# Table of Contents



# Feedback | Group 1

## Milestone 1 | 2Oct-13Oct

1. **Define the problem:** done
    - Please switch from .docx to .md format by Milestone 2 Good
2. **Finalizing roles:** done
3. **Create a product roadmap and prioritize functionality (items):** done
    - overall, the roadmap is quite realistic
    - you only included must and could haves
    - I would recommend going with Thompson Sampling; there is no need to compare algorithms at this stage. Remember we need to make a skate or a scooter.
4. **Creating the GitHub repository included readme.md and `.gitignore` (for Python) files:** partially done
    - during the remote repository initialization, you should have selected add `.gitingorne` with the Python option
5. **Create a virtual environment in the above repo and generate requirements.txt (`venv` must be ignored in git)** Not Done
    - it seems you did with conda create, without adding `--no-default-packages` option. As a result, we have a bunch of extra packages.
    - please fix it and push it to GitHub by the end of Milestone 2
6. **Push *point 1, point 3, point 5 (requirements.txt)*.** not done :
    - see point 5
7. **Complete the first chapter of Developing Python Packages:** done
    - completed by everyone
8. **Create a private Slack channel in our Workspace and name it Group-{number}** done
9. **Schedule a call with me and Garo or come during the office hours:** done

By the end of the Milestone 2, you must complete the tasks mentioned above. Feel free to reach out if you have any questions.

- Fix requirements.txt file (virtual environment)
- Fix `.gitignore`

**Grade:** 7/10

# Milestone 2 | 16Oct-27Oct

## Fixes From the Milestone 1

I can see that you have managed to fix:

- The requirements.txt
- `gitignore`

## Milestone 2

1. **DB developer:**
    - Design the database using Star schema (provide ERD): done
    - Insert Sample to data done
2. **Data Scientist:**
    - Complete data generation/acquisition/research: done
    - Select data from DB: done
    - Insert data to DB: done
3. **API developer:**
    - Select data from DB not done
    - Insert data to DB not done
    - Update data in DB not done
4. Finish the second chapter of Datacamp course
5. Finalize file/folder structure: relative imports must work properly
    - docs folder: putting all the documents there
    - models folder: putting modeling-related classes, functions
    - api folder: api related stuff
    - db folder: db related stiff
    - initialize `__init__.py` files accordingly (see Datacamp assignment chapter 1 and chapter 2)
    - logger folder: I will provide this module

I can see Anahit's, Naira's and Mher's contributions.

I would recommend:

- to replace `CallTracking.sql` with python code, in order to make it easier on the long run.
- make package name relevant (search the available names in pypi)
- Use `logger` module, instead of the `print()`: start with debug level

By the end of the 3rd Milestone you must:

1. API Developer: add api part and test (select, insert,update)
2. DB developer: add complete update part in order to pass the function to api developer

Overall you have done great job: **15/20**

Kindly note if you manage to complete the remaining parts by friday, I will give you **20 points**

# Milestone 3 | 30Oct-10Nov

1. Complete things from *Milestone 2* restructuring is done
2. Finish the **third** chapter of Datacamp course (please complete only the 3rd one) completed by everyone
3. **API Developer:**
    - Create a `run.py` file for an API (find the minimum workable example here). done
    - Test it on swagger done
    - following request types must be available to test (GET, POST, PUT), will provide more details on Friday. done
4. **DB developer:**
    - complete the methods from `SQLHandler()` class Not done: there are still methods to complete
    - finalize the documentation for `schema.py` by using `pyment` package
    - finalize the documentation for `SQLHandler()` by using `pyment` package
5. **Data Scientist:** start working on algorithms by
    - completing the bandit and experiment part: `BernoulliThomsponSampling()` Note done
    - create `BernoulliReward.ipynb` outside of the package and show the output Not done

I can see multiple contributors I could not see the application of an algoirithm on outside of the package. It has been expected to have something like the homework.

In order to manage to finish by the deadline we need to concentrate only on one use case provided in `cases.md` file.

Merge(fetch) API with the main branch in order to have updated modelling/api part

**Grade:** 15/30

# Milestone 4 | 26 Nov-6 Dec

1. Complete things from *Milestone 3* Done
2. Finish the final(4) chapter of Datacamp Assignment Done
3. Finish Thompson Sampling Done
    - algorithm: must be imported in api
    - api:
        - get an arm from db (unless its the first time)
        - post (manually on swager) 1/0 to db
        - put: update the reward for the bandit in db
    - db: in `SQLHandler()` here you need to have following methods
        - `insert_one()`
        - `update_one()`
        - `select_one`
4. Apply only for one case
5. In readme.md file provide end to end steps for package usage Done
6. Finalise the documentation. Instead of `pyment` you can can use this extantion as well. Done

# Final Feedback

## Group Project Scope

- Finding a Marketing related problem
- Understanding the methodology of the analysis
- Building a Python package with following mandatory modules:
    - Predictive Model (component)
    - DB
    - API
    - Logging (provided by me)
- Post to `Pypi.org`

## Submission format

In the Github Repository, the following structure must be available

```
| GitHubRepo
    | Docs
    | PackageName
        |SubPackage_1
            modlule1
            __init__.py
        |SubPackage_2
            module2.py
            __init__.py
        __init__.py
        utils.py
    other files (.gitignore, *config files)
    readme.md
    requirements.txt
    setup.py
    example.py/ipynb (Demonstrate all the funcionality)
```

Submision format is correct:

## Grading Methodology

Group Project is going to be graded according to the following poins:

1. **Topic Relevancy:** matched
2. **Team Work:** I can see the contrubtutions from each member
3. **Availability of Documentation:** Perfect
    - Description of each `function()/method()`:
        - Parameters: description/docstrings
        - Returns: what do you expect as a return?

    - Description of Classes:
        - Use *dunder methods:* `__repr__`, `__str__`, for nice Class formulation
        - Describe the class
    - converting into a webapp using `mkdocs` or any alternative
4. **The code must run without any errors:** OK
    - logical
    - syntax
    - runtime error
5. **The availability of a Predictive Element** Updating reward
6. **Endpoints solving/touching the business problem** OK
7. **Successfully hosted on** `Pypi` Done

## Final Feedback

Technically you have done everything which was required. However you could have done way better by simply trying to understand what do you need to do at first.

# Grade

- **Grade from the Milestones:** 77
- **Grade from the Presentation:** 300/300
- **Final Grade: 377**