

## Projet

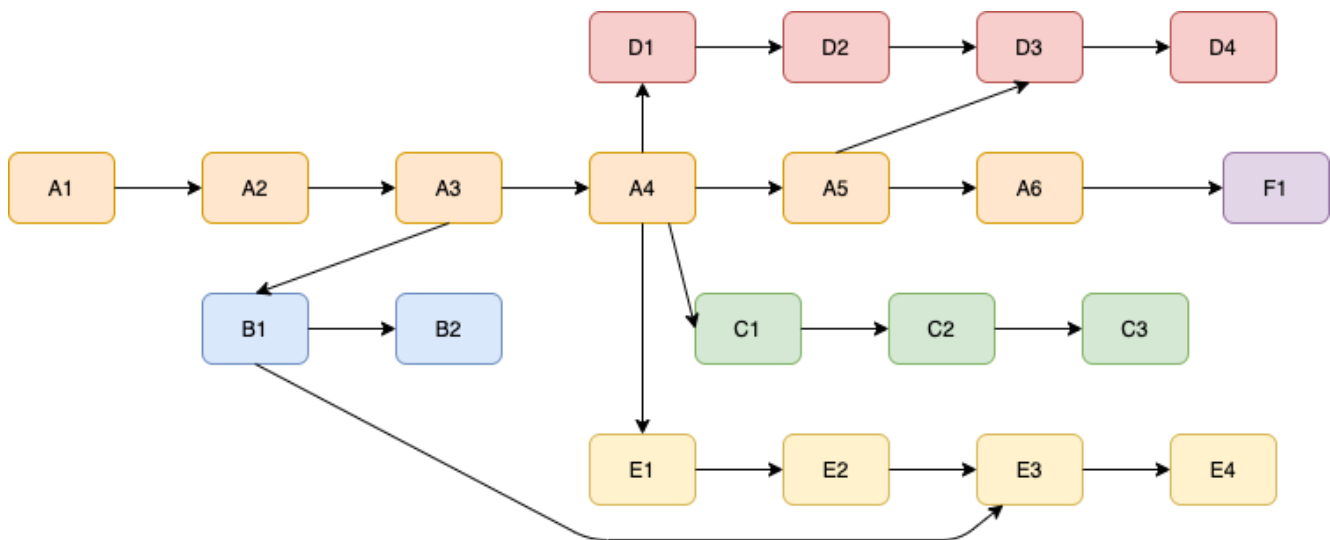
Le projet consiste à récupérer un ensemble de données provenant de la ville de Montréal et d'offrir des services à partir de ces données. Il s'agit de données ouvertes à propos d'extermination de punaises de lits..

Vous avez la possibilité de construire un projet selon vos préférences. Les diverses fonctionnalités vous offrent un certain nombre de points d'expérience (XP) et vous ouvre le chemin vers d'autres fonctionnalités à développer. Si vous faites le travail individuellement, vous devez accumuler un minimum de 100 XP. Si vous faites le travail en équipe de 2 personnes, vous devez accumuler un minimum de 200 XP. Vous n'êtes pas tenu de développer toutes les fonctionnalités mais lorsque vous choisissez une fonctionnalité, vous devez la faire au complet.

Le point de départ est la fonctionnalité A1. Les fonctionnalités ont des dépendances entre elles. Par exemple, il faut avoir complété A1 avant de pouvoir faire A2; il faut avoir complété D2 et A5 avant de pouvoir faire D3.

### Fonctionnalités

Voici une carte représentant les dépendances entre les fonctionnalités :



Voici les exigences pour chaque fonctionnalité :

#### **A1 10xp**

La liste des déclarations est obtenue en format CSV à l'aide d'une requête HTTP et son contenu est stocké dans une base de données SQLite. La modélisation de la base de données est à votre discrétion.

Les données sont accessibles à partir de l'adresse suivante : <https://data.montreal.ca/dataset/49ff9fe4-eb30-4c1a-a30a-fca82d4f5c2f/resource/6173de60-c2da-4d63-bc75-0607cb8dcb74/download/declarations-exterminations-punaises-de-lit.csv>

Pour choisir le lien « Déclarations exterminations punaises de lit » en format CSV au lien suivant : <https://donnees.montreal.ca/ville-de-montreal/declarations-exterminations-punaises-de-lit>

À ce point-ci, vous ne devez faire qu'un script Python qui télécharge les données et insère les données dans la base de données. Vous pouvez assumer que la base de données existe déjà lors de l'exécution du script (le script ne doit pas créer la base de données, ni la vider). Vous devez également fournir le script SQL pour créer la base de données ainsi qu'une base de données déjà créée mais vide pour des fins de tests.

