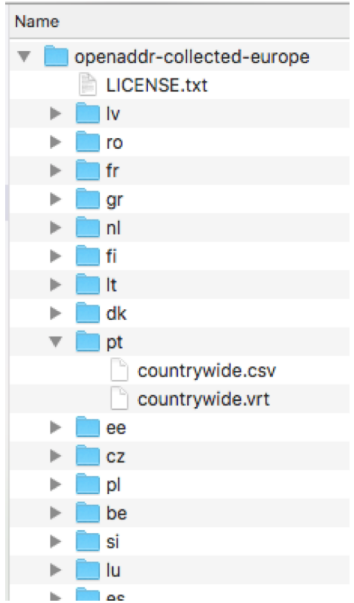
Use OpenAdresses Data To Validate Adresses

Practical Exam

Goal

OpenAddresses.io provides regulary exports of worldwide adresses (we will focus on europe for now):

- http://results.openaddresses.io/
- https://data.openaddresses.io/openaddr-collected-europe.zip




```
LON,LAT,NUMBER,STREET,UNIT,CITY,DISTRICT,REGION,POSTCODE,ID,HASH
13.1688262,52.5078776,9,Potsdamer Chaussee,,Berlin,,,13593,,5ef12a3087c99461
13.1689364,52.507843,9 A,Potsdamer Chaussee,,Berlin,,,13593,,a9f40456e296699a
13.1692019,52.5078327,9 B,Potsdamer Chaussee,,Berlin,,,13593,,b6ece1ead310fc9f
13.1693468,52.507773,9 C,Potsdamer Chaussee,,Berlin,,,13593,,839a7d07f6e663ae
13.1694613,52.5077415,9 D,Potsdamer Chaussee,,Berlin,,,13593,,ccfb7c1285bcd45a
13.1698781,52.5076352,11,Potsdamer Chaussee,,Berlin,,,13593,,b86e62e6fb76b33a
13.2591722,52.4354736,1,Potsdamer Straße,,Berlin,,,14163,,31017dd3e09e6930
13.2589704,52.4355088,2,Potsdamer Straße,,Berlin,,,14163,,6516a41eb899cd75
13.2577856,52.4357575,3,Potsdamer Straße,,Berlin,,,14163,,edaee5c1994b3281
13.2574511,52.4358362,4,Potsdamer Straße,,Berlin,,,14163,,9b87f7456c37c259
13.2570002,52.4359144,6,Potsdamer Straße,,Berlin,,,14163,,b2759ff7fe5a960f
13.2570961,52.4363525,7,Potsdamer Straße,,Berlin,,,14163,,5a41311f751606f3
13.2562084,52.436783,7 A,Potsdamer Straße,,Berlin,,,14163,,b86af65860315c1c
13.255622,52.4359221,7 B,Potsdamer Straße,,Berlin,,,14163,,b47315f8ba8452b6
13.2550108,52.4358296,8,Potsdamer Straße,,Berlin,,,14163,,cdc7b606fcbf62e1
…
```

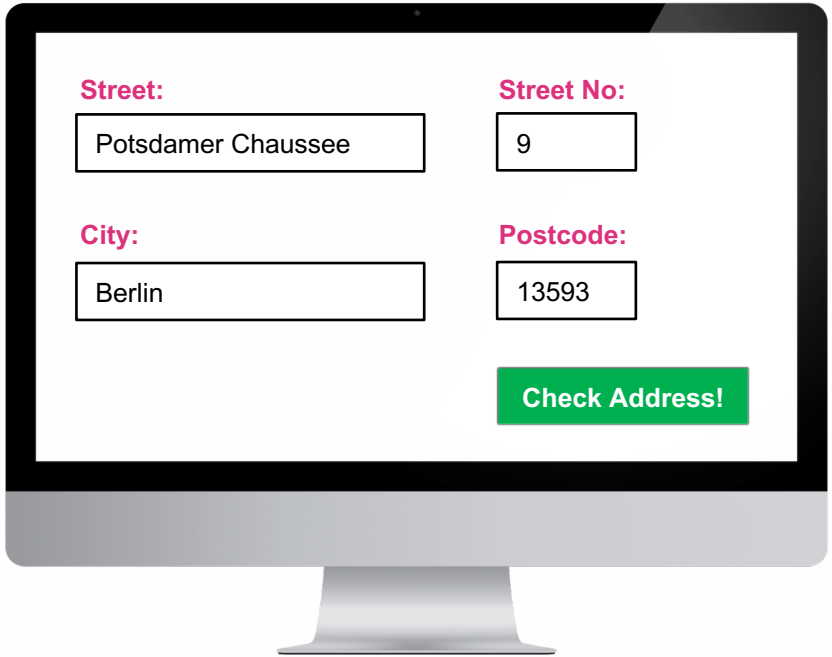
berlin.csv

Goal

We want to make use of this data to validate adresses entered on a website, to check whether they are real or not.

