Money Rocket

A Robo-Advisor Investment Wizard

MIE479 Intermediate Progress Report

Ling Long, 1002516618

Ahmed Nawaz Amanullah, 1001325773

Karl Chen, 1001198660

Andre D'Souza, 1002268397

David Gonder, 1001410920

November 5th, 2018

Universe of Investible Securities

The major change from Money Rocket's original plan has been to expand the asset universe. The original asset universe only composed of S&P 500 equities, as well as select US Treasuries of varying duration. It has been decided that this decision would limit strategy and global exposure. Therefore, the decision has been made to expand the asset universe using ETFs, to increase asset exposure without greatly increasing the overall quantity of assets. Problems with the original asset universe include:

- The portfolio would only be concentrated in the U.S. Markets. While the largest markets, investors may benefit from global exposure in periods where European and Asian equities outperform their North American counterparts. To combat this, ETFs tracking both European and Asian markets will be used to diversify the possible portfolios.
- The S&P only provides access to large cap and mid-large companies. In order to allow for optimal growth and value strategies, mid cap and small cap companies should be included. It is often the case that the most attractive returns are found from small cap companies due to their higher growth potential. S&P companies are typically over $6 Billion USD market cap excluding several lucrative opportunities.
- Direct U.S. Treasury access is less liquid and requires carrying off the run bonds if holding for long periods. To mitigate liquidity concerns instead of buying direct Treasuries of varying duration (ie. T-bills or notes), an ETF will be used to add fixed income exposure. The chosen ETF will only be comprised of U.S. Treasuries to maintain the risk-free profile desired.
- Emerging Market Bonds offer higher returns than U.S. Treasuries and another asset class opportunity for returns. With a different risk profile, emerging markets may be useful to control volatility or seek high returns than U.S. Treasuries while remaining in the fixed income space.

Asset Universe List:

i. <u>S&P 500 Equities (Security Level)</u>

The stock universe will be composed of the securities comprising the Standard & Poor's 500 index in November 2018. As a side note, the historic price data spanning back may include periods that a security was not included in the index. The S&P 500 index is only comprised of companies listed on the New York Stock Exchange and Nasdaq, simplifying data collection. In the case security level historic prices older than recent periods are not available (ie. recent IPO), the asset will be removed from the list of available equities.

ii. <u>U.S. Large Cap ETF</u>

While strongly correlated with the S&P 500 index, the SPDR S&P 500 ETF (Ticker SPY) will be used to add diversified exposure to US large cap assets. As the largest ETF in the world, low spreads and high liquidity can be found. Industry leaning portfolios (ie. Technology, Manufacturing) can still be formed using assets from the S&P 500 universe, while the ETF provides a lower risk profile.

iii. <u>U.S. Mid Cap ETF</u>

Mid cap companies offer a trade off between large and small caps, with more potential upside and risk than large caps, but safer than small caps. This will greater allow a growth strategy to be implemented, while still keeping volatility low. The ETF in use will be the iShares Core S&P Mid-Cap ETF (Ticker IJH). Being one of the most popular ETFs on the market, there is no liquidity concern with over 1.5M average daily volume and low historic spreads.

iv. <u>U.S. Small Cap ETF</u>

Small cap companies offer maximum growth potential, but also the greatest risk in the equity markets. Often neglected compared to other companies, small caps may offer diamonds in the rough with returns impossible to see in large caps due to an already high market cap. The high growth potential comes with increased risk and volatility, with higher chances of failure and less liquidity. However, these are less concerning when accessing through an ETF. The choice will be the iShares Russel 2000 (Ticker IWM). The popularity again mitigates liquidity concerns in the small cap space, with an average volume over 23M shares.

v. <u>European Index</u>

To add global exposure, the Vanguard FTSE Europe ETF (Ticker VGK) will be used. With a combination of several European securities and the Financial Times Stock Exchange 100 (FTSE) index, it offers diversification from the U.S. while still traded on U.S. exchanges. The FTSE index tracks the London Stock Exchange with representation of about 80% the LSE’s market cap, providing a strong correlation to the United Kingdom’s economic outlook, and offering diversification compared to U.S. only concentrated portfolios.

vi. <u>Eurozone Index</u>

Using the iShares MSCI Eurozone ETF (Ticker EZU), exposure to mid and large cap companies in Eurozone countries will be gained. Most notably, the United Kingdom, Sweden and Switzerland are excluded, adding diversification from VGK. The assets comprising EZU are concentrated in France and Germany. It will allow developed market exposure with liquid access to Eurozone stocks, and similarly add international diversity.

vii. Asian ETF

The Vanguard FTSE Pacific ETF (Ticker AAXJ) will be used to increase global portfolio exposure. This ETF is composed of all cap range securities from both developed markets in the Asia-Pacific region, offering another growth profile. With high volume well over 1M and low spreads, liquidity risk is mitigated. The fund has increasing Small Cap holdings, adding growth potential from Asian markets.

viii. U.S. Treasuries

The iShares U.S. Treasury Bond ETF (Ticker GOVT) is a direct gauge of the U.S. Treasury market. GOVT offers exposure to risk-free bonds ranging 1-30 years, and low cost liquid access to the U.S. treasury market. It can be used to add stability to the portfolio, in the case risk aversion required volatility to be controlled. The tradeoff involves low return potential, however, it offers strong liquidity with over 1M daily volume, and can be useful in the case of hedging impending recessions, adding benefits to equity only portfolios.

ix. Emerging Market Bonds

Contrary to U.S. Treasuries, emerging markets offer higher returns with higher risk. The iShares JPMorgan USD Emerging Markets Bond ETF (Ticker EMB) will be used. It is comprised of US Dollar denominated bonds sovereign debt issued by emerging markets, with a range of 2+ maturity. The strong liquidity EMB offers is a major benefit over direct emerging market debt, and can be used to add stability with less of a drag on returns. Unlike U.S. bonds, emerging markets remain volatile and may offer worse risk profiles than equities.

Business Logic

Preprocessing + Metrics + Parameter Estimation + Financial Modelling

Parameter Estimation

Fama-French 3 factor model

Using accuracy, explanatory power and computational efficiency as our design considerations, we opted for the Fama-French 3 factor model as the estimation tool of the parameters for our financial optimization model. This approach allows for an enhanced explanatory performance over the simpler CAPM factor modelling, while at the same time retains the advantage of having an improved computational efficiency over the sample statistics approach. Using this approach, the excess expected return of an asset is estimated using the expression:

$$E(r_i) - r_f = a_i + \beta_i[E(r_M) - r_f] + s_i E(SMB) + h_i E(HML)$$

where,

- $E(r_i) - r_f$ is the excess expected return of an asset *i*.
- $E(r_M) - r_f$ is the excess expected market return.
- $E(SMB)$ and $E(HML)$ are the expected returns of the small minus big (SMB) and high minus low (HML) portfolios.
- The factor values will be obtained from Kenneth R. French's Data library [1].
- The factor coefficients for asset *i* are determined by solving a system of 3 linear equations:

$$\sigma_{r_i f_j} = \sum_{k=1}^{3} b_{ki} \sigma_{f_j f_k}$$

  - Use numpy.linalg.solve method for this [2].

- a_i is estimated from the mean return of asset *i*, i.e. $a_i = (\overline{r_i} - r_f) - \beta_i[\overline{r_M} - r_f] - s_i\overline{SMB} - h_i\overline{HML}$.

The mean asset returns as well as the asset covariances and variances are subsequently estimated using the expressions:

$$\overline{r_i} = a_i + \sum_{j=1}^{n} b_{ji}\overline{f_j} = r_f + a_i + \beta_i[E(r_M) - r_f] + s_i E(SMB) + h_i E(HML)$$

$$\sigma_{ij} = \begin{cases} \sum_{l=1}^{3}\sum_{k=1}^{3} b_{li}b_{kj}\sigma_{f_l f_k}, i \neq j \\ \sum_{l=1}^{3}\sum_{k=1}^{3} b_{li}b_{kj}\sigma_{f_l f_k} + {\sigma_{e_i}}^2, i = j \end{cases}$$

Note: Will use the statistics methods available within the numpy library to compute the necessary sample statistics for our purposes above (e.g. mean asset returns, mean factors etc.) [3].

Black Litterman parameters

While not our most preferred portfolio optimization technique, we will compare its performance in relation to the portfolio optimization strategies we have chosen in our proposal (CVaR minimization, Most Diverse (Bucketing) MVO and MVO w/ Cardinality), and, if necessary, re-evaluate the appropriateness of our decision. Within this approach, we need to first estimate the implied equilibrium returns of the assets:

$$\Pi = \lambda \Sigma w_{mkt}$$

where,

- Π is a NX1 column vector containing the implied equilibrium returns of the assets.
- λ is the risk aversion co-efficient, which is calculated by dividing the expected excess return by the variance of the market portfolio,
$$\lambda = \frac{E(r) - r_f}{\sigma^2}$$
- Σ is the covariance matrix of the excess returns (NXN matrix).
- w_{mkt} is a NX1 column vector containing the market cap weight of the assets.

