Attendance
- Our Team:
  - Blake, Cole, Lee, Nick
- Other:
  - Ben

Questions going into the meeting
- What are the components we need
- Review goals that need to be met this semester

Overview on what was Discussed
- Broke down the goals for this semester and system design

Discussion
- Goals for the System
  - 0.3) Must be done, absolute minimum
  - 0.7) Meets requirements
  - 1.0) Stretch Goals
  - Optimization (Backend)
    - 0.3)
      - Unconstrained optimization w/ historic μ (mean), σ (variance) estimates
    - 0.7)
      - Generate multiple portfolios based on fixed range
      - Constrained optimization
        - Property type
        - Market
        - Property + Market
      - Handling of missing data (Statistical imputation)
    - 1.0)
      - Revisit μ (mean), σ (variance) estimates
  - Dash Shell (Frontend)
    - Navigation (How the user moves around the app, etc)
    - Layout Design (Button and feature placement, colors, sizes, etc)
    - User Input
    - Backend Communication (How the frontend utilizes the backend)
    - 0.3)
      - Gather user Input for optimizer from user, stale data from NCREIF file
      - Launch Optimizer

- - - Basic Layout to meet minimum user requirements
      - Send email
    - 0.7)
      - Input of new market file w/ current holdings
      - File written to DB for consumption by PBI / backend
      - Multiple iterations of UI w/ end users (Portfolio managers)
      -
    - 1.0)
      - User specific risk range
      - User guide
  - PowerBI (Frontend)
    - API
    - Dashboard Design
    - 0.3)
      - Current Holdings shown in PBI app, embedded in Dash
    - 0.7)
      - Current and Optimal Portfolios shown in PBI in Dash.
      - Visualize efficient frontier
      - Multiple iterations based on user feedback
    - 1.0)
      - Current vs Optimal Comparison w/ recommended actions
  - System
  - 0.3)
    - Self (local) hosted
  - 0.7)
    - EC2 hosted
  - 1.0)
    - Automated deploy script
- Data output?
- Different sessions?

- System Goals

1) Optimization

0.3) Unconstrained opt
w/ historic μ, σ estimates

0.7) + generate multiple pfo's based
on fixed risk range
+ constrained optimization
  • prop type
  • market
  • prop + mkt
+ handling of missing data (stat imputation)

1.0) Revisit μ, σ estimates

2) Dash shell

0.3) • Gather input for optimizer
from user, using stale market
data from NCREIF file.
• Launch optimizer
• Basic layout to meet min.
user requirements

0.7) + input of new market
file w/ current holdings.
+ file written to DB for
consumption by PBI / backend
+ mult. iteration of UI w/
end users

1.0) + user specifies risk range
+ user guide

3) Power BI

0.3) Current holdings
shown in PBI app
embedded in Dash

0.7) Current & optimal
portfolios shown in PBI
in Dash. Multiple UI iterations
based on user feedback.
• Viz efficient frontier
• Current vs. optimal

1.0) Current vs. optimal
comparison w/
recommended actions

Backend comm
Front end

weight assign — Dash Shell — Power BI
User Input — Navigation — API — Dashboard Design

- System Design

NCREIF
market
p type
yyq
tret
aret
---
market value

Asset
id
name
portfolio id
expected return
---
opt weight
current weight

Portfolio
id
name

System Div

Dash
SQLite → Opt Eng. ⇄ Power BI Embedded ⇄ Dash U.I.

Power BI
Python
Data pull
script

HTTP

System
0.3) Self (local) h
0.7) EC2 hosted
1.0) automated depl
script

  - Dash
    - Optimization Engine
    - Power BI Embedded (HTTP)
    - Dash UI
  - Database
    - NCREIF
      - Market, ptype, yyq, tret, aret

- - - // market value
      - Asset
        - Id, name, portfolio id, expected return
        - // current weight, optimal weight(market value)
      - Portfolio
        - Id, name
  - Power BI (HTTP)