Project Release Summary

GitHub repository monitoring tool

(https://github.com/abhandal/SOEN341-G4)

Professor: Dr. René Witte, P.Eng.
Teacher assistant: Tamara Dittimi

Name	Student id	GitHub id	Number of story points that member was an author on
Nikita Baranov	40012854	NikitaBaranov	
Aman Bhandal	27390858	abhandal	
Raymart De Guzman	40010443	tramyardg	
Dmitry Kryukov	40029645	b5n	
Charles-Philippe Labbé	40002442	CharlesPhilippeLabbe	
Andy Pham	40006071	andypham29	
Ksenia Popova	40029623	Lyncis	
Batoul Yehia	40010912	batoulyehia	

Montreal, QC, Canada
2017

Project summary:

The GitHub repository monitoring tool helps the teaching assistants to monitor and grade each group and student, by analyzing a repository information in an easy-to-use and easy-to-understand way.

Main Features:

- Secure login with GitHub account
- Get an information from available repositories
- Show the information in a convenient way (text, charts ...):
    - Contributors (students)
        - a number of events: overall and per student
        - a percentage of events per student
    - Events:
        - Commits
        - Pull requests
        - Issues
        - Comments
- Write a feedback from TA that will be stored in teams' repositories

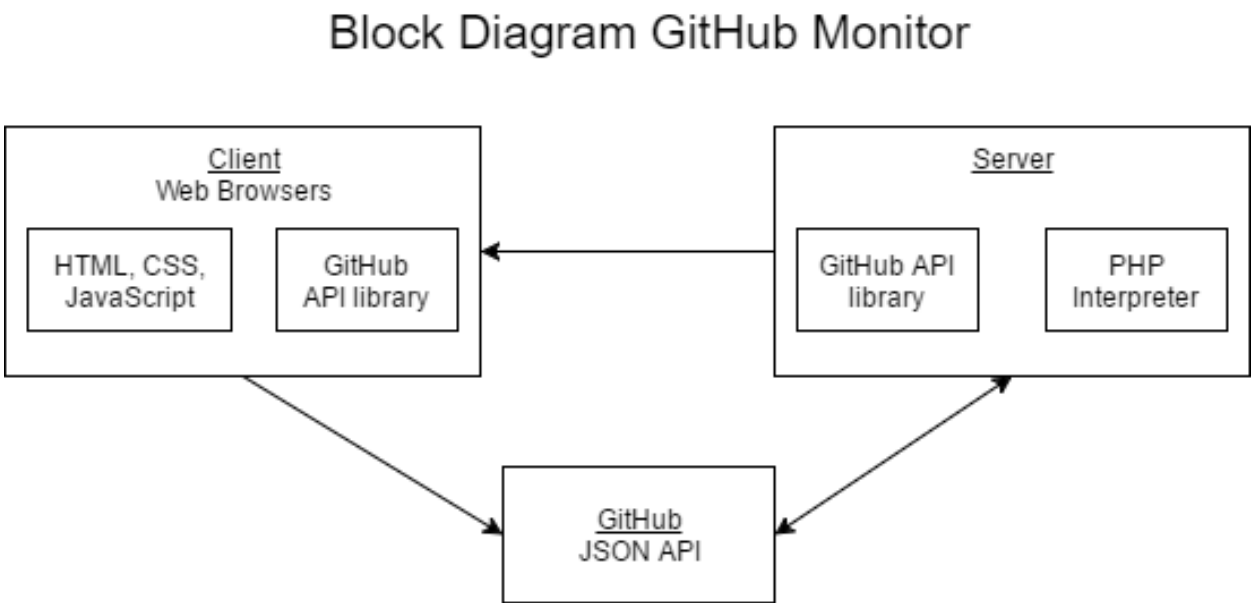
Velocity and a list of user stories and non-story tasks for each iteration:

Total: X stories, X points over X weeks
Iteration 1 (X stories, X points)
US #X: US name [X points] [Status: Done, Removed, Pushed, or Splitted]
Iteration 2 (X stories, X points)
Log In (3)
Events Separation (3)
Adding repositories (2)
Generation Information (3)
Overall Report
Function of Activity Percentage (2)
Tabs
Data Selection (1)
Iteration 3(X stories, X points)
Burndown Chart (?)
Comments (3)
Selecting Repository (5)
Sorting the information of the user interface
Weekly Report (3)
Implement a graphing tool (Coffee)
Report Type
Iteration 4(X stories, X points)
Iteration 5, Release(X stories, X points)

Overall Arch and Design

Show us the overall architecture in your system with architecture diagram.

