CSC4001 Software Engineering
Project Requirement Specification

17 January 2022

1. Objective

The objective of this course project is to practice what you are learning in this CSC4001 Software Engineering course by specifying, designing, implementing, testing, and documenting a Web based client-server application (e.g., web application, mobile-cloud application). The project serves as a vehicle to sharpen your knowledge in Software Engineering and to develop your relevant skills. This course project also introduces students to teamwork and project management, which are keys for successful large-scale software development. Besides, you can take the project opportunity to develop a tech-savvy, customer-driven, and eye-catching software product (such as a mobile application), as if you are founding an IT start-up company.

The project consists of four phases: (1) project design document, (2) initial code, (3) completed code and demo, and (4) commented code and final report. After the submission of completed code in (3), project team needs to make a demonstration and answers questions regarding the project during the demo on the Demo Week(s).

2. Project Grouping

Each project group is composed of 4 students for the whole duration of the project. All students in a group work together on the same project based on the project requirements defined below. By now you may have chosen group members by yourselves. If so, please sign up your project group accordingly in our Google Form:
https://forms.gle/wCPXVopgmvCp2gPL8

Note that you can sign up a group only when you have 4 members in the group already. At the end of project grouping, we will randomly assign remaining students to form additional groups or assign them to existing groups which include less than 4 (i.e., 2 or 3) members. Once the group is assigned, you should remain in the group and work with your team members closely for the entire project duration.

3. Project Requirements:

The goal of the project is to develop an advanced Web based client-server application with an arbitrary topic defined by your group. The whole project is composed of two parts, implementation and documentation. Here we just list the requirements, for detailed grading criteria, please refer to **Section 5**.

3.1. Implementation

For the implementation part, you can imagine that this project is the killer application of a start-up company created by your team. Thus, you can try to design and implement as many fancy features as you like, making use of the full power of software engineering techniques, under your topic. Meanwhile, there are some basic requirements that you *must* include in your project:

- Web-based access under client-server architecture

The application must be a Web-based client-server system, actively accessible over the Web using a Web browser (e.g., Chrome). Your application will provide useful and friendly functionality/service to the human users via Webpage interfaces. There is no particular technological requirement on Web access. However, we encourage you to employ modern Web technologies, such as HTML5, CSS3.0, JQuery, Ajax, Node.js, WebSocket, Vue, AngularJS, ReactJS, React Native, etc.

- Database

Either SQL database (e.g., MySQL, or SQLite) or NoSQL database (e.g., MongoDB, or Redis) must be employed by your application for storing data. If you are not using a database, please explain why in the *project design document*.

- Clear UI

Your application should at least have a clear graphical UI design. The UI should be consistent and easy to understand. Users should be able to use the application without the help of your developers.

- User management

Your application should also have comprehensive user management functionalities. Specifically, you should let users to sign up, login/logout, upload profile photo, and change password. Verification emails are required to activate the user account.

- Admin User

Your application should also have admin user account, who is responsible for the management of the whole application. The major functionality of an admin user is to manage the database of the whole application. Specifically, he/she should be able to modify all the content in the database, e.g., user profile.

By fulfilling these basic requirements, you can get **60%** of marks in the "completed code and demo" part. For the remaining 40%, you can add any fancy features you want. For example, you can also port your applications on mobile devices, provide a pretty UI design, adopt some advanced features like recommendation, get your application deployed on the cloud, employ machine learning techniques for better user experience, and so on. Any interesting feature is welcome, just broaden your mind.

3.2. Documentation

For the documentation part, you are required to use techniques (e.g., requirement analysis, DFD,

UML) taught in the lectures in your project development process and provide a comprehensive documentation for your design and implementation. In this project, you will provide two documents, a high-level design document and a final report. You are recommended to employ as many software engineering techniques taught in the lectures as possible. However, the followings are the basic requirements that you *must* include in your documents:

- DFD (Data Flow Diagram)

Requirement specification is fundamental to your project. You are expected to submit your project requirement specification by English description and DFD techniques taught in the class, which explain the key components of your project. This is normally done before the project design phase, but for this project we simply require it in your Final Report, when you have completed your demo and got a comprehensive idea of your project. You are recommended to employ DFD as detailed as you like in your Final Report, with proper revision to reflect your actual software system specification.

