# Software developers guide

[Software developers guide 1](#_Toc448907917)

[Project directory structure 1](#_Toc448907918)

[Phase 3 solution architecture 3](#_Toc448907919)

[Data ingestion module 4](#_Toc448907920)

[Web server module 4](#_Toc448907921)

[Initial data generation 5](#_Toc448907922)

[Systems access, user IDs and credentials 5](#_Toc448907923)

## Project directory structure

|  |  |  |
| --- | --- | --- |
| Folder/file | Comment | Phase |
| . | Project root. |  |
| ├── app | Runnable applications |  |
| │   ├── clf | Phase 3 application (web service) | 3 |
| │   │   ├── config.json | Configuration file for app | 3 |
| │   │   ├── init\_database.sh | Initializes databases | 3 |
| │   │   ├── \_\_init\_\_.ipynb | IPython notebook with flask code | 3 |
| │   │   ├── init\_pyfiles.sh | Converts necessary ipynb files to plain python | 3 |
| │   │   ├── offers\_suboffers.csv | Unused. Offers and corresponding suboffers | 3 |
| │   │   ├── README |  | 3 |
| │   │   ├── static | Static files to serve with web server | 3 |
| │   │   ├── templates | Templates of webpages | 3 |
| │   │   └── websocket.ipynb | Unused. Web-socket POC for real-time updates | 3 |
| │   ├── IAP | Application from Phase 1 | 1 |
| │   ├── init\_shiny.r | Applications and startup scripts from Phase 2 | 2 |
| │   ├── init\_shiny.sh |
| │   ├── lm |
| │   ├── runiap.r |
| │   ├── runlm.r |
| │   ├── run\_shiny.sh |
| │   ├── runwm.r |
| │   └── wm |
| ├── data | Data for applications from Phase 2 | 2 |
| │   ├── behavior |
| │   ├── customer |
| │   ├── darlene |
| │   ├── data\_generation |
| │   ├── data\_merging |
| │   ├── summary |
| │   └── transaction |
| ├── data\_analysis |  |  |
| │   ├── data\_processing | Data preparation for WM (Phase 2) | 2 |
| │   ├── influence |  | 3 |
| │   │   └── get\_circle.ipynb | Script gathers information about person’s influence from their twitter network | 3 |
| │   ├── model\_draft |  |  |
| │   │   ├── churn\_rate\_and\_NBA | POC models for Churn rate & NBA | 3 |
| │   │   │   ├── 10\_churn\_rate\_histohram.ipynb | Unused. Helper for churn rate analysis. | 3 |
| │   │   │   ├── 1\_preprocess\_data.ipynb | Preprocess customers data from CSV files | 3 |
| │   │   │   ├── 2\_fit\_cox\_model.ipynb | Fit Cox model (Churn rate) | 3 |
| │   │   │   ├── 3\_fit\_rft\_NBA\_model.ipynb | Fit RF model (NBA/NBO) | 3 |
| │   │   │   ├── 4\_apply\_trained\_cox\_model.ipynb | Apply previously fitted model to new data | 3 |
| │   │   │   ├── 5\_apply\_trained\_rft\_NBA \_model.ipynb | Apply previously fitted model to new data | 3 |
| │   │   │   ├── readme.md |  | 3 |
| │   │   │   ├── spark\_streaming\_churn.ipynb | Apply previously fitted model in Spark | 3 |
| │   │   │   └── spark\_streaming\_NBA.ipynb | Apply previously fitted model in Spark | 3 |
| │   │   └── clustering |  |  |
| │   │   ├── clara.r | POC from Phase 2 | 2 |
| │   │   ├── darlene.r |
| │   │   ├── dbscan.r |
| │   │   ├── fuzzy.r |
| │   │   └── k-means.r |
| │   ├── model\_impl | Models in R from Phase 2 | 2 |
| │   ├── PCI | Predictive customer intelligence POC | 3 |
| │   │   ├── data\_preparation\_pci.ipynb |
| │   │   ├── init\_server.sh |
| │   │   ├── install\_server.sh |
| │   │   ├── kwords.json |
| │   │   ├── parse\_query.ipynb |
| │   │   └── queries.csv |
| │   └── spark\_pipeline | POC for sentiment analysis in Spark | 3 |
| │   ├── AFINN-111.txt |
| │   ├── english\_stopwords.txt |
| │   ├── init\_table.py |
| │   ├── README |
| │   ├── shell\_init.py |
| │   ├── spark\_twitter\_streaming.ipynb |
| │   ├── taxonomy\_dict.json |
| │   └── test |
| │   ├── run\_streamer.sh |
| │   └── stream\_tweets.py |
| ├── data\_ingestion | POC for sentiment acquisition and processing with Kafka | 3 |
| │   └── twitter |
| │   ├── README |
| │   ├── test\_kafka\_consumer.sh |
| │   ├── test\_kafka\_producer.sh |
| │   └── twitter\_data.py |
| ├── data\_marts |  |  |
| │   ├── customer\_data | Auxiliary data used to generate datasets for Phase 3 application | 3 |
| │   │   ├── CSV\_Database\_of\_Last\_Names.csv |
| │   │   ├── generate\_mongo\_data.ipynb |
| │   │   ├── names.csv |
| │   │   ├── README |
| │   │   ├── states.csv |
| │   │   ├── states.json |
| │   │   ├── uci\_dataset |
| │   │   │   ├── bank.csv |
| │   │   │   ├── bank-full.csv |
| │   │   │   └── bank-names.txt |
| │   │   └── us\_states\_stats.csv |
| │   ├── customer\_matching | Unused. Draft for customer matching process (not completed) | 3 |
| │   │   ├── a01\_init\_database.ipynb |
| │   │   ├── init.py |
| │   │   ├── README |
| │   │   └── shell\_init.py |
| │   └── neo4j\_graph | Sample neo4j database with influence graph | 3 |
| │   └── neo4j\_db.zip |
| ├── docs | Description of data stored in MongoDB | 3 |
| │   └── mongodb\_schema.json |
| ├── local\_version | Files for running local version of Phase 3 app | 3 |
| │   ├── AFINN-111.txt | Dictionary with words sentiment | 3 |
| │   ├── data\_ingestion.ipynb | Captures tweets from tweeter stream, analyzes them and saves to PostgreSQL | 3 |
| │   ├── data\_preparation.ipynb | Create initial sample records in PostgresQL | 3 |
| │   ├── english\_stopwords.txt | Stopwords | 3 |
| │   ├── init.sh | Initialization scripts | 3 |
| │   ├── init.sql |
| │   ├── taxonomy\_dict.json | Dictionary for tweet clustering | 3 |
| │   └── vm\_image | VirtualBox image-specific information | 3 |
| │   ├── installs |
| │   └── root\_password |
| ├── mobile | Mobile app from Phase 1 & 2 | 2 |
| │   ├── android | Android app | 2 |
| │   └── webservice | Accompanying web service | 2 |

