Programming Exercise using PySpark

Background:

A very small company called **KommatiPara** that deals with bitcoin trading has two separate datasets dealing with clients that they want to collate to starting interfacing more with their clients. One dataset contains information about the clients and the other one contains information about their financial details.

The company now needs a dataset containing the emails of the clients from the United Kingdom and the Netherlands and some of their financial details to starting reaching out to them for a new marketing push.

Since all the data in the datasets is fake and this is just an exercise, one can forego the issue of having the data stored along with the code in a code repository.

Things to be aware:

* Use Python **3.7**
* Only use clients from the United Kingdom or the Netherlands.
* Remove personal identifiable information from the first dataset, **excluding emails**.
* Remove credit card number from the second dataset.
* Data should be joined using the **id** field.
* Rename the columns for the easier readability to the business users:

| **Old name** | **New name** |
| --- | --- |
| id | client\_identifier |
| btc\_a | bitcoin\_address |
| cc\_t | credit\_card\_type |

* The project should be stored in GitHub.
* Save the output in a **client\_data** directory in the root directory of the project.
* Add a readme file explaining on a high level what the application does.
* Application should receive three arguments, the paths to each of the dataset files and also the countries to filter as the client wants to reuse the code for other countries.
* Use **logging**.
* Should create generic functions for filtering data and renaming.

Recommendation: Use the following package for Spark tests - <https://github.com/MrPowers/chispa>

* If possible, have different branches for different tasks that once completed are merged to the main branch. Follow the GitHub flow - <https://guides.github.com/introduction/flow/>.
* **Bonus** - If possible it should have an automated build pipeline using  [https://www.travis-ci.com/](https://travis-ci.org/) for instance.
* **Bonus** - If possible log to a file with a rotating policy.
* **Bonus** - Code should be able to be packaged into a source distribution file.
* **Bonus** - Requirements file should exist.
* **Bonus** - Document the code with docstrings as much as possible using the reStructuredText (reST) format.