Hectre Python back-end ML technical challenge

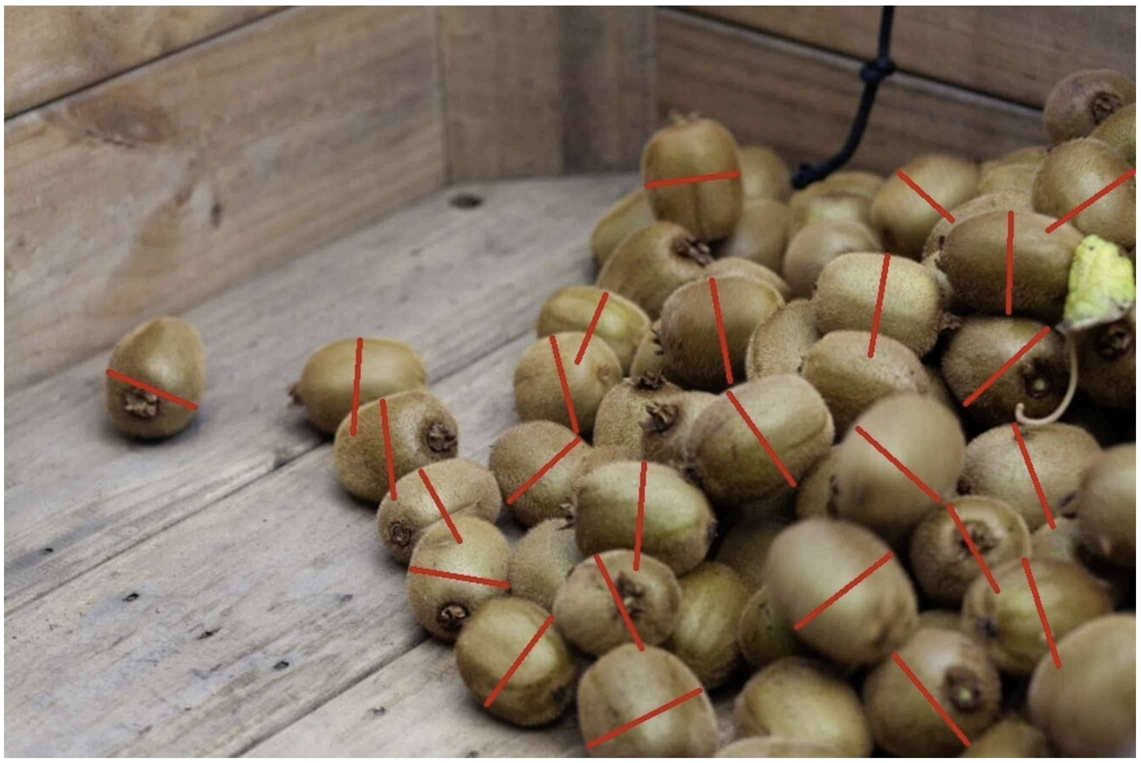
This document describes a technical challenge that forms part of the Hectre recruitment process for Spectre back-end engineers.

It should not take more than one working day. Don't worry if the project isn't complete by then - just email back what you have with some notes on where you got to and what's left to do.

Background

Spectre AI tool is one of Hectre's products that utilises machine learning, computer vision, and cloud computing to provide size & colour distributions for fruit bins within seconds from uploaded images/videos.

Here is one example of uploaded images for Kiwifruit with expected diameter lines.



Requirements

Your task is to design and deploy a backend machine learning solution that could allow the frontend to upload image(s) and save the information to a structured database. The tasks are:

1. Design the solution/service architecture (ideally combining with cloud platform)
2. Deploy one Python backend API endpoint for uploading one or many images with at least the following information:
    - Image ID (Primary key)
    - Created date (UTC format)
    - Fruit type
    - User email
    - Tenant ID

    And save this information into a relational database.

3. Demo test with simple Unit test(s)
4. Document the detailed steps on how to deploy and run the code in Mac/Linux environment
5. Deploy the one of core blocks of solution on cloud with Infrastructure as code (IaC) (In case of designing solution with cloud)
6. One of Spectre AI features is grading fruit size. Assume that we have to develop a machine learning solution to find the pixel diameter of Kiwifruits. From the visualization image above, can you describe your practical solution:
    - Which kind of ML model should you use? Why?
    - Any idea about data, training ML?
    - Any idea about organizing/deploying ML model on production?
    - Post-process algorithm idea to the get expected diameter size
    - How can we best deal with the flat shape of a Kiwifruit?

Deliverable

The deliverable product should contain:

- A design solution with diagram(s) and explanations
- A Python application
- A .git folder with the commit history for the project
- A README that explains how to install and use the project

  The project should work on macOS or Linux.
  Feel free to document any assumptions made, or ideas for improving the project. Don't upload the project to a public repo since it might be unfair for other candidates.

Application notes

- The Python scripts, packages are compatible with Python 3.6,3.7,3.8.
- Use Flask API or other blueprint for backend APIs
- Use a PostgreSQL or other structured database, including DB script/code for creation.
- It is better to use ORM (Object Relational Mapper) and a suitable backend design pattern.
- Ideal unit test is Pytest but not limited to this option