Real Estate Portfolio Optimization

sdmay19-07

http://sdmay19-07.sd.ece.iastate.edu/

Advisor Chinmay Hendge

Client Principal Financial Group

Principal℠

Meet the Team

Blake Roberts

Project Lead / Backend

Software Engineering

Kevin Johnson

Quality Control / Frontend

Computer Engineering

Nickolas Moeller

Report Manager / Backend

Software Engineering

Leelabari Fulbel

Meeting Facilitator / Frontend

Software Engineering

Colton Goode

Meeting Scribe / Backend

Computer Engineering, Management of Information Systems

Principal℠

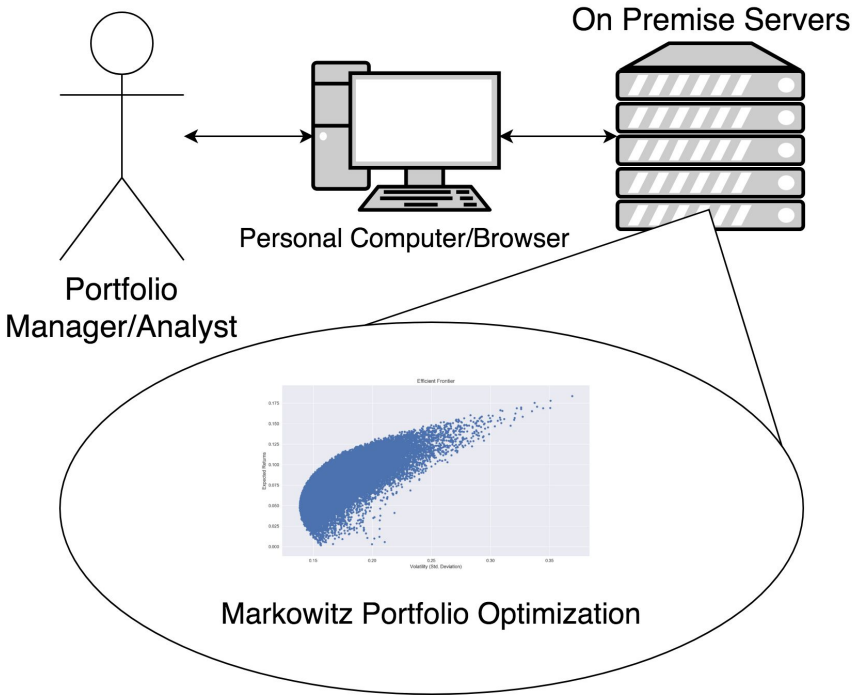
Project Plan

Problem Statement

- Portfolio managers (PM's) must decide the composition of assets within their real estate portfolio
- Asset, market and portfolio level analysis helps form this decision
- Principal utilizes a third party, Costar, to gain portfolio insight
    - Reports are long, inefficient, non-customizable, and pricy

Conceptual Sketch

- Input market data and portfolio holdings
- Perform Markowitz portfolio optimization
- Output analytical data visualization



Functional Requirements

User can …

1. Generate optimal portfolios via configurable constraints such as market and property type
2. Visualize efficient frontiers compared to markets, property types, and the current portfolio
3. Visualize current holdings by geography, property type, expected return, and risk
4. Visualize differences between optimal and current holdings in maps, plots, and other charts
5. Receive recommend buy and sell decisions given the current portfolio holdings
6. Share results via file export and/or email

Non-functional Requirements

- The system will use only open source libraries and frameworks
- Principal data must not pass through non-vetted third party systems
- Optimization must take no longer than 5 seconds to calculate

Technical/Other Constraints/Considerations

- Principal has servers to host internal applications
- The application must be easy to use and should not require training
- Future project will be maintained by Principal's data science team

Market Survey (What Makes Us Unique)

- An inhouse tool keeps private data safe, and allows iteration and new feature development
- Most often, portfolio optimization and analysis tools are for the stock market
- Allows “what-if” real time analysis

