change during the past days so that you can track on your moods and remind yourself to adjust moods time by time. With some configurations, Today will display the moods of your friends. You can also ask after your friends. Besides, Today will do moods matching between friends and even strangers to let you find someone with moods like you.

Here we will not consider how moods data are stored in the server, which will be designed and implemented as another class project. But we know some web service interfaces with which we can access the functions provided with server.

## 2. Use-cases

This document use use-cases to describe the requirements of functions and related user interface.

As a social networking application, some functions related with users management are necessary. A super user is predefined to act as administrator of the system. Other users must register as member users to access the system. With the first version, users registration and the functions related with administrator only are available on a web application, which is not developed in this project. Here we focus on the member users and their actions. An integrated use-case diagram is shown below (Dark parts will be realized in this project). The rest parts of this section give detailed use-case descriptions.



### a) User Login

The users must have registered as member users of the system. Users need supply their login credentials in order to gain access to the system.

**Precondition:** User is not logged in, but is a member of the system.

1. Application displays "login screen".
2. User enters id/name/email and password to login in the system.
3. Application connects to server to authorize the user.
4. If successful, the user is authenticated with proper rights and the application displays "main screen".
5. If failed, application displays an error message and returns to login screen.

**Post Condition:** User is logged in and application displays main screen.

**User Interface:**

1. Login screen:
    a. Input: id/name/email, password
    b. Button: login, close
2. Main screen:
    a. Mood icons list: click to enter mood
    b. Self mood: self-mood display
    c. Friends mood: friends' moods display
    d. Button: add friends, remove friends, moods matching

### b) User adds friends

The system's notation of "friends" is people that are able to share moods each other. In order to become a friend, a user must confirm you.

**Precondition:** User is logged in.

1. Application displays "Add friends screen"
2. User searches for an existing user by their id, name or email.
3. Application displays the matches with a list and check boxes
4. User selects the users they are looking for and request to add as friends.
5. System sends an email to the selected user requesting to be friends.
6. When other user confirms the "friendship", system sets these users to be friends.

**Post Condition:** User has new friends associated with his account.

**User Interface:**

1. Add friends screen:
    a. Input: id/name/email
    b. List: matched users, with check boxes
    c. Button: search, add friends, close

### c) User removes friends

The application allows users to remove friends from their list.

**Precondition:** User is logged in.

1. Application displays "remove friends screen".
2. Application displays current friends list with check boxes.
3. User selects the friends they are removing and remove friends.
4. Application asks the user to confirm the removal of the friends.
5. System removes friends after confirm.

**Post Condition:** The selected friends have been removed from the user's friend list.

**User Interface:**

1. Remove friends screen:
    a. List: current friends, with select boxes
    b. Button: remove friends, close

### d) User enters current mood

Application allows user to enter current mood in "main screen".

**Precondition:** User is logged in

1. Application always displays moods icon list in "main screen".
2. User clicks the proper icon to enter current mood.
3. System saves the entered mood into the database.

**Post Condition:** A new record of user's mood is stored in the database.

**User Interface:**

1. Main screen
    a. Mood icon list: a list of icons corresponding to some feelings, like excited, very happy, good, just so so, sad, desperate, and so on.

### e) Displays user's self mood

The application always displays user's recent mood in "main screen".

**Precondition:** User is logged in

1. Application always displays current user's mood in "main screen".
2. User may click the "mood-curve" to view details.
3. Application displays the details of user's mood in a new screen.

**Post Condition:** User's recent "mood-curve" is displayed and details can be viewed.

**User Interface:**

1. Main Screen
    a. Self-mood: a icon denotes user's current mood and a "mood-curve" denotes the mood changing during the past days.

### f) Display friends' moods

The application allows users to share moods with their friends. The user can view the "mood-curve" of his friends, also his friends can view his "mood-curve".

**Preconditioned:** User is logged in.

1. Application always displays a list of friends' mood in "main screen".
2. User selects one friend that he is interested to viewing details.
3. Application displays the details of user's mood in new screen.

