Brian Chan
Edwin Prakarsa
Rob Mantzoros
Jaideep Juneja
Neil Amin
Shreyaansh Bassi

CS307 Team 20 Design Document

TeamUp

Table of Contents

Description	Page
Purpose	3
Design Outline	3
Design Issues	4
Design Details	5

1. Purpose

This is an android application that allows teams to communicate with each other efficiently to get objectives done. Team users can create and manage groups, projects, tasks, as well as text message each other as a group or privately.

2. Design Outline

The components of this system will consist of a client-server model. The client will be an android application and the server will be created using Node.js, WS WebSockets, and a MongoDB database for easy setup and deployment. Creating an android application was chosen because of its large user base, high availability, and excellent medium of communication. The server is needed because it stores and sends all data related to messages, current groups, projects, and task list for the projects.

App

1. Team members will use this to interact with each other.
2. The app will display current tasks and deadlines for a project.
3. The app will make requests to the server.

Server

1. The server will synchronize users and deliver messages.
2. The server will keep track of tasks and report deadlines to users.
3. Handle incoming requests and access the database if need be.

Database

1. The database will hold current tasks and store messages.
2. The database will keep track of team members and groups.

3. Design Issues

Functional :

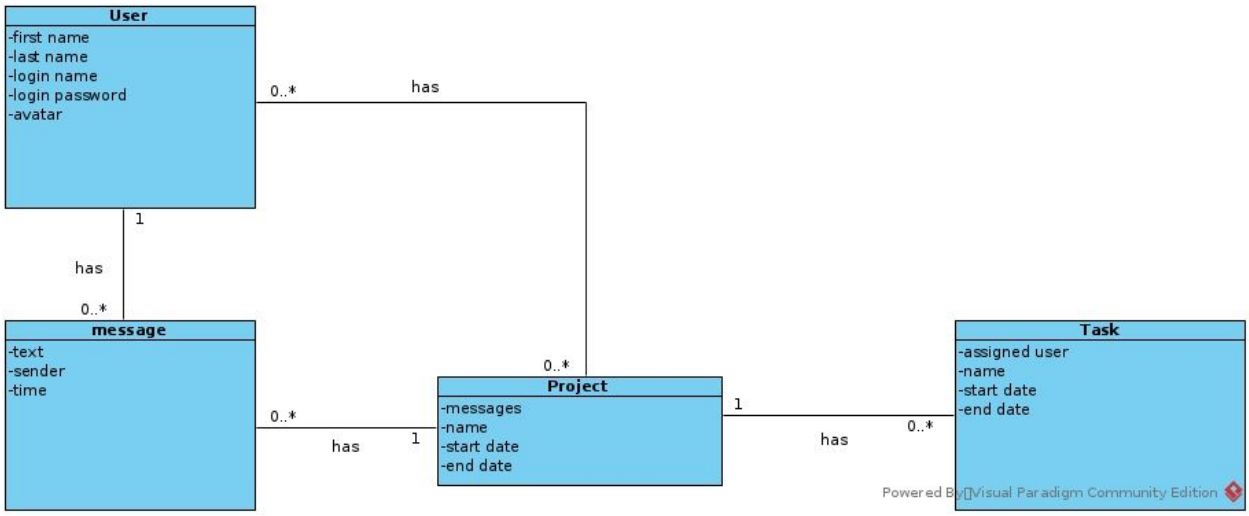
- We also need to decide how we should let a user to create his/her account, either go with simple user password or let the user create an account using facebook. We think that we will just go with simple user password because to time issues, if at the end of sprint 3 we still have a lot of time, we will go with the implementation of creating account using facebook.
- Because we are using simple user password with no email registered, we have another design issues of how if the user forgot the password. Either we implement the creating account option with email or facebook or we could just use some security questions options like what is your mother maiden name or the name of your high school, etc. We think that security question options will be much more simpler than implementing the create account option using facebook or email.
- One issue is how to let the user backup their local data, we might let the local data such as the task information and the messages to store in their device memory or save it into the server. We discussed that it will be better to save it into the server because by saving into the server, the user can access their data from any devices as long as they have their account connected.
- Group authorization is a critical thing in messaging group, deciding whether member has full authorization of the group such as adding task, adding member, kicking member, setting deadlines of the task and so on. We decided that only leader has a full control of the group while member don't because if member has a full authorization, the group will have a difficult hard time if one member change the task while others not aware of it.