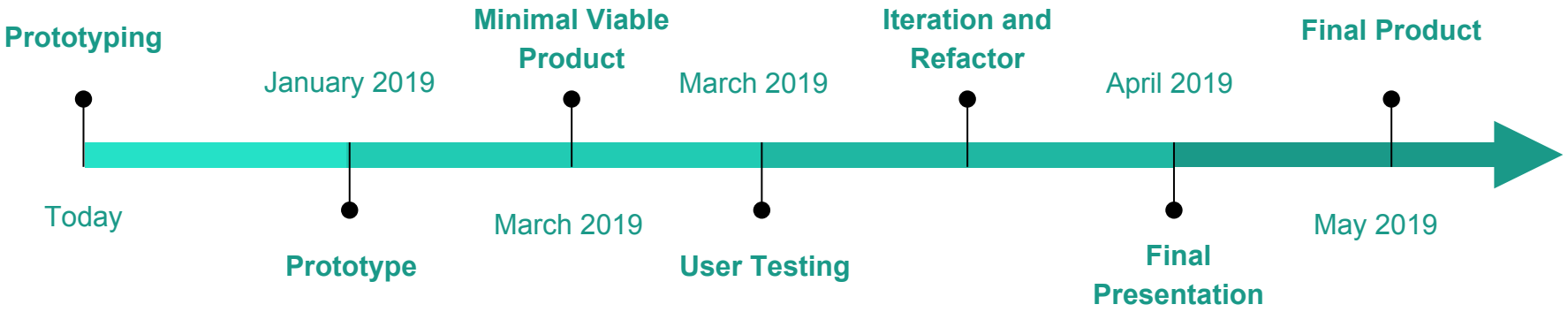
Potential Risk & Mitigation

- The system is too slow and will reduce user experience
  - Portfolio optimization is not resource intensive
- Results are not in a format easily usable by the PMs
  - Coordinated user testing with Principal (Spring 2019)
- The system consumes sensitive data and may leak it
  - Only integrate Principal vetted systems

Resource/Cost Estimate

- The project is expected to incur little to no cost
- *May* need additional Power BI Azure licence.
- *May* use Principal's existing AWS RDS/EC2 instances for SQL DB

Project Milestones & Schedule



Principal[SM]

System Design

Detailed Design

Input specification:

- NCREIF market data
- Portfolio holdings

Output specification:

- Optimized portfolio
- Efficient frontier

UI and software specifications:

- JavaScript (runtime compiled)
- Modern browsers (Chrome)

