**Short summary**

Your job is to create a simple backend-only application where an *investor* can *track* and *analyze* his *investments*.

**Description**

An investor can invest his money in different financial instruments. One of them are Loans, money borrowed to customers or businesses. To simplify the situation, we’ll say that each loan has one single expected repayment (this is sometimes called bullet repayment). This means that the investor invests some money in the loan (lends money to the customer/business) and then expects a repayment on the maturity date of that loan.

**Main entities**

* Loan - contains the information of a loan in which the investor has invested.
* CashFlow - an event in the lifecycle of the loan.
  + We’ll consider two types of cash flows: funding, repayment

**Entities attributes**

| ***Entity*** | ***Attribute*** | ***Type*** | ***Source*** | ***Description*** |
| --- | --- | --- | --- | --- |
| Loan | identifier | string | Input | A unique identifier of the loan) |
| issue\_date | date | Input | The date in which the loan was created by the respective institution (bank etc.) |
| total\_amount | float | Input | The total amount of the loan |
| rating | integer | Input | The rating of the loan, from 1 to 9. |
| maturity\_date | date | Input | The date in which the repayment is expected. |
| total\_expected\_interest\_amount | float | Input | The total interest amount expected to be repaid on the maturity date from the debtor. |
| invested\_amount | float | Calculated | The amount invested in the loan by the investor |
| investment\_date | date | Calculated | The date in which the investment was placed. |
| expected\_interest\_amount | float | Calculated | The interest amount expected to be repaid to the investor on the maturity date. |
| is\_closed | boolean | Calculated | Indicates if the investment is still completed, or it is still open. |
| expected\_irr | float | Calculated | The expected Internal Rate of Return, a percentage indicating the expected return rate. |
| realized\_irr | float | Calculated | The Internal Rate of Return, calculated when the investment is closed. |
| CashFlow | loan\_identifier | string | Input | The identifier of the loan to which this cash flow is related. |
| reference\_date | date | Input | The date in which the event happened |
| type | string | Input | Funding/Principal Repayment/Interest Repayment |
| amount | float | Input | The amount funded/repaid. |

**Fields calculation:**

1. investment\_date
   * Calculated only when the loan is created.
   * Equal to the reference\_date of the funding cash flow of the loan.
2. invested\_amount
   * Calculated only when the loan is created.
   * Equal to the amount of the funding cash flow of the loan.
3. expected\_interest\_amount
   * Calculated only when the loan is created.
   * total\_expected\_interest\_amount \* (invested\_amount / total\_amount)
4. is\_closed
   * Calculated whenever a new cash flow for the loan is received.
   * If the loan has received a total repaid amount equal to the expected amount (invested\_amount + expected\_interest\_amount) or higher, it is considered closed. Otherwise it is still open.
5. expected\_irr
   * Calculated only when the loan is created.
   * Use the XIRR function for this purpose.
   * *Hints:* 
     + IRR is a percentage, representing the rate of return of the investment.
     + You can use the XIRR function from any Python library or any implementation on the web.
     + The XIRR function takes an array of (date, amount) tuples, representing cash flows (realized or expected) and returns a float between 0 and 1.
     + For the expected IRR, the cash flows that should be used are:
       1. The funding
          - date: reference\_date
          - amount: funding amount (negative)
       2. The expected repayment:
          - date: maturity\_date
          - amount: invested\_amount + expected\_interest\_amount (positive)
6. realized\_irr
   * Similar to the expected\_irr
   * Calculated only when the loan is closed.
   * The cash flows that should be used are:
     + The funding
       1. date: reference\_date
       2. amount: funding amount (negative)
     + The repayment
       1. date: reference\_date
       2. amount: repayment amount (positive)

**Functional Requirements**

* Allow the user to upload the data of his investments in the application.
  + When doing this, the user will provide 2 CSV files:
    - One containing the loans.
    - One containing the cash flows of the loans.
  + Calculate the loan fields as necessary.
* Allow the user to create a repayment for a loan, not through the CSV file.
  + The necessary loan calculations should take place in this case as well.
* Allow the user to view his loans and filter them by any attribute.
* Allow the user to view the cash flows and filter them by any attribute.
* Allow the user to view statistics on his investments:
  + Number of Loans
  + Total invested amount (all loans)
  + Current invested amount (only open loans)
  + Total repaid amount (all loans)
  + Average Realized IRR
    - Weighted average of realized IRR, using the loan invested amount as weight.
    - Consider only closed loans.

**Other Requirements**

* The application should expose a REST API for the required functionalities.
* The application should support authentication and authorization.
  + 2 types of users:
    1. Investor - Can do anything on the application.
    2. Analyst - Read-only permissions.
* The processing of the CSV files should happen asynchronously.
* The statistics should be stored in a cache.
  + Whenever new loans/cash flows arrive, the cache should be invalidated.
* The project should be set up with the usage of docker and docker-compose.

**Technologies:**

* [Django](https://www.djangoproject.com/)
* [Django REST framework](https://www.django-rest-framework.org/)
* [drf-spectacular](https://github.com/tfranzel/drf-spectacular)
* [Celery](https://docs.celeryproject.org/en/stable/getting-started/introduction.html)
* [Redis](https://github.com/jazzband/django-redis)
* [docker + docker-compose](https://docs.docker.com/)

**Example files:**

* loans.csv
* cash\_flows.csv