Non-Functional :

- One design issues is choosing what languages we are using as a server, is it either javascript or java based server. We think that javascript is the best option for this issues because by using WS websockets implementation for nodeJS, we can easily send information back and forth much easier than the regular java server.
- User Interface is really important in developing an applications. We are having some issues of what languages we should choose for the user interface design. We have

choices between Java and design the user interfaces in XML or another alternative is Corona SDK that build on LUA programming languages that is much simpler than Java. We chose java because we had learned it before and it comes with ADT bundle which includes the SDK and the Android-Platform-tools with the Android emulator, but the important thing is it's free.

4. Design Details



The system is composed of 4 classes: user, message, project, and task.

User:

The user is the person who is actually using the android app. Each user has a name, login, and optional avatar. Users can create projects and add other team members to them, but will not be part of any initially. Users can receive or send private/project specific messages. Every project has zero (ie. future projects) or more users, and any task belongs to only one project.

Message:

The message class represents each message. These will be sent between users and stored on a database.

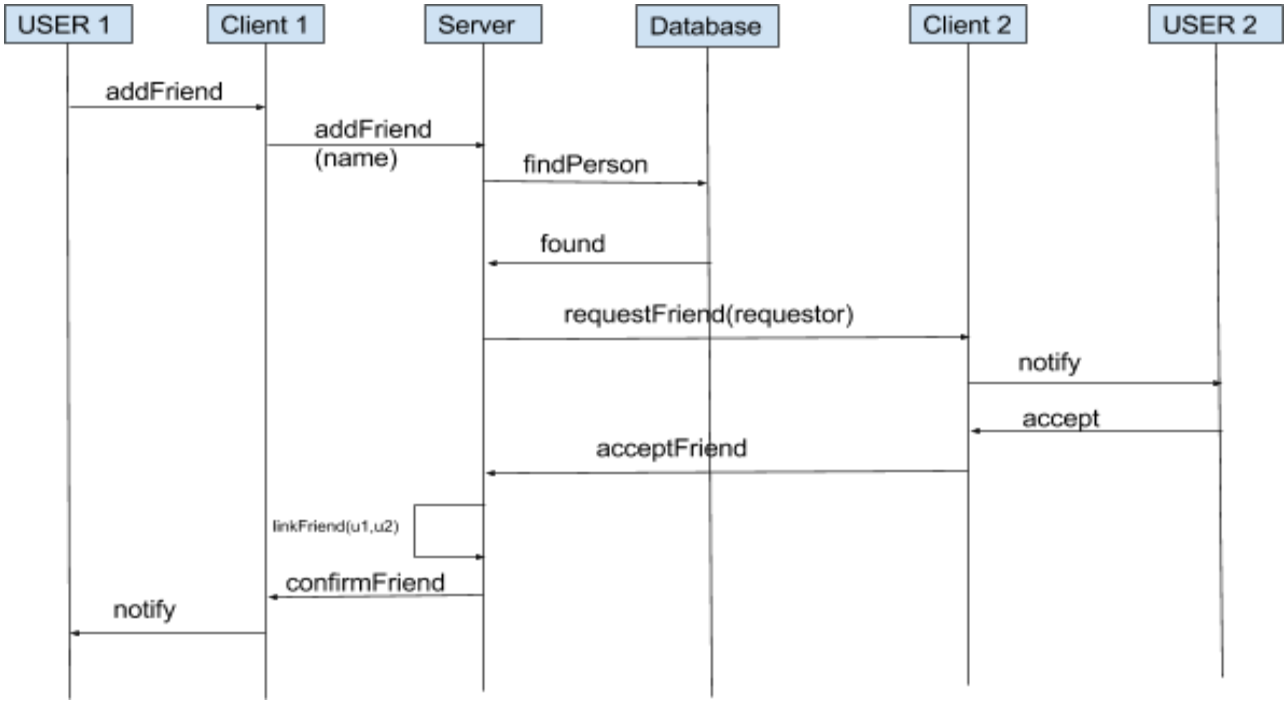
Project:

A project is basically a collection of the other classes. It will contain tasks, messages, and users.