- UML

UML must be employed to describe the key components of your project design. This will also be described in the class. You are required to employ UML in your Final Report, so as to reflect the final detailed design of your software system.

- Test Cases

You must design some test cases to test the key functionalities of your project. The design of test cases includes black box testing (required) and white box testing (optional), which will be covered in the class.

Apart from these basic requirements, you could also include **Finite State Machine** in your documentation to describe non-trivial software system behaviors. These are requirement specification techniques, which will be covered in the class. Trivial system behavior, such as the typical login process, will not count. You are encouraged to specify the Finite State Machine of your system in the Final Report, to describe your system behavior in detail.

3.2. Other requirements

When designing your Web application, the most important project feature to keep in mind is that the software product you will develop should require a reasonable programming effort. In order to learn good design, the process must include actual implementation of the design. Therefore, the project must contain a decent programming exercise. Please note that designing passive HTML pages with no communications between server and clients is not programming. Although your project can include Web page design, that is not enough. On the other hand, designing active Web pages using JQuery or other advanced frameworks like Vue, AngularJS and React for dynamic pages is programming and is perfectly acceptable for the class project. In addition to the programming effort, keep in mind that one key purpose of this course is that you learn how to do modular design of software and how to document the design using symbolic representations, i.e., UML diagrams.

No joint work over any technical aspects of the project is allowed between any two groups/teams. Any problem about the project requirements should be directed to the tutors through electronic mails, or tutorial sessions. The reason for this policy is to enforce team separation for proper credits. This project should be considered as if there is only one single team, namely your team, responsible for your whole project development. No plagiarism is allowed regarding any aspect of the project. Reusing existing designs or codes as part of your project (such as those from the open source) is allowed, but you **need to document the reused code and automatically generated code clearly**. We will also employ professional tools to verify the percentage of existing codes in your project. Note that you should not reuse existing code excessively. Your project mark will be deduced significantly if the percentage of reused existing code exceeds a certain threshold.

4. Project Phases:

There are four project phases described as follows:

(1) Project Design Document

In this phase, each project group will prepare and submit a high-level design document to provide high-level descriptions on functionalities, features, and architectural design of your application. Project background, architecture diagrams and brief descriptions of the key system components should be provided. Please refer to Appendix 1 for guidance. Feedbacks will be provided on your high-level design, and you should consider and possibly revise the project goals before going on to the next phase. You are required to submit your project design report by **23:59:59 of 4 March, 2022 (Friday)**. You should then continue to develop the high-level design of your project into a detailed design, preparing for the next phase. We do not require you to submit your detailed design, but it should be included in your Final Report as a comprehensive documentation of your project. Note that you are encouraged to start this phase and the next phase early since we only have around 2 weeks for the next phase. If you want to get our feedbacks earlier, feel free to ask the tutors and the instructor.

(2) Initial Code

In this phase, you will work as the programmers to implement your own design and collaborating using the **git** version control system. The objective of this phase is to get you familiar with **git** version control system. At the end of this phase, you are required to create a public repository on GitHub. You are required to submit your initial code and report by **23:59:59 of 18 March, 2022 (Friday)**. Your code will be handed in by providing the URL of your repository. For more information on the **git** version control system and GitHub, please refer to Appendix 2.

(3) Completed Code and Demo

In this phase, you are completing your project and are required to submit your final code of the project. Your final code should be self-contained and working. You will need to make a demonstration after the submission of the final code. Project Demo Day is scheduled on **week 12 and week 13**. Detailed project demo arrangement will be announced in the course website, and the demo schedule (about 15-minute per group) will be signed up accordingly (please note our announcement on Blackboard). Your final code for demo, a 5-min demo video, and slides for demo should be submitted by **23:59:59 of 18 April, 2022 (Monday)**. During the demo, in addition

to playing the video, we will also try to play with your software to test the functions described.

(4) Commented Code and Final Report

After the demonstration, you are required to prepare and submit a Final Report and commented code of your project by **23:59:59 of 13 May, 2022 (Friday)**. In addition to reporting your comprehensive project, which includes your finalized detailed design, the Final Report should also show what software engineering techniques/tools you have applied in the project, possible future enhancement of your project, and whatever lessons you have learned. Detailed requirements of the Final Report are provided in Appendix 1. The revised final code should be commented as detailed as possible, with all known bugs removed.

