Web Programming – Final Project

Winter 2014-15 CS@Haifa University

Course Instructor: Dr. Haggai Roitman, IBM Research - Haifa

Publication date: 1/1/2015, Deadline: 28/2/2015, Version: 1.2

Project scope: during the project you will explore, design and implement a web application that is built on top of the various client-side and server-side technologies that were thought during this semester.

Project general description: the goal of this project is to implement a simple microblogging service (http://en.wikipedia.org/wiki/Microblogging) – a popular online social communication form on the Web. Popular microblogs such as Twitter, Tumblr, FriendFeed, etc., serve as an important media for knowledge exchange, situation awareness, information discovery and social online interaction.

The microblogging service, to be implemented in this project, would allow various users to chat together via simple user message exchange. Updates to various "user discussions" may be refreshed in the user view in a "real-time" manner. Users can post messages to newly started discussions or reply (via comments) to existing ones. Users may have the options to discover discussions, contribute updates of their own or comment. A user may decide to follow discussions made by other users. User popularity in the system is derived from such "followership", where the more followers a user has, the higher is her popularity.

Outline

The followings include description of the project structure (requirements, guidelines, tips, etc.).

- **Functional requirements** are a set of features that you are expected to implement during the project. **Please make sure you have covered all requirements.**
- **Implementation requirements** are a set of requirements that define what web technologies you are allowed to use and which are a must. In addition, general development guidelines are provided. Please pay attention.
- **Design and UX requirements** are a set of requirements about the UI design of your application and its user experience. Please pay attention.
- **Deployment and Environment requirements** are a set of requirements for properly preparing your project for submission. Please pay attention.
- **Bonus requirements** are a set of requirements that you need to fulfill to deserve an extra credit in your project (**up to 10 points can be earned in final grade! Final project grade can't be higher than 100, though**☺). Project submissions that will not fulfill the basic requirements will not be given a bonus. Therefore, don't waste your precious time before you finish those requirements.
- **Submission guidelines** are a set of guidelines of what should be submitted, in what format, using what naming convention and where to submit, etc**.** Please pay attention.
- **Tips** are a set of guidelines that should "ease" your life. Such tips are partially given throughout this document, but some others may pop-up as you shall work on this project. Don't ignore those tips, keep tracking for updates at class or in the course online forum.
- **Grading guidelines** is the approximate grading scheme that will be used to judge the quality of your work in this project. Please pay attention.
- **Project integrity guidelines** is a set of integrity rules about this project; basically, a list of "do and not do" (especially to make sure your work is genuine). **Integrity guidelines will be strictly enforced**. Please pay attention.

Functional requirements:

The following is the list functional requirements that your microblogging application should support **(use this requirements list as your check-list; make sure you have covered each one and one of these requirements!)**

a. A user should log into the system using a username and password
b. In case the user doesn't exist, the user should be able to register to the service.
    i. The user will provide the following details for registration:
        1. **Username** (required, up to 10 characters) – username represents the user internally in the system.
        2. **Password** (required, up to 8 characters)
        3. **User nickname** (required, up to 20 characters) – nickname is a public name that will be displayed to other users in the system.
        4. **Short description** (optional, up to 50 words).
        5. **Photo** (optional, size 50x50) – link to photo should be provided by **URL**. This photo will be linked in the application. **No need to support image upload from the user file-system!**
    ii. Upon registration, the user will be shown the main application's UI.
c. Registered users will be shown the main application's UI.
d. The UI will allow the user to perform the following operations:
    i. Discover users/interesting discussions
        1. The discover option is the most basic option a user has. The user can use this option so to discover other users or discussions which the user may be interested to follows or participate in.
        2. Upon selection of this option, the ``discover" UI will show the user a list of 10 top latest messages of any kind, posted by any other user (i.e., messages made by the current user should not be displayed in this UI option).
        3. The user will have an option "**Display only messages of users who I follow"** to see only messages from user who she follows.
        4. Messages to be displayed will be chosen according to the following formula: *log(2 + number of followers of message*

