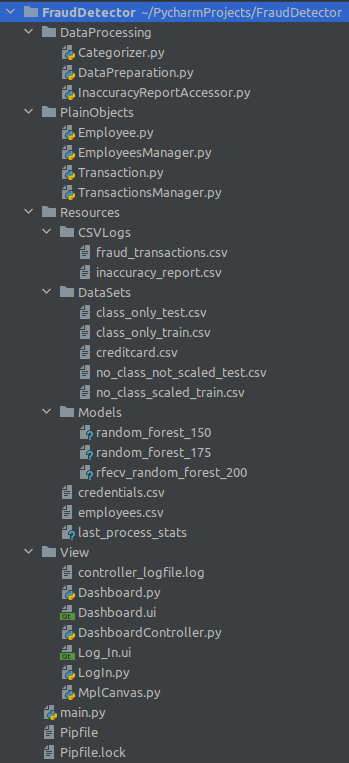
Application Files

* Data Processing
  + **Categorizer.py**
    - trains model
    - prepares testing data set and “future transactions” for predictive model
    - runs predictive model
    - adds newly detected fraud transactions to fraud\_transactions.csv
    - evaluates predictive model
    - logs evaluation statistics to last\_process\_stats file
  + **InaccuracyReportAccessor.py**
    - logs false negatives and false positives reported by the user
    - returns count of false negative or false positive when requested
  + **DataPreparation.py**
    - splits data set crediccard.csv into training and testing sets
    - applies SMOTE to the training set
    - scales time and amount fields
    - creates class\_only\_test.csv, no\_class\_not\_scaled\_test.csv, class\_only\_train.csv, no\_class\_scaled\_train.csv.
* PlainObjects
  + **Employee.py**
    - provides a container for employee information such as employee id, name, number of cases assigned. It also provides a method to increase the number of assigned cases.
  + **EmployeeManger.py**
    - uses data from employees.csv and to create employee objects
    - method for retrieving all employee objects
    - method for retrieving a specific employee based on employee id
  + **Transaction.py**
    - provides a container for transaction information such as transaction id, amount, whether or not it has been assigned, the employee assigned to it, who assigned it, the time assigned. Provides get methods for all fields and set methods for all fields except transaction id and amount.
  + **TransactionManager.py**
    - uses data from fraud\_transactions.csv to create transaction objects
    - keeps a list of unassigned and assigned transactions
    - provides a method to get all unassigned transactions
    - provides a method to get all assigned transactions
    - provides a method to get a specific unassigned transaction based on the transaction ID
    - returns the length of unassigned or assigned list when requested
    - provides a method to assign a transaction to an employee
    - provides a method to save changes to fraud\_transaction.csv
* Resources
  + CSV\_Logs
    - **fraud\_transactions.csv** - logs data about detected fraud transactions
    - **inaccuracy\_report.csv** – logs data about false positive or false negatives reported by users
  + Data\_Sets
    - **class\_only\_test** - test data set containing only the class attribute
    - **class\_only\_train** - training data set containing only the class attribute
    - **creditcard.csv** – original data set from Kaggle (Machine Learning Group, 2016)
    - **no\_class\_not\_scaled\_test.csv** - test data set containing all attributes except class
    - **no\_class\_not\_scaled\_train.csv** - training data set containing all attributes except class
  + Models
    - **random\_forest\_150** - A random forest model with 150 trees.
    - **random\_forest\_175** - A random forest model with 175 trees. It is the performed the best, and therefore is the model used by app.
    - **rfecv\_random\_forest\_200** – A random forest model with 200 trees using fields selected by recursive feature elimination with cross-validation.
  + **credentials.csv** - This contains a list of authorized employees' employee IDs and encrypted passwords.
  + **employees.csv** – a list of analysts’ employee id, employee name, and number of assigned cases
  + **last\_process\_stats** – contains evaluation data of predictive model
* View
  + **controller\_logfile.log** – logs users’ actions
  + **Dashboard.py** – python code version of Dashboard.ui
  + **Dashboard.ui** – It contains generated code from QtDesigner for all screens except for the log-in screen. QtDesigner is an application that provides a GUI editor to create a GUI. It is part of the pyQt5 framework.
  + **DashboardController.py** – connects the back-end code to the GUI
  + **LogIn.py** – python code version of Log\_In.ui.
  + **Log\_In.ui** – generated code from QtDesigner for the log-in screen.
  + **MplCanvas.py** – creates a matplotlib canvas object and turns it into a widget usable in the pyQt5 framework.
* main.py – starts the program
* Pipfile – contains a list of dependencies for the program
* Pipfile.lock – This is an auto-generated file created by pipenv. It allows the user's computer to know which version of each dependency is required for the application.