Show applicable parts (at least one) from the 4+1 logical/physical/etc. model, with appropriate UML techniques we covered. You can also include these diagrams in your stories on GitHub (by providing urls).



Infrastructure

Server side:

- PHP (http://php.net)

PHP makes it easy to work with headers, allowing the authentication of a user to be written with less code. It also allows to pass the information of a repo in the header (such as the name of the repo and the owner).

- Composer (https://github.com/composer/composer)

The Composer was needed to make a request to the Github api for the authentication of the user. By doing it this way, it allowed to store the Github oauth token in a user session which is useful for passing information from one page to another.

- Amazon Web Services (AWS) (https://aws.amazon.com)

Free of charge reliable web hosting with PHP support.

Client side:

- HTML, CSS (https://www.w3.org/html)
- JavaScript (http://www.ecmascript.org)
- Chart.js (http://www.chartjs.org)

Open source simple and flexible JavaScript charting library.

- JQuerry.js (https://jquery.com)

Fast, small, and feature-rich JavaScript library for HTML document traversal and manipulation, event handling, animation.

- Material Design (https://material.io)

Simple, ready to use UI library.

- Bootstrap (http://getbootstrap.com)

JavaScript framework for developing responsive website.

- Github.js (https://github.com/github-tools/github)

This Github API wrapper handles the method calls and promises very well, and is very easy to understand. Other Javascript wrapper needed Node.js, but this one only required JQuery.

Automation testing:

- Travis CI (https://travis-ci.com)

Free for students flexible testing tool with immediate repetitive testing.

Name Conventions

The code conventions of the Java programming language were followed:
(http://www.oracle.com/technetwork/java/codeconvtoc-136057.html)

Code

File Path	Purpose
Integration/js/repo.js https://github.com/abhandal/SOEN341-G4/blob/master/Integration/js/repo.js	Getting the information of the Github API and converting it to useful information.
Integration/repoSelection.php https://github.com/abhandal/SOEN341-G4/blob/master/frontend/repo_selection.html	Main application page
Integration/admin.php https://github.com/abhandal/SOEN341-G4/blob/master/Integration/admin.php	Handles Github login and session timeout. Sets up objects and variables.
Integration/js/addRepo.js https://github.com/abhandal/SOEN341-G4/blob/master/Integration/js/addRepo.js	Lists available repositories and lists all selected repositories on the sidebar.
Integration/css/styles.css https://github.com/abhandal/SOEN341-G4/blob/master/Integration/css/styles.css	Creates the style of the project.

Testing and Continuous Integration

5 most important unit test.
(https://github.com/abhandal/SOEN341-G4/blob/master/test/unit/gh_api_test.js)

Line Number in unit test file	What is it testing
Description Test Line 26 - 30 See Appendix 3.1	If the description is empty than it will return false, failing the test. Otherwise, it will return true which is a pass.
Burndown Test Line 46 - 50 See Appendix 3.2	If the burndown data is empty than it will return false, failing the test. Otherwise, it will return true which is a pass.
Commit Test Line 60 - 66 See Appendix 3.3	If the user has 0 or less than 0 commits then the test will fail, since the user does have more than 0 commits in this SOEN341 repo. Additionally, if an undefined user is given then the test will pass as it ensures than an undefined user is caught.
Week Test Line 68 - 73 See Appendix 3.4	Tests whether there are less than 0 weeks in the first week and if a non-existent week returns undefined.
Collaborator Test Line 52 - 58 See Appendix 3.5	Tests whether there are 10 collaborators in the SOEN341-G4 repo, in this case there are exactly 10 users within the repo. Additionally, it checks whether there are 0 or less

	than 0 collaborators within the repo for which if it returns true then the test will fail.

