Executive Summary

PaperCloud is an internet application on the world wide web, built for the purpose of organizing words from different researchers' publications into a word cloud. The publications come exclusively from IEEE and ACM. This application will allow for the generation of 2 types of clouds and as such, two types of searches for each cloud. The first is to enter the full name of the researcher. PaperCloud will then generate a word cloud (if the input is valid) based on all the MaxWord most common words from the researcher's papers. The second search option allows the user to enter a keyword phrase which will result in a wordcloud consisting of the MaxWords most common words from all the documents searchable (from the two publications mentioned) containing the keywords. PaperCloud offers certain other features to simplify further searches and word cloud generations which are described in the rest of this document.

1 Introduction

1.1 Purpose

The purpose of this document is to clearly describe every aspect of the second group project for the CSCI-310 course taken at the University of Southern California in the Spring semester of the year 2015. The intended audiences of this software requirements specification document includes, but is not limited to: the members of group six whose names are listed on the front page of this document and by whom this document is prepared, the professor of the course Dr. William G. Halfond, and the teaching assistant Sonal Mahajan. Other possible audiences include: students who are at the time of this publication taking the course in which this project is assigned or any future students of this course who may read this document should it become available to them by the course instructor Dr. William G. Halfond or by any other means which cannot be predicted at this time. This software development project will hereafter be referred to as PaperCloud. It is intended for any user with an internet connection-either through a mobile or stationary platform.

1.2 Overview

This document is to present the management and development structure of the first iteration of software under construction. The different processes are explained in detail in the following sections. Furthermore, the project serves as guide on how to build software using the SCRUM and Extreme Programming practices of the Agile collection of processes.

The content regards the two Agile collection development processes described above. Section 2 elaborates on our use of SCRUM which includes our project

management plan and organization. Section 3 and 4 details our requirements, design, and implementation planning. Section 5 describes the review process in which we reevaluate the effectiveness of our strategy and implementation. The Appendix includes some artifacts and further details on our development process as a whole. ## 1.3 References

[1] Cunningham, Ward. "Extreme Programming". O'Reilly Media. July 2003.

[2] Wells, Don. "Extreme Programming". October 10,2013. April 5, 2015.http://www.extremeprogramming.org/.

1.4 Terminology

IEEE- Institute of Electrical and Electronics Engineers ACM- Association of Computing Machinery Development team (use same as before) SCRUM XP- Extreme Programming Asana- Planning and time tracking software available on the internet. arXiv- application provide by Cornell university that helps with searching for research papers across a wide range of subjects. Gmail- Email messaging software from Google Inc. Used for communication purposes by development team Clients/Customers - Professor William G. Halfond and Sonal Mahajan

1.5 Scope

PaperCloud is a web based application for generating a horizontally positioned text word cloud generated from the most commonly occurring words from a collection of research papers. There are two search options each correlating to two different clouds that can be generated. In the first option, the cloud is generated based on the first and last name search for a particular author. The second search option allows the user to enter a keyword phrase which results in a word cloud generated based on words from all articles that contain the keyword phrase.

PaperCloud will be hosted and available to the World Wide Web and will require no user registration or membership. To access the PaperCloud service, a user only needs an internet connection and one of the commonly used web browsers. PaperCloud arXiv API to generate the results from

1.6 Documents

We used certain third party software products to assists with project management and implementation. Information for access to these products can be found below. Asana is used to keep project backlogs and iteration related scheduling and task completion details for each sprint. Each member of the development

team will chose tasks from the sprint log to complete and can interact with other member and update information about task progress on Asana.

For full access to SCRUM meeting notes refer to the Group 6 discussion board on the course BlackBoard link under the "Iteration 1 Team Activity Log" thread. For a quick reference, refer to the appendix Section 6.1.

For full access to Asana, the clients are suggested to refer to their University of Southern California Gmail account where they will find an invitation to view PaperCloud's Asana account.

For full access to source code, please refer to the project's public Github page at https://github.com/C-Lyrics/PaperCloud

2 SCRUM Management

2.1 Process Introduction

The primary purpose of using Scrum is to work together to develop the product required from the client. In doing so, we broke up the requirements created in the back log. This way, we were able to prioritize tasks for the sprint log to better understand which tasks needed to be completed before others. By building the product in smaller broken down pieces, the development team was able to give feedback and changes if necessary.