The second step of the approach involves deviating the portfolio from the equilibrium market portfolio based on the subjective views of an investor regarding the asset data. Firstly, there is the computation of the Combined Return Vector:

$$E[R] = [(\tau\Sigma)^{-1} + \mathrm{P}'\Omega^{-1}\mathrm{P}]^{-1}[(\tau\Sigma)^{-1}\Pi + \mathrm{P}'\Omega^{-1}\mathrm{Q}]$$

where,

- $E[R]$ is a NX1 column vector containing the combined returns of the assets.
- τ is a scalar and there are different ways to calculate it in literature. Lee suggests a value between 0.01 and 0.05 [4]. Satchell and Scowcroft suggest a value of 1, while

Blamont and Firoozye claim that τ is approximately equal to 1 divided by number of observations [4]. We will try all 3 alternatives and compare the performance difference when backtesting.

- Σ is the covariance matrix of excess returns (N x N matrix).
- P is the view matrix, specifically it specifies the assets involved in the views (it is a K x N matrix, with each row pertaining to a view). We plan to take our asset return estimates from our factor modelling as our views. Another option within our mind is to generate the return views using random normal generation, and generate portfolios based on those.
- Ω is a diagonal covariance matrix that represents the uncertainty in the views (K x K matrix). Using Izdorek's algorithm [4], this is computed by solving the optimization problem:

$$min \sum (w_{k,\%} - w_k)^2$$

subject to $\omega_k > 0$
where

$$w_k = [\lambda\Sigma]^{-1}[(\tau\Sigma)^{-1} + p_k{}'\omega_k{}^{-1}p_k]^{-1}[(\tau\Sigma)^{-1}\Pi + p_k{}'\omega_k{}^{-1}Q_k]$$

The $w_{k,\%}$ in the optimization problem specified above is obtained by following the first 5 steps within Izdorek's algorithm:

1. Calculate the combined return vector assuming each view is 100%. That is, for each view implement the following formula:
$$E\left[R_{k,100\%}\right] = \Pi + \tau\Sigma p_k{}'(p_k\tau\Sigma p_k{}')^{-1}(Q_k - p_k\Pi)$$
2. Calculate the weights associated to the 100% confident views using:
$$w_{k,100\%} = (\lambda\Sigma)^{-1}E[R_{k,100\%}]$$
3. Calculate the deviation of the 100% confidence weights from the market:
$$D_{k,100\%} = w_{k,100\%} - w_{mkt}$$
4. Calculate the desired tilt for each view based on confidence specified by user:
$$Tilt_k = D_{k,100\%} * C_k$$
5. Estimate target weight vector based on tilt:
$$w_{k,\%} = w_{mkt} + Tilt_k$$

- Q is the KX1 view vector that contains the investor's subjective views on asset returns. For example, in our approach to take the return estimates from our factor modelling as our subjective views, this vector will simply contain the factor modeling's return estimates for each asset.

After the Combined Return Vector is obtained, the portfolio's weights are estimated by the expression:

$$w = (\lambda\Sigma)^{-1}\mathrm{E[R]}$$

Performance Metrics

We will be using several metrics to assess and compare the performance between different portfolios, including

- Portfolio return

$$r_P = \sum_{i}^{n} x_i r_i = \boldsymbol{x}^T \boldsymbol{r}$$

$$\mu_P = \sum_{i}^{n} x_i \mu_i = \boldsymbol{x}^T \boldsymbol{\mu}$$

  Purpose: To compare the gain of the portfolios over time.

- Portfolio volatility

$$\sigma_P = \sqrt{\boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x}}$$

  Purpose: To compare the portfolio risks

- Sharpe Ratio (ex-post)

$$SR_P = \frac{\mu_P - R_f}{\sigma_P}$$

  Purpose: The ex-post variant of the Sharpe ratio is useful for comparing the risk-adjusted gain performance of the portfolios.

- Value at Risk (VaR) [5]

$$VaR_\beta(X) = \min_\beta \{V : P[-X > V] \leq \beta\}$$

  Purpose: To compare potential loss between different portfolios.

- Conditional VaR (CVaR) [5]

$$CVaR_\beta(X) = E[-X | X \leq -VaR_\beta(X)]$$

  Purpose: Same as VaR.

- Information Ratio (IR) [4]

$$IR = \frac{R_p - R_i}{S_{p-i}}$$

Note: Here R_p is the return of the portfolio, R_i is the return of the index/ benchmark and S_{p-i} is the standard deviation of difference between returns of the portfolio and the returns of the index (i.e. tracking error)

Purpose: For assessing validity of portfolio generated using BL approach. Bevan and Winkelmann suggest the Information Ratio of the generated portfolio should be at most 2.0 [4]. If Information Ratio is above 2.0, decrease the weight given to the views (by decreasing value of τ) [4].

Preprocessing

We can reduce the amount of computations necessary for the application to perform its base functionality by pre-calculating (and storing) data and estimates that will be used regularly within our financial modelling. This includes:

- The mean asset returns
- The asset covariances and variances
- The mean factors
- The asset-factor covariances
- The parameters/ estimations necessary within Black Litterman

Changes + comments

- The project proposal described the method to calculate the small minus big and high minus low portfolio factor values. Since the data for the factor is readily available from Kenneth R. French's Data library, we can skip this previously mentioned calculation step.
- Previously did not elaborate on how we would construct our views matrix and vector in our testing of the BL approach. As mentioned previously, we plan to take our asset return estimates from our factor modelling as our views. Another option within our mind is to generate the return views using random normal variable generation, and generate portfolios based on those.
- Added some additional details regarding computation steps/ methods (e.g. using numpy methods to solve systems of linear equations and computing sample statistics).
- Our biggest change since our proposal is expanding/ elaborating on our asset universe. This will have no impact on our parameter estimation methodology (i.e. our previous approach applies for these newly included assets as well).

Portfolio Optimization tools

In our proposal, we decided to go with CVaR, MVO with cardinality and Most Diverse (Bucketing) MVO as our optimization model.

Most Diverse (Bucketing) MVO [6]

One of the downsides to the basic MVO formulation is the over-concentration in a few assets in comparison to the asset universe being considered. The Most Diverse MVO is an optimization technique that helps mitigate the over-concentration issue of the basic MVO formulation. It does so by limiting the number of assets to build the optimal portfolio from, while ensuring the assets are selected such that correlation is minimized. The problem is formulated as follows:

$$\max_{z,y} \sum\sum \rho_{ij} z_{ij}$$

$$\sum y_j = k$$

$$\sum z_{ij} = 1 \; for \; i = 1, \dots \dots ., n$$

$$z_{ij} \le y_j \; for \; i = 1, \dots \dots, n; j = 1, \dots \dots ., n$$

$$z_{ij} \in \{0,1\}, \quad y_j \in \{0,1\}$$

where,

- ρ_{ij} is the correlation coefficient between assets *i* and *j* and is computed as $\frac{\sigma_{ij}}{\sigma_i \sigma_j}$.
- y_j is an integer variable which indicates if asset *j* is picked to be in our portfolio.
- z_{ij} is an integer variable that indicates if asset *j* is representative of asset *i*. Note that each asset can have at most one representative, so in other words this variable indicates if *j* is most similar to *i* within our portfolio.
- k is our asset restriction, i.e. the problem permits a selection of *k* assets. Each of these assets are buckets that contain assets that are most similar to it, and thus by selecting the buckets we make it so that the assets we are picking are most uncorrelated to each other.

We plan to use this approach to:

- Limit the assets our optimization model needs to consider, thereby improving computational efficiency. Additionally, since the basic MVO formulation tends to over-concentrate weight allocation in a few assets regardless, restricting the number of assets the model has to consider somewhat mitigates that.
- Satisfy the investor if they choose to have fewer assets within their portfolio by limiting the number of assets our model can allocate weight to.
- Alternatively, the optimal weight to be allocated to each asset according to this approach is given by the expression:

$$x_j = \frac{\sum_i^n V_i z_{ij}}{\sum_i^n V_i}$$

where,

- x_j is the optimal weight for asset *j*.
- V_i is the market capitalization for asset *i*.

Even though our proposed plan was originally to use this approach to simply select the assets our optimization model is to consider, we plan to use this to compare model performance when backtesting.