5. Grading Criteria:

The followings are the project schedule of different phases:

Phase Deliverables	Weightings	Due Date
0. Team Formulation	--	**Feb. 14 (23:59:59)**
1. Project Design Document	**10%**	**Mar. 4 (23:59:59)**
2. Initial Code	**5%**	**Mar. 18 (23:59:59)**
3. Completed Code and Demo	**50%**	**Code submission: Apr. 18 (23:59:59), Demo Day: week 12 and week 13**
4. Commented Code and Final Report	**35%**	**May. 13 (23:59:59)**
Total	**100%**	

5.1. Project Description

The Project Design Document should be written with text font Times New Roman and size 11. The main body of the design document should be **no more than** 10 pages. Marks will be deduced if the format requirements are not met. The document will be graded upon the innovativeness of the proposed application and the clarity of the presentation. Please refer to the Appendix for more information.

5.2. Initial Code

The initial code will be graded on the availability of your GitHub repository. The implementation of your project is **not** required.

5.3. Completed Code and Demo

The Completed code and Demo will be graded upon the functionalities of your application. The grading criteria is listed as follow:

- Required Project Features (60%):
  The required project features are functionalities that all projects should meet. There are 10 functionalities in total. You will get 6 points for each functionality you implemented. We

provide a checklist for your reference.

Functionality	Requirements
User Interface	
Basic UI Design	A graphical UI design.
Database	
Database Integration	Integrate a database like MySQL.
User Management	
User signup	Create a new user profile.
User Login & Logout	Let user login and logout your application. Users should be able to conduct more operations after login.
Email Verification	Send email to verify a user identity when a u=ser signup.
Profile Photo	Each user should be able to upload a profile photo.
Password Modification	Users should be able to modify their passwords.
Admin User	
Admin user interface	The admin user should have different interface from normal users.
Reset user password	The admin user should be able to reset the password of any normal users.
List all users	The admin user should be able to access the user profile of all normal users.

- Advanced Project Features (40%)
  - Project specific functionalities
  - Deployed on the Cloud
  - Mobile Integration
  - Pretty UI
  - …

During the demo, we will check all the required project features listed in the checklist above. For the advanced project features, you will be graded based on your project specific functionalities and other advanced techniques you employed. You can show as many interesting features in the demo as possible and your grade will be considered accordingly.

5.4. Final Report and Commented Code

The final report should also be written with text font Times New Roman and size 11. The main body should be **no more than** 30 pages. Marks will be deduced if the report doesn't satisfy the format requirements. The reports will be graded based upon the technical content and the clarity of the presentation, and the final revised code will be graded according to its modular structure,

comments, and cleanness. However, it is not enough to meet all the listed requirements to receive the maximum grade. For example, having a perfect report for a trivial project will result in a very low overall grade. Thus, the overall quality and functionality of the project is the key scaling factor for all other aspects of the grade. Moreover, the techniques and tools you used to develop the project and the test cases and scenarios you designed to verify the system are also important factors considered in your project grade.

Although generally the project grade will be based for the whole team and will not be assigned individually to the members, each team member must be aware that a major part of his or her final project grade depends on the teamwork. Failures to cooperate with other team members and to invest equitable amount of effort can lead to undesirable outcomes, particularly when other team members raise complaints about the non-participating members. In the project Demo Day, we will verify with each team regarding fair contribution of team members to each project.

6. Submission

There are two report submissions (i.e., Project Design Document and Final Report), three code submissions (i.e., Initial Code, Completed Code, and Commented Code), and one demo package submission (i.e., demo video and slides). The submissions need to meet the following requirements:

6.1. Report submission and demo package submission

Each project group should submit the softcopy of the report to Blackboard System before the deadlines. The followings are the required names of the attached documents of different phases:

"Group** Project Description Document"
"Group** Final Report"
"Group** Demo Package"

Please replace the "**" with your group ID.

6.2. **Code submission.**

ALL your project stuff (including source code, images, flashes, databases files, etc.) should be conducted by **Git**. Git actions play an important role in evaluating your project coding phases. You should take advantage of the version control system to support the development and documentation of your project. You MUST submit your project via **Git**, and faithfully record your coding activities. We will NOT accept any code submissions via other approaches. Moreover, the tutors will check your version control logs when marking your coding efforts.

Appendix 2 provides a guide for code submission. Further information will be provided in the related emails or information on the website.