## Phase 3 solution architecture

Twitter

data\_ingestion.py

PostgreSQL

Neo4J

MongoDB

Web server (\_\_init\_\_.py)

Web Browser

Solution has two executable modules: data\_ingestion.py and \_\_init\_\_.py (flask-based webserver).

### Data ingestion module

Module authorizes against twitter.com with OAuth.

Connect to twitter stream with given tracked keywords.

Every message from tweeter is being analyzed in endless loop.

For each message we calculate sentiment score, taxonomy (cluster group), and matching with existent customer in our database. Tweet and all additional information than are saved to PostgreSQL database.

### Web server module

User works from their browser. From “Welcome Page” he/she has access to all use-cases implemented in Phase 3:

1. Closed Loop Feedback
2. Wealth Management
3. Predictive Customer Intelligence\*
4. Loyalty Management
5. Voice of Customer
6. Next Best Offer
7. 720 Customer View

Module services requests with following specifications:

|  |  |
| --- | --- |
| *@app.route('/')* | Returns Welcome page |
| *@app.route('/loopfeedback')* | Returns CLF page |
| *@app.route('/reply', methods=['POST'])* | Used to post tweet reply from CLF page |
| *@app.route('/graph')* | Returns JSON with twitter influence graph from Neo4J. This URL is used from CLF page |
| *@app.route('/data', methods=['GET'])* | Returns JSON with twit stream to display on CLF page |
| @app.route('/cluster') | Returns “Cluster analysis” page. Linked from NavBar |
| @app.route('/download/customers') | Returns CSV file with customer information. Linked from CLF page |
| *@app.route('/wm', methods=['GET'])* | Returns Wealth Management page |
| @app.route('/pci') | Returns Predictive Customer Intelligence page |
| @app.route('/send\_pci\_query', methods=['POST']) | Called from PCI page. Returns JSON with chart information and corresponding data as was queried by user |
| @app.route('/loyalty') | Returns Loyalty page |
| @app.route('/voc') | Returns Voice of Customer page |
| @app.route('/map') | Returns NBO first page (with map) |
| @app.route('/map/get\_states\_data') | Returns JSON with churn rate information about states. Also contains GEO information (polygons) |
| @app.route('/map/get\_customers/<state>') | Returns JSON with customer information from specified state |
| @app.route('/nbo/<int:userid>') | Returns NBO page for specified user |
| @app.route('/send\_offer', methods=['POST']) | Used to send E-mail from NBO page. Returns html snippet with result. |
| @app.route('/720view/<name>') | Returns 720 view as a page without NavBar or footer. Can be included as iframe to other pages |
| @app.route('/720page/<name>') | Displays NavBar, footer, and iframe with 720view between them |

### Initial data generation

Test datasets are generated for NBO use case and for PCI use case.

Script for generating NBO use case data is “data\_marts/customer\_data/generate\_mongo\_data.ipynb”

Script for generating PCI use case data is “data\_analysis/PCI/data\_preparation\_pci.ipynb”

Data are generated in such way that it analogize with real trends. (Male/female proportions, US states average age, average income and so on).

## Systems access, user IDs and credentials

1. IBM Cluster. Public key authentication is used for logging in to IBM cluster as root user. Public key provided by CapGemini team.
2. Git repository. Public key authentication is used for working with central git repository. Gitolite is used for hosting git repository (<http://gitolite.com/gitolite/gitolite.html#basic-admin>). Home directory for “gitolite” is “/home/git” on IBM cluster.
3. John Smith twitter account. Login: “capgbanktest”, Password: “p@ssword123”.
4. CapGBank twitter account. Login: “CapGBank”, Password: “p@ssword123”.
5. Gmail account. Login: “capgbank@gmail.com”, Password: “my!Smart”.
6. Jupyter notebook. If you run jupyter notebook from smartanalytics user account, you will be asked for password. The password is “p@ssword123”.