Assignment 02: Building an Application for Task Management amongst multiple Users

Barry Denby

February 17, 2022

1 Assignment Information

Course:	MSCC/MSCBD
Stage / Year:	1
Module:	Cloud Computing
Semester:	2
Assignment:	2 of 3
Date of Issue:	2022-02-17
Assignment Deadline:	2022-04-17 @ 23:55 (End of week 9)
Assignment Submission:	Upload to Moodle
Assignment Weighting:	16% of Module

2 Introduction

NOTE: read the whole assignment brief first before implementing it contains very important information

In this assignment you will be building a task management system that can deal with tasks of multiple users. Users will have the ability to create task boards which are collections of tasks. Users will also have the ability to invite other users to see and interact with their created boards.

For example if I create a board I can add users to it and anyone who creates a task can assign that task to any user who is part of that board. Users should be able to edit and delete tasks as well as mark them complete. The only person who can delete the entire board though is the original creator.

For this application you will also be exploiting the parent child relationships available through the NoSQL database. You will also be required to build this application without using a query of any kind. If you organise and manage keys correctly you should be able to directly access everything without the use of a query.

NOTE: This is an individual assignment. It is not a group assignment. You can discuss ideas/algorithms but you cannot share code/documentation

3 Submission and Penalties

You are required to submit two separate components to the Moodle

- An archive containing your complete Google App Engine Python project. The accepted archive formats are: zip, rar, 7z, tar.gz, tar.bz2, tar.xz. The use of any other archive format will incur a 10% penalty before grading.
- A PDF containing documentation of your code. **If you do not provide documentation your code will not be marked.** Copying and pasting code into a PDF does not count as documentation.

There are also a few penalties you should be aware of

- Code that fails to compile will incur a 30% penalty before grading. At this stage you have zero excuse to produce non compiling code. I should be able to open your project and be able to compile and run without having to fix syntax errors.
- The use of libraries outside the SDK will incur a 20% penalty before grading. You have all you need in the standard SDK. I shouldn't have to figure out how to install and use an external library to get your app to work
- **An omission of a git repository attached to your email address that is registered for GCD will result in your application and documentation not being graded.**
- The standard late penalties will also apply

You are also required to submit as part of your archive a working Git repository.

- When I unpack your archive there should be a .git directory as part of it.
- This should be a fully working **local** git archive. It should not require access to a remote repository
- You are not permitted to upload your work to Github, Gitlab, or any other publicly visible git repository (assignment will be marked as a zero if it is)
- If you need a remote git repository the only permitted one is the college provided Gitlab which can be found at gitlab.griffith.ie
- There must be a minimum of seven commits in the git repository, one per completed bracket.

Very Important: Take note of the groups listed below. These are meant to be completed in order. Groups must be completed in full before the next group will be evaluated. Completed will mean that all tasks in the groups are visible and testable. If a single one is not visible and testable further groups will not be considered. e.g. if there are four tasks in Group 1 and task 3 is skipped or not visible or testable then Groups 2, 3 and 4 will be ignored. Documentation

will be treated separately irrespective of how many Groups you have completed.

You should also be aware that I will remove marks for the presence of bugs anywhere in the code and this will incur a deduction of between 1% and 15% depending on the severity. If you have enough of these bugs it is entirely possible that you may not score very many marks overall. I want robust bug free code that also validates all user input to make sure it is sensible in nature. Please be aware of the major bugs section. If any of these bugs are present in your application you will lose 20% for each one up to a maximum of 60%

4 Plagiarism

Be aware that we take plagiarism very seriously here. Plagiarism is where you take someone else's work and submit it as if it was your own work. There are many different ways plagiarism can happen. I will list a few here (this is not exhaustive):

- Finding something similar online (full implementation or tutorial) that does the same job and submit that.
- Finding something similar online (full implementation or tutorial) and transcribing (i.e. copying it out by hand)
- Working together on an individual assignment and sharing code together such that all implementation look the same.
- Getting a copy of someone else's code and submitting/transcribing that
- Paying someone to do your assignment
- Logging into someone elses Moodle account, downloading their assignment and uploading it to your own Moodle account.

I've had to deal with many cases of plagiarism over the last six years so I can spot it and diagnose it easily, so don't do it. To prevent plagiarism include but not limited to the following:

- Do all your code by yourself
- Don't share your code with anyone, particularly if anyone looks for a copy of your code for reference.
- Don't post your code publicly online. Remember the use of GitHub, Gitlab, BitBucket etc is prohibited.
- If you need to find information online only query about very specific problems you have don't look for a full assignment or howto.
- Change the default password on your Moodle account. The default password can be determined if someone is connected to you through social media or they get one or two details from you.

Be aware that if you submit your assignment you accept that you understand what plagiarism is and that your assignment is not plagiarised in any way.

5 Coding Tasks (80%)

- Group 1 (20%)
  1. Write the shell of an application that has a working login/logout service.
  2. Write a model to describe a task board (collection of tasks and users), a task, and a user.
     - Create a new user object when a user logs in for the first time.
     - Make sure appropriate types are used.
     - Make sure proper parent child relationships are used.
  3. Add the ability for a user to create a task board. If a board is created it should be visible to the created user.
  4. Add the ability for a user to click into a task board. This should go to a separate page and display the task board.
- Group 2 (40%)
  5. Add the ability to invite a user to a task board.
     - This should be automatically added and visible to that user.
     - Only the creator of the board can add a user to that board
  6. Add the ability to add a task to a task board.
     - A task should contain a title, a due date, and a checkbox or similar to mark completion (should be defaulted to not complete).
     - It should be possible to assign the task to one of the users that is currently on the board.
  7. Modify the task board list to include boards that a user has been added to.
  8. On the board add in the ability to check a task. If the task is checked it should be marked as completed and a date and time marking completion should be displayed and persisted.
- Group 3 (60%)
  9. Add in the ability to edit and delete tasks for anyone who is a member of the board.
  10. Add the ability for the board owner to rename the board.
  11. Add the ability for a board owner to remove a user from a taskboard.
      - If a user has been removed from a board then mark all of their tasks as unassigned and highlight them red.
      - The user should not be able to see the board anymore.
      - Only the creator of the board has the ability to remove a user from the board
  12. Add in a marker to indicate if a user is the creator of a board on the task board list.
- Group 4 (80%)

13. If an unassigned task is then edited and assigned another user drop the red highlighting.
14. Add in counters to display the number of active tasks, number of complete tasks, total tasks, and total completed today.
15. Add the ability to remove a task board this should only be done by the creating user and only if all non-owning users and all tasks have been removed.
16. UI Design: Well thought out UI that is easy and intuitive to use.

- Major bugs (20% reduction for the presence of one of these bugs upto a maximum of 60%)
  - A task with the same name can be added twice to the same board.
  - a non-owning user can rename the board
  - a non-owning user can remove another user from a board
  - a non-owning user can remove a board
  - a board can be removed if there is a task still present
  - a board can be removed if there is a non-owning user present

6 Documentation Brackets (20%)

NOTE: Documentation should be around 1,500 words in length total

1. (0 to 10%): Document every method in your code from a high level perspective. i.e. give an overview of what the method does. Do not copy and paste code you will be penalised for this.
2. (10 to 20%): Document every datastructure and model you have used in your code and why you chose them.