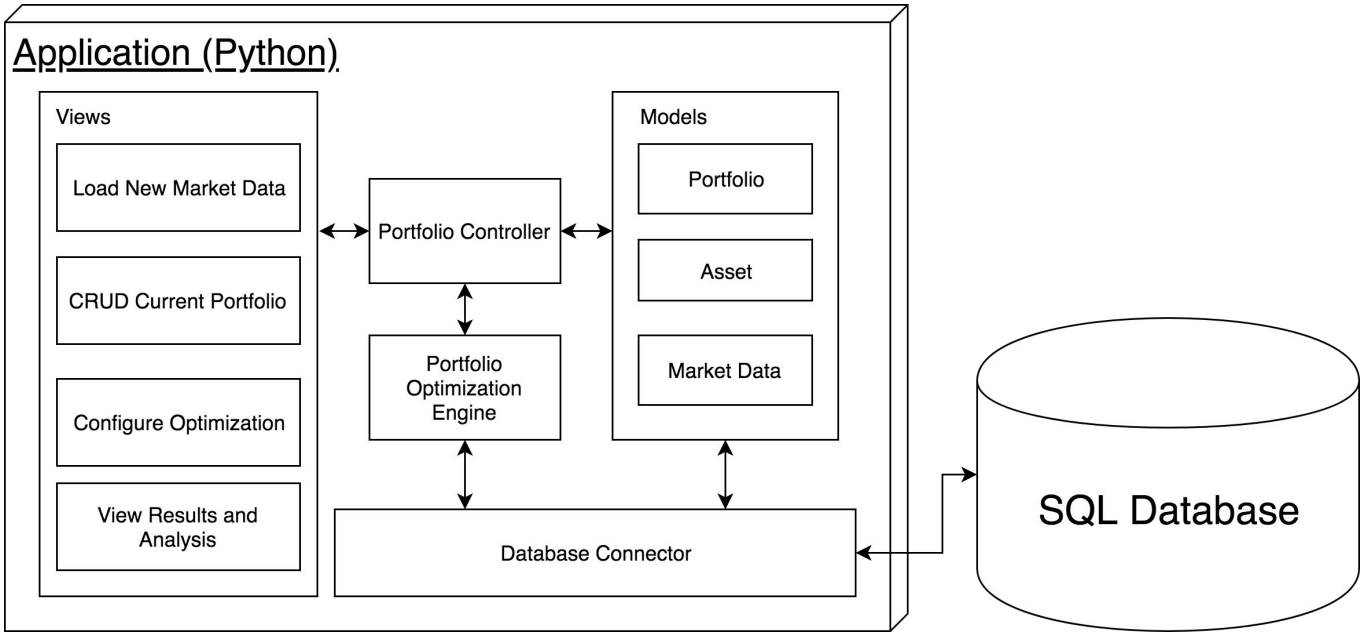
Software/Technology Platforms Used

- Python
- Flask
- SQLite
- Dash
- Plotly
- Power BI

System Description / Operating Environment

- Operating System Agnostic
- Web application written in Python utilizing the Flask framework
- Client views built with Dash and Power BI
- SQLite or on premise SQL Database

