# DSCI 551 – Fall 2025

## Homework 1: JSON & Firebase

**Due: 11:59pm, September 15, 2025, Monday**

**NO Late Submissions will be accepted!**

Suppose we want to build an app that uses Firebase real-time database to manage a “todo” list, that is, a list of tasks to be completed. The architecture of the app can be depicted below.

The app consists of three parts.

1. Firebase server: we assume that the tasks in the todo list are stored at <https://dbxxx-firebaseio.com/tasks>, where <https://dbxxx-firebaseio.com/> is the root URL of the database server.
2. Todo DB client, which talks to the database server to perform varied functions on the list. The client is implemented as a Python module, named TodoClient.py, which contains a class TodoClient that encapsulates all the functions.
3. Todo user interface, which takes requests from users and calls the relevant functions in the DB client to fulfill the requests. We assume a simplified UI where the code is in a Python file called “todo.py”. See the example attached.

A diagram of a flowchart

AI-generated content may be incorrect.

Your job in this homework is to implement the TodoClient.py. Your implementation should be based on the provided template by completing the codes. In particular, you need to supply the codes for: clearing the todo list, add or delete tasks, mark a task as completed, retrieve all tasks or tasks with specific status. Note that a task is “pending” (status) when just added, and changes to “completed” when marked as completed. Note also that:

* add\_task should report error when the task already exists;
* delete\_task and mark\_task\_as\_completed should report error when the task does not exist in the todo list;
* get\_all\_tasks should return tasks and their status in a dictionary;
* get\_task\_by\_status should return a list of tasks in the given status, and empty list if no such tasks exist.
* See the attached template for more information on the methods and their required behaviors.
* You can have additional codes in TodoClient.py, other than supplying the code for the body of required functions.
* You need to rename “TodoClient-template.py” to “TodoClient.py”.

You can assume that the tasks are stored as follows in Firebase. Note that “tasks” is a child node of the root node of database. Note that “do homework” and “eat dinner” are task names, while “pending” and “completed” are their statuses.

A black text on a white background

AI-generated content may be incorrect.

You can assume that the index on the values of tasks (i.e., statuses) has been created for you when we test your code.

A computer code with text

AI-generated content may be incorrect.

**Tips:**

* If you are using Google colab to develop your code, you should restart the session whenever the new version of the module (TodoClient.py) is uploaded to the file folder.
* Requests library: <https://requests.readthedocs.io/en/latest/api/#requests.Response>

**Submissions:**

1. Your completed TodoClient.py. You are allowed to use “requests” and “json” libraries in the code.
2. A notebook file, todo.ipynb, that contains the code testing the functions of the client program. For each function (e.g., add\_task), you should test its different input and output cases (e.g., success or error). Please separate individual tests in separate cells of the notebook and show the output of tests. See example notebook attached. Your testing cases (e.g., tasks and sequence of operations) should be different from that in the example.
3. Add a Google colab link for your todo.ipynb file at the beginning of the file (as comment, see example notebook). Please make sure the link is accessible.