The Scrum process is comprised of three different procedures: pre-game, mid-game, and post-game. The pre-game process involves engaging in planning and high-level design. This included elicitation of the requirements from the client, and the Scrum spring cycle. This comprised of the delegation of the three main roles, such as Scrum master, Product owner, and team, and organizing meeting times with the development team. After clarifying the requirements, the development team created the product backlog and sprint log. Every member present at each meeting that occurred answered the workday questions to make sure the everyone was on track for task completion. The mid-game process involves developing, wrapping, reviewing, and adjusting. This is primarily abiding by the sprint logs that were created from the backlogs. We developed and wrapped code, reviewed code by pair programming, and make adjustments accordingly. The post-game process involved closure of the first iteration, which includes the sprint review and sprint retrospective.

2.2 Organisation Planning

2.2.1 Roles

The group divided the roles in the following manner:

- SCRUM Master: Justine
- Product Owners: Sonal Mahajan
- Customers: Sonal Mahajan, William Halfond
- Development Team: Sebastien, Mark, Milad, Justine, Jeff, Kelsey

Justine was chosen by consensus to be our SCRUM master because she would be available to attend all meetings and she has had experience using SCRUM techniques.

2.2.2 Teams

The teams are allocated as the following:

- Frontend: Sebastien, Kelsey, Justine
- Backend: Mark, Milad, Jeff

These teams were chosen based on the team allocations we used in the previous project, as well as each group member's individual coding knowledge and experience. We felt that the division of frontend and backend made sense as we could assign requirements for implementation on the frontend Javascript and HTML code and the backend PHP code.

2.2.3 Product Backlog and Sprint Logs

Asana was used to create a product backlog of project requirements that were determined based on meetings with the customer. From this product backlog, we narrowed the list down to a sprint log for our first iteration of code. The sprint log was further broken down into implementable tasks that were outlined on Asana, which allowed us to track the development progress of our first sprint.

- Product Backlog
  - Search by keyword
  - Search by researcher
  - Autocomplete for searching by researcher
  - Ability to see history of searches
  - Display word cloud with common words from top N pages
  - Progress bar for progress of generating word cloud
  - Click on word in cloud lets you make new search with that word

  - Click on word in cloud lets you see all research papers with that word in it and how frequently the word appears in each one
  - Display list of research papers that contain selected word in them with their authors listed next to them in MLA format
  - Click research paper title takes you to a page with a link to download the paper
  - Click author name in research paper list takes you to word cloud generation of common words from that author's research papers
  - Select research papers to export references in .txt and .pdf format
  - Ability to get BIBTEX reference of each research paper upon hover/ button press
  - Navigation buttons between all pages

- Sprint Backlog: Because the product owner was not present at our sprint planning meeting due to meeting logistics, we were not able to produce a sprint backlog during this meeting. However, we requested a prioritized list of requirements and were able to create the sprint backlog laid out below at our first scrum meeting on March 27th.

  - Search by researcher: implement researcher search function, make API call with researcher name
  - Search by keyword: implement keyword search function, make API call with keyword, parse PDF's with PHP, get abstract from API
  - Display word cloud with common words from top N pages: implement function to gather common words from top N pages, display word cloud
  - Progress bar for progress of generating word cloud
  - Click on word in cloud lets you make new search with that word: make wordcloud results clickable, run search function with new keyword, make API call with new keyword

@TODO add screenshot of asana

2.3 Meetings

Our team had a sprint planning meeting on Thursday March 26th in which we set up various future group meeting times for the week of our implementation phase. These meeting times are documented in Asana. All team members then signed up to participate in four to five meetings based on their availabilities. In this process, we were able to have rotating teams to allow for variance in our pair programming practices while keeping in line with having the team meet every day.

@TODO add screenshot of asana

3 Extreme Programming Practices

Extreme programming (XP) is an agile software engineering methodology which enforces several practices. It is a natural companion to the SCRUM process, as both of them are iterative and organized in sprints and iterations.

In addition to extending Agile practices to the extreme, XP also promotes its own values and assumptions. The values include:

- **Communication**: Communication is essential for adjusting to feedback and implementing constant changes. Honest, regular communication should not only happen amongst developers, but also amongst their customers.
- **Feedback**: Feedback is the response from the customer to clarify what is required and wanted. By asking questions and making adjustments accordingly, the development team will be able to ensure that the code complies with the customer's specifications.
- **Simplicity**: The development team should only focus on what really needs to be built and only solve the current problems of the day.
- **Courage**: In this case, courage means making difficult decisions when necessary. It could be easy to disregard an issue because it would make several people unhappy. Courage implies speaking up and pointing out difficulties.