System Block Diagram



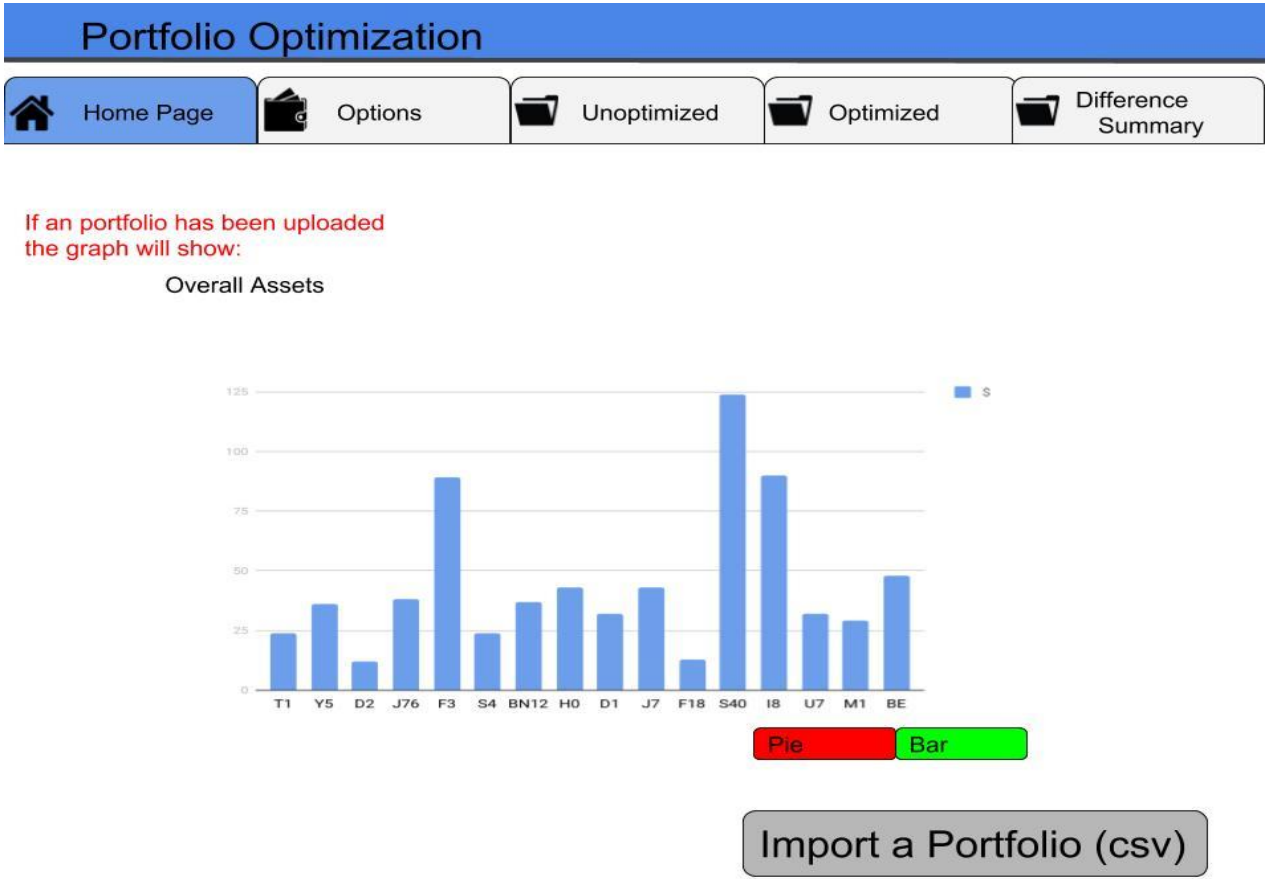
User Interface

Dash, a Python library built on top of Plotly, compiles JavaScript data visualizations (i.e. charts and graphs)

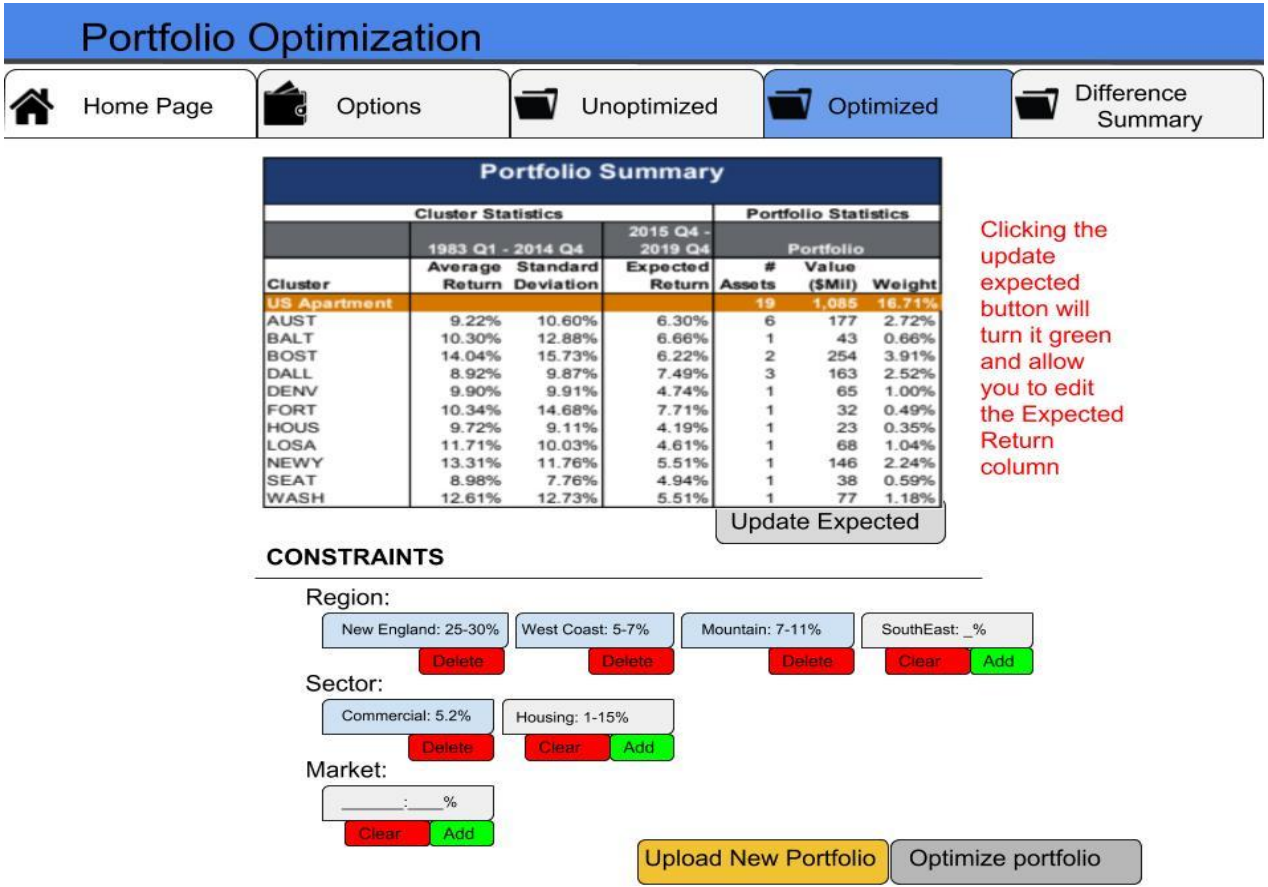
The application consists of five major components:

