Syllabus projet

Année : **2021-2022**

Enseignant(s)	Email(s)
CHAPONNEAU Brice	bchaponneau@myges.fr

Application d'emailing

1 Matières, formations et groupes

Matière liée au projet :

Formations : -

Nombre d'étudiant par groupe : **2 à 3**

Règles de constitution des groupes: **Libre**

Charge de travail estimée par étudiant : **20,00 h**

2 Sujet(s) du projet

Type de sujet : **Imposé**

Application d'emailling

Créer une application d'emailing :
- 1 API (framework de son choix)
- 1 base de données (de son choix)
- n fichiers html
- 1 fichier de config .env

+ Utiliser les fichiers en pièce jointe pour le MCD et l'ARCHI attendue

=== TODO ==========
+ Construire la base de données (local ou hébergée (ex: elephantsql)) via le MCD fournit

+ Ajouter un répertoire templates avec des fichiers html qui seront les modèles de messages

+ Créer un fichier de config .env qui spécifie la chaine de connexion et autres variables globales

+ Créer une API Rest qui doit répondre aux critères suivants :

  + CRUD de contact :
    - id / nom / prénom / mail / date de création auto à la date du jour

  + CRUD d'une liste de contact :
    - id / nom / date de création auto à la date du jour / description

  + CRUD de modèles qui son les templates :
    - id / nom / filename
    - faire un replace de contenu avec les valeur de message :
    ex : Bonjour M/Mme {{firstname}} {{lastname}}...

  + CRUD des états de message :
    - id / label
    ex : brouillon, à valider, prêt, envoyé

  + CRUD de Message :
    - id / objet / corps / dateEnvoie / heureEnvoie / dateEnvoyé / heureEnvoyé

  + Endpoint statistique
    - total utilisateurs dont x admin et y user
    - total contacts
    - total liste
    - dernier message envoyé (id)
    - nombre total de messages envoyés
    - nombre de messages envoyés par type

  + créer les fichier http ou postman pour les domaines (CRUD)

  + créer un swagger pour au moins un domaine (CRUD complet)

+ Construire un CRON (job auto) en Node.js :
  - tous les x temps appelle l'API pour connaitre les prochains messages prêts à être envoyés.
  - envoie les mails (peut être une simulation avec un timeout aléatoire) :
    - en cas de succès, fait un changement d'état via un appel API
    - en cas d'echec, re programme l'envoi pour l'heure suivante via un appel API

+ Le CRON doit absolument être référencée via un token application JWT auprès de l'API

=== QUALITE ==========
- code propre
- code commenté
- code fonctionnel : api + cron + http...
- commits avec des messages explicites

=== TECHNIQUE ==========
+ TypeScript peut être utilisé
+ Un autre framework qu'Express peut être utilisé
+ Toute amélioration au sujet initial apportera des points (ex: page ou CLI d'administration du server / mise en place d'une pipeline avec variables d'environnement...)

=== ORAL ==========
+ Savoir expliquer son projet, son découpage, ses modules : schématiser
+ Des questions de cours seront posées
+ Chaque membre du groupe doit présenter quelque chose. Il n'y a pas qu'un seul présentateur

=== DELAIS ==========
Le projet doit être envoyé au plus tard le : 21 avril 2022 minuit

Comment envoyer le projet :
- Supprimer les répertoires node_modules de chaque répertoires
- Zipper le projet global (client, server, http... et surtout le .git à la racine)
- Uploader le zip sur myGES

OU

- Lien vers un repo OUVERT git

3 Détails du projet

Objectif du projet (à la fin du projet les étudiants sauront réaliser un...)

Maitriser Node.js et son ecosystème

Descriptif détaillé

Ouvrages de référence (livres, articles, revues, sites web...)

Outils informatiques à installer

4 Livrables et étapes de suivi

1	Rendu final	Upload du travail	**jeudi 21/04/2022 23h59**

5 Soutenance

Durée de présentation par groupe : **20 min**

Audience :

Type de présentation :

Précisions :