*author user) *log (2 + number of times the message was republished).*

5. New popular messages posted by users should be reflected in the UI in a continuous "real-time" manner. Therefore, if a new popular message was posted and its popularity is above the message with the lowest popularity in the current displayed list, the later should be replaced by the new message and displayed to the user.

ii. Post a new message

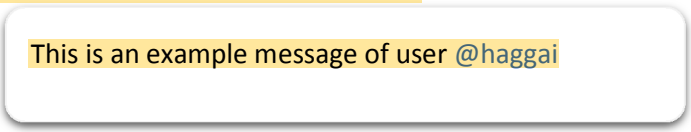
1. Each message length should be no more than 140 characters.
2. A message that starts with **@** would count as a **reply**.
3. A word within a message that is prefixed with **#** would count as a discussion **topic**.
4. A word within a message that starts with **@** (yet that word is not the first one in the message) would account as a **mention** (i.e., reference to other user).
5. To be account as a reply/mention, word that starts with **@** should be followed by a legal user nickname in the system (e.g., @haggai); otherwise, the application will treat such word as a regular text word.
6. Mentions and Topics text should be highlighted in posted message when displayed to the user after the user submitted the message to the application (e.g., this is a **#topic**, wow…).
7. A newly posted message will be displayed to the user at the top of the current list of displayed messages.
8. A newly posted message will be displayed to the user with the following details and options:
    a. Time message was posted using the label: "*[time] ago*"
        i. If the message was posted within the last minute, the time to be displayed is in seconds (e.g, "40 seconds ago").
        ii. If the message was posted within the last hour, the time to be displayed is in minutes.
        iii. If the message was posted within the last day, the time to be displayed is hours (e.g., "today, 4 houts ago").

iv. If the message was posted more than a day ago, the exact message time will be displayed in dd/mm/yyyy hh:mm format (e.g., 1/1/20014 at 10:04).

b. Name of the message author (i.e., the user nickname in format [usernickname]) .

c. Message content.

d. Example:



iii. **Reply to a message**, by pressing "**reply**" option next to any given message.

1. The replied message content, will start with the user nickname of the user that her message has been replied (e.g., *"@haggai I think bla bla.."*).
2. A newly replied message will be displayed to the user at the top of the current list of displayed messages.

iv. **"Republish"** a message of another user by pressing "**republish**" option.

1. The republished message content, will start with the prefix "RE:" followed by the original message content (e.g., if original message is *"WHAT a nice day..",* a republished message content should be: *"RE: WHAT a nice day..").*
2. A newly republished message will be displayed to the user at the top of the current list of displayed messages.

v. **"Follow user":** the user will be able to choose to follow messages made by a certain message author by pressing a **"Follow"** option.

vi. **"View user profile"**

1. Each displayed user nickname (e.g., @haggai) should be clickable.
2. Upon click, the **User Profile View** will be displayed and include the following details and options:
   a. User profile details.
      i. User photo (or show some default photo if none was provided).

ii. User nickname.
iii. User's Short description.

b. Last 10 messages published by that user.
c. Number of followed users "**Following"** by that user.
    i. A click on "**Following**" will display the list of top-10 users that are being followed by this user. The 10 users will be chosen for displayed based on their "popularity" in the system. User popularity is defined by *log(2 + the number of followers)* that the user has.
    ii. For each user in that list, the current user will have the option "**Follow**".
d. Number of user followers **"Followers"**
    i. A click on **"Followers"** will display the list of top-10 users that follow this user. Again, follower users are chosen by their popularity (defined the same as above).
    ii. For each user in that list, the current user will have the option "**Follow**".

vii. **"View topic":**
1. Each displayed topic (e.g., #topic) should be clickable.
2. Upon click, the **Topic View** will be displayed.
    a. The list of 10 **last messages** (i.e., according to their creation time) that included topic #topic in their message content will be displayed.
    b. The list of messages will **auto-refreshable**
        i. Upon new message in the system that was posted by some user in the system that contains this topic, the UI will notify the user that new related messages have been posted.
        ii. The user will have an option to request to view the new added messages by clicking **"See new updates"**.
            1. The view will add the new messages to the list of 10 last displayed messages. I.e., older messages will be replaced.