- uploading portfolios
- creating constraints
- viewing the portfolio before and after it is optimized
- comparing both versions

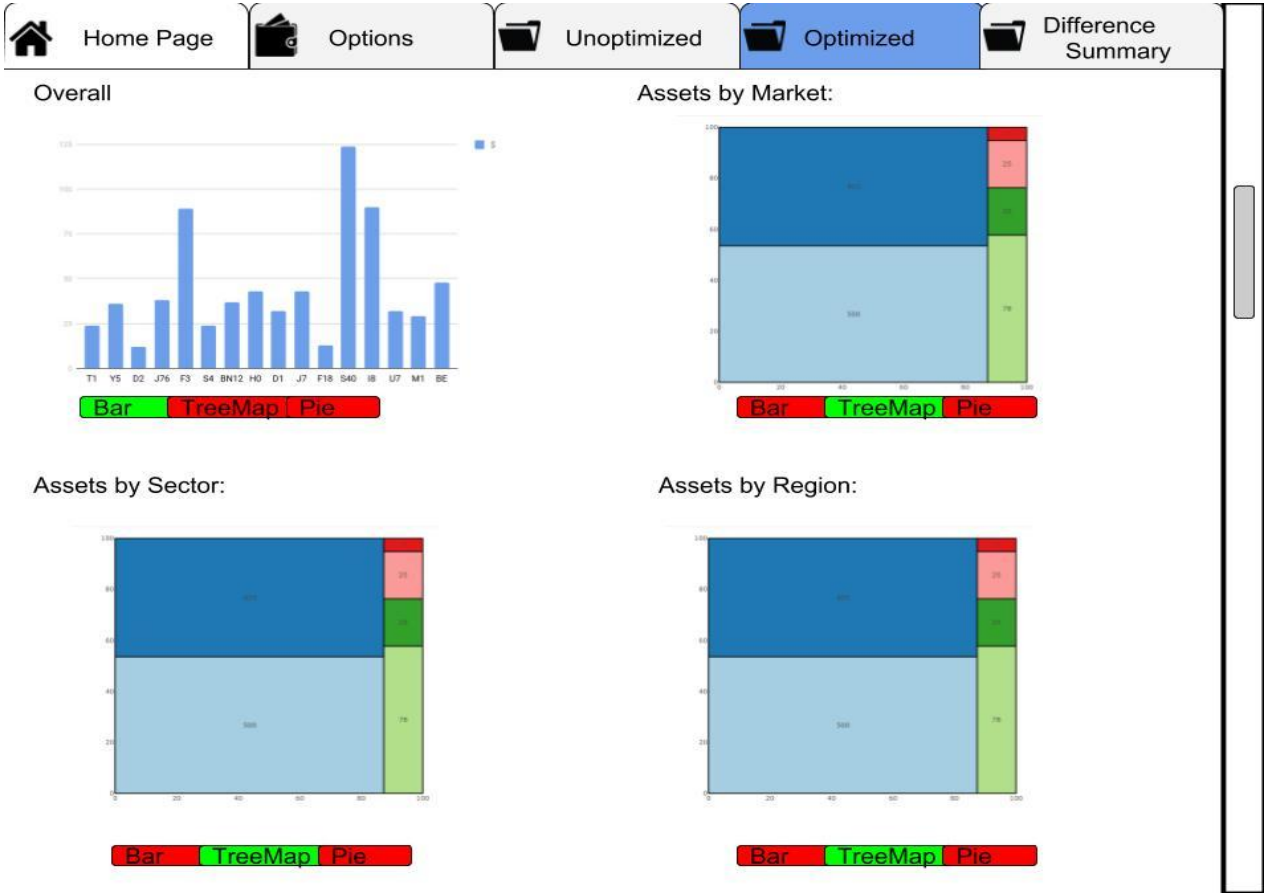
Home Page


Portfolio Optimization
Home Page
Options
Unoptimized
Optimized
Difference Summary
If an portfolio has been uploaded the graph will show:
Overall Assets
125
100
75
50
25
0
$
T1 Y5 D2 J76 F3 S4 BN12 H0 D1 J7 F18 S40 I8 U7 M1 BE
Pie
Bar
Import a Portfolio (csv)

