CS3300: Introduction to Software Engineering

Project Plan for Team Chat

Version 0.0: 5/28/15

Team: Web Client

Table of Contents

1 Introduction

1.1 Overview (Executive Summary)

Team Chat is an application providing intra-team communication for projects. It is a client-server application, with a REST API for web, smartphone, tablet, and smartwatch client-side applications. Our team, within the Team Chat application, is the Web App team. In this document, the Web App team provides a description of our management structure, how we will manage risk, our schedule, and the technologies we will use. Highlights include our use of the agile software process, how we will maintain communication, our schedule, and the use of technologies AngularJS and express.

1.2 Definitions and Acronyms

Web App - Web Application
MVC - model view control
DOM - Document Object Model
CSS - Cascading Style Sheets
HTML - Hyper Text Markup Language
OOCSS - Object Oriented Cascading Style Sheets
SASS - Syntactically Awesome Style Sheets
npm - Node package manage

2 Management Structure

2.1 Project Life-cycle

We will mostly be using agile as our development method. We will keep our methodology mostly informal and in person where we meet weekly to discuss the status of the current sprint and any current issues.

The timeline of this project is laid out and can be broken down into three major phases. The first phase is design documentation. In this phase, we complete all relevant technical documentation as well as form a design for our project. The second phase of this project is implementation where we actually create our project. The final phase is integration where we integrate our project module with the rest of the modules to form the final application.

2.2 Roles and Responsibilities

Role	Responsibility
Project Manager	Hunter Brennick
Planning and Tracking Lead	Andrew Chow
UI Design Lead	Adam Beaton
Implementation Lead	Scott Vermeyen
Quality Assurance Lead	Andrew Chow
Development Engineers	All team members.
QA Engineers	All team members will QA their own code and other team members code.

2.3 Communication

In order to maintain contact, we have already created a Google Hangouts. All of our team members have already communicated using this hangout. As a fallback, we have each other's emails and all attend classes regularly. With these 3 mediums of exchange, we should be able to maintain contact with each other.

Additionally, we plan to meet once a week (if necessary) to coordinate, plan, make decisions, test, and integrate software when necessary. We will meet on campus.

3 Risk Management

3.1 Risk Identification

Risk	Probability	Severity	Description
Requirements changing	Low	High	Would demand a front-end refactor.
API changing	Low	Med	API changing midway through the project. Would demand front-end refactor.
Blocked by API	Medium	Med	If this happens and we get blocked on the front-end we will have to take the time to implement a temporary local endpoint to buy us time.
File Upload	Low	Med	Could be tricky depending on how supported by API.
Other course's workload	High	Low	All of our team members are taking multiple classes, and often this leads to team members getting inundated with work from other classes sporadically.

3.2 Mitigation Plan

Be AGILE. Essentially implement in a highly modularized so if requirements change, API changes, etc we can isolate the refactor to a specific module. This will significantly reduce refactor costs.

4 Planning and Control

4.1 Milestones

- May 14, 2015: Initiate Project
- May 19, 2015: Project kickoff meeting
- May 26, 2015: Project Feasibility Finalized
- May 28, 2015: Project Plan Finalized
- June 2, 2015: Requirements Finalized
- June 4, 2015: Project Design Finalized
- June 9, 2015: Project Test Plan Finalized
- June 16, 2015: Module Finalized - Begin Integration With Others
- June 25, 2015: Go live with full app

4.2 Work Breakdown Structure

User Stories that represent assignable tasks

1) As a user, I want to be able to log in
2) As a user, I want to be able to search for another user
3) As a user, I want to be able to select a user and send him/her a msg
4) As a user, I want to be able to see previous msgs and reply
5) As a user, I want to be able to create a group and edit its members
6) As a user, I want to be able to send msgs to a group
7) As a user, I want to be able to upload/share docs in a chat
8) As a user, I want to be able to view docs from a chat
9) As a user, I want to be able to delete one of my groups
10) As a user, I want to be able to remove someone from my group
11) As a user, I want to be notified when I receive a msg
12) As a user, I want to be notified when I receive a group invite
13) As a user, I want to be able to edit my profile

4.3 Schedule

A more detailed schedule/chart can be found in the ProjectPlanSchedule excel file.

4.4 Tracking and Control

After a task is completed, we will look over the code and check for any bugs. We will always keep a stable/working version backed up while moving forward. As for tracking time spent in development, we will keep a log of our time spent on certain tasks using our schedule chart. During status

updates, we can just simply review the chart and see the progress everybody has made. From there, we can discuss what needs to happen next and prioritize tasks.

5 Technologies to be used

IDEs

- intelliJ / webstorm
    - intelliJ required, webstorm by choice

Markup

- HTML5
- CSS3

JS frameworks

- jQuery (jLite) - AngularJS dependency
- AngularJS - high level MV* framework
    - Really clean DOM manipulation.
    - Highly supported community.
    - Group members with experience.

CSS pre-processors

- SASS or LESS - for OOCSS
    - OOCSS promotes modularization and code re-usability.

JS tools

- npm - package manager
    - A package manager furthers project implementation efficiency.
- bower - package manager
    - Because npm won't support all the JS packages we will want to use (potentially).
- express - web app framework
    - Will be used for local development primarily (local HTTP server).
- grunt - task runner
    - To support different development environments.
    - Potentially will be used builds, minification, uglification, and a variety of other tasks.
- yeoman - scaffolding our web application

Version Control

- git - required