Appendix 1: Guidance for Documentation

The reports should be submitted by the whole team (one report per team) and the report will be graded as a whole for the team members.

Project Description Document focuses on the high-level system functionality and architectural design. In typical software engineering project, there should be a detailed design document which include detailed classes diagrams, component descriptions, pseudocodes, and programmers should be able to implement the system based on this design document. We skip this for reducing project development workload (except for DFDs to show your detailed design), and defer the detailed design to the Final Report. In the Final Report, besides the design details, the work assignment of project members, code statistics (e.g., lines of code, number of functions, etc.), project highlights, test case design and results, and lessons learned should be reported.

Each document should contain three parts: cover page, table of contents, and detailed contents.
The Cover page must contain the following information

- Name of Document
- Project Title (You can create your own project name under the assigned topic)
- Document Version Number
- Printing Date
- Group ID
- member names and SID
- School & University

The followings are the detailed outlines for the two reports:

1.1 Project Description Document (Main body *no more than* 10 pages)

The project description document is mainly focused on the purpose of the product you are designing, and its high-level descriptions. You should describe the objective, expected customers and market, features, and architectural design of your Web application project. Moreover, the project background, architecture diagram and brief descriptions of the system components should be provided. The recommended outline is listed as follows.

1 INTRODUCTION
    1.1 Project Overview
    1.2 Objective
    1.3 System Features
2 BACKGROUND
    Emphasize why your design the product, and its most attractive functionality/features.
3 SPECIFICATION (e.g., DFD)

4 SYSTEM ARCHITECTURE
- 4.1 Architecture Diagram
- 4.2 System Components

1.2 Final Report outline (Main body *no more than* 30 pages)

Your Final Report should include five sections (introduction, system architectural design, detailed description of components, user interface design, and test plans/cases). The detailed requirements of each section is listed below. A recommended report structure is attached at the end of this document. You are not restricted to the recommended report structure, but your own structure should contain AT LEAST the following listed sections and sub-sections.

Introduction

In this section, you should describe the general overview of your project. Contents that you should report in this section include background, group members and the workload of each group member, the problem to be solved, your motivation, project key features, etc.

In the *highlights* sub-section, you can include major attractive functionality and advanced features in your project. The expected customers and users can also be included, and you can elaborate why they will enjoy your project.

In the *project statistics* sub-section, you should indicate the LOC (lines of code) and McCabe's number (to be taught in class) of your project, including the main function and other major functions. These metrics can be as extensive as possible, and down to module level. Present your project statistics clearly, such as using a tabular format.

System Specification

In this section, you should describe the system specification of your project. You have to include the DFD diagrams to introduce the operational specification of your project. Besides, you are encouraged to use FSM to explain non-trivial system operational behaviors, for the whole or some selected parts of the system.

System architectural design

In this section, you should describe the architectural design of your project. Contents that you should concern in this section include: whole system architecture overview, key components, key interface description etc. Moreover, you should indicate the connections among different key components.

Detailed description of components by UML

In this section, you should use UML to provide the detailed description of at least the key components. Contents that you should concern in this section: UML diagrams (including use-case diagram, class diagram, and sequence diagram) of the major class, functionality of the component, list of major function, etc.

User interface (UI) design

In this section, you should present the UI of your project. This section could be like a user manual of your project. You should teach and guild the users by walking through your UI operations. The use of screenshots is needed to indicate your UI overview and the result after certain user actions.

Testing

In this section, you should present the test plan, test case design, and test result of your project.

In the *test overview and test plan* sub-section, you should contain your test approach, features to be tested, plan for testing, testing tools if any, and the testing environment.

In the *Case-N* sub-section, you should concern the objective of each test case, input, expected outputs, Pass/Fail criteria, and so on.

Lessons learned

In this section, please describe what lessons you have learned from the software engineering exercise of this project, what would you do differently if you can re-do the project again, and any positive and negative experience you have with the project.