Task:

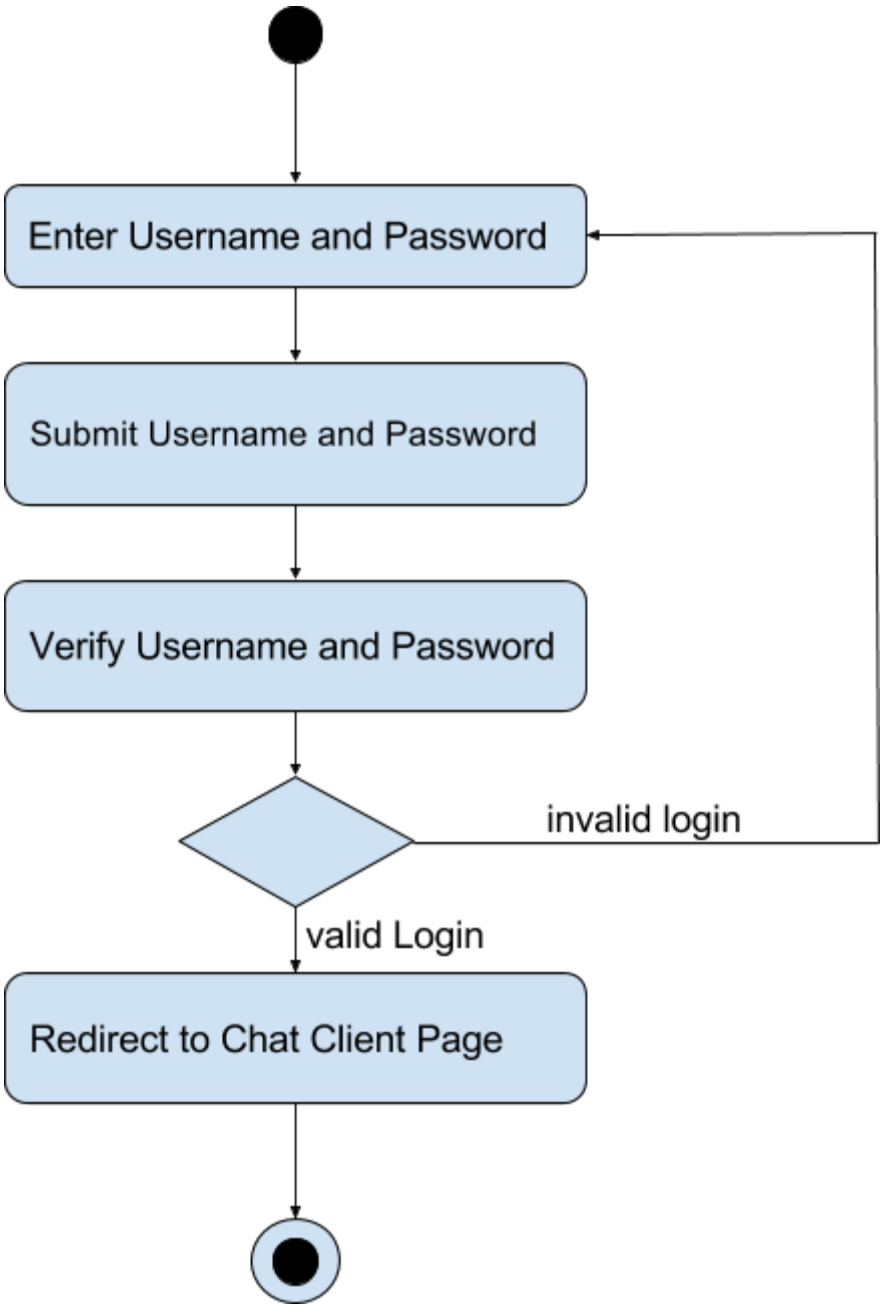
Tasks are created by users.

Figure 1. Creating a Project



This is a sequence diagram of how adding friend in TeamUp works. First, user 1 looks for his friend in his client, then notifies the server by the name. The server then look into the database to find the person. After it is found, the database notifies the server which will be passed to the client and notify user 2. When user 2 accept the friend request, the client will tell the server to link between them. Then the server will confirm it with notifying the user 1.

Figure 2. Login Procedure



This is the activity diagram for login procedure. The user needs to enter his or her username and password, then press the login button. Then the server verifies the username and password. If the username and password is incorrect, it will ask the user to enter the correct username and password. If the username and password is correct, it will redirect the user to the chat client page.