



IUT DU LITTORAL CÔTE D'OPALE

UE216 COMPLÉMENTS INFORMATIQUES

---

## Discord MemeBot

---

*Enseignant :*  
Antoine Gamelin  
Département Informatique

*Groupe :*  
Isaac Ruchalski  
Guillaume Le Louarn  
Théo Helary

## Table des matières

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Projet</b>	<b>3</b>
2.1	Bot Discord . . . . .	3
<b>3</b>	<b>Outils</b>	<b>3</b>
3.1	API imgflip . . . . .	3
3.2	node-rest-client . . . . .	3
<b>4</b>	<b>Contributions</b>	<b>4</b>
4.1	Commande makeMeme . . . . .	4
4.2	Déploiement sur Heroku . . . . .	5
<b>5</b>	<b>Introspection</b>	<b>6</b>
5.1	Difficulté d'intégrations . . . . .	6
5.2	Notion apprises . . . . .	6

# 1 Introduction

Discord étant un logiciel permettant à plusieurs utilisateurs de se rejoindre dans un chat vocal ainsi dans un salon de discussion permettant d'échanger en temps réel. De plus, l'application permet de créer des robots répondant à certaine commande quand on précise leurs préfixe.



**Figure 1** – Logo Discord

## **2 Projet**

### **2.1 Bot Discord**

Le projet MemeBot utilisant l'API Discord permet de créer un bot qui répond à des commandes prédéfini. Ce bot permettra à chaque utilisateur qui invitera le bot sur son serveur de créer à sa guise des images à but humoristique pour faire rire petits et grands.

## **3 Outils**

### **3.1 API imgflip**

L'API imgflip quand appelé avec get-memes permet de générer une liste de memes sous format JSON avec quelques propriétés tels que l'ID, name , url et le box-count. Toutes ces informations sont utiles afin de généré un meme dans les meilleures conditions.

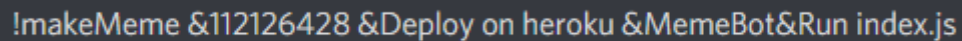
### **3.2 node-rest-client**

node-rest-client est un package node.js qui permet une connexion à n'importe quelle API REST d'obtenir en retour une valeur en JSON.

## 4 Contributions

### 4.1 Commande makeMeme

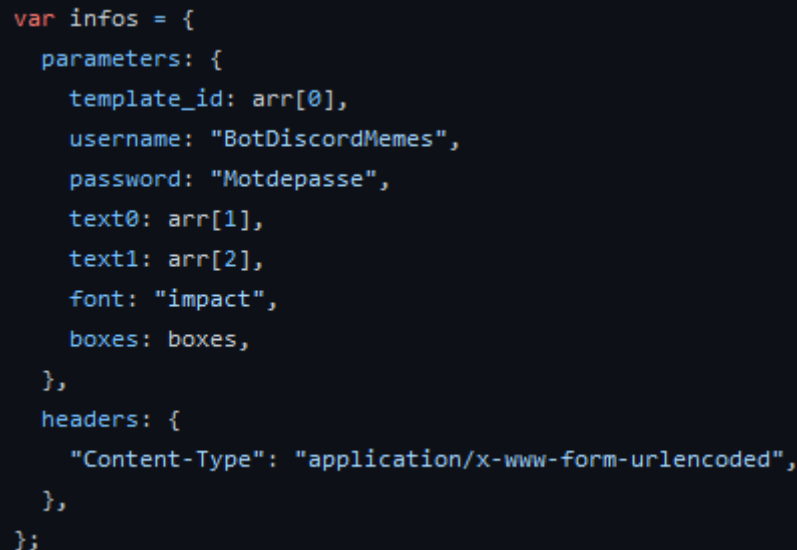
La commande makeMeme intervient en dernière lors de la procédure de création d'un meme. Cette commande va réunir plusieurs éléments ensemble. L'ID du template utilisé ainsi que les textes souhaités par l'utilisateur. On commence par appeler la commande avec le préfixe du bot. Cette fois-ci pour différencier les arguments dans la commande on va ajouter un caractère spécifique lequi va nous permettre de split les arguments à chaque lecture de ce caractère.



```
!makeMeme &112126428 &Deploy on heroku &MemeBot&Run index.js
```

**Figure 2** – commande !makeMeme et split

Après avoir split, il faut vérifier que le premier argument est bien un int qui correspond à l'id du template. Après cela, l'API imgflip n'utilisant pas de JSON mais des paramètres HTTP. Il nous faut donc générer une requête avec nos arguments.



```
var infos = {
  parameters: {
    template_id: arr[0],
    username: "BotDiscordMemes",
    password: "Motdepasse",
    text0: arr[1],
    text1: arr[2],
    font: "impact",
    boxes: boxes,
  },
  headers: {
    "Content-Type": "application/x-www-form-urlencoded",
  },
};
```

**Figure 3** – HTTP request

On peut observer ici le passage de paramètres "username" et "password". Permettant à l'API d'accepter ces paramètres pour l'authentification. L'API n'acceptant pas les tokens de connexions. Afin de garder une trace des requêtes effectués.

## 4.2 Déploiement sur Heroku

Heroku étant une solution cloud principalement pour le déploiement d'API. Déployé le bot était donc une étape importante pour la dépendance du bot. Avant il fallait exécuter l'`index.js` sur l'une de nos machines pour qu'il fonctionne.



Figure 4 – Meme

## 5 Introspection

### 5.1 Difficulté d'intégrations

J'ai rencontré quelques difficultés d'intégrations sur les fonctionnalités que j'ai intégrées. La commande `makeMeme` pour bien afficher les templates à plusieurs textes, l'API `imgflip` utilisant des boxes, j'ai créé un objet `boxes` pour ensuite `push` à l'intérieur de celui-ci mon objet `json` qui comporte avec les textes.

Ayant déjà déployé une API sur le cloud d'Heroku. Cependant, une spécificité m'a posé problème. En effet, il fallait préciser dans les fichiers `config` `profile`. Permettant de créer un worker qui va `run index.js` à chaque nouvelle instance du bot. Le problème venant du fait sans ce worker le bot ne pouvait `run` qu'une fois et donc sur un seul serveur ou alors le crash du bot.

### 5.2 Notion apprises

Ayant déjà utilisé une API sur un autre projet, cela m'a permis d'effectuer quelque révision. Cependant, lors de mes recherches pour optimiser ma requête HTTP, j'ai appris que certaines API utilisant une interface d'authentications autorise l'utilisation d'un token de connexion au lieu de devoir envoyer les informations d'`username` et `password` à chaque requête HTTP. Hors l'API `imgflip` n'autorise pas les tokens. Pour permettre un meilleur aperçu sur les logs de leurs requêtes.

De plus, j'ai appris quelques notions sur lors de la rédaction de mon rapport LaTeX sur `overleaf`.