**Technical Assessment Notes**

* Provided Resources:
  + Source Code on GitHub
  + Postman Collection
  + Swagger Doc
  + How to run the application walk through
* Technologies Used:
  + Python
  + MongoDB
  + Fast API
* Security Considerations:
  + The .env file is kept for testing purposes. At production level, the variables should be bound to the repository (stored securely – Git variables).
  + HS256 (HMAC with SHA-256) symmetric key hashing algorithm is used as a default. For higher security requirements, an RSA-based algorithm would be recommended.
  + Session/Cookies and database dependencies can be added for an extra level of authentication.
  + The encryption function should be removed from the class, but it is kept for testing purposes.
  + The generate\_secret\_key.py file should also be removed due to security purposes but is kept for testing.
* Implementation Details:
  + There can be an alternative configuration to connect to MongoDB through pymongo.
  + A sample Dockerfile has been provided for production.
  + A health check endpoint is created to check if the APIs are up and running.
* Folder Structure:
  + helpers folder contains:
    - db\_helpers.py: functions that manipulate the database.
    - utils.py: stand-alone functions with no dependencies.
    - api\_helpers.py: functions that connect the db\_helpers and utils.
    - custom\_exceptions.py: custom exceptions created for the app.
    - response\_format.py: a default response format returned by the API for predictable responses and better testing.
    - crypto folder: contains files that should be deleted on production but kept for testing purposes. Only the decryption function should be left and can be stored in utils.py.
    - models folder contains the schemas of the collection inputs as well as the ORM.
* Configuration:
  + Configurations for the database and other app configurations are found under the config.yml file.
  + settings.py contains the loaded configurations and environment variables.
* Testing:
  + A mock database would have been preferable for the Pytests, but a different approach has been made to satisfy the current duration.
  + More tests should be created for all functions used in the project, but this step has been removed for demo purposes.
  + Not all APIs have a unit test since there should be a test access token that must be created. This access token is required to be sent in the headers for the APIs to work. For requirement purposes, this has been removed as well.
  + An integrational test is recommended for the login flow.