# ASSIGNMENT 4
# Git, GitHub, APIs and Group Work

(handout for students)

## Total: 75 marks

This Git, GitHub, APIs and **Group Work Assignment is creative**. We will look for and mark:

1. Good use of Git and GitHub
2. Demonstrating teamwork
3. Good code practices (e.g. comments, formatting, consistent capitalisation, naming, etc.)
4. Creativity that is taken to implement the answer
5. Using constructs taught in API sessions (Flask, Requests etc.)
6. Effective use of coding constructs (correct use of techniques in a way that makes sense in the picked theme)

Total marks will be awarded for demonstrating concepts fully, there will be partial marks available for attempt.

**There are 2 exercises for this assignment.:**

1. **Git and GitHub team exercise**
2. **Build your own API exercise**

## What to submit:

Submit a **link** to your team's GitHub repository and **add** all/the marking **instructor as a collaborator** to this repository. Make a **pull request** for Question 2 that the instructor can review and comment on as it would be in your future workplace code review. Make sure to invite instructors to your individual repositories too for further submissions.

## Top tips:

- Start early
- Start with Q1 and then Q2
- Work together on all parts
- Make sure to schedule some meetings with your group outside of class to set up the GitHub repository together and to work on code using pair programming
- Read all requirements carefully before starting
- Check the mark distributions as provided below

## Question 1
## (33 marks)

**Set up GitHub accounts for yourself and create a repository for your team to collaborate in.** Demonstrate Git and GitHub usage for collaborative projects using pull requests, reviews and more! Take **screenshots** and save them either to a markdown file or a PDF file on the repository.

**Make sure to invite your Assigned instructor/All instructors to your individual repository and team's repository for marking and review purposes.**

Marks:
1. 6 marks for demonstrating Git and GitHub command usage (one of the requirements)
2. 4 marks for team collaboration with Git and GitHub (one of the requirements)
3. 4 marks for creativity
4. 19 marks for meeting the remaining requirements

You should:
+ Create **GitHub accounts** (one for each member, can use existing ones) and make a note of them/ display them for easy verification
+ One student to create a **private team repository** for this Assignment
+ Create a **group name and a short slogan**
+ Create a **README.md file** and use markdown to make something interesting like showing off the group's skills and interests
+ Use at least **6 different markdown** text formatting **features.** See: [https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax](https://docs.github.com/en/get-started/writing-on-github/getting-started-with-writing-and-formatting-on-github/basic-writing-and-formatting-syntax)
+ Demonstrate a scenario of how to **work as a team on GitHub**
+ Show the whole **workflow** (from git add to opening a pull request and closing it), each correct step gives a point
+ Use **meaningful commit messages**
+ Creating **branches** for each member or feature and using meaningful naming
+ Add a file to a branch
+ **Merge branches**
+ Create **.gitignore** (can be empty) and briefly explain what it is for
+ Create **requirements.txt** (can be empty) and briefly explain what it is for

Each of the requirements mentioned above is worth **1-6 points.**

## Question 2
## (42 marks)

**Design and implement a simple API as done in class.**
Remember to come up with a unique creative problem or scenario and show real-life expected use of your database!

Marks:
1. 5 points for creativity
2. 4 points for commenting and code quality
3. 33 points for meeting the requirements (coding concepts taught and their use)

You should:

+ Implement API endpoints with appropriate functionality
+ Implement one **additional endpoint** of your choice (can be POST or GET but with a different implementation)
+ Implement client-side for the **3 API endpoints**
+ Create a **MySQL database with at least 1 table**
+ Have a **config** file (do not leave your private information here)
+ Have **db_utils file and use exception handling**
+ Use appropriate **SQL queries** to interact with the database in your Flask application, and demonstrate at least **two different queries**.
+ In main.py have a **run( ) function**/call the functions to simulate the planned interaction with the API, this could include welcome statements, displaying etc., (hairdressers booking example from lesson)
+ Have correct but minimal imports per file (do not import things you do not use in the file)
+ **Document** how to run your API in a **markdown** file including editing the config file, any installation requirements up until how to run the code and what is supposed to happen.
+ Submit in GitHub as a **Pull Request**

Each of the requirements mentioned above is worth **1-5 points.**

If time permits:
*Test your API (not part of the marking guide but testing and Test Driven Development (TDD) are essential topics)*

**If you cannot come up with a creative idea** you can use this scenario, but **5 points out of the total mark are for creativity** and you will receive 0 for creativity:
*"Restaurant reservation system API. This API should allow users to make reservations for a restaurant, view available tables, and cancel reservations if needed. Each reservation should include the customer's name, reservation date and time (saved in the DB), and the number of guests in the party. Additionally, the API should keep track of the available tables and their seating capacity."*