Recommended Final Report structure

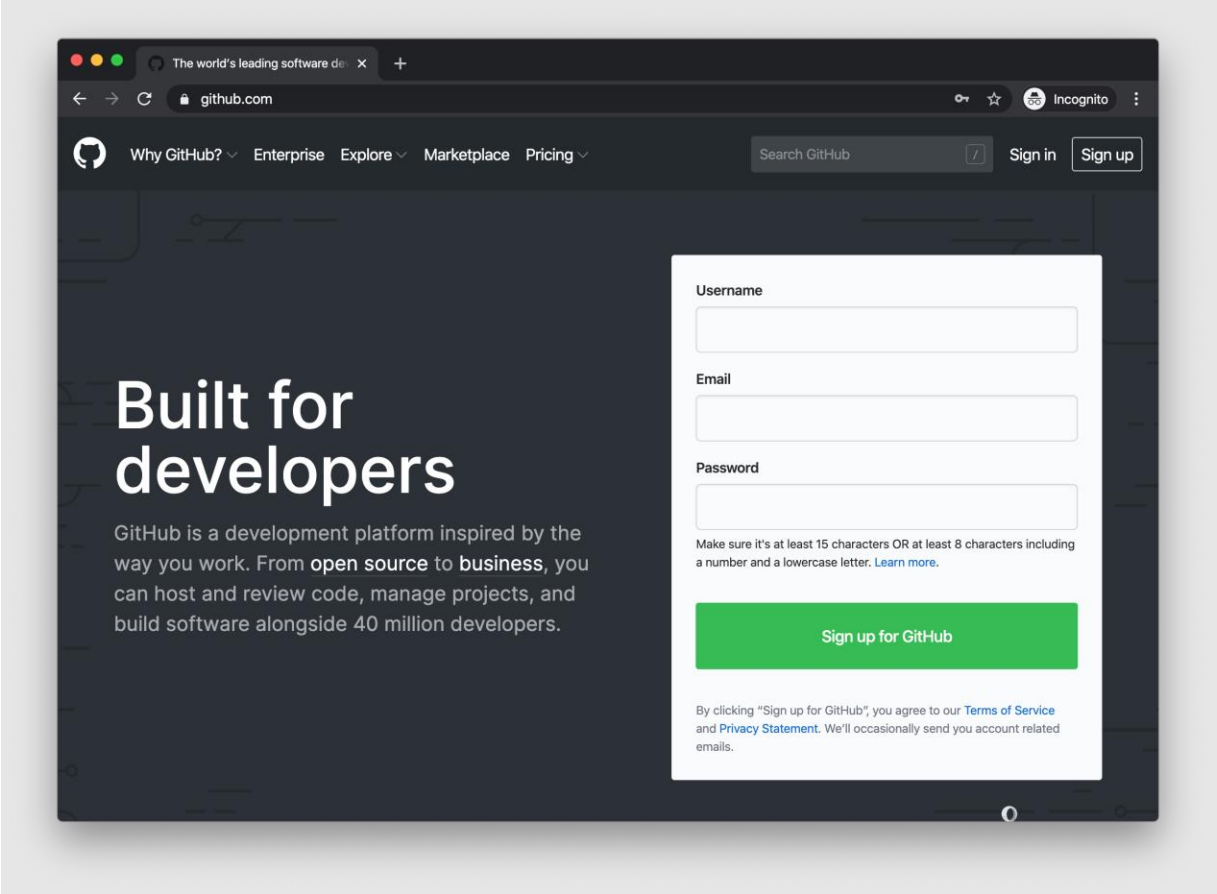
Cover Page
Table of Contents
1 INTRODUCTION
    1.1 Project Overview
    1.2 Objective
    1.3 Highlights
    1.4 Project Statistics
2 SYSTEM ARCHITECTURAL DESIGN by DFD
    2.1 System Architecture
    2.2 DFDs
3 DETAILED DESCRIPTION OF COMPONENTS by UML
    3.n Component-n
        3.n.1 Structural Diagram
        3.n.2 UMLs
        3.n.3 Functionality
        3.n.4 Procedures and Functions
4 USER INTERFACE DESIGN
    4.1 Description of the User Interface
    4.2 Screen Images
    4.3 Objects and Actions
5 TEST
    5.1 Test Overview and Test Plan
    5.n Case-n

Appendix 2: Guidance for Code Submission

You MUST submit your project via **GitHub.com**, and faithfully record your coding activities.

1. Create a GitHub account (if you don't have one)

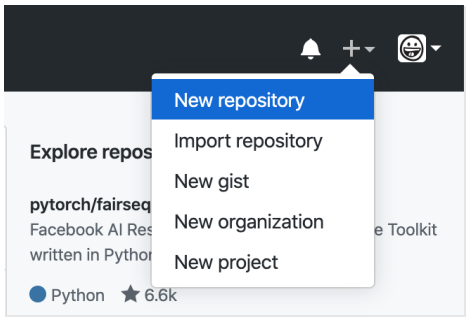
GitHub is the leading code management platform worldwide. If you do not have an account, you can navigate to https://github.com and Sign up.



