INF5190 – Programmation web avancée
Projet de session – Hiver 2023

Projet

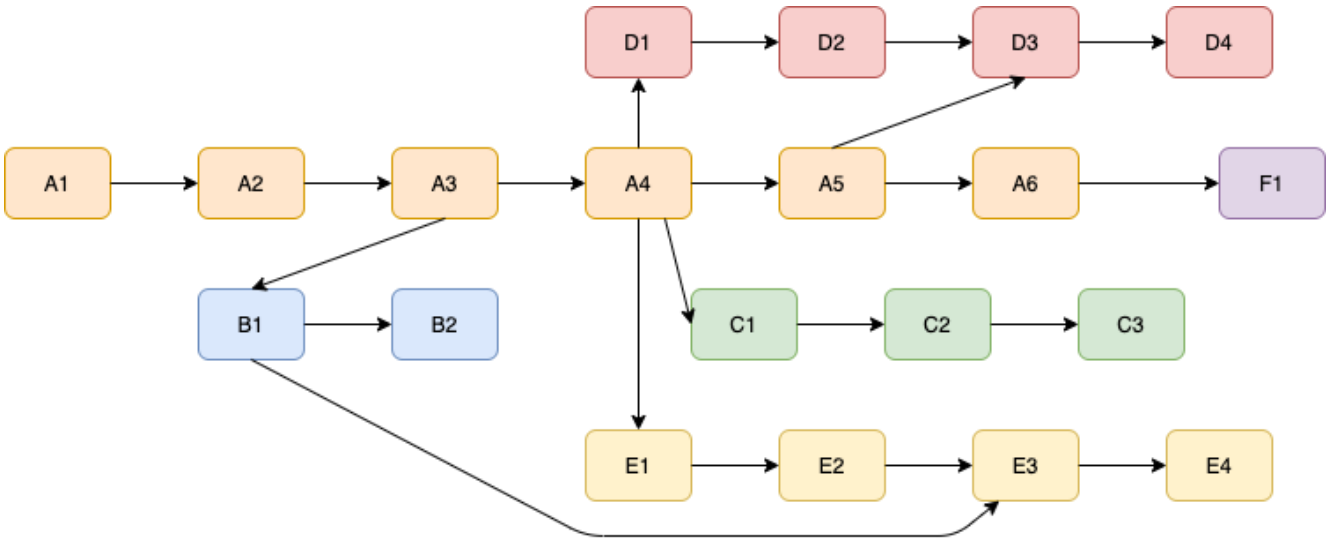
Le projet consiste à récupérer un ensemble de données provenant de la ville de Montréal et d'offrir des services à partir de ces données. Il s'agit de données ouvertes à propos d'établissements ayant reçu des constats d'infraction lors d'inspections alimentaires.

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 contrevenants est obtenue en format XML 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/05a9e718-6810-4e73-8bb9-5955efeb91a0/resource/7f939a08-be8a-45e1-b208-d8744dca8fc6/download/violations.csv

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 contrevenants par :

- nom d'établissement;
- propriétaire;
- rue (par exemple, tous les restaurants sur le boulevard Rosemont).

Les résultats de la recherche doivent s'afficher sur une nouvelle page. Pour chaque contrevenant, on affiche toutes les données disponibles sur une contravention. Il est possible qu'un restaurant apparaisse plus d'une fois, s'il a eu plusieurs sanctions.

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 contrevenants ayant commis une infraction entre deux dates spécifiées en paramètre. Les dates sont spécifiées selon le format ISO 8601. Les données retournées sont en format JSON.

Ex. `GET /contrevenants?du=2020-05-08&au=2022-05-15`

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

A5 10xp
Sur la page d'ailleurs 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 2 colonnes :

- le nom de l’établissement;
- le nombre de contraventions obtenues durant cette période de temps.

A6 10xp
L'application du point A5 offre un mode de recherche par nom du restaurant. La liste de tous les contrevenants est prédéterminée dans une liste déroulante et l'utilisateur choisira un restaurant 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 infractions du restaurant.

B1 5xp
Le système détecte les nouveaux contrevenants 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
Les noms des nouveaux contrevenants sont publiés automatiquement sur un compte Twitter.

C1 10xp
Le système offre un service REST permettant d'obtenir la liste des établissements ayant commis une ou plusieurs infractions. Pour chaque établissement, on indique le nombre d'infractions connues. La liste est triée en ordre décroissant du nombre d'infractions. 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 demande d’inspection à la ville. Le document JSON doit être validé avec json-schema. Le service doit permettre de recevoir les données suivantes :

- le nom de l’établissement;
- l’adresse;
- la ville;
- la date de la visite du client;
- le nom et prénom du client faisant la plainte;
- une description du problème observé.

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

Ensuite, une page de plainte doit permettre à un visiteur sur le site web de faire une plainte à propos d'un restaurant. La page de plainte doit offrir un formulaire et la page doit invoquer le service REST de création d'une demande d'inspection. Indice : la requête doit être envoyée au backend par du Javascript.

D2 5xp
Le système offre un service REST permettant de supprimer une demande d'inspection. Le service doit être documenté avec RAML sur `/doc`.

D3 15xp
Modifier l'application faite en A5 afin de pouvoir supprimer ou modifier les contrevenants retournés par l'outil de recherche. 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 contrevenants (modification et 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 modification et suppression de D3 uniquement à un utilisateur prédéfini. Assurez-vous de spécifier clairement les données d'authentification pour la correction.

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 d'établissements 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 d'établissements 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'un nouveau contrevenant est détecté, un courriel est automatiquement envoyé à tous les utilisateurs qui surveillent cet établissement.

E4 10xp
Le courriel envoyé au point E3 contient un lien pour se désabonner du restaurant. 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 restaurant 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 2
- SQLite 3

Vous pouvez utiliser les librairies 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 ».

pycodestyle

Il est attendu que l'exécution de pycodestyle sur votre code source ne soulève aucune erreur et aucun avertissement.

Évaluation de la qualité du logiciel

Un document nommé "Évaluation de la qualité du logiciel" est disponible sur Moodle. Tous les éléments dans ce document sont des exigences à ce travail.

Base de données

Vous devez modéliser vous même votre base de données. Un répertoire doit être créé avec le nom « /db » et il doit contenir un script de création de tables avec le nom « db.sql ».