CS3300: Introduction to Software Engineering

Feasibility Study for Team Chat

Version 0.0: 5/26/2015

Team: Web Application

Table of Contents

1 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 in Team Chat is the Web App team. In this document the Web App team provides a description of the services provided by the web client piece, including authentication, group formation, message storage and delivery, and the transfer of documents. Like all other client-side apps in the Team Chat application, the web client will interact with the REST API. The Web App team in particular will use Javascript frameworks AngularJS and ExpressJS, Object Oriented CSS language SASS, and tools like Bower and Grunt in order to develop the web client. Because the Messaging application space is already saturated, we will try to find a niche with a simple team communication application and focus on that segment of the market. In order to build the client on time, we define a preliminary schedule. Finally, we recommend that Team Chat approves the web chat client proposal and begins project development.

2 Description of Products and Services

Team Chat aims to create a chat application that functions concurrently across different devices. Specifically, this chat application will provide a simple interface in which all users have easy access to through their phone, smart watch, tablet, or any other device that can access a web browser. This chat application will support chat rooms, private messages, and the ability to share files. The main goal of this project is to make it as consumer friendly as possible by providing solutions for most modern devices.

3 Technology Considerations

Given there is no current existing code base for the web client team to build on within Team Chat, the web client team's front-end technology stack decisions comes from a place that won't have to consider legacy architecture. Without the burden of a legacy architecture, the web client team is able to hand-select a technology stack that will allow us to deliver their module in the most efficient, reusable manner possible. We will build a Web Application that will function as our web client; the Web App will have to be supported by the majority of the web browsers - preferably IE9+.

First, we recognized several main JavaScript tools that would be required for the Web App: a "high-level" MVC JavaScript framework (Angular, Ember, GWK, Knockout, etc.), a JavaScript package manager, a JavaScript task runner, a JavaScript scaffolding tool, and a JavaScript web framework for local development.

The JavaScript tools we chose for the Web App piece of Team Chat are based on a balance between our teams collective level of experience with the tool and the strength of the tools acceptance in the web development community. Without getting specific on the decision behind each JavaScript tool, we have chosen the following tools: AngularJS as

our MVC JavaScript framework of choice, bower and npm as our JavaScript package managers, GruntJS as our task runner, Yeoman as our scaffolding tool, and ExpressJS as our web framework for local development.

Continuing with our emphasis of reusability and modularity, we recognized that we need a CSS extension framework that will allow us to style our Web App in a much more efficient way utilizing Object-Oriented CSS (OOCSS). This decision really comes down to two frameworks: SASS and LESS. We have opted with SASS on this decision because of our team's experience with the framework.

Because all five Team Chat modules operate in completely independent silos, our front-end technology decisions for the web client module bear no impact on any other module. All other front-end modules communicate via Team Chat's REST API. The REST API defines the contract between the front-end modules and the back-end module.

4 Product/Service Marketplace

Currently, there are many applications that facilitate intra-team communication. In many workplaces around the world, people use email as a primary means of communication and document exchange. Many other applications are being used in the modern workplace, including Yammer, Pie, Unison, Basecamp, and Zoho. Many companies use services like Dropbox, Box, and Google Drive to exchange documents.

Students, in addition to using some of the same technology mentioned above, also use services such as Facebook Messenger, Facebook Groups, WhatsApp, GroupMe, SMS, Google Docs, and Google Hangouts to get organized, communicate, and share documents.

Although this market is heavily saturated, we hope to create an application that is better suited to the specific needs of students, rather than for the corporate or social world. In this way, we can find a niche for the Team Chat app.

5 Marketing Strategy

For Team Chat to make an impact in this market, our application has to bring something new to the table. To accomplish this, Team Chat will design the most user-friendly app available for group coordination purposes. Current competitors offer systems that make even simple document sharing overly complicated. Team Chat users will have the ability to engage in normal one-on-one conversation and even private group discussions.

Team Chat will implement custom group creation in order to keep users and their discussions organized. Additionally, Team Chat will allow document sharing between users. This will appeal to users working in group projects needing to edit project-critical documents on the fly. Team Chat will also maintain a user database in order to organize

its target customer groups. Another important consideration of Team Chat's online distribution is profit. Electronic distribution costs are relatively low, and our marketing gains will outweigh them.

It is important to note that Team Chat does not currently have a marketing team.

6 Organization and Staffing

N/A

7 Schedule

Team Chat's development process is expected to take approximately one month.
The following is a high level schedule of some significant milestones for this application:

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 Integration
June 25, 2015: Go live with app

Upon approval of this project, the assigned project teams will create a detailed schedule to include all tasks and deliverables.

8 Financial Projections

N/A

9 Findings and Recommendations

Based on the information presented in this feasibility study, it is recommended that Team Chat approves the web chat client proposal and begins project development. The findings of this feasibility study show that this application will be highly beneficial to the organization and has a high probability of success. Key findings are as follows:

Technology

- Will utilize a modern technology stack that promotes efficiency and reusability.
    - Absence of legacy code allows decisions to be made without previous architectural knowledge.

- Reusable components will be integrated to keep our system structure neat, easily maintainable and scalable.
- Will implement each module independently in order to keep module-specific functionality from impacting other modules.

Marketing

- This application will allow Team Chat to reach large number of target users.
- Team Chat can expand user base easily through electronic distribution.
- The marketplace for online communication and document sharing is continuing to grow at a steady pace.

Organizational

- Minimal staffing additions are required.
- No new facilities or capital investments are required.