2. Create a new repository

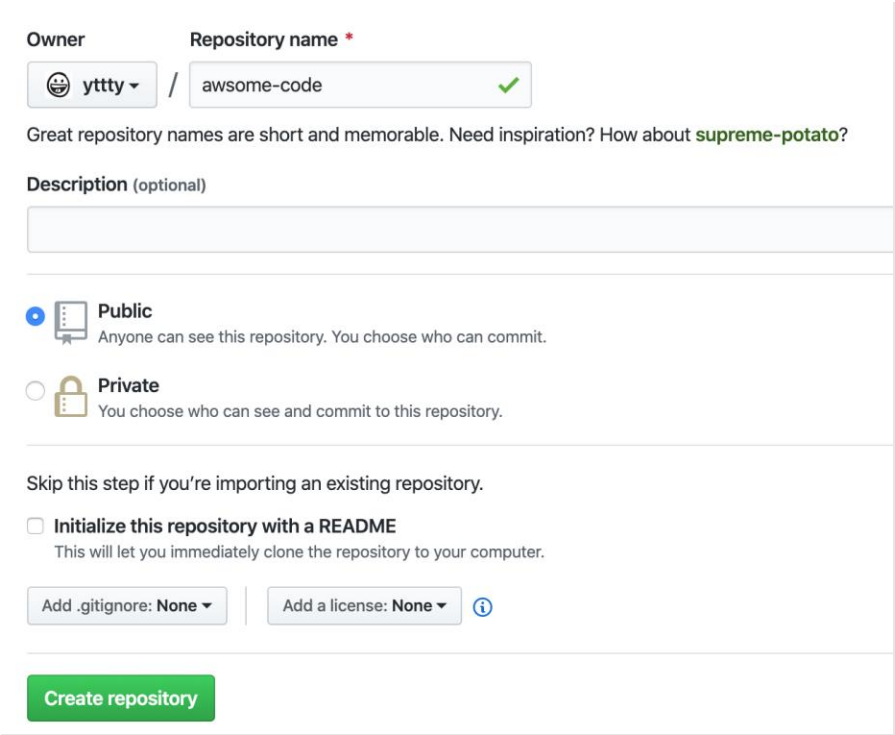
To facilitate submission, you are required to set up a git repository on GitHub.com.

Click the **"+"** on the upper right corner of GitHub, and click "New repository"

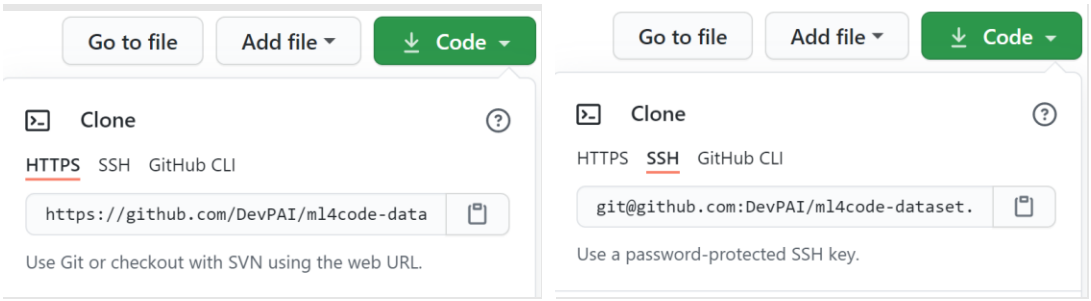


Then enter the name of your repository and choose the visibility. You can set up either public or private repository. **For public repository, you need to submit HTTPS URL of your repository (e.g., https://forms.gle/rPxUpm4CAb42JGWu5). For private repository, you need to follow the guide below to add ssh key and submit SSH URL of your repository (e.g.,**

git@github.com:yttty/my-awsome-project.git).



You are required to submit the **HTTPS URL** or **SSH URL** of your repository to the Google form (**https://forms.gle/DFjHVpL14WYysjGS9**) **two days before Phase 2 due (i.e., 16 Mar.)**. As specified in the project specification, we will pull the latest code on the due date of **Phase 2, Phase 3, and Phase 4**.