MVO w/ cardinality

MVO w/ cardinality is a formulation that builds on top of the basic MVO framework by adding constraints to limit the amount of assets the MVO can allocate weights to. Therefore, in addition to the variables that denote the asset weight allocation, x_i, we add binary variables that denote whether to include asset *i* within our selection, y_i. Then we add the following two constraints on top of the basic MVO formulation:

$$\sum_{i}^{n} y_i = k$$

$$x_i \leq y_i \ \forall i = 1,2, \dots, n$$

The first constraint ensures the model picks *k* assets to allocate weights to, while the second one forces to allocate weight only to assets that are picked. It is also possible to introduce lower and upper bounds on the weight allocation for each asset by tweaking the second condition as follows:

$$L_i . y_i \leq x_i \leq U_i . y_i \ \forall i = 1,2, \dots, n$$

Which is an option we may consider assessing during backtesting as a means of mitigating overconcentration.

Note that since we added binary variables to the model we have converted a formerly quadratic programming case to a mixed integer programming scenario, which reduces the efficiency. As stated formerly in our proposal, we plan to use the Most Diverse MVO formulation to reduce the assets this model needs to consider, thereby improving efficiency.

We plan to use this optimization technique to cater to the investors' needs in case they want to put a restriction on the amount of assets they want to build a portfolio with.

Conditional Value at Risk (CVaR) Optimization

The CVaR optimization formulation selects a portfolio that minimizes the loss/downside risk. The model can be expressed as follow:

$$\min_{x} CVaR_{\beta}(x)$$

$$s.t. x \epsilon \chi$$

where,

- χ represents the constraints relating to budget, short selling, and target returns.
- $CVaR_{\beta}(x)$ can be defined as $\gamma + \frac{1}{1-\beta}\int_{f(x,y)\geq VaR_{\beta}} f(x,y)^{+}p(y)dy$, where
    - β is the confidence level (note that in our performance metrics section the β in the VaR and CVaR definitions denoted the risk tolerance).
    - γ is the Value at Risk with respect to β
    - $f(x,y)$ is the loss function (in our case it would be the portfolio loss). Therefore $f(x,y)^{+}$ is portfolio loss in excess of γ.
    - $p(y)$ is the probability density of the loss.
- For a finite number of scenarios, the problem can be alternatively expressed as:

  $$\min_{x} \gamma + \frac{1}{(1-\beta)S}\Sigma z_s$$

  $$s.t. z_s \geq 0, z_s \geq \ f(x,y) - \gamma, x\epsilon\chi$$

  Here, z_s represents a generated loss scenario in excess of γ.
    - We plan to use Monte Carlo Sampling to generate the scenarios for CVaR optimization.
        - It is assumed stock returns follow a normal distribution, such that $r_t = \mu + \sigma\epsilon_{t-1}$, allowing individual price steps to be formulated as:

          $$S_{t+1} = S_t \exp\left[\left(\mu - \frac{\sigma^2}{2}\right)dt + \sigma\sqrt{dt}\epsilon_t\right]$$
        - We repeat the same expression above for a number of steps to generate stock paths (for this assignment, assume a single step is taken to estimate the stock price change in 6 months).
        - To account for correlation, introduce element $\zeta = \ L\epsilon, where\ L = chol(\rho,' lower')$:

          $$S_{t+1}^{i} = S_t^{i} \exp\left[\left(\mu - \frac{\sigma^2}{2}\right)dt + \sigma\sqrt{dt}\zeta_t^{i}\right]$$

          where *i* represents asset *i*.

If the user does not require the application to restrict the number of assets to allocate weights to, we plan to use CVaR optimization to build the portfolios.

Additional comments

We considered five other portfolio optimization models in our proposal that we did not end up choosing for use in our application. We plan to implement each of these models and compare their performance amongst themselves as well as with the results generated by the portfolio optimization models of our selection. This will contribute to help validating our design choices or review our design choices as necessary during backtesting. These five other models are as follows:

1. <u>MVO</u>

   The basic MVO formulation can take any of three forms:

   i. Minimize portfolio variance subject to expected return and budget constraints:

   $$\min_{\boldsymbol{x}} \boldsymbol{x}^{\boldsymbol{T}} \boldsymbol{Q} \boldsymbol{x}$$
   $$s.t. \boldsymbol{\mu}^T \boldsymbol{x} \geq R$$
   $$\mathbf{1}^T \boldsymbol{x} = 1$$

   ii. Maximize risk adjusted expected return:

   $$\max_{\boldsymbol{x}} \boldsymbol{\mu}^T \boldsymbol{x} - \boldsymbol{x}^{\boldsymbol{T}} \boldsymbol{Q} \boldsymbol{x}$$
   $$s.t. \mathbf{1}^T \boldsymbol{x} = 1$$

   iii. Maximize expected return subject to risk and budget constraints:

   $$\max_{\boldsymbol{x}} \boldsymbol{\mu}^T \boldsymbol{x}$$
   $$s.t. \boldsymbol{x}^{\boldsymbol{T}} \boldsymbol{Q} \boldsymbol{x} \leq \sigma_P$$
   $$\mathbf{1}^T \boldsymbol{x} = 1$$

   Among these formats, we have chosen to implement the first formulation as it will allow us to directly evaluate the performance of the basic MVO in relation to the third base functionality that must be accomplished (i.e. generate a minimal risk portfolio that achieves a stated return target).

   We are also thinking of implementing the basic MVO framework on an asset universe reduced using the Most Diverse MVO framework, which will be an especially good comparison test for evaluating how the MVO w/ Cardinality framework performs relatively. This will give us a good idea on if the MVO's optimality is affected adversely by introducing the cardinality constraint.

2. <u>Robust MVO</u>

Robust MVO is an optimization technique that mitigates the input sensitivity issue of the basic MVO framework. Since market data is insufficient to provide accurate estimates of the asset mean returns (and by extension the factors required to estimate the returns in our factor modelling), the idea is to add an uncertainty set to the basic MVO formulation that consists of constraints that incorporate a range of values of the uncertain parameter.

As the ellipsoidal uncertainty set is the preferred formulation of the robust MVO framework [6], we plan to implement that specific formulation:

$$\min_{\boldsymbol{x}} \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x}$$

$$s.t\ \boldsymbol{\mu}^T \boldsymbol{x} - \epsilon_2 y \geq R$$

$$y^2 = \boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x}$$

$$\boldsymbol{1}^T \boldsymbol{x} = 1$$

$$y \geq 0$$

$$(x_i \geq 0, i = 1, \dots, n)$$

where ϵ_2 is a measure of the maximum distance from our estimated expected returns. One option we plan to use is the concept of confidence regions, that is set ${\epsilon_2}^2 = {\chi_n}^2(1 - \alpha)$ [6] for a confidence region of $1 - \alpha$, where ${\chi_n}^2$ is the inverse cumulative distribution function of the chi-squared distribution with n degrees of freedom (corresponding to number of assets in portfolio). One value we can try out is $\alpha = 0.05$ for a confidence level of .95 or 95%.

Since the Black Litterman approach is another alternative that attempts to mitigate the input sensitivity issue of the basic MVO framework, the comparison of this model's results with the BL approach will be very illuminating during backtesting, which will give us options to consider if our optimization model of choice does not perform as required during backtesting.

3. <u>Resampling MVO</u>
Resampling MVO is an alternative robust MVO approach. The idea diverges from the typical robust MVO in that instead of incorporating the parameter uncertainty into the optimization model deterministically, we generate portfolio weights for multiple

scenarios of parameter estimations and take the average of all the portfolio weights generated. The approach we intend to take is as follows: [6]

i. Estimate the expected asset returns and co-variance matrix.
ii. Assume a normal distribution for the asset returns, with the parameters being the estimations computed in step i. Generate a sample of ~100 observations using random normal variable generator.
iii. Use the generated observations to generate a new set of expected asset returns and covariance matrix.
iv. Optimize portfolio using the new set of expected asset returns and covariance matrix. Save the weights of the generated portfolio.
v. Repeat steps ii-iv ~100 times and save the weights generated from each iteration. Take the average of all sample portfolios as optimal portfolio.

As brought up earlier, comparison between the BL approach, the typical robust MVO approach and resampling MVO approach during backtesting will provide us valuable data in case we later consider incorporating designing against parameter uncertainty in our optimization models.

4. <u>Risk Parity</u>
Risk parity is a portfolio optimization strategy that forgoes the estimation of the asset returns entirely and seeks to build portfolios with equal risk contribution from each asset (so it joins the BL approach, the typical robust MVO approach and resampling MVO approach as techniques to mitigate the parameter uncertainty drawback of the basic MVO formulation). We can partition the risk of the portfolio into risk contributions from individual assets as follows:

$$\sigma_p(x) = \sum_{i=1}^{n} RC_i(x) = \sum_{i=1}^{n} x_i \frac{\partial \sigma_p}{\partial x_i} = \sum_{i=1}^{n} \frac{x_i (Q\ x)_i}{\sqrt{x^T Q\ x}}.$$

The formulation uses a least squares concept as its base (i.e. in trying to minimize the sum of the squared difference in risk contribution between the assets):

$$\min_{\boldsymbol{x}} \sum_{i=1}^{n} \sum_{j=1}^{n} (x_i(\boldsymbol{Qx})_i - x_j(\boldsymbol{Qx})_j)^2$$

$$s.t.\ \ \boldsymbol{1}^T\boldsymbol{x} = 1$$

$$\boldsymbol{x} \geq 0$$

Note that with this formulation, we have a highly non-linear objective with a gradient that is difficult to compute. So, we have two approaches in re-defining this formulation to obtain a more efficient model to work with.