Implementation requirements:

1. You are only allowed to use web technologies that were taught during the semester.
2. You should use HTML 5 (with UTF-8 encoding) and CSS 3 (most browsers support both). You are allowed to use advanced HTML 5 or CSS 3 features as you wish.
3. Keep correct usage of the technologies, i.e., content should be given in HTML, style and layout in CSS (or Bootstrap), logic should be located in JavaScript, Servlets, data in the database, etc.
4. Don't define inline or embedded style rules. Instead, define such rules in a linked CSS file.
5. You are not allowed to use any third-party (or open source) libraries of any kind for the server side code; except those required by Eclipse Luna, Tomcat and Java Servlets (the later should be included in Eclipse Luna J2E distributions).
6. You are not allowed to use any third-party (or open source) libraries of any kind; especially JavaScript libraries (except for the **basic** jQuery, Bootstrap and AngularJS libraries).
7. You are not allowed to use any other J2E server-side technology except for regular Java code, Servlets and JDBC features (e.g., don't use JSP, EJB, JSP-Tags, etc.).
8. All communication with the server should be done using a **RESTful** API. POST messages should contain parameters sent to the server within a JSON format.
9. Data transferred from the server to the client and backwards must be in JSON format. For server side JSON java utilities use: https://code.google.com/p/google-gson/
10. Mark your servlets with a `SingleThreadModel` interface.
11. You should only use the Apache Derby open source Database.
12. Use proper coding:
    a. Follow Java code conventions (see http://www.oracle.com/technetwork/java/codeconventions-135099.html)
    b. Follow JavaSscript code conventions (see http://javascript.crockford.com/code.html)
    c. On client side, document with `<!--comments -->` your HTML code when needed for clarity.

d. Use //comments in JavaScript to document your code
e. Use /*comments*/ in CSS to document your code
f. Use Javadoc documentation guidelines:
    i. Use /**….*/ for public (user) comments
    ii. Use /*…*/ or //… comments for developer comments
    iii. Document method input parameters (*@param*) , method outputs *(@return*) and exceptions (*@throws*)

Design and UX requirements:

1. You client-side UI should be completely developed with Bootstrap.
2. UI must be responsive.
3. You are not allowed to use any additional (third-party or open source) UI features.
4. You are not allowed to use ready templates. **Usage of such a template will be easy to trace, and your project will be disqualified**.
5. **Usability**: make sure your application correctly follows the usability guidelines we have learned during the semester (e.g., make sure images have <alt> tags; fonts, colors, etc. are properly chosen, etc.).

Deployment and Environment requirements:

1. The following third-party/open source libraries should be used in your project for development and deployment. **Do not use any other code/software packages etc.**
    a. Java 7 JRE http://www.oracle.com/technetwork/java/javase/downloads/jre7-downloads-1880261.html
    b. Eclipse IDE (Luna) for J2E: https://eclipse.org/downloads/packages/eclipse-ide-java-ee-developers/lunar
    c. Apache Tomcat v.7: http://tomcat.apache.org/download-70.cgi
    d. Bootstrap: http://getbootstrap.com/getting-started/#download
    e. jQuery: http://code.jquery.com/jquery-1.11.2.min.js
    f. AngularJS: https://ajax.googleapis.com/ajax/libs/angularjs/1.2.28/angular.min.js

   g. Apache Derby: http://db.apache.org/derby/releases/release-10.11.1.1.cgi
2. You may want to use the JUnit tool for testing your java code: http://help.eclipse.org/luna/index.jsp?topic=%2Forg.eclipse.jdt.doc.user%2FgettingStarted%2Fqs-junit.htm
3. **Make sure your application pass W3C Validation**: http://validator.w3.org/

Bonus requirements:

The following are four options to earn extra credit to the final project grade. You may choose to implement up to two out of four such options.

