CS3300: Introduction to Software Engineering

Requirements for CS 3300 Messaging App

Version 1.0: 6/2/15
Team: RESTful API

Table of Contents

1. Introduction

1.1 Overview (Executive Summary)

This document outlines the requirements necessary to host a RESTful API for a real-time, messaging application supported by four clients. If the reader is unfamiliar with web servers, refer to section 1.2 for acronym definitions. Our service will provide the back-end support for the web, Android, and smart-watch clients. The API being developed will give clients access to a web service that can store or transfer their information and uploads. In addition, it will support the integration of four interactive bots. The messaging application will rely on the API for managing the information it needs in order to run. The architectural properties of a RESTful API will provide clients with an easy to use system for naming their resources. No server-specific methods will be needed to refer to the server's resources. This assures that any client that uses HTTP can communicate with our service.

1.2 Definitions and Acronyms

HTTP: Hypertext transfer Protocol
RESTful: Service that accepts http requests (Representational State Transfer)
GET: Requests a resource from the server. (Web page or image)
POST: Similar to get but places additional info is sent in the body of the request.
PUT: Upload a resource to the server
DELETE: Delete a resource from the server
Spring: Java Developmental Framework
IntelliJ: Interactive development environment (Will be used with Spring)
Maven: Software project management tool
Jackson https://github.com/FasterXML/jackson : Java JSON parsing library
URI: Uniform Resource Identifier
status codes (Ex: 404, 500) - responses to network requests
JSON: Javascript Object Notation.
API: Application programming interface. Specifies how software components should interact
Slack: The communication method set by the team. This application will help us organize team tasks and documents.
GroupMe: The communication method set by the team. GroupMe is an application available on mobile and web platforms.

2. User Requirements

2.1 Software Interfaces

MySQL Server 5
MySQL
Version 5.6.22
The system must use SQL Server as its database component. Communication with the DB is through JDBC connections. The system must provide SQL data table definitions to be supplied to the other teams (tablet,watch,web...etc.) for setup.

Amazon Web Service
AWS
The API being developed will use AWS to host it's web service and SQL database. Amazon will securely store all of our data, as well as deploy our application.

Spring
Version 1.1.2
Spring is a collection of libraries used in the source code of the project, so it has intimate knowledge of our application's data and source code.

Maven
Version 3.3.3
Maven will use an XML file to build the project. The XML file will have information about the project's dependencies, build order, package architecture, and required plug-ins.

2.2 User Interfaces

Our users (client-side teams) will enter URIs to place information in our server or retrieve information from our server. There will be no graphical user interface. Our users can prompt the server for information via command line if necessary.

2.3 Product Functions

The product will utilize HTTP methods to transfer resources. This can include text, images, or pdf files. The commands used to for this method are defined below.

GET: Users issue GET commands to get information from the server
DELETE: Users issue DELETE commands to delete information from the server
POST: Users issue POST commands to post new information to the server or update existing information

2.4 User Characteristics

Our users will be the developers of the front-end clients for the messaging application. This would require a higher-level of education and technical expertise. Because of this, we are able to develop of a higher-level service. We will use JSON formatting for our responses and requests. JSON is a common data format used across many APIs that can be easily integrated into all our users applications.

2.5 Assumptions and Dependencies

Based on the user characteristics, it is assumed that users will be able to send and receive JSON requests and responses. If a team cannot send/receive with the JSON format, we will need to provide a different data format option, such as CSV.

2.6 Apportioning of Requirements

The time available for fully developing the API is minimal. Although bots add a unique feature to the application, they are a non-essential function for the system. For this reason, bot integration may be delayed in order to ensure that more important features are implemented in time.

3. System Requirements

3.1 Functional Requirements

- The system shall take information from GET requests and return the corresponding data from the server
- The system shall take information from DELETE requests and remove the corresponding data from the server
- The system shall take information from PUT/POST requests and update the server with the information
- The system shall correctly handle authentication

3.2 Non-Functional Requirements

3.2.1 Software Quality Attributes

- The system shall always be up and running
- The system shall keep messages and conversations private
- The system shall have enough space to handle all its users
- The system shall scale to the size of its user base
- The system should have performance on par with competing messaging services

3.2.2 Other Non-functional Requirements

N/A