Options Page



Portfolio Page



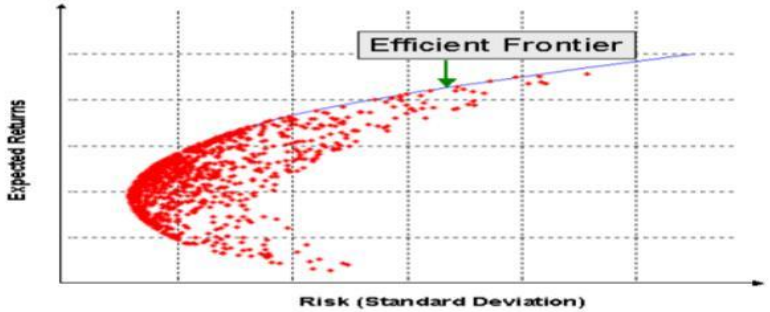
Principal℠

Portfolio Page (continued)

Bar | TreeMap | Pie

Bar | TreeMap | Pie

Efficient Frontier



Summary:

Sharpe Ratio: ******

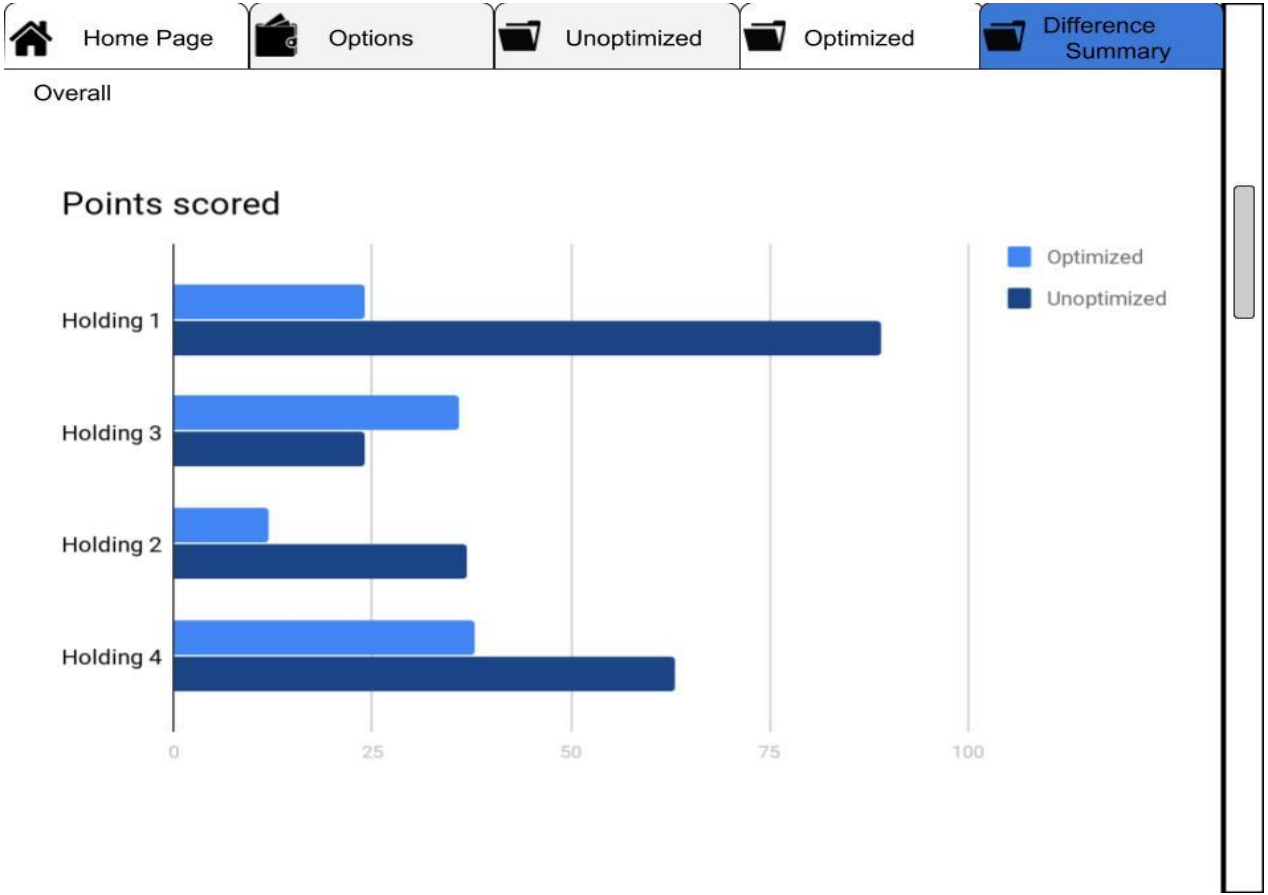
Expected Return ******
/ Expected Risk ******