5 most important integration tests.

Test File path	What is it testing
Tabs test https://github.com/abhandal/SOEN341-G4/blob/master/test/integration/tabs_test.js	The test runs through all the tabs and checks for anchor objects that confirm that this tab works
Authorization test https://github.com/abhandal/SOEN341-G4/blob/master/test/integration/login_test.js	The test verifies the authorization of the user with the correct and incorrect credentials
Add repo test https://github.com/abhandal/SOEN341-G4/blob/master/test/integration/get_repo_test.js	The test checks to see if the user has at least one repository to add
Get issues test https://github.com/abhandal/SOEN341-G4/blob/master/test/integration/get_repo_issues_test.js	The test checks whether it is possible to get an object with the number of issues
Get commits test https://github.com/abhandal/SOEN341-G4/blob/master/test/integration/get_repo_commits_test.js	The test checks whether it is possible to get an object with the number of commits

5 most important Acceptance tests

User Story	Expected Results
Adding repositories	Add repository to be availiable in the system.
Overall Report	The system should provide the overall report of the repository
Weekly Report	The system should provide the weekly splits of overall report of the repository
Burndown Chart	The system should provide the burndown chart for the repository
Comments	Write a comment and save it to the abserved repository.

Describe your continuous integration environment. Include a link to your CI.

Describe the choice of the static analysis tool and how do you run it. The static analysis tool should analyze the language that are used in the majority of the your source code.

Attache a report as appendix from static analysis tool by running the static analysis tool on your source code.

Challenges

Aman Bhandal

The main challenge areas involved setting up the testing environment and assigning story points. The test environment was difficult as it involved learning QUnit, a JavaScript unit testing framework, and creating test cases for each of the JavaScript functions. Learning and understanding the steps involved in setting up QUnit were facilitated by the framework's documentation and assistance from members of our team. Additionally, the creation of test cases were carried out using the partition method which assisted in ensuring that outlier and normal cases were tested accordingly. Another challenge area involved assigning story points to user stories, especially during our first planning poker session where we did not understand our system nor our ability to implement the features. However, as we moved through the sprints and development cycles were able to assign story points more accurately and we were able to deliver on our individual tasks.

Raymart De Guzman

I faced two different challenges during the course of the project. First, learning Material Design Lite (MDL) framework for the front-end development. This is framework is all new to me. However, I did quite a bit of work in Bootstrap, a framework similar to MDL. I've been using Bootstrap in my own projects. So, this was not a surprise for me. It took me a couple of hours to learn the basics and finally applied them to the project. Second, understanding and learning about web application programming interface (API) especially GitHub API. GitHub API is the main web API used in the project, thus, I had to read the API documentation and make sense how are we going to employ their usage in our project. Fortunately, playing around with GitHub's OAuth authorization API helps me understand API application better, as well as its relevance in other projects.

Ksenia Popova

I met a few challenges during this project.

1. Work with Promises when I needed to use the data from them and it was not visible outside Promises.
2. Chart type switch, because I needed to destroy the existing chart before that, but charts plugin.
3. Sorting of the data, because it is represented in object with different properties and it is not possible to make data sorted with one of the object properties.

Dmitry Kryukov

In this project, the main challenges involved experience with asynchronous javascript and also work with selenium webdriver library for creating integration tests.
I spent a lot of time studying how promises work and how to test asynchronous code.

Charles-Philippe Labbe

The most difficult part of the project, was to learn to work with asynchronous Javascript. Not having a lot of experience with any of the web-programming languages meant that I had to work extra hard to understand the code I was writing and learn how to make it work. At first, I started implementing the backend using PHP, but when it was all done, I realized that it was too slow to convert Github data from JSON to PHP back to JSON objects. Hence, after a couple of weeks of implementing the PHP part of the program, I had to convert everything to Javascript. Since AJAX on its own is not the most convenient way to work with api calls, I opted to go with JQuery. Now it is much faster, but working with promises can be finicky at times.