In the first approach, we add two auxiliary variables, ζ and θ, to improve computational efficiency and remove non-linearity from our objective formulation:

$$
\begin{aligned}
\min_{x,\theta,\zeta} \quad & \zeta \\
\text{s.t.} \quad & \zeta \geq \theta - x_i(Qx)_i, \quad i = 1, ..., n, \\
& \zeta \geq x_i(Qx)_i - \theta, \quad i = 1, ..., n, \\
& 1^T x = 1, \\
& x \geq 0
\end{aligned}
$$

The second approach involves replacing the objective function with a less non-linear function that at optimality evaluates to give the result we desire (i.e. equal risk contribution from each asset). One such formulation is as follows [6]:

$$
\min_{\boldsymbol{y}} \frac{1}{2}\boldsymbol{y}^T\boldsymbol{Q}\boldsymbol{y} - c\sum_{i=1}^{n} ln y_i
$$

$$
s.t. \boldsymbol{y} \geq 0
$$

where c is some positive scalar (note that the solution is independent of this value, so choice of this value does not matter) and the solution, $\boldsymbol{y}^*$**,** contains the un-normalized portfolio weights (so we get the optimal weights as $x_i^* = \frac{y_i^*}{\sum_{i=1}^{n} y_i^*}$).

We plan to implement both approaches and evaluate which performs better during backtesting. Additionally, it must be noted that risk parity guarantees uniqueness of optimality given we have a long-only constraint (so we must not allow short selling when testing this model).

5. <u>Sharpe Ratio Optimization</u>

Since we will be using the Sharpe Ratio as a performance metric for comparing the performance of different portfolios, we will also try to find an optimal portfolio for which the Sharpe Ratio (the ex-ante variant, to be specific) is maximized and compare the performance of this portfolio with the portfolios generated from the other models, especially with the results demonstrated by the optimization models we proposed to use.

The model is formulated as follows:

$$\max_{\boldsymbol{x}} \frac{\boldsymbol{\mu}^T \boldsymbol{x} - R_f}{\sqrt{\boldsymbol{x}^T \boldsymbol{Q} \boldsymbol{x}}}$$
$$s.t.\ \mathbf{1}^T \boldsymbol{x} = 1$$
$$\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{b}$$
$$(\boldsymbol{x} \geq \mathbf{0})$$

We can reformulate the model to ease the computations by making some transformations. The reformulated model is as follows:

$$\min_{\boldsymbol{y},\kappa} \boldsymbol{y}^T \boldsymbol{Q} \boldsymbol{y}$$
$$s.t.\ (\boldsymbol{\mu} - \boldsymbol{R_f})^T \boldsymbol{y} = 1$$
$$\mathbf{1}^T \boldsymbol{y} = \kappa$$
$$\boldsymbol{A}\boldsymbol{y} \leq \boldsymbol{b}.\kappa$$
$$\kappa \geq 0$$
$$(\boldsymbol{y} \geq \mathbf{0})$$

The optimal portfolio weights are then obtained as $\boldsymbol{x}^* = \frac{\boldsymbol{y}^*}{\kappa^*}$.

6. Equally-weighted portfolio
   While not a model we are seriously contemplating using, we will utilize this model to validate our portfolio optimization model choices when backtesting. In constructing this portfolio, we simply assign each asset an equal weight.

Furthermore, it must be noted that these models will use the asset means and covariances/variances estimated using our factor modelling of choice (i.e. the Fama and French 3 factor model), with the exception of step iii of our resampling MVO algorithm.

Preprocessing

As mentioned previously in the parameter estimation section, we can perform some preprocessing steps in the hopes of reducing the computational burden when the application is to perform its base functionalities. In addition to the preprocessing steps mentioned in the parameter estimation section, these preprocessing steps include:

- The computation of the asset correlations for using the Most Diverse MVO approach.
- The reduction of the asset universe using the Most Diverse MVO approach.

Changes + comments

- We plan to use either of Python's CVXPY [7] or CVOXPT [8] packages for executing our optimization models. Both packages provide tools for solving convex optimization problems, which covers the optimization problems we need to solve.
- We added some details, specifically describing our approaches to implementing each optimization model and steps we plan to take in the subsequent future.
- As mentioned in the parameter estimation section, our biggest change since our proposal is expanding/ elaborating on our asset universe. Particularly, our current plan is to add some ETFs that cover a variety of asset universes (such as US Medium Cap, US Small Cap, Emerging Markets to name a few) in addition to our original asset universe of S&P500 stocks and US Treasury bonds. Our preprocessing steps will include using Most Diverse MVO to narrow the universe of S&P500 stocks our optimization model needs to consider.

Portfolio Rebalancing and Back-testing

Portfolio rebalancing is essential to maintain the desired risk-return profile of a portfolio. Rebalancing also has significant transaction cost associated with it. For our portfolio wizard we are going to assume a 0.5% cost per rebalancing. The user will be allowed to specify a specific number of rebalancing times a year, but the default will likely be either once or twice a year. The final number for the default will be determined in testing to see how often portfolios deviate from their calculated risk-return profiles. In either case the cost of rebalancing will have to be added to the desired return in order to properly match the users' desired return level.

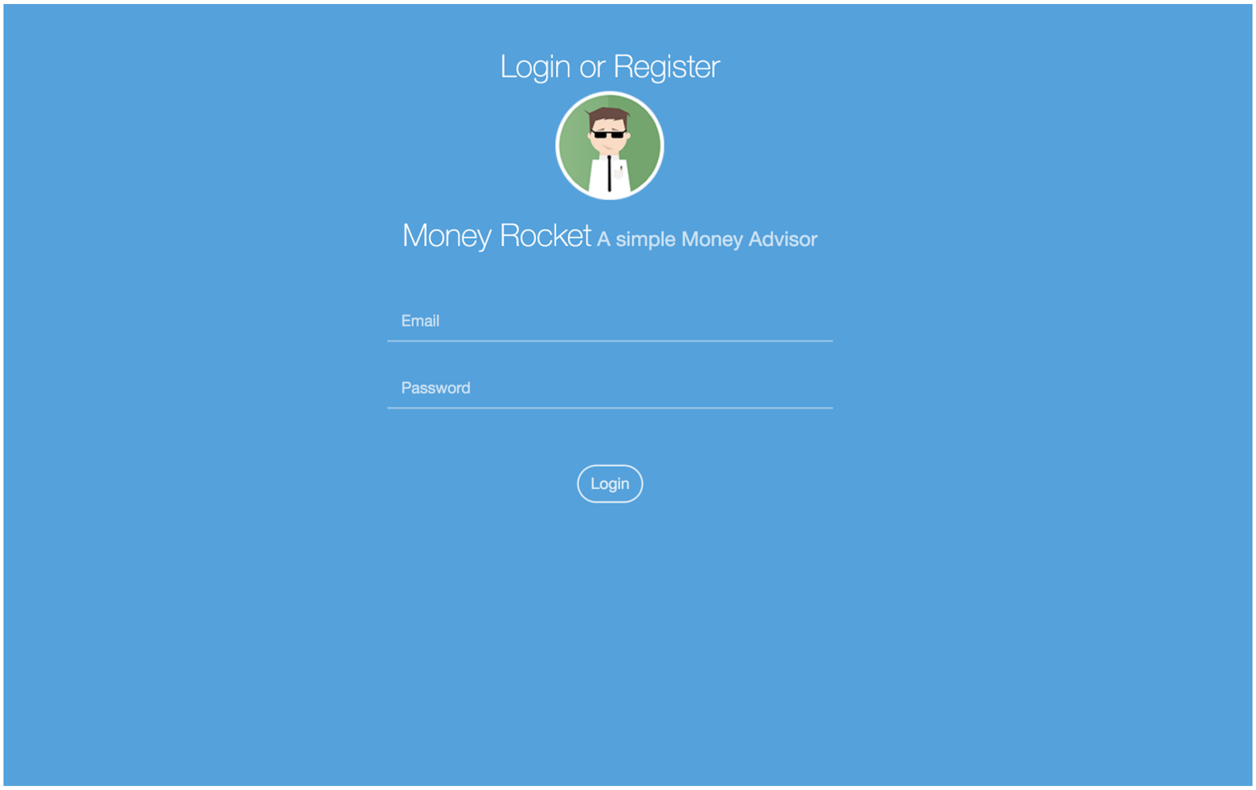
Back-testing a portfolio optimization strategy involves determining how that strategy would have worked in previous time periods. For our strategy, this would involve first recalculating our factor model using some amount of the available historical data. This length is expected to be between 1 and 5 years of history to determine the factors, but the final number will be determined in testing. Once the factors have been calculated, we will run the portfolio generation strategy to determine the optimal portfolio for the specified risk-return profile. The results of this portfolio will then be tracked using the actual stock prices until we reach the next rebalancing portfolio. We will then deduct the rebalancing cost, recalculate the factors, generate a new portfolio and continue tracking the portfolio value for several rebalancing periods. The number of rebalancing periods for which we back-test has not yet been determined, and we plan to determine the number of periods through testing.