Extreme programming not only assumes that each stakeholder will incorporate the mentioned values, but also makes the following assumptions:

- **Enough Time and Resources**: Instead of rushing through the coding process, XP enables developers to work at their normal paces. By working in very short cycles, it reduces the length of time between actions and their effects. It also adjusts the project to fit the available resources.
- **Constant Cost of Change**: XP ensures a constant change of cost. This means that implementing a functionality now versus in a year will take the same amount of effort. By making this assumption, it allows developers to focus on current tasks without worrying about future features.
- **Developper Effectiveness**: Extreme programming assumes that in order to have great software, the developers should be great as well.
- **Freedom to Experiment**: Every team member (including managers, developers, and customers) should have the opportunity to experiment. This means asking question such as "Is there a better way to solve this problem ?" and keeping an open mind towards the answers.

3.1 Coding Practices

3.1.1 Simple Code and Design

Simple code and design was practiced to make code more efficient and straightforward. The development started by defining what frameworks to use and choosing the ones that are most flexible with respect to change of direction for long term purposes. These frameworks were chosen to be AngularJS and Slim. The development team also implemented features using libraries and reused code, taking out unnecessary functionalities when needed. This is shown in the comparison of code from C-Lyrics - a team's previous project - and PaperCloud, as well as the use of jqcloud2. However, the two pages for listing display were created, although they are not XP-friendly.

3.1.2 Refactor Mercilessly

The purpose of refactoring is to have better and more efficient code. This is a crucial step in order to obtain readable code and reduce the cost of change. The developing team's last programming session was entirely devoted to refactoring. The goal was to improve code's readability, document some of the main functions, and remove superficial or unused functionalities. Refactoring can be seen in the last commits on the git repository. @TODO: refactor commit image

3.1.3 Coding Standards

Coding standards are required to enable readability and communication through code. By having everyone writing code in the same manner, the team ensures that any developer will have some degree of familiarity with the whole codebase. The developers used custom code formatting templates to ensure homogeneity over the entire code. The code is openly available on GitHub as mentioned in Section 1.6, and the formatting templates are available at:

- https://github.com/seba-1511/st_settings

3.1.4 Common Metaphor

The idea behind having a common metaphor is to describe the project as it evolves. It allows the development team to share a vocabulary that will define well-understood relationships. For this project, the development team used a previous project as a metaphor. While not being the most creative, this was highly useful as every team member was able to clearly visualize the different parts of the application and how they related to each other. The specific project was the creation of a lyrics retrieval website which offered a lot of similar functionalities to the current project.

3.2 Developer Practices

3.2.1 Test Driven Development

Before implementing any functionality, we first wrote unit tests for functional requirements or end-to-end tests for visual requirements. This practice is underlined in our GitHub commit history (See @TODO add github commit reference). Due to the absence of a customer in our team, we do not consider these tests as acceptance tests. However, the visual tests were developed using the same framework and methodology that the customer would be using for acceptance testing.

@TODO add image

3.2.2 Pair Programming

Pair programming is the process of programming in pairs with separate roles: driver and observer. The driver writes code actively, while the observer passively guides and discusses the code being written with the driver. During all of our team meetings, we made sure that at least two team members were engaging in pair programming. This practice provides two significant benefits: uniformization of code throughout the project and collective code ownership.

@TODO Proof is in the BB thread, as well as K and J's picture.

3.2.3 Collective Code Ownership

Collective code ownership was created within respective teams, for example Kelsey, Sebastien, and Justine worked on the same GitHub repository as a part of the frontend team. Ultimately, there was little communication between the frontend and backend teams of our group. The team essentially only defined what the API should look like and worked on separate code repositories.

3.2.4 Continuous Integration

To apply continuous integration, our team set up a GitHub account and TravisCI. This allowed us to consistently make sure that we all shared the same code base, and it allowed us to monitor which commits broke our tests.

@TODO Proof is in the constant use of GitHub.

3.3 Business Practices

3.3.1 Customer Team Member

One of the main obstacles of adhering to Extreme programming practices was the fact that it was not possible to have the customer as a team member due to the special circumstances of the project being completed in a class environment. Instead of working closely with the customer during our meetings, the team focused on separate meetings with the customer and inferred preferences from conversations through Blackboard and in-person meetings.

