[CS209A-22Fall] Final Project (100 points)

> Question Design: Yida Tao
>
> Demo & Data: Chi Xu, Xiaofeng Wu
>
> Evaluation: Xiaofeng Wu
>
> Git & Code styles: Chi Xu

Early submission: Dec.20, 23:55pm (Tuesday at week 15).

Final submission: Dec.27, 23:55pm (Tuesday at week 16).

Late submissions after the final deadline will NOT be accepted.

Developing a software application is not an easy task. It involves multiple team players and complicated development efforts, and any careless error or mis-communication could directly affect productivity and software outcome. Hence, it's crucial for a developer team to constantly monitor their commits, bug fixes, communications, and other development activities.

In this final project, we'll use Spring Boot to develop a web application that stores, analyzes, and visualizes development data, with the purpose of answering some of the critical questions raised during development and team management.

Basic Requirements (60 points)

GitHub is a website for developers to store and manage their code. Developers could also use GitHub to track the releases, versions, issues, commits (code changes) and discussions of their projects. For a given GitHub repository, we are interested in the following questions.

Developers (10 points)

- How many developers have committed to this repo?
- Which developers are the most active (i.e., who committed the most)?

Issues (20 points)

- How many issues are open and how many are closed?
- What is the typical issue resolution time (i.e., the duration between issue open time and issue close time) for this repo?

Releases and Commits (30 points)

- How many releases are there in this repo?
- How many commits are made between each release?
- At which time (e.g., weekday, weekend, morning, evening, etc.) do developers made commits?

Your web application, opened in a browser, should be able to answer these questions with proper visualization. It's your job to design the web application and choose proper visualization, so that users could

comfortably use your application to get the answers they want.

Requirements for Implementation (25 points)

Repo Selection

There is no restriction on which GitHub repo you should use. You could choose any repo that you are interested in as the subject of your project.

Data Collection & Storage (10 points)

You should collect corresponding data from GitHub to answer the above questions. Please check the official GitHub REST API documentation to learn the REST API for collecting different types of data.

- You need to create a GitHub account in order to use its full REST API service.
- Different types of API requests are subject to different rate limits. **Please carefully design and execute your requests**.
- Connections to GitHub REST service maybe unstable sometimes. So, **please start the data collection ASAP!**

It is recommended that you use a database (e.g., PostgreSQL, MySQL, etc.) to store the data. However, it is also fine if you store the data in plain files.

Web Framework (10 points)

You should use `Spring Boot` as the web framework .

Frontend (5 points)

Frontend functionalities, such as data visualization and interactive controls, could be implemented in any programming language (e.g., JavaScript, Java, JSP, HTML, CSS, etc.) with any 3rd-party libraries or framework.

Advanced Requirements (12 points)

Multiple repositories (3 points)

Your web application could handle multiple GitHub repos. Users could navigate through different repos to see the corresponding visualizations.

REST services (4 points)

You could also build a *web service* that answers the above questions, so that users may use RESTful APIs to get the answers they want. The web service may include questions defined in this documentation, or you may also define new questions. Nevertheless, your web service should provide at least 3 different RESTful endpoints that answer 3 different types of questions (e.g., https://your.rest.server/{repo}/issues?status=open will return the number of open issues for the specified repo).

Issue topics (5 points)

A GitHub repo has an "issues" tab, in which there are many issue threads. In a single issue thread, developers often have many rounds of discussions on why the issue raises and how to solve it. For instance, in the

`spring-boot` repo, you could see a list of issues here or click a single issue thread to see all of the discussions.

It seems interesting to know "*which topics/keywords/problems are often discussed in this repo?*". To answer this question, your application should collect a repo's issues and analyze their textual data (including issue title, description, and/or follow-up comments). Finally, you should choose a proper visualization to answer this question.

Documentation (3 points)

You should provide a written report that describes the GitHub repos you selected for this project. The written report should also introduce the architecture design of your project, as well as the important classes, fields, and methods. Finally, your report should highlight the insights you obtained from the data analysis results, e.g., what are the answers to the above questions, what can we learn about the repo according to your answers, and what can be improved about this repo, etc.

Teamwork

We encourage you to work in a team for this final project. The preferred team size is 2, while a team of 3 or a team of only 1 student is also allowed. However, teams of size 3 CANNOT be consisted of only CS students. In addition, teams of size 3 will get a 90% discount on their project scores, because the average workload for each student decreases. Teams of only 1 student will NOT get a bonus, because s/he doesn't have to make the communication efforts that are costly but crucial for a teamwork.

Please find your teammates as soon as possible, and fill in your team information in this form: 【腾讯文档】CS209A-22F-项目组队 https://docs.qq.com/sheet/DQ2JYcVRNWFhWdWd3?tab=BB08J2

Sample Data & Demo

We provide sample data in the attached `.zip` file. You could also find a simple demo of the project here.

However, try NOT to directly reuse the exact data or visualization in your project. **Use your imagination**. **We hope that each team could deliver a unique, creative, and insightful project.**

Submission

Please submit a zip file named "StudentID-Name-Project.zip" to Sakai. The submitted zip should include two parts:

1. The project folder, which includes all the source code and other relevant files necessary for running the project.
2. A written report (.pdf).

Presentation

Each team should present your project during the lab sessions on either Dec.21 (Week 15) or Dec.28 (Week 16). We have two lab groups. You may team up with someone from another lab group. However, all team members must be present for the project presentation. Please check your calendar and arrange your presentation time accordingly.

To present at Dec. 21 (Week 15), your team needs to submit the project before the early submission date (Dec.20). Teams that have submitted and presented the project at week 15 will get a 1 point bonus to the **overall** course grade.

In addition, teams that perform well at week 15 will have a chance to present the project during the lectures (Tuesday) at week 16. Such teams will get an additional 2 points bonus to the **overall** course grade.

Evaluation

- **Functionalities**: Each team must present the project during lab sessions (see above), and we'll check whether you've accomplished the required functionalities onsite.
- **Version Control**: You should use `GitHub` to manage the code changes of your project (see lab 1 for further details of how to use `git`). You should made **at least 2 commits**. Your remote repo on GitHub **should be set to `private` before deadline, so that no one else will see your code.**
- **Coding Style**: You should pay attention to write readable and maintainable code along the way. See lab 1 for how to use `CheckStyle` for that purpose. **After the deadline**, you can set your GitHub repo to `public`, and we'll check whether any of your commits have reduced `CheckStyle` warnings according to `google_checks.xml`.