### **A2 10xp**

Construire une application Flask pour accéder aux données de la base de données. La page d'accueil offre un outil de recherche qui permet de trouver les déclarations par :

- nom de quartier;
- nom d'arrondissement.

Les résultats de la recherche doivent s'afficher sur une nouvelle page. Pour chaque déclaration, on affiche toutes les données disponibles sur la déclaration.

### **A3 5xp**

Mettre en place un BackgroundScheduler dans l'application Flask afin d'extraire les données de la ville de Montréal à chaque jour, à minuit, et mettre à jour les données de la base de données. Une fois par jour, les données doivent être synchronisées avec celles de la ville.

### **A4 10xp**

Le système offre un service REST permettant d'obtenir la liste des déclarations ayant eu lieu entre deux dates spécifiées en paramètre. Les dates sont spécifiées selon le format ISO 8601. Les données retournées sont en format JSON.

Ex. GET /declarations?du=2018-05-08&au=2020-05-15

Une route /doc doit être disponible et afficher la représentation HTML de la document RAML du service web.

### **A5 10xp**

Sur la page d'accueil du site web, ajouter un petit formulaire de recherche rapide permettant de saisir deux dates. Lorsque l'utilisateur lance la recherche, une requête Ajax contenant les deux dates saisies est envoyée à la route définie en A4. Lorsque la réponse Ajax revient, l'application affiche la liste des contrevenants dans un tableau. Le tableau contient 3 colonnes :

- le nom de l'arrondissement;
- le nom du quartier;
- le nombre de déclarations faites durant cette période de temps pour ce quartier.

**A6 10xp**

L'application du point A5 offre un mode de recherche par nom du quartier. La liste de tous les quartiers est prédéterminée dans une liste déroulante et l'utilisateur choisira un quartier parmi cette liste. Lorsque l'utilisateur lance la recherche, une requête Ajax est envoyée à un service REST que vous devez créer à cet effet. Lorsque la réponse Ajax revient, l'application affiche l'information des différentes déclarations.

**B1 5xp**

Le système détecte les nouvelles déclarations depuis la dernière importation de données, en dresse une liste sans doublon et l'envoi par courriel automatiquement. L'adresse du destinataire du courriel est stocké dans un fichier de configuration en format YAML.

**B2 10xp**

Le nombre de nouvelles déclarations par quartier est publié automatiquement sur un compte Twitter.

**C1 10xp**

Le système offre un service REST permettant d'obtenir la liste des quartiers contenant une ou plusieurs déclarations. Pour chaque quartier, on indique le nombre de déclarations connues. La liste est triée en ordre décroissant du nombre de déclarations. Le service doit être documenté avec RAML sur `/doc`.

**C2 5xp**

Le système offre un service permettant d'obtenir exactement les mêmes données que le point C1 mais en format XML. L'encodage de caractères doit être UTF-8. Le service doit être documenté avec RAML sur `/doc`.

**C3 5xp**

Le système offre un service permettant d'obtenir exactement les mêmes données que le point C1 mais en format CSV. L'encodage de caractères doit être UTF-8. Le service doit être documenté avec RAML sur `/doc`.

**D1 15xp**

Le système offre un service REST permettant de faire une déclaration d'extermination à la ville. Le document JSON doit être validé avec json-schema. Le service doit permettre de recevoir les données suivantes :

- le nom du quartier;
- le nom de l'arrondissement;
- l'adresse;
- la date de la visite de l'exterminateur;
- le nom et prénom du résident;
- une description du problème observé.

Le service doit être documenté avec RAML sur `/doc`.

Ensuite, une page doit permettre à un exterminateur de faire une déclaration à la ville. La page doit offrir un formulaire et la page doit invoquer le service REST de création d'une déclaration. Indice : la requête doit être envoyée au backend par du Javascript.

