Backend Engineer Challenge

At refive, our systems receive and process receipt documents. Our goal is to maximise the information we can extract out of them. A receipt contains several blocks each one with different details about the transaction. In order to better understand the data contained in the receipt, we want to extract each block, so we can locate where a piece of information lies within the receipt. Each block in the receipt is delimited either by an empty line or a full line containing a separator (for example a sequence like '————-').

As we want our team to have an easier time making sense of this data, we'd like to build an utility for them. This tool is a frontend where our team upload receipts, which then are sent to a backend to be analysed, the resulting blocks of the receipt are then rendered in the window.

The goal is to implement the backend for the application mentioned above, which is composed of:

- An API to submit receipts documents (.txt files), attached a sample file.
    - POST `/receipts`
      Request `{"receipt": text}`
      Response

      ```
      {"blocks": [{"begin_row": int,
                   "begin_col": int,
                   "end_row": int,
                   "end_col": int},
      ]}
      ```
- The algorithm invoked by the API which:
    - Extracts all the blocks within the receipt and the position of each block.

      A **block** is composed of two pairs of coordinates:

        - **begin -** upper left corner denoting the row and column in the receipt where the block begins
        - **end** - lower right corner denoting the row and column in the receipt where the block ends

  Blocks are separated either by: more than one new line character or a full line containing a separator (for example the character '-')

- The necessary models to store the results of the processed documents.
- Authentication - this is up to you to decide an approach

Please use Django and Django Rest Framework to implement the backend.

What we will take into account when reviewing your submission:

- The submitted solution must work
- The code follows the clean code principles
- Dependency management
- You've taken into account situations that can lead to error (for example, what happens if I try to upload a binary file)
- How you decide to represent this problem entities
- Your design would be easy to extend, imagine the scenario where we find new separators, we would need to include those in the algorithm
- You have included the tests you deem necessary

For the submission you can send a zip with the Python source code files and required dependencies, alternatively you can create a private repository on GitHub and invite cristian.martin@leaflabs.de as a collaborator.