PDF Rendering Application

In Rossum, some customers are processing tens of thousands of documents every month. The very first step before AI is applied to a document is input file normalization and rendering, so that subsequent processing may be performed. Right now, rendering is part of the data extraction process, but it could be extracted to a separate service.

Assignment

Create a service that accepts PDF files containing one or more pages. These pages should be rendered to normalized png files: they should fit into a 1200x1600 pixels rectangle.

The service should be accessible through a REST API and offloads all file processing to asynchronous tasks (e.g. using dramatiq library), so that it is easy to scale.

Create the following REST API endpoints (go for an easily scalable design when creating these endpoints):

- Endpoint for document upload
  - Is being used to upload a file
  - Must return at least document ID in the response
- Endpoint for getting information about the uploaded document
  - Must return at least the information on the current status of the document and number of its pages
- Endpoint for getting the rendered image PNG
  - Must return at least the rendered image png

Implementation

- Use Python 3, Flask, PostgreSQL
- Use message queues and libraries (e.g. rabbitmq, celery, dramatiq)
- Add tests for all API endpoints
- Include a simple Dockerfile and docker-compose.yml to enable easy testing
- Create OpenAPI documentation for these endpoints

Notes

- API should use common practices, e.g. return proper status codes, content-type, etc.
- Consider software development best practices

- If you would need to have a Rossum account for the purpose of this application, you can create a new Rossum account here. To create the Rossum account, you can use your own email address or you can create some temporary email for this purpose, for example at this website.