Feedback

Aman Bhandal

The project provided an abundance of experience in the realm of software development and enforced the importance of project planning. This was my first time using the agile development method and working with a large team in developing an application from the ground up. Our team's ability to effectively coordinate tasks and utilize the strengths of individual team members allowed for a seamless planning and implementation of the e-learning application. Due to the positive impact of this project I will use the methods learned and utilize it in future development projects.

Ksenia Popova

This project was a very valuable experience for me for many reasons. First of all, I improved my skills in Javascript. My task was to display the data from the GitHub with charts and tables, so I needed to figure out how to do it and create some algorithms. Secondly, the project idea was to apply Scrum model to our development process, and this should be a very useful experience in the future. And finally, we have a great team. We had a lot of meetings and communicated with Slack, so we always stayed in touch. Everyone could get help with any part of the project and everyone did what he was supposed to do. We completed everything on time without any conflicts or delays, what is why I am very satisfied with this project.

Charles-Philippe Labbe

Overall, I found this project very fulfilling for many reasons. Primarily because I did not have any experience with web development; and because I never worked with external libraries and APIs before. These two reasons in particular meant that I had to work overtime in certain cases, but it was enjoyable nonetheless.

Dmitry Kryukov

I received a lot of useful experience during this project. It was a good work such as a team on a real project, I got a lot of experience in the software development, upgraded skills in Javascript and knew a lot of new about testing, improved my communication skills. We were working together without conflicts and I was enjoying the project all the time.

Appendix 1 – Final Acceptance Test table.

Project:	GitHub repository monitoring tool	Browser:	Chrome
Written By:	Nikita Baranov	Description:	Full User story test
Tested By:	Aman Bhandal	Tested On:	12-Apr-17

#	Date	User Story	Expected Results	Actual Results	Pass
login to the System					
1	12-Apr	Log In	Should get to home screen using Github credentials	User logged in using GitHub credentials	Yes
2	12-Apr	Selecting Repository	Look throught available repositaries, select some of them to be available on system.	User got a list of available repositaries. Selected 2 and got them available on left panel.	Yes
3	12-Apr	Adding repositories	Add repository to be availiable in the system.	User selected repository to be added and got them available on left panel of the system.	Yes
Overall functionality					
4	12-Apr	Tabs	The information should be presented in different tabs	The system has tabs for different reports	Yes
5	12-Apr	Generation Information	The system should provide the general information about the repository.	General info seen on the General InfoTab	Yes
6	12-Apr	Overall Report	The system should provide the overall report of the repository	Owerall report is on the Overall Report Tab	Yes
7	12-Apr	Weekly Report	The system should provide the weekly splits of overall report of the repository	Owerall report is on the Weekly Report Tab	Yes
8	12-Apr	Burndown Chart	The system should provide the burndown chart for the repository	Burndown Chart is on the Burndown chart Tab	Yes
9	12-Apr	Comments	Write a comment and save it to the abserved repository.	Wrote a comment and save it to the abserved repository. The page did not refreshed.	Yes
The functionality of the Report Tabs					
10	12-Apr	Sorting the information of the user interface	Table with information could be sorted in descending order	The information cold be sorted by clicking on the Sort button.	Yes
11	12-Apr	Function of Activity Percentage	The information should be seen in precentage	The information in tables are shown with precentages.	Yes
12	12-Apr	Events Separation	The information should be separated to commits, issues, and comments.	Different events are shown separately.	Yes
13	12-Apr	Implement a graphing tool	The information should be presented in graphical view	There is Donut, Pie, Line and Bar charts.	Yes
14	12-Apr	Report Type	The required information should be displayed. In table forms or charts, or both combined.	The table and Charts are on the pages.	Yes
15	12-Apr	Data Selection	The system should display only selected information.	The information could be chosen using checkboxes on the top of the page.	Yes