@TODO (Blackboard/Meetings) Proof is in the conversations. (Show screenshot of conversations)

3.3.2 Planning Game and User Stories

The planning game was played through the scrum master, Justine. She made an early decision on which functionalities would be implemented for this iteration, according to the customer's defined product backlog. We stated a set of sprint logs that was a filtered version of the product backlog to complete high priority tasks first. These sprint logs also served to break down backlog requirements into implementable processes.

3.3.3 Regular Releases

After this first iteration, the team can now submit a fully working and tested system. This will hopefully be the case for next iterations, which will allow for regular releases. As outlined in the iteration review, we overestimated our capacities with respect to our sprint log. However, after the final refactoring, the requirements we decided to tackle are fully completed, and modifying the implemented functions will only occur if new functionalities are to be added that conflict with pre-existing functionalities.

3.3.4 Sustainable Pace

The team held meetings every day in the past week, for an average time of 2 hours. We did not necessarily have to "rush" or overwork ourselves. This is underlined by GitHub's graph, which shows a steady pace which can be shown in the figure below. @TODO Proof is on contribution pages of Github. Interestingly, however, as several team members were progressing on different tasks, most of tasks were marked as complete at the same time. This is shown in the figure by the Burndown chart in Asana. @TODO Proof by Asana Burndown chart.

@@@NOTE: Chart doesn't list that tasks were assigned on March 31

4 Artifacts

4.1 Story Cards

Although the customer was not necessarily available for the writing of story cards, the following cards have been created from the project requirements addressed in this iteration. If story cards were used as a strategy for requirements engineering, the customer would have arranged these in order of importance for prioritization.

Users should be able to search by keyword for articles Users should be able to search by researcher name for articles A wordcloud should appear after a search has been completed Clicking on a word in the wordcloud should perform a new search with that keyword A progress bar should show what progress of the word cloud being generated

4.2 The Bullpen

In order to simulate an effective bullpen, our team booked rooms in Leavey library for our meetings. In accordance with bullpen strategies, these rooms are isolated, and team members have laptop computers to move around freely and engage in pair programming. We were not, however, able to have the customer in the room with us to ask clarifying questions.

5 Iteration Review

5.1 Sprint Review

Our sprint review took place at a development team meeting on April 5th with all members present. We discussed the completed items on our product and sprint backlogs as well as the timeline for future progress with this project.

While the sprint review traditionally has the product owner present, this was not plausible for us because our meeting occurred after all changes to the code were made on Sunday night when the product owner was not available. To compensate for this, we plan to get input from the product owner on Monday April 6th.

5.1.1 Completed Product Backlog Items

Below is a list of completed product backlog items: * Search by keyword * Search by researcher

5.1.2 Completed Sprint Backlog Items

Below is a list of completed sprint backlog items: * Search by keyword: implement researcher search function, make API call with researcher name * Search by keyword: implement keyword search function, make API call with keyword, parse PDF's with PHP, get the abstract from API

5.1.3 Timeline

Based on the current state of the product backlog, the development team is behind schedule. They will need to improve the efficiency of meetings by adhering to more XP principals and by sticking closely to all of the scrum and agile development guidelines. Next sprint, the development team plans to catch up on the product and sprint backlogs, as well as continue adding functionality to the system.

5.2 Sprint Retrospective

The development team's sprint retrospective took place at a development team meeting on April 5th at 6:00pm with all members present.

5.2.1 Identify What Went Well

The development team was successfully able to have at least three members meet every day. This allowed the development team to make steady progress towards completion of this sprint's goal. The team worked well together and were able to split up jobs based on each team member's strengths to optimize time.

5.2.2 Identify What Needs Improvement

In the next sprint the development team need to break down the sprint log into much smaller, more manageable tasks. Because there were large tasks in the backlog, different people worked on each one, finishing the majority of tasks at the same time instead of finishing a steady stream of log items within the sprint. This will change the structure of the sprint meeting minutes because we will be able to be more specific in our goals for the day instead of making general statements about what we have worked on and what we plan to work on that day. Doing so will help the team adhere to agile development much more closely and will hopefully increase efficiency and help to keep all of the team members better informed on each other's work. The team will also be able to see project progress much more clearly and will know how to adjust the sprint backlog for the next iteration to match the work pace to productivity.

