Nathanaël Depauw

Services web

TD2

1 principe et utilisation

La tâche donnée était de créer un service de prêt immobilier. Pour y répondre, j'ai créé un service un client et un service composite, communiquant avec deux services particuliers. Pour l'activer, il faut lancer les programmes service_banque.py, service_visit.py, service_composite.py et listener.py puis déposer la demande dans le dossier demands (les données fournies sont suffisante pour traiter demand1.txt. Toute demande dont le client ou l'adresse ne font pas partie des données placées dans les dossiers data et visit essuieront un refus.).

Les programmes service_banque.py, service_visite.py et service_composite.py fonctionnent grâce à fastapi et requierent d'utiliser uvicron. Les commandes conseillées sont :

```
$ uvicorn service_banque:app --reload --host="127.0.0.1" –port=8002
$ uvicorn service_composite:app --reload --host="127.0.0.1" –port=8000
$ uvicorn service_visit:app --reload --host="127.0.0.1" –port=8001
```

2 Formats

L'entrée principale est un fichier de texte de la forme :

```
Nom
Prix
Adresse
Description
```

Dès que ce fichier est placé dans demands, le listener va se déclencher.

Les fichier de la banque sont représentés par des json :

actuel_database.json de la forme :

{"[nom]": {"adresse": "[adresse]", "revenus_mens": [revenus mensuels], "depenses_mens": [dépenses mensuelles]}}

historique_database.json de la forme :

{"[nom]": {"credit": [crédit actuel], "nbPaymentRetard": [nombre de payements en retards], "faillite": [nombre de déclarations de faillite]}}

Le fichier des visite est de la forme :

{"[adresse]": {"valide": [si la propriété est vendable], "comment": "[commentaire sur la propriété]"}}

La sortie est un fichier texte nommé Reponse.txt qui contient le verdict et résonnement sur cette demande.

3 Listener et client

Le listener (listener.py) n’est pas une application web à proprement parler. Il s’agit d’un programme qui attend qu’une demande soit déposé dans le dossier demands et active le client quand c’est le cas. Le client lit la demande et envoie au service composite les informations qu’elle contient via une enveloppe SOAP qui inclus le nom, prix, adresse et description du client. Il retourne ensuite la réponse au listener, qui l’affiche dans le terminal.

4 Service composite

Le service composite envoie ensuite une request get incluant le nom du client au service banque et une autre au service visite, incluant l’adresse. Ces deux services renvoient un avis (approbation ou refus + raisonnement) au service composite en utilisant le corps du retour. Ce texte, sous format JSON, contient un booléen qui dit si le prêt est acceptable d’un point de vue bancaire ou de la visite, et le raisonnement en cas de refus. Le service composite décode , le JSON avec la bibliothèque json, vérifie que la demande est bien acceptable et écirt dans Reponse.txt un texte contenant l’approbation ou le refus et le raisonnement au client.

5 Service banque

Le service banque (géré par le programme service_banque.py) reçoit une request get incluant le nom du client. Il consulte les fichiers actuel_database.json et historique_database.json.

actuel_database.json est un dictionnaire dont l’entrée au nom du client indique son adresse actuelle, ses revenus et dépenses mensuels. Si ses dépenses sont plus grande que ses revenus, le service indique que la demande n’est pas acceptable d’un point de vue bancaire.

historique_database.json est un dictionnaire dont l’entrée au nom du client indique son crédit et son nombre de paiement en retard et de faillite. Si le client a été en retard sur un payement ou en faillite, le service indique que la demande n’est pas acceptable d’un point de vue bancaire.

Enfin, le service envoie le retour au service composite. Cette enveloppe contient un texte sous format json contenant son approbation/refus sous format booléen et son raisonnement.

6 Service visite

Le service banque (géré par le programme service_visite.py) reçoit une request get incluant l’adresse de la propriété que le client veut acheter. Il consulte le fichier data.json dans le dossier visit.

Ce fichier est un dictionnaire dont l’entrée correspondant à l’adresse de la propriété indique, si elle est vendable et laisse un commentaire sur la propriété.

Enfin, le service envoie le retour au service composite. Cette enveloppe contient un texte sous format json contenant son approbation/refus sous format booléen et son raisonnement.