1. **[5 bonus points] Real-time search:** support a real-time search feature over messages, topics and users. You are free to decide how to implement such feature and what will be the UI and user experience for that purpose. The only requirement is that search results will be rendered in online fashion (see how Twitter search works for example).
   a. Use Lucene's real-time indexing capabilities for the implementation.
   b. You should use Lucene version 4.10.X
2. **[5 bonus points] Visualization:** using the Data-Driven Documents, D3 framework (http://d3js.org/) implement a visualization that can be useful for users in such a microblogging setting using the current application you have developed.
3. **[5 bonus points] Personalization**: using the Apache Mahout TASTE package, implement a simple personalization service (e.g., recommender system) for your application. https://mahout.apache.org/users/recommender/recommender-documentation.html
4. **[5 bonus points] Creativity track:** you are encouraged here to suggest a cool feature of your own! Be creative!

Project submission guidelines:

Due date (NO EXTENSIONS!): 28/2/2014

You will need to submit the following material enclosed inside a directory named "**webappproject-ID1-ID2**", where ID1 and ID2 are the student ids of the submitting team. Further zip your submission (using only **.7z** or **.zip** compressions!).

Your submission should be sent to haggair@gmail.com with the title: "*web proj. sub.*" Submissions will get confirmation from me upon receive. **You may submit only once**.

1. **Students.txt**: will include details of your **ids** and **names**.
2. **ERDdiagram.ppt:** a power-point (or image file) that will include the ERD model of the data entities and relationships of your database.
    a. **Please keep the rules of ERD that were taught in class.**
3. **DBSchema.sql**: all SQL DDL (i.e., CREATE TABLE…) and SQL (i.e., SELECT …) commands that you have used for creating and manipulating your Derby database. Please document any assumptions you made on the data.
4. **webapp.zip or webapp.7z:** an archive file (**in .zip or .7z compression file formats only**) of your project that was developed in Eclipse.
    a. **Java code should be well documented using Javadoc commands** (i.e., General description of classes, methods, @param, @return, @throws, etc. No need to documents getters and setters.)
    b. Your project archive file should contain all resources (i.e., static web pages, .js, .css, .class, web.xml, java sources, etc) files that your web project depends on for proper running. Use eclipse **Export to Archive File** option for this purpose. DO NOT SEAL THE PACKAGE. I WILL IMPORT IT INTO MY ECLIPSE ENVIROMENT DURING PROJECT GRADING.
5. **javadoc.zip:** the Javadoc of your project. The following instructions show how to generate a Javadoc for your project in Eclipse: http://help.eclipse.org/juno/index.jsp?topic=%2Forg.eclipse.jdt.doc.user%2Freference%2Fref-export-javadoc.htm

Project grading:

The following grading guidelines will be used to evaluate your project (max project grade is 100):

1. Requirement fulfillment: 70-75%
2. UI and UEX (Usability): 10%
3. Documentation: 5%
4. Proper coding style: 5%
5. Creativity: 5-10%

Project integrity guidelines:

1. Project must be prepared in pairs only.

2. Teams should not copy or share code between eachother.
3. Code or template usage from third-parties or open source is completely forbidden.
4. **Failing to follow this basic guidelines will disqualify your project**.

Last and for all:

We are here to enjoy what we have learned. This can be only done through "diving into the deep water". If you get stuck, have questions, etc., try to utilize the following resources:

1. Online communities: I warmly recommend to post questions/look for answers on StackExchange, CodeAcademy and Quora.
2. General topics may be discussed in the course forum, **though you are not allowed to share code through this medium**.
3. I will devote 15 minutes at the beginning of every lecture to discuss issues related to the project.
4. If you had some important assumption you have made during development, document it or ask me for confirmation.

GOOD LUCK!