5.2.3 Plan to Improve Process Next Sprint

For the next sprint the development team plans to meet with the product owner earlier in the process and get a better idea of what is reasonable to add to the sprint backlog. They will also attempt to have scrum meetings with the whole team as opposed to only having meetings with the three plus people that worked on the project that day. In doing so the development team will be better equipped to adjust the sprint backlog throughout the iteration to ensure that the team is neither overwhelmed or not working to its full potential.

5.3 Scrum Meetings Review

All scrum meeting minutes can be found with timestamps on the online blackboard forum for team 6 in the "Iteration 1 Team Activity Log" thread. Additionally, all of the meeting minutes are listed in the Appendix in section 6.1 for the stakeholders' convenience.

5.4 Customer Integration Review

The development team met with the customers, Professor William G. Halfond and Sonal Mahajan, to obtain the necessary requirements for the intended project. After getting the initial set of requirements from both of the customers, the development team used this information to build the product backlog. The development team then asked for the customers to prioritize all of the requirements so they could build the sprint backlog for this iteration. The sprint backlog was checked with the customers to ensure that they agree that the decided tasks are reasonable for this sprint.

Later in the sprint the development team met again with Sonal, the product owner, to confirm that none of the product requirements had changed. While there were changes to be made to the product backlog, none of them affected the current sprint backlog so the development team didn't integrate any of this new information into this iteration. The updated product backlog is listed below in full, including items that were completed during this iteration.

- Updated Product Backlog Search by keyword (ACM and IEEE publications only) Search by researcher (ACM and IEEE publications only) Autocomplete for searching by researcher Ability to see history of searches Display word cloud with common words from top N pages User selects how many papers included in top N, we decide how to order the top N (ex. alphabetical or temporal) Progress bar for progress of generating word cloud Click on word in cloud lets you make new search with that word Click on word in cloud lets you see all research papers with that word in it and how frequently the word appears in each one Click on Conference

or Journal name creates list of all papers from that journal or conference. Display list of research papers that contain selected word in them with their authors listed next to them in

format Click research paper title takes you to a page with a link to download the paper Click author name in research paper list takes you to word cloud generation of common words from that author's research papers Select research papers to export references in .txt and .pdf format Ability to get BIBTEX reference of each research paper upon hover/ button press Navigation buttons between all pages

6 Appendices

6.1 SCRUM Meeting Minutes

- Scrum Meeting 1
- Date: Friday March 27th, 3pm
  - Justine
    - Q1: Last time I helped decide the requirements list for our project which will form the project backlog and I helped to create our meeting schedule for this first sprint.
    - Q2: This time I set up the project repo and peer programed with Seb to start the skeleton for the frontend and the corresponding first tests.
    - Q3: No, I am eager to start this project and thus have not encountered any problems yet.
  - Seb
    - Q1: Last time I helped decide the requirements list for our project which will form the project backlog and I helped to create our meeting schedule for this first sprint.
    - Q2: Defined the API between the client and the server with Mark and peer programed with Justine to start the skeleton for the frontend and the corresponding first tests. Implemented first test in testing architecture.
    - Q3: No, I am eager to start this project and thus have not encountered any problem yet.
  - Mark
    - Q1: Last time I helped decide the requirements list for our project which will form the project backlog and I helped to create our meeting schedule for this first sprint.

        * Q2: Setup the repository for the backend data while discussing the data limits of the current API. I am still looking for potential APIs that will help us get the data faster. Loaded dependencies and registered the backend on Travis CI.
        * Q3: The current API has limitations, so I will need to find either an entirely new one or find a second one that can fulfill the missing requirements.

- Scrum Meeting 2
- Date: Saturday March 28th, 5 - 7pm
    - Justine
        * Q1: Last time I set up the project repo and peer programed with Seb to start the skeleton for the frontend and the corresponding first tests.
        * Q2: Today I am going to decide which requirements we will focus on for iteration 1 and I will start revising the Implementation document from the previous project. I also will peer program with Seb on tests.
        * Q3: I still haven't encountered any problems yet.
    - Seb
        * Q1: Last time I defined the API between the client and the server with Mark and peer programed with Justine to start the skeleton for the frontend and the corresponding first tests. Implemented first test in testing architecture.
        * Q2: While Justine is taking care of her scrum master tasks, I will set up a complete testing environment. (black- and white-box) Then I will pair program with her and write the first tests the team will have to pass on the front page.
        * Q3: Having the environment to work did not go as smoothly as I expected, yesterday. This might become a problem.

