# PTON2 - TP2

Arthur Darcet (arthur@darcet.fr)

## 1 Setup

**Python environment:**

- Create and active a new virtualenv
- Install the `pymongo` package
- Create a `todo` package

**Mongo:**

- Create an account on `https://mlab.com/`
- Create a "new sandbox deployment" (the provider doesn't matter)
- Create a user/password for this DB

## 2 Backend

In the `todo` package, write a `db` module with the following functions:

- `create('my todo text')`
- `find()`: returns a list of
  `{'_id': 'some unique *string* id', 'text': 'my todo text', 'done': False}`
- `toggle('some unique *string* id')` toggles the `done` flag on a ToDo
- `remove('some unique *string* id')`

**Bonus:** make the `toggle` function atomic

## 3 Command line interface

Write a `__main__.py` file so that you can use your package with the following commands:

```
# python -m todo show
# python -m todo create "My note"
# python -m todo flag 59cbbac8d228eb0f672a5b9e
# python -m todo remove 59cbbac8d228eb0f672a5b9e
```

**Bonus:** add a `python -m todo repl` command that displays a shell input and takes show/create/flag/remove commands

# 4 HTTP server

Write an HTTP server that controls our ToDo app with `http.server`.
The API should accept the following calls:

- `GET /` should return, in JSON, a list of all the existing ToDo:
  `[{'text': 'My ToDo 1', '_id': '59cbbac8d228eb0f672a5b9e', 'done': True/False}, ...]`
- `POST /` creates a ToDo. The title of the ToDo is the body of the request
- `PATCH /59cbbac8d228eb0f672a5b9e` toggles the flag on the corresponding ToDo
- `DELETE /59cbbac8d228eb0f672a5b9e` removes it

**Notes:**

- **All** your API responses should include the two following headers:
  `Access-Control-Allow-Origin: *`
  `Access-Control-Allow-Methods: GET,POST,PATCH,DELETE`
- Your API should listen on `localhost:8000`
- You should be able to access **your** API through this webpage:
  `http://darcet.fr/tp2.html`

```
class HelloWorldHandler(http.server.BaseHTTPRequestHandler):
    def do_GET(self):
        self.send_response(200)
        self.send_header('Content-Type', 'text/plain; charset=utf-8')
        self.end_headers()
        body = 'Hello world!\n\nReceived headers:\n'
        for k, v in self.headers.items():
            body += '{}: {}\n'.format(k, v)
        self.wfile.write(body.encode('utf-8'))
```

**Bonus:** return proper HTTP errors codes, make the whole server as resistant to improper queries as possible (the server should not crash when the API is used improperly)