Front End

Overview of Front-end

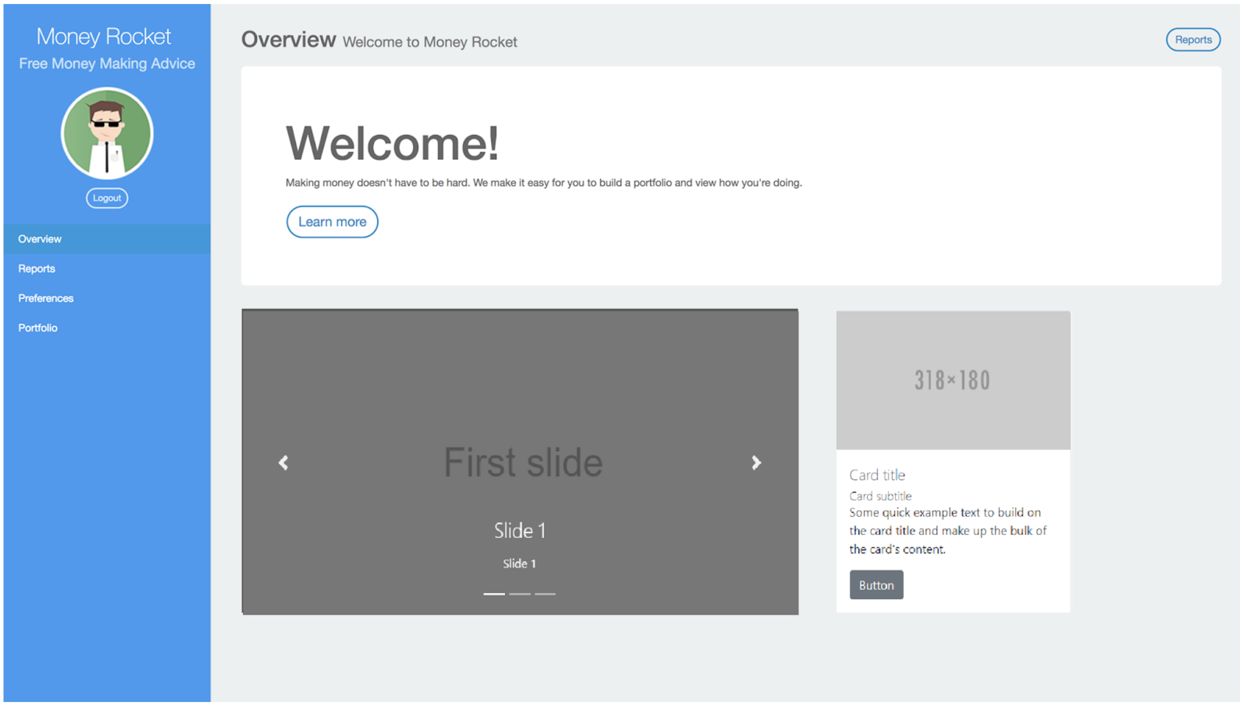
In the previous deliverable, we justified our decisions to make the front-end as user friendly as possible. Now, in order to continue the trend of maximizing usability of our design, we outline the entire website with wireframes and BPMN diagrams. We refer to the BPMN diagram in our wireframe descriptions. Additionally, we develop a testing protocol for the front-end so that we can accurately test how usable our interface is.

Finalised WireFrames with descriptions



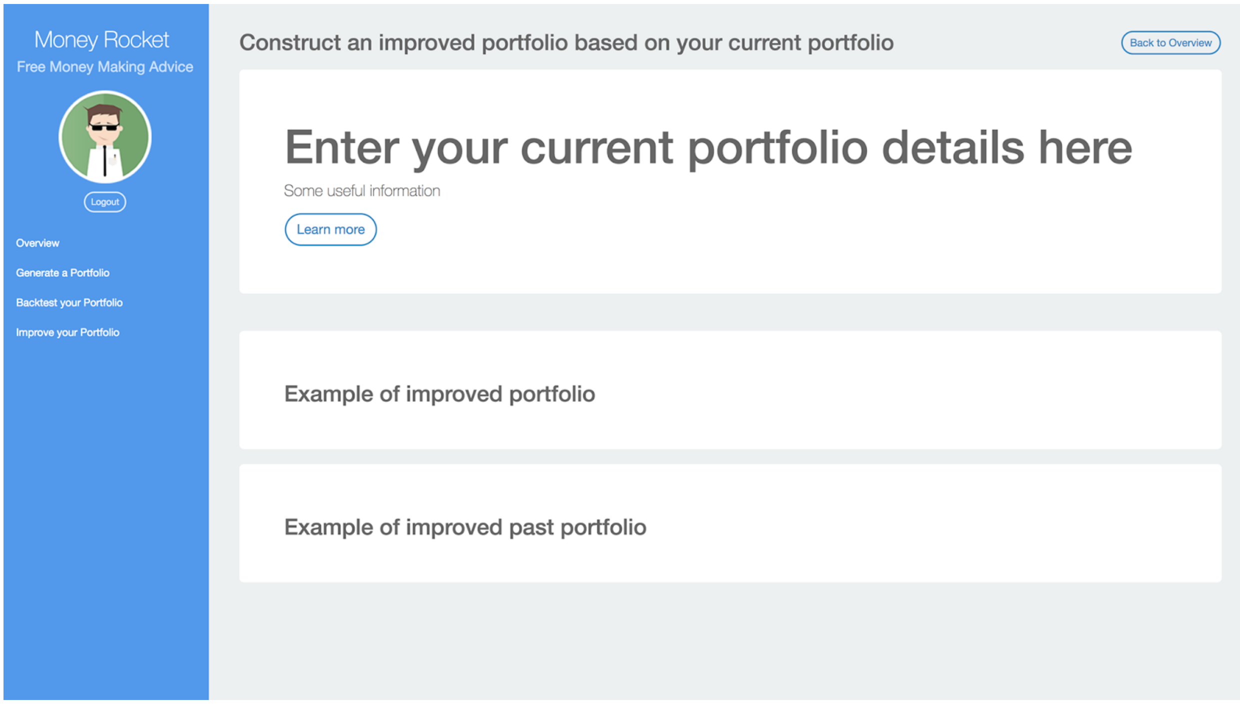
Login page:

This page is where the user enters when they first land on the page, if they have not previously registered for the site or logged in. The site attempts to access cookies to log the user in, but if the user has no cookies for logging on, then it redirects to this page. On this page there is a simple login interface where the user can register or log in with a simple Username Password credentials system. Upon submission of an email/password, the credentials are sent to the database to be validated, and if found the user is logged in.