Top **5** Holdings:

1. ******
2. ******
3. ******
4. ******
5. ******

Export as CSV | Send to Email

Difference Page



Test Plan

- Utilize Python's unittest package
- Build database with known correlation
- User testing in Spring 2019

Principal℠

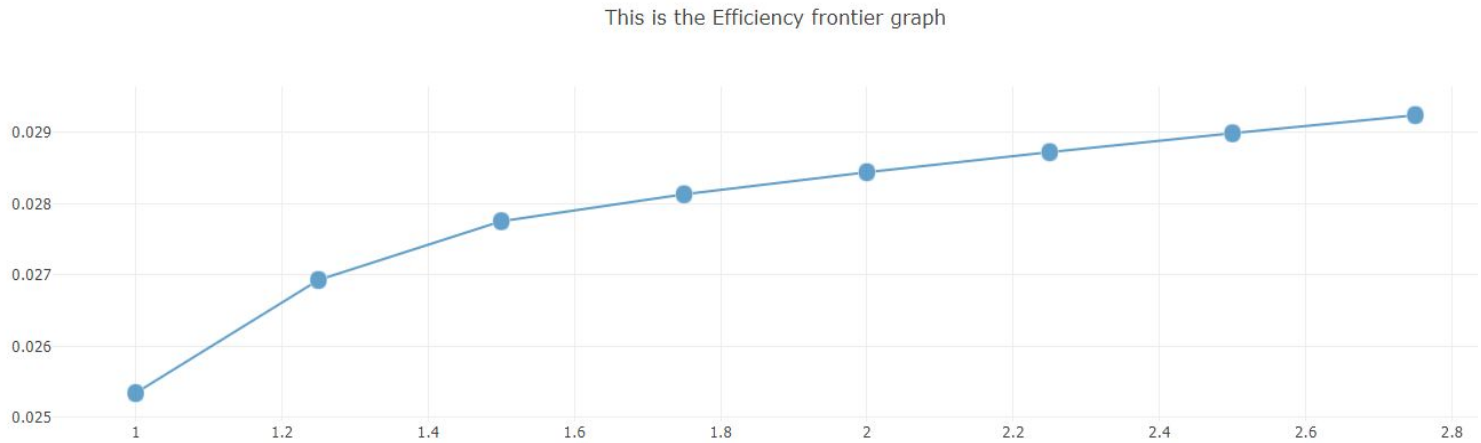
Prototype Implementations

```
constraints = [Constraint.market_max('Houston', 0.5)]
risk_data, ret_data, weight_data = markowitz(db, user_constraints=constraints)
```

```
run(7): max_risk=7.5625   return: 2.848%   risk: 2.75%
    ...
    Houston_________50.00%    Jacksonville_____0.00%    Lake County_____0.00%
    Los Angeles_____0.00%     Miami___________14.08%    Minneapolis_____0.00%
    ...
    San Jose________0.00%     Santa Barbara___35.92%    Seattle, WA_____0.00%
```

Prototype Implementations

Efficient Frontier Graph



Principal℠

Conclusion

Project Status

Things We Have:

- Server (Flask/Dash Application)
- Markowitz implementation w/ market constraints

Things We Need:

- Fleshed out UI/UX
- Markowitz w/ property type and market/property type pair constraints

Plans For the Next Semester

- Prototype - January 2019
- Minimal Viable Product - March 2019
- User Testing - March & April 2019
- Automation of Deployment - April 2019
- Finalize Documentation and Project Handoff - May 2019

Principal℠

Questions?