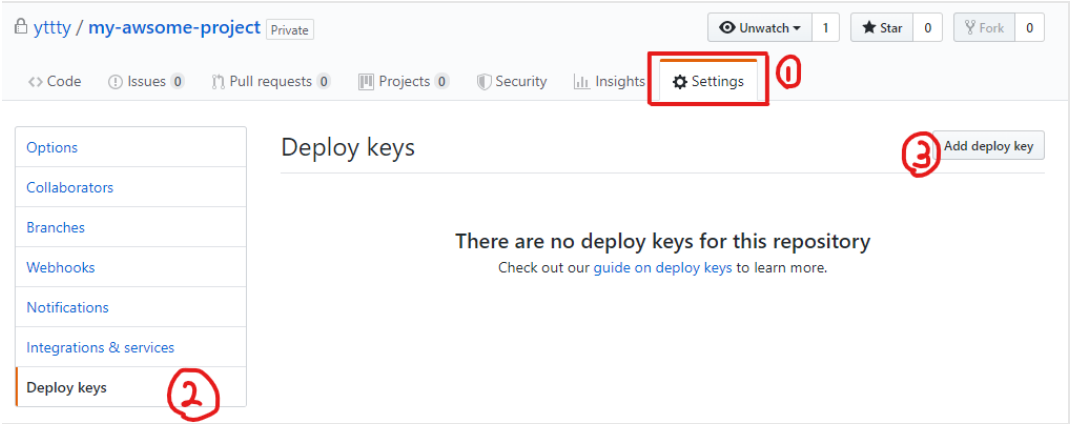


3. Add deploy key to the private repository

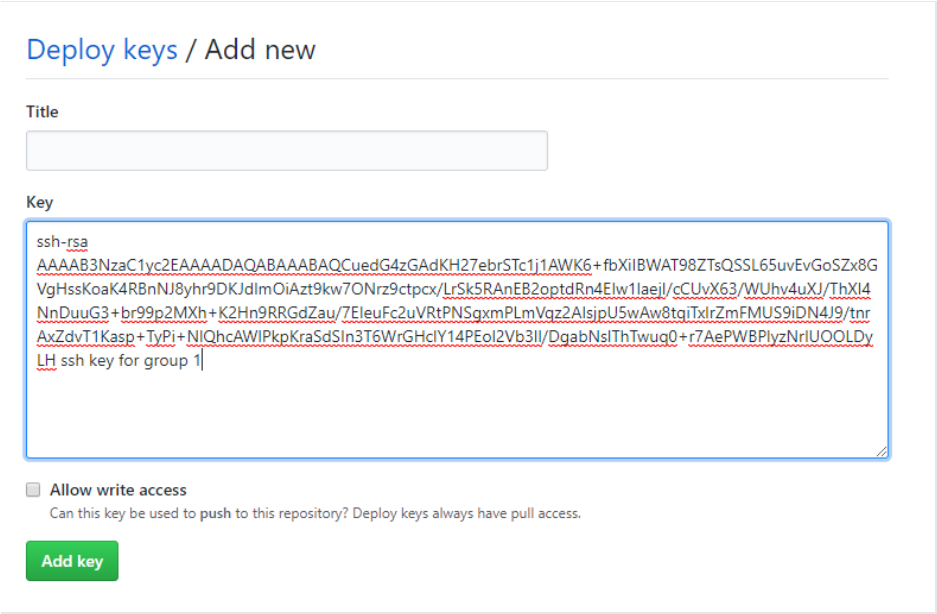
If you choose to create a private repository, you need to add deploy key to your project so that TAs can access your code. Please be reminded the member capacity of a private repository of a free account is limited to 3. You should apply for Student Account to increase the member capacity to enable everyone to contribute code.

First, download the key from the project page of BB:Content; extract the zip file; open the public key of your group with a text editor and copy the corresponding public key **of your group**. Please ensure you add the right public key, otherwise TAs cannot access your private repository.

Then, go to your GitHub repository, navigate to settings->Deploy keys->Add deploy key



Paste the public key you just copied, then click Add key.



4. Read More

- If you are new to git, start with this simple guide.
- The official git documentation: https://git-scm.com/documentation.
- Git and GitHub Tutorial https://www.youtube.com/watch?v=xuB1Id2Wxak