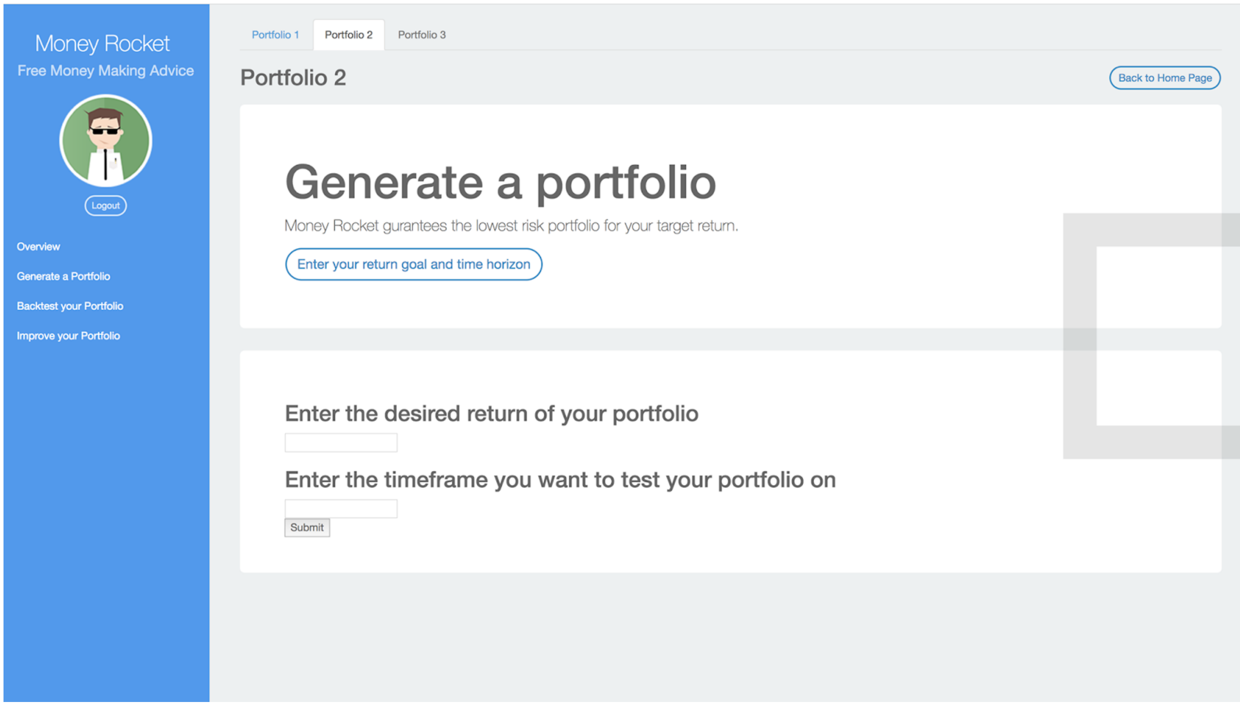
Welcome Page:

This page is where the user lands upon loading the website if they are logged in, or after logging in/registering. This page greets the user with some initial information, and contains the main layout of the website, a side bar with links to all the different functionalities of the app. On the main area of the landing page, there's some useful help information for first time users, and there are some infographics/news articles on the bottom to help the user become acquainted with both finance as well as the usage of our site. There will also be information to teach the user about how the actual models we chose work.



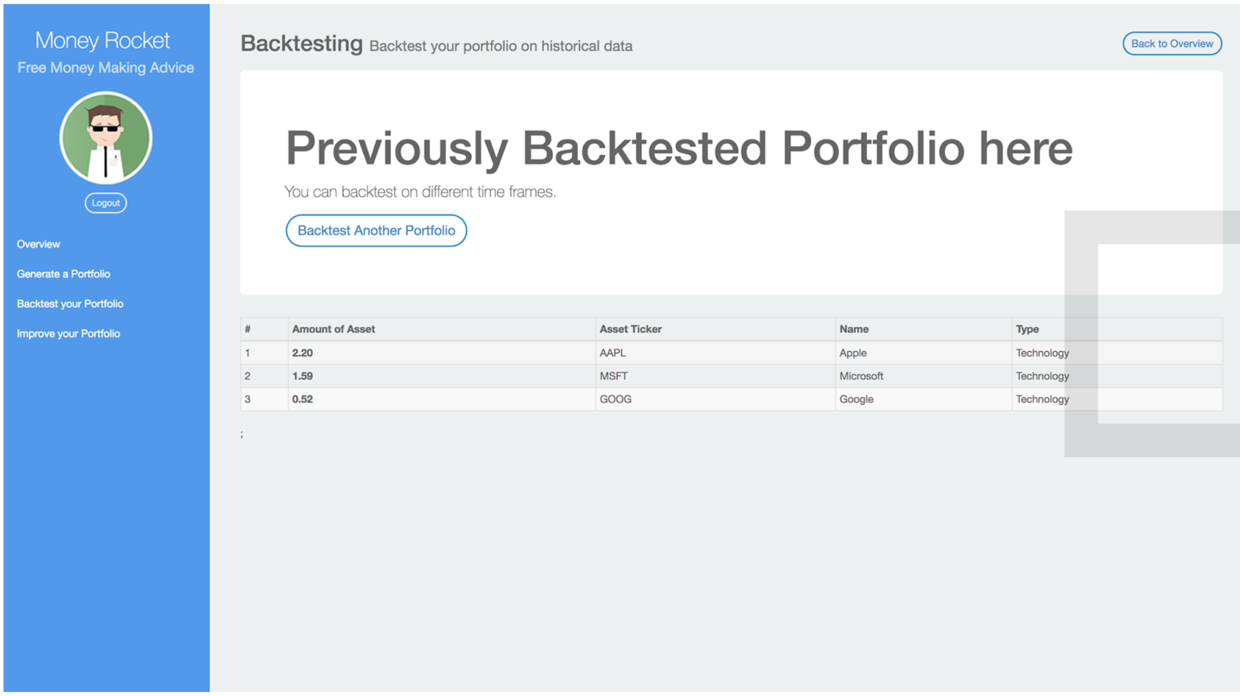
Improve your portfolio page:

According to the functionality requirements, "given a portfolio to be held for some time horizon, the investment wizard will seek to find a portfolio of assets that dominates it". Here, we provide a convenient page to connect the business logic of improving a portfolio. This gives the user the option to enter their current portfolio details, as well as the time frame which is entered in. After processing the portfolio and sending details to the business logic/accessing the database, the website returns a new portfolio which dominates the one we just entered over that time frame. Here, this can be displayed in the form of many different visuals and text, so that the user can easily comprehend what the improvement in the new portfolio is.



Generate a new portfolio:

This page allows you to enter a desired return and a timeframe to test on and will spit out a portfolio of assets which has your desired return on that timeframe. It will accept these inputs in a form with a nice visual format, and will run the business logic on the backend. After the business logic is done, the portfolio details will be spit out and the page will change to display some visuals/attributes of the new portfolio, so that it is easy to comprehend what the generated portfolio is, what the return is, and what assets are in the generated portfolio.



Back testing:

This page allows the user to enter a portfolio of assets and choose a time horizon on which to back test it on. It will then run that portfolio against historical data on the business logic side and return the returns and other statistics about the portfolio. It will display the portfolio details and results in a visual and convenient way and will allow the user to retest/change up time windows relatively easily so that back testing can be done multiple times on multiple portfolios.

Front End Usability testing protocol

In order to properly determine how usable our website is, we develop a simple usability testing protocol to test at each iteration of the protocol the website is.

The protocol contains the following attributes:

- Objectives of Usability Testing Protocol
- General testing methodology

General Testing Methodology

Based on the objectives outlined later in this document, we prepare a testing methodology which allows us to go through and measure our success in each of the many usability metrics.

The plan is to have sample of approximately 10 random participants (University of Toronto Engineering students) and brief them as testers for our user interface. They will be each presented with a laptop, a working internet connection, and the link to the website. Half of the participants will receive a set of basic instructions to navigate through the site, so that they can investigate all different features:

- Load up the website
- Click on log in in order to test the log in functionality.
- Investigate the homepage and help page to learn about the web site.
- Navigate to each of the 3 sidebars, and test out each of the functionalities
- Attempt to backtest a simple portfolio(given)
- Attempt to generate an improved portfolio given a simple portfolio(given)
- Attempt to create a portfolio with a simple set of parameters for variance.

The other half of students will not receive any instructions about the website, and be prompted to simply explore the website to their liking and not be given any background of how the website functions.

During this process of following the instructions, the testers will be prompted to give feedback and rate functionality of the websites as given in the above objectives. If at any stage they struggle with any of the instructions, this will be noted and added as a negative towards the usability of that specific aspect of the product.

After the feedback is collected for each person, scores will be averaged amongst all participants of the usability survey, and the aspects of the project that are bad will be identified according to the next section of this document. Depending on the group that struggles with navigating the website, different changes will be applied to different parts of the interface.

Objectives of Usability Testing Protocol

During the testing, the following objectives are to be measured and quantified in order to see the overall success of our UI.

- *Identify if users can complete specific tasks successfully*

Logging in: The user must easily be able to access the login page, and the login page must load in case the user is a first-time user to the site, or if cookies have been cleared. The user must easily be able to register their account with unique information in the registration page without difficulty. Success in logging in is measured if the user can easily find the log in/register buttons on the web page.

Orienting themselves amongst the webpage: Upon loading in, the user must have an intuitive understanding of how to navigate the web page. For example, the user must be able to navigate to the help page if needed. The user must be able to navigate to each of the specific functionalities as given in the instructions, and not be confused as to how to get from page to page at any time.

- *Pinpoint changes to the design that might need to be made to address any shortcomings to improve performance.*

For each of the tasks and other objectives that we've identified, anything that scores lower than a 7 on the scale from 1-10 needs to undergo some improvement. It is expected that the people testing the website leave some feedback on the positives and negatives of the interface, and as such we as designers will be able to identify all the changes to the design. If the user doesn't leave feedback but rates one of the objectives or the product poorly,

- *Determine if the product is effective*

Based on the primary objectives of the interface as outlined in a previous section, does that product allow the user to accomplish all the objectives needed, without lots of hassle. Does the product load everything that is expected with bugs?