**Post Condition:** The recent moods of friends are displayed in a screen.

**User Interface:**

1. Main Screen
    a. Friends' mood: a friend list with mood of each friend like self mood display.

### g) Moods Matching

The system can analyze other users' moods and find users with similar moods. The user may make friends with these "matched" users.

**Preconditioned:** User is logged in.

1. Application displays "moods matching screen".
2. Application displays a list of matched users with check boxes.
3. User selects the interesting user to views details of matching.
4. User selects interesting users and adds as friends.
5. The process to add as friends is same as former.

**Post Condition:** User finds some friends with certain moods.

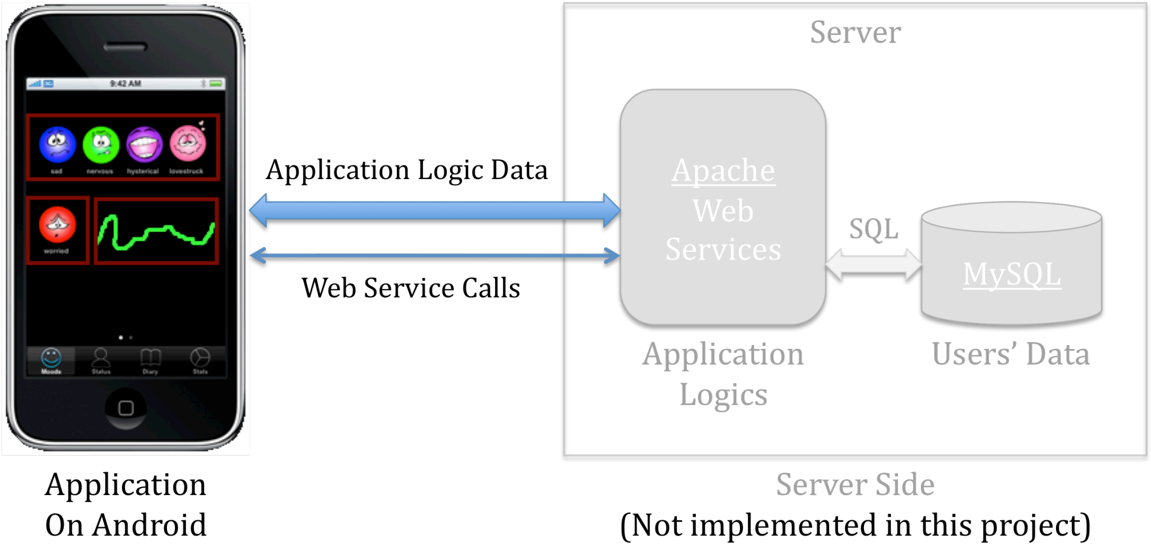
**User Interface:**

1. Moods matching screen
    a. List: matched users with check boxes
    b. Button: refresh, add as friends, close

# III. Technical Analysis

## 3. Software Structure

Overall, the system includes two parts: server side and client side. On server side, central database is deployed to store the users' data. Also, some web services are deployed to offer data access and application logic service. On client side, an Android application is developed to interact with the user and interact with the server to collect, record, analyze, and display the user's moods. The architecture is shown below.



This project will complete a mobile application running on Android platform, which includes below logic modules:

- An Android application framework
- Application logics
- User Interface
- Web service access

### 4. Development Platform

- Running environment: Android platform and Android Dalvik VM
- Development interface: Android SDK
- Programming language: Java

## IV. Development Issues

### 5. Development Landmarks

Here is not a complete software development plan but some key times that act as checkpoints to verify the development progress and to adjust the development plan.

| | Task | Output | Due | Memo |
|---|---|---|---|---|
| 1 | Problem identification and initial analysis | Abstract | Sep. 27 | |
| 2 | Requirements Analysis and initial design | Requirements | Oct. 11 | |
| 3 | Design and framework coding | Designs | Oct. 25 | |
| 4 | Coding | Prototype | Nov. 8 | Demo |
| 5 | Coding<br>And integration test | System and documents | Nov. 29 | |