Group Project - 7CCSMGPR
Dr. Laurence Tratt

Development of a Chatting Platform

T. Avula, S. Bhatt, P. Bharti,
S. Mohan, F. Mosca, M. Smith

February 7, 2017

1 Project Description

1.1 Main goals

The project will deliver a distributed chat system consisting of a server and two clients that run in two different domains (e.g. mobile and web), in addition to an interim project report and a final project report.

Table 1 shows the mandatory features of the chat system. These ensure that the project meets the project requirements outlined in the specification.

Deadlines	**Mandatory Feature List**
18/02	Send text based messages to one or more users
18/02	Receive text based messages from one or more users
18/02	Secure communication between users
25/02	Search for other users on the server
25/02	User Registration (New User)
25/02	User authentication (Log in / Log out)
04/03	Chat client for mobile
04/03	Chat client for desktop

Table 1: Mandatory features

When these project features have been implemented, with accordance to time availability, the optional features outlined in Table 2 can be added.

Furthermore, one of the main aims is to employ rigorous testing. The test driven development approach is the project team's ideal approach to testing. The main testing strategies will be unit testing. As well as, functional, regression and integration testing of the clients and server software.

Optional Feature List	
Set Status	Emojis
Send Images or Files	Backup Chat / Chat Log
Set name / Profile Picture	Remove messages
Delete user account	Chat time stamps
Contact List	Customised Chat background

Table 2: Optional features

Additionally, the project aims to equip each person with the experience of working on a distributed software project as members of a team; also, to become au fait with some of the tools currently used in industry, especially in the areas of team collaboration and software versioning.

1.2 Implementation Strategy

Following the project initiation, the project team undertook background research to understand how to implement a distributed client server chat application. Some of the strategies investigated were Restful APIs, utilizing web frameworks such as Django and as well as socket programming. It was ultimately decided to employ sockets to develop the application, since the other methods employ several layers of abstraction. Socket programming allows the project team to implement network communication as close as possible to the operating system without rewriting the network stack.

The background research culminated in the development of a basic client server prototype written in python. This has enabled the project team to grasp how programming with sockets work and provides a very clear understanding of how to achieve the implementation requirements.

Moving forward the project team will employ an agile or rather its own incarnation of an agile software development approach. In the project team's planned approach, a subset of features to be implemented will be selected from the prioritized list of features on a weekly basis. These features will then be implemented, tested and documented, being ready for review at the next weekly team meeting. As the project progresses each week, new features may be added and some features deleted as the project team sees fit.

As noted earlier, the project consists of three components. The project team will be implementing the server in python as a multithreaded socket server. The first client will be implemented as a mobile application for the android platform and the second client will be developed as a web application. One of the planned approaches is to split the project team into sub-teams in order to implement feature sets for each component concurrently during the weekly coding sprints.

During each weekly coding sprint, the rational for implementation will be documented, since an agile approach is being taken. The documentation will also be updated each week to include updated UML diagrams, updated use cases and updated test cases. This documentation including the interim report will form the basis for the final report.

2 Project Organisation

2.1 Working together as a team

The forms of communication used between the team members are the following:

- An initial kick-off meeting was held where the scope and requirements of the project were discussed and the mandatory/optional features that can be built into the system were outlined.
- Physical meetings are held every Tuesday at 5pm. These meetings will involve discussing and showcasing individual updates to the project.
- Development meetings will be organised at periodic intervals. They will involve working together to code certain aspects of the system. These will be necessary especially when testing of the system features implemented is needed.
- Skype/virtual meetings will be organised on-demand and if needed between physical meetings, especially when any major issue needs to be escalated.
- There is a WhatsApp group for instant communication. This provides an on-demand, always available, easy-access chat platform for the group members to communicate with each other. Generally, meetings are organised in this group chat.

Each member will be responsible for coding, testing and code reviewing of the project. This means there will be no designated roles for each person in the team. However, certain members will be focussed on certain aspects of the project (i.e. certain people will work on the UIs and some people will be responsible for managing the server). The individuals designated to work on each feature are yet to be decided. The GitHub coordinator will be Sagar Mohan.

GitHub's feature branches will allow the team members to work on different parts of the system (e.g. two feature branches will be created for the UI of the system, one for the web client and one for the Android client); then, as each developer finishes working on said feature, that feature can be pushed to the central repository. Before this happens though, the developer working on the feature would need to submit a pull request, letting the rest of the team know that they are done working on that specific feature. Other developers in the team will receive the pull request and then they can decide to make changes if they wish, or integrate the feature into the project.

The use of this GitHub workflow would mean that conflicts can be resolved as the code can be reviewed by each person in the team.

2.2 Resolving conflicts

The final marks will be assigned equally.

Regular code reviews by at least two members of the group are planned before merging any code to the master branch of the GitHub repository. This would ensure that any major/minor bugs are avoided early. Also, through this, team members could keep a track of what others are contributing to the code.

In terms of conflicting ideas on the implementation and requirements of the system, open discussions will be held during meetings where an individual may raise any concerns/new ideas they would have, and a vote will be held where the majority consensus prevails. Few of the potential sources of conflicts could be:

- Difference of opinions on prioritization
- Technical and design disagreements
- Disagreements on schedule or timeline
- Lack of consensus on unified process methodologies.

In order to address such situations, few solutions that we expect to adopt are:

- Mutual cooperation and taking ownership of any proposed idea
- Having a consensus on the decision taken by the group and abiding by it.