Stock validation application  
Database schema local SQL lite database

Tables

* User
  + # one-to-one -> (A User) - has - (One Configuration) .ie json stored app settings
    - # one-to-many-> (A User) – has (multiple portfolios) . two types [Paper\_ portfolios, Real\_ portfolios] ie. TFSA, Risky-bets, value-strategy etc
  + Paper\_ portfolios
    - # one-to-many-> (A Paper\_ portfolios) – has (multiple Holdings)
  + Real\_ portfolios
    - # one-to-many-> (A Real\_ portfolios) – has (multiple Transactions)
  + Portfolios
    - # one-to-many->(Any portfolio) - has (multiple Stocks)
* Stocks
  + # many-to-many -> (A Stock) - belongs to - (Sectors)
  + # one-to-many -> (A Stock) - has - (Dividends over time monthly )
  + # one-to-many -> (A Stock) - has - (Historical Prices) monthly or yearly
  + # one-to-many -> (A Stock) - has - (Historical Metrics) upon analysis
* Sectors
  + # many-to-many -> (A Sector) - contains - (Stocks)
* Documents
  + Many-to-one -> (many Documents) will be associated with a (A Stock)
  + Many-to-one -> (many Documents) will be associated with a (A Sector)

General thoughts:

* Decouple backed from frontend with the application running locally in docker with exposed port, and external volume for persistent data
  + Personally not worried about data in a file but leave space to change db.
* Use streamlet for the frontend and rapid prototyping
* Continuing pytest
* Do not worry about the documents table integration with a llm

TODO:

* Update libraries
* Split portfolios
* Try to get a user interface asap running in docker