Appendix 2 – User Story Screen Shots

1. Log In
   As a user, I want to log in using my GitHub account.
2. Adding repositories
   As a user, I want be able to add repositories on my page, so the data from it can be displayed.
3. Selecting Repository
   As a user, I want to select a repository from a GitHub account.
4. Generation Information
   As a user, I would like to have a general information page about a repository.
5. Comments
   As a user, I want to add comments to each repository.
6. Sorting the information of the user interface
   As a user, I would like to sort the charts and the tables from the least commits to the most.
7. Overall Report
   As a user, I would like to have an overall report of all the collaborators' activities.
8. Weekly Report
   As a user, I would like to see a weekly report of all the collaborators' activities.
9. Function of Activity Percentage
   As a user, I would like to see the percentage of each collaborator's activity.
10. Events Separation
    As a user, I would like to see a separation of events to commits, issues, and comments.
11. Burndown Chart
    As a user, I would like to have access to the burndown chart of the team.
12. Tabs
    As a user, I want to see different tabs in order to have access to general information, reports, burndown charts, and comments.
13. Implement a graphing tool
    As a user, I would like to see the breakdown of events into tables and charts.
14. Report Type
    As a user, I want to choose how the required information will be displayed. In table forms or charts, or both combined.
15. Data Selection
    As a user, I want to be able to select the data shown to me.

Appendix 3 - Unit Test Code Blocks

1. Code Block 1

```
repo.description.then(function(description) {
  QUnit.test("details", function(assert){
      assert.notEqual(description, "", "Description is not empty");
  });
});
```

2. Code Block 2

```
repo.burndown.then(function(burndown) {
  QUnit.test("getBurndown", function(assert){
      assert.notEqual(burndown, "", "Burndown is not empty");
  });
});
```

3. Code Block 3

```
repo.commits.then(function(commits) {
  QUnit.test("getCommits", function(assert){
      assert.notEqual(commits['abhandal'], 0, "Repo 'abhandal' has greater than 0
commits");
      assert.equal(commits['blablabla'], undefined, "Repo undefined returnes
undefined");
      assert.notEqual(commits['abhandal'], -1, "Repo 'abhandal' does not have less
than 0 commits");
  });
});
```

4. Code Block 4

```
repo.weeklyInfo.then(function(weeks){
  QUnit.test("getWeeklyInfo", function(assert){
      assert.notEqual(weeks[0]['abhandal'], -1, "Number of events in first week is not
less than 0");
      assert.equal(weeks[-1], undefined, "Number of events in non-existant week is
undefined");
  });
});
```

5. Code Block 5

```
repo.collaborators.then(function(response){
 QUnit.test("getCollaborators", function(assert){
     assert.equal(response.length,10,"There are 10 collaborators in the SOEN341-G4
repo");
     assert.notEqual(response.length,-1,"There are not less than 0 collaborators in the
SOEN341-G4 repo");
     assert.notEqual(response.length,0,"There are not 0 collaborators in the
SOEN341-G4 repo");
  });
});
```

Old Report

User Stories

Extra functions

1. Periodical Email Summaries
   As a user, I want the system to send me an email with an overall summary report on activities in repositories so that I can spend less time to get an overall information on a class process.
2. Time Range
   As a user, I want to select the time range for the activities for the team and each person: I can select a starting date and an ending date.
3. Separation by sprints
   As a user, I want to specify sprints as a time range to be able group information in charts by sprints.
4. Grading space
   As a user, I would like to publish grades for each sprint so that the team could see it.
5. Repository Description
   As a user, I want to be able create my own description to each repository in my list, that I could recognize repository in my own association.
6. Managing repositories
   As a user, I should be able to manage repositories on my page, that I could edit list of repositories.

What should be in the doc (delete before submission)
TA meeting:
Final report: description, US, Architectural blocks and some feedback.

Course outline:
complete report integrating all sprints of the project,
demonstration of software, and quality of application