- Scrum Meeting 3 (3/29 not documented here)
- Scrum Meeting 4
- Date: Monday March 30th, 8 - 9pm
    - Justine
        * Q1: Last time I worked on the Implementation and Testing redo docs.
        * Q2: This time I plan on working on both the layout and functionality of the main page.

    * Q3: There are some holes in my knowledge of JavaScript which I foresee to be a problem later on, so I will try to do some self teaching before my next team meeting.
  - Mark
    * Q1: Last work day I found an API to use to make the data requests. I also set up the work environment and created the github repo.
    * Q2: I will find an Atom parser for php today and get it working with the API endpoints.
    * Q3: The atom parsers can be difficult to implement, so I will have to find a simple way to make it work.
  - Milad
    * Q1: I edited and fixed my assigned sections of the implementation document. Section 3.3.1 and section 4. I added section 3.3.2.
    * Q2: Start building php code using WAMP server downloaded last time before my computer screen broke. Get up to speed on PHP coding.
    * Q3: Still a bit confused in Asana, need to understand how to use it better.

- Scrum Meeting 5
- Date: Tuesday March 31th, 8:00pm - 9:30pm
  - Kelsey
    * Q1: Last time I worked on creating the meeting schedule for the first sprint.
    * Q2: Today I will start on the main page and make sure the tests that Seb creates will pass.
    * Q3: I have not encountered any problems thus far.
  - Seb
    * Q1: Last time I wrote tests for the display of the search bars and their related functionalities.
    * Q2: Today I will work on writing unit tests for those funcitonalities and tests for the server factories.
    * Q3: I have not encountered any problems thus far.
  - Milad
    * Q1: Last time I set up the WAMP server environment to make sure it is working. I also met with Sonal to discuss
    * Q2: I am going to work on APIs and how to get information on research papers.

        * Q3: Asana is still confusing, but I am slowly understanding it.
    - Jeff
        * Q1: Last meeting worked on meeting schedule for the first sprint.
        * Q2: I am going to work on the sprint logs.
        * Q3: I have not encountered any problems thus far.

- Scrum Meeting 6
- Date: April 1, 7:30-9pm
    - Justine
        * Q1: Last time I focused on getting a better understanding of JavaScript and gaining familiarity with the code we have so far.
        * Q2: This time I plan on working on the word cloud generation.
        * Q3: There are no new problems yet.
    - Mark
        * Q1: Downloaded SimplePie to parse the Atom data from the arXiv API.
        * Q2: Work on the search by keyword and search by name data retrieval.
        * Q3: There were problems with the parser, but I was able to resolve them. No new ones so far.

- Scrum Meeting 7
- Date: April 1, 9:00-10pm
    - Justine
        * Q1: Last time I worked on the home page and tried to figure out how to generate the wordcloud.
        * Q2: This time I plan on continuing to work on the word cloud generation and peer programed with Kelsey.
        * Q3: There are no new problems yet.
    - Kelsey
        * Q1: Last time I worked on creating the search bars and search buttons for the main page.
        * Q2: This time I will work on generating a space for the word cloud by peer programing with Justine.
        * Q3: I have not encountered any problems thus far.
    - Mark
        * Q1: Got the search by keyword functionality working. The search by author functionality is partially working but with bugs.

        * Q2: I will get the search by author working, along with the auto complete.
        * Q3: I have a problem with parsing strings, but it will be fixed soon.

- Scrum Meeting 8
- Date: April 3, 3:30-5:30pm
    - Justine
        * Q1: Last time I tried to understand how the word cloud generation worked for our previous project and how to adapt it to this project.
        * Q2: This time I plan to continue working on word cloud generation.
        * Q3: I am having problems understanding the code from our previous project and figuring out how to reuse it.
    - Kelsey
        * Q1: Last time I worked on generating a word cloud by peer programming with Justine.
        * Q2: This time I will work with Justine, pair programming.
        * Q3: I am having problems with understanding code from the previous project
    - Sebastien
        * Q1: Last time I wrote test code for the search bars and server factories
        * Q2: Today I will write the factories' code so that it passes the tests. I will also help Justine and Kelsey understanding angular directives.
        * Q3: No main problem today. Everything is looking bright!