Workflow:

- **Gather data** from OpenAddresses.io
- **Save raw data** (*CSV files*) to HDFS (partitioned by country
- shortcut, e.g. *de*, *fr*, *it*…)
- **Optimize**, **reduce** and **clean raw data** and save it to **final** directory on HDFS
- **Export** address data **to end-user database** (e.g. MySQL, MongoDB…)
- Provide a simple **HTML Frontend** which is able to:
  - read from end-user database
  - process user input (Street, City, Postcode…)
  - validate user input against OpenAddress data in end-user database
  - Display result (real or non real address)
- The whole data workflow **must be implemented** within an ETL **workflow tool** (e.g. **Pentaho Data Integration** or **Airflow**) and **run automatically**



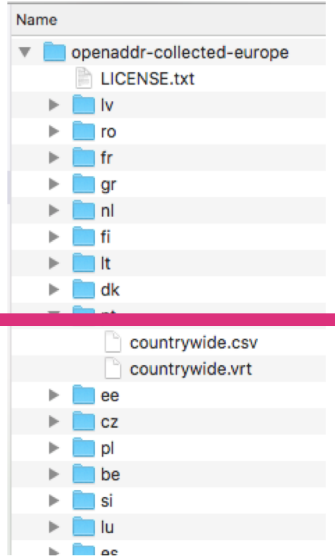
Dataflow: 1. Get Address Data

Get https://data.openaddresses.io/openaddr-collected-europe.zip

1

http://results.openaddresses.io/

2



/user/hadoop/openaddresses/**raw**/fr/*.csv
/user/hadoop/openaddresses/**raw**/it/*.csv
/user/hadoop/openaddresses/**raw**/de/*.csv
…

Dataflow: 2. Raw To Final Transfer

/user/hadoop/openaddresses/**raw**/fr/*.csv
/user/hadoop/openaddresses/**raw**/it/*.csv
/user/hadoop/openaddresses/**raw**/de/*.csv
…

- move data from *raw* to *final* directory
- optimize and reduce data structure for later query purposes if necessary
- remove duplicates if necessary
- …

/user/hadoop/openaddresses/**final**/fr/*
/user/hadoop/openaddresses/**final**/it/*
/user/hadoop/openaddresses/**final**/de/*
…

Dataflow: 3. Enhance Data And Save Results

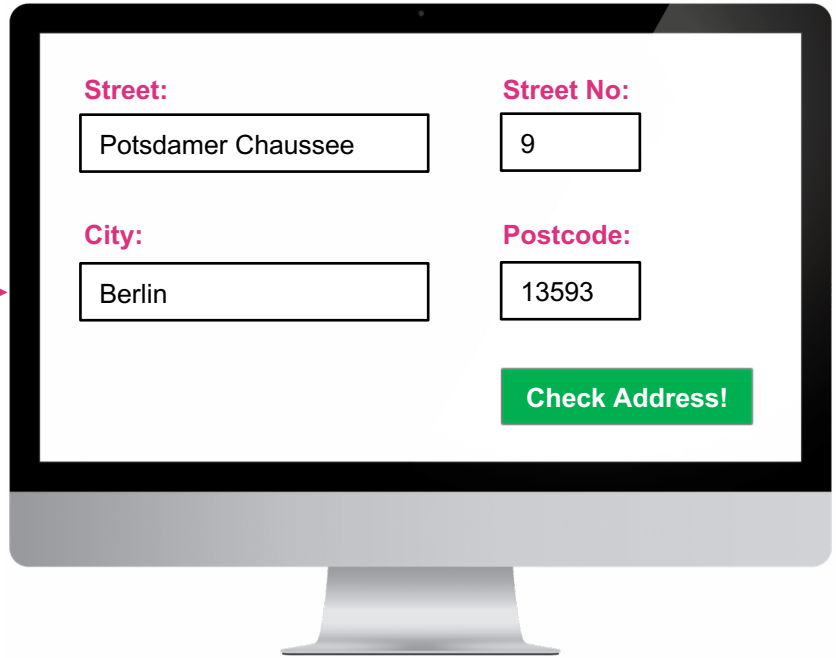
/user/hadoop/openaddresses/**final**/fr/*
/user/hadoop/openaddresses/**final**/it/*
/user/hadoop/openaddresses/**final**/de/*
…

1

- enhance data (e.g. add missin g entries of street no‘s)
- use *Hive*, *Spark* or *PySpark*
- save everything to a end-user database (e.g. *MySQL*, *MongoDB*)

Dataflow: 4. Provide Simple Web Interface

1



- Provide a simple **HTML Frontend** which is able to:
  - read from end-user database
  - process user input (Street, City, Postcode…)
  - validate user input against OpenAddress data in end-user database
  - Display result (real or non real address)