## **D2 5xp**

Le système offre un service REST permettant de supprimer une déclaration. Le service doit être documenté avec RAML sur `/doc`.

## **D3 15xp**

Modifier l'application faite en A5 afin de pouvoir supprimer les déclarations retournées par l'outil de recherche. Ainsi, on peut supprimer d'un seul coup toutes les déclarations d'un quartier complet. L'application doit invoquer des services de votre choix avec des appels Ajax et afficher une confirmation en cas de succès ou un message d'erreur en cas d'erreur. Les changements aux déclarations (suppression) doivent être préservés en tout temps, même lors d'une synchronisation quotidienne. Les services doivent être documentés avec RAML sur `/doc`.

## **D4 15xp**

Le système offre une procédure d'authentification du type « Basic Auth » et permet de restreindre l'accès aux fonctionnalités de suppression de D3 uniquement à un utilisateur prédéfini. Assurez-vous de spécifier clairement les données d'authentification pour la correction.

Attention : Une authentification de type Basic Auth n'est pas la même chose que l'authentification avec salt et hachage vu en classe. Une recherche sera nécessaire.

## **E1 15xp**

Le système offre un service REST permettant à un utilisateur de se créer un profil d'utilisateur. Le service reçoit un document JSON contenant :

- le nom complet de l'utilisateur;
- l'adresse courriel de l'utilisateur;
- une liste de noms de quartier à surveiller;
- le mot de passe.

Le document JSON doit être validé avec json-schema. Le service doit être documenté avec RAML sur `/doc`.

## **E2 15xp**

Le système offre une page web pour invoquer le service fait en E1. Le système offre également une option d'authentification. Après l'authentification, une page permettant de modifier la liste des noms de quartier à surveiller est disponible. L'utilisateur authentifié peut également téléverser une photo de profil (qui sera sauvegardée dans la base de données). Vous devez accepter les formats jpg et png.

## **E3 5xp**

Lorsqu'une nouvelle déclaration est détectée, un courriel est automatiquement envoyé à tous les utilisateurs qui surveillent ce quartier.

## **E4 10xp**

Le courriel envoyé au point E3 contient un lien pour se désabonner. Le lien amènera à une page HTML qui demandera une confirmation à l'utilisateur. Si l'utilisateur confirme le désabonnement, une requête Ajax invoquera un service REST pour supprimer le quartier du profil de l'utilisateur.

## F1 15xp

Le système est entièrement déployé sur la plateforme infonuagique Heroku. Pour compléter ce point, vous pouvez transformer votre projet pour utiliser la base de données PostgreSQL. Il est possible de le faire avec SQLite, c'est donc à votre discrétion.

## Remise et correction

Le répertoire de travail contenant les fichiers doit être archivé dans un fichier zip et nommé selon les codes permanents des auteurs. L'archive doit être remise par Moodle (la date de remise est sur Moodle). Aucun retard ne sera accepté et une pénalité sera appliquée pour une archive non conforme sans les codes permanents.

Si vous voulez travailler en équipe, vous devez mettre votre logiciel sous gestion de sources dans un dépôt **privé** git (ex. github, bitbucket) et m'en donner un accès en lecture. Cette mesure sert à m'assurer que les coéquipiers ont partagé l'effort équitablement. Un coéquipier n'ayant pas offert une participation significative n'aura pas les points pour le travail. Il est exigé que chaque membre de l'équipe contribue à un minimum de 100 XP.

Un fichier 'correction.md' rédigé en markdown doit être placé à la racine du projet. Vous devez y inscrire tous les points que vous avez développé et comment tester chaque point. Si un point a été développé mais qu'il n'est pas présent dans ce fichier, il ne sera pas corrigé. Pour le point F1, nous n'avez qu'à fournir l'URL de votre application.

## Technologies

Voici les technologies imposées :

- Python 3
- Flask
- SQLite 3

Vous pouvez utiliser les bibliothèques et packages Javascript/Python de votre choix. Même chose pour HTML/CSS. Il est permis d'utiliser des templates CSS tant que la licence d'utilisation est respectée.

## Pondération

Fonctionnalités (XP) : 60%

Qualité du projet : 40%

La qualité du projet est évalué en conformité avec le document « Évaluation de la qualité du logiciel ».