Python Developer Assessment

Data Pusher

Overview

You have to create a Django web application to receive data into the app server for an account and send it across different platforms (destinations) from that particular account using webhook URLs.

Modules

1. Account Module: Each account should have email id (Mandatory field & unique), account id (unique to each account), account name (Mandatory field), App secret token (automatically generated), Website (optional)

2. Destination Module: A destination belongs to an account. An account can have multiple destinations. A destination has URL (mandatory field), HTTP method (mandatory field) and headers (mandatory field & have multiple values).

Ex for headers:

```
{
    “APP_ID”: “1234APPID1234”,
    “APP_SECTET”: “enwdj3bshwer43bjhjs9ereuinkjcnsiurew8s”,
    “ACTION”: “user.update”,
    “Content-Type”: “application/json”,
    “Accept”: “*”
}
```

3. Data handler: Data handler receives only JSON data in the POST method. While receiving the data, it should have an app secret token sent through the header (header key is CL-X-TOKEN). Based on the secret token, the account should be identified. Using any Python http client, send the data across different available destinations (use the url, http method and headers) for that particular account.

Note: If the destination’s HTTP method is **get**, then the incoming JSON data should be sent as a query parameter. If the method is **post** or **put**, send the data as it is (JSON).

Process:

1. Create a Django web application.
2. Create JSON rest APIs
    a. CRED operations for an Account

    b. CRED operations for Destinations.
       Note: An account can have multiple destinations. For example, if an account is deleted, the destinations for that account should also be deleted.
3. Create a URL to get destinations available for the account when the account id is given as input.
4. Create an API for receiving the data.
    a. The API path is **/server/incoming_data**. The data should be received through the post method in the JSON format only.
    b. The app secret token should be received while receiving the data.
    c. If the HTTP method is GET and the data is not JSON while receiving the data, send a response as “Invalid Data” in JSON format
    d. If the secret key is not received then send a response as “Un Authenticate” in JSON format.
    e. After receiving the valid data, using the app secret token, identify the account and send the data to its destinations.

Instructions:

1. Write the code in the latest version of Python Django web application.
2. Upload the code to Github in a private repository and give collaborator access to id: ***vishnuixm***
3. Attach the sample APIs as files in the same Github repo.
4. Reply to this email with the Github link.