- *Determine if users enjoy using the product*

One of the main objectives in terms of user experience is an overall positive rating from the users. That is, when the users go through a cycle of using each of the features of the web applet, do they have a net positive impression of the UI, and were there enough positive features that outweigh the negative for this specific user. What things could the user do to make the product more enjoyable, both in the product and the user interface.

In order to quantify the objectives, we score each of the common usability metrics based on the following criterion, with a subjective scale from 1-10, 1 being the worst and 10 being the best. In our actual testing, we present every testing candidate with each of these metrics and ask them to score our website on the scale.

Quantifiable User Metrics to be scored

Visibility of system status

The system should always keep users informed about what is going on, through appropriate feedback within reasonable time. We will have a page that displays an error message if the site is down, and with a reasonable simulation of the real world, if a portfolio is performing poorly, notify the user. If the back-end is broken, the front-end must communicate it somehow with an insightful error message.

Match between system and the real world

The system should speak the users' language, with words, phrases and concepts familiar to the user, rather than system-oriented terms. We employ informal language with a conversational tone to be as close to speaking with a real human as possible. Essentially, we explain everything in rudimentary terms so that a person with the least amount of knowledge can understand everything, including financial terms.

User control and freedom

Users often choose system functions by mistake and will need a clearly marked "emergency exit" to leave the unwanted state without having to go through an extended dialogue. The site will have buttons to return to the home page, log out, view help, and exit to previous page on every page.

Consistency and standards

Users should not have to wonder whether different words, situations, or actions mean the same thing. The site will follow the same style theme as well as layout, as we follow a single page application format with a common navbar on the left side of the site.

Error prevention

We want to make a careful design which prevents errors from occurring in the first place. The site will not have errors due to the front end due to a carefully constructed React Hierarchy. The backend will interact with the front-end so that there should be no errors related to data APIs or similar.

Recognition rather than recall

We want to minimize the user's memory load by making objects, actions, and options visible. The site will have easy access to all pages, and a help menu which tells the user about previous actions on the site, and the ability to see previous preferences and settings.

Flexibility and efficiency of use

We will have a first time use onboarding page, as well as a permanent landing page in order to make it easy to navigate as well as start using the product. There will be a help menu in case the user is confused.

Aesthetic and minimalist design

The design will not incorporate extra elements which are not pleasing, or text which is unnecessary. Therefore, minimal elements and text are displayed on the website, rather visual components and insights are prioritized, and text elements are used only to convey very important points.

Help users recognize, diagnose, and recover from errors

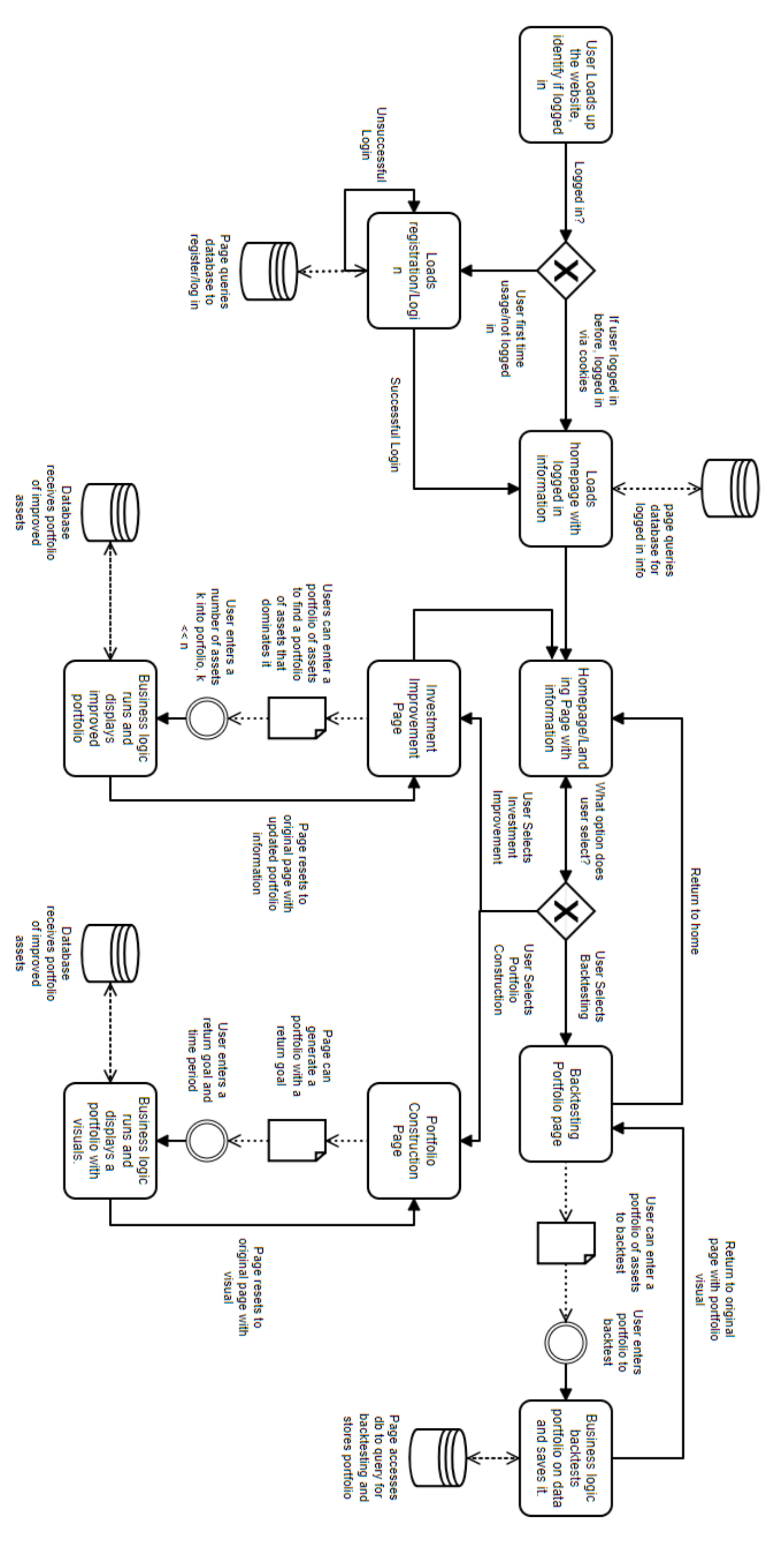
Error messages/warnings will again be in conversational language and easy to understand. They will also be insightful and tell the user what is going on.

Help and documentation

A help page and information about the site will be provided to the user, with frequently asked questions and an option to ask the creators about what to do. The help and user page must be easy to access and find.

All of these objectives/goals will be quantified and discussed after testing in order to create the best possible user experience. If it is found that many of these objectives are not being reached by the site, the user experience will be revamped, and a new website will be developed and tested along the same guidelines.

BPMN DIAGRAM



The BPMN diagram describes the overall flow through the business logic and website. Choices between different loading screens that are decided in the code are denoted by a diamond X, and web pages are denoted by a rectangular icon. Arrows indicate the workflow of the website when things are clicked/typed, and the cylinder indicates a database access/query. A page document icon indicates business logic being run, and a circle indicates a user decision specific to business logic. Overall, the website flows from a login page to a main landing page, and from the main landing page, the user has 3 different choices of pages to click on, which allow for three different functionalities:

- Backtesting
- Generating a new portfolio
- Improving an old portfolio

For each of these pages, the user has some option to put in their own inputs, as well as return to the home page. If they choose to access the functionalities of these pages, the business logic is run on the back end via Python, and the page gets reupdated. Otherwise, it just returns to its previous state.

Back End

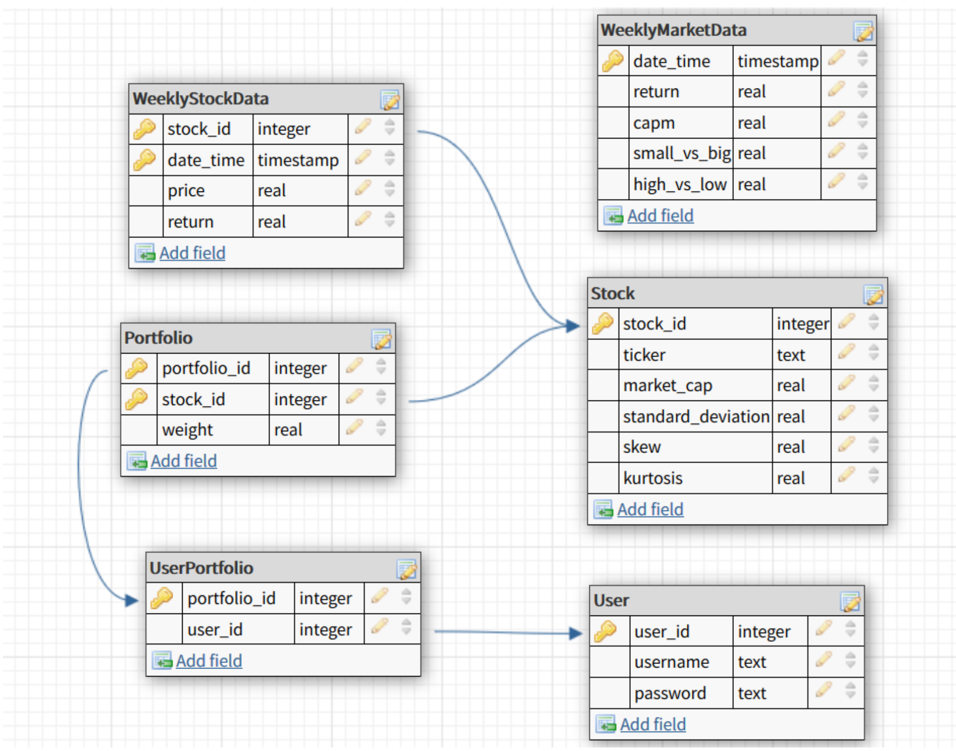
SQLite Schema Script (*schema.sql*):

```
DROP table IF EXISTS User;

DROP table IF EXISTS Stock;

DROP table IF EXISTS WeeklyStockData;

DROP table IF EXISTS WeeklyMarketData;

DROP table IF EXISTS UserPortfolio;

DROP table IF EXISTS Portfolio;


CREATE TABLE User (

  user_id INTEGER PRIMARY KEY AUTOINCREMENT,

  username TEXT UNIQUE NOT NULL,

  password TEXT NOT NULL

);


CREATE TABLE Stock (

  stock_id INTEGER PRIMARY KEY AUTOINCREMENT,

  ticker TEXT NOT NULL,

  market_cap REAL NOT NULL,
```

```
  standard_deviation REAL NOT NULL,
  skew REAL NOT NULL,
  kurtosis REAL NOT NULL
);

CREATE TABLE WeeklyStockData (
  stock_id INTEGER NOT NULL,
  date_time TIMESTAMP NOT NULL,
  price INTEGER NOT NULL,
  return REAL NOT NULL,
  PRIMARY KEY(stock_id, date_time),
  FOREIGN KEY(stock_id) REFERENCES Stock(stock_id)
);

CREATE TABLE WeeklyMarketData (
  date_time TIMESTAMP PRIMARY KEY,
  return REAL NOT NULL,
  capm REAL NOT NULL,
  small_vs_big REAL NOT NULL,
  high_vs_low REAL NOT NULL
);

CREATE TABLE UserPortfolio (
  portfolio_id INTEGER PRIMARY KEY AUTOINCREMENT,
  user_id INTEGER NOT NULL,
  FOREIGN KEY(user_id) REFERENCES User(user_id)
);

CREATE TABLE Portfolio (
```

```
 portfolio_id INTEGER NOT NULL,
 stock_id INTEGER NOT NULL,
 amount REAL NOT NULL,
 PRIMARY KEY(portfolio_id, stock_id),
 FOREIGN KEY(portfolio_id) REFERENCES UserPortfolio(portfolio_id),
 FOREIGN KEY(stock_id) REFERENCES Stock(stock_id)
);
```

Schema Visualization:



Data Utilization:

Schema	Description
User	This table contains login information so that users can create a profile and save portfolios they generate.
Stock	This table contains stock information: its market ticker, daily price and historical statistical standard deviation, skew and kurtosis. The statistical information, along with the expected return, estimated with the Fama French Three Factor model, is used for portfolio optimization.
WeeklyStockData	This historical data is used to determine the historical statistical information for stocks and may be periodically updated so indicators are up to date. It is not referenced while generating the portfolios.

WeeklyMarketData	This historical data is used to determine the Fama French Three Factor modeled returns for stocks and may be periodically updated so indicators are up to date. It is not referenced while generating portfolios.
UserPortfolio	This table describes which portfolio belongs to which user.
Portfolio	This table contains the portfolios, including which stocks it contains and how much. The information, along with stock data, can be used to determine the estimated return and variance of the portfolio.

Updated Timeline

The following outlines what has been **completed in green, delayed in red**, and what is left to be done:

Task	Deadline (2018)
Have business logic decision making completed (including universe of assets, models, parameter estimation etc.) (i.e. Team Meeting # 1).	Oct. 15
Universe of assets, required data and data source, necessary parameters and their estimation methods confirmed.	Oct. 11
Optimization model and necessary optimizer to go with confirmed. Re-confirm parameters necessary for model-choice. Results and how results are displayed chosen	Oct. 14
Have details for front-end, back-end and business logic worked out (if possible, should have coding started by this time) (i.e. Deliverable 3).	Nov. 5
Extract data of universe of assets that will be used, as well as the necessary data for parameter estimation. (Extracting additional ETF data)	Oct. 28
Document any data-pre-processing that is necessary as well as the data storage/ database schema requirements.	Oct. 21
Document details of optimization model to be used. This includes the form of the model, the parameters that will be used, the optimizer that will be used etc.	Oct. 24
Document the data that need to be retrieved for each step from the back-end and when.	Oct. 26
Document any necessary re-balancing/ re-optimization steps	Oct. 28
Document what the application needs to output and the form of display (i.e. how the output should be displayed).	Oct. 30
Confirm if the objectives are met with regards to output. Confirm if base objectives are satisfied and/or if more features/ functions are to be included.	Nov. 1
Document tests that will be used to validate output and accomplishment of objectives.	Nov. 4
Business logic and the integration of front-end and back-end details must be ready and firmly in place (i.e. Team meeting #2).	Nov. 19
Perform the necessary data extraction, preprocessing and storage.	Nov. 8

(Possibly) use different modelling and estimation techniques on extracted data and compare the validity and efficacy of each technique.	Nov. 11
Test out different sets of constraints and assumptions and fine-tune model (for example, whether to allow short-selling, whether to and/or how to restrict the number of assets model uses etc.).	Nov. 14
Ensure model produces the necessary output.	Nov. 16
Iron out how each sub-component interacts with each other (which data is stored/ retrieved for each step etc.).	Nov. 18
Application must be complete (or near complete, with only minor debugging/ troubleshooting remaining) (i.e. Team Meeting # 3).	Nov. 26
Start integrating the sub-components.	Nov. 22
Perform testing and debugging and trouble-shooting as necessary.	Nov. 25
Draft of final report due (i.e. Deliverable # 4).	Nov. 30
Document all the details, techniques used and steps involved for the application.	Nov. 27
Have diagram, visual aids and tables for description of results and application ready.	Nov. 29
Presentation day (i.e. Deliverable # 5).	Dec. 3
Work out member roles for the presentation (i.e. what segment each member will cover).	Dec. 1
Rehearsal of presentation ideally performed.	Dec. 2
Final report due (i.e. Deliverable # 6).	Dec. 5
Work out what needs to be added to and polished for transforming the draft report into the final report.	Dec. 4

Bibliography

[1] K. R. French, "Current Research Returns," Dartmouth College, 4 November 2018. [Online]. Available: mba.tuck.dartmouth.edu/pages/faculty/ken.french/data_library.html. [Accessed 4 November 2018].

[2] SciPy, "numpy.linalg.solve," SciPy.org, 4 November 2018. [Online]. Available: https://docs.scipy.org/doc/numpy-1.15.0/reference/generated/numpy.linalg.solve.html. [Accessed 4 November 2018].

[3] SciPy, "Statistics," SciPy.org, 4 November 2018. [Online]. Available: https://docs.scipy.org/doc/numpy-1.14.1/reference/routines.statistics.html. [Accessed 4 November 2018].

[4] T. Idzorek, A Step-By-Step Guide to the Black Litterman Model, Chicago: Ibbotson Associates, 2005.

[5] D. Luenberger, Investment Science, Toronto: Oxford University Press, 2014.

[6] G. Costa, "MIE377 Winter 2018 Lecture Notes," [Online].

[7] S. Diamond, E. Chu, A. Agrawal and S. Boyd, "Welcome to CVXPY 1.0," CVXPY, 4 November 2018. [Online]. Available: www.cvxpy.org. [Accessed 4 November 2018].

[8] M. S. Andersen, J. Dahl and L. Vandenberghe, "CVXOPT," CVXOPT, 4 November 2018. [Online]. Available: https://cvxopt.org. [Accessed 4 November 2018].