Closing the World's Investment Gap

Final Design Report

Design Team
Ameer Shaikh
Amr Mahmoud
Daniel Kecman
Stefan Momic

Contents

Problem Statement

Money matters – love it or hate it, money has become the cornerstone to survival and society. Despite all the claims that "money doesn't buy happiness" money is a critical component allowing one to pursue the life they want. Money provides us with the ability to satisfy our basic survival needs (i.e. food, shelter, clothing, etc.), creates opportunities for choices/flexibility, allows you to live your dreams and/or fund the causes you care about. By this point, we believe that the importance of money does not need to be stressed any further as we can appreciate that everyone had some concerns about money at one point or another. Rather, we are here to help everyday people make more money, so they can live an even better life.

Fundamentally, there are only two ways to make money – by working and/or by having your assets work for you. We all know the importance of working with most people having jobs to satisfy their monetary needs, but having your assets work for you is a concept that is often overlooked. We are here to change that. Alpha Factory aims to reduce the stigma around investing and make it accessible and understandable for everyone. Ultimately, Alpha Factory aims to *close the world's investment gap* by making investing activities part of everyday life!

Simply put, investing is the act of committing money or other assets to an endeavor with the expectation of obtaining an additional income or profit in the future. [1] Investing allows one to further enhance their wealth as it provides the opportunity to earn a return on investment (i.e. additional money in the future). Conversely, if you don't invest you are missing out on the opportunity for financial growth and/or asset appreciation.

There are many reasons to invest with as many different investment goals as there are people. Common reasons people invest money include, but are not limited to the following:

1. Growing their money – the return on investment allows money to build, creating additional wealth over time
2. Saving for retirement – saving money for retirement, and better yet, investing retirement savings allows people to live off these funds after they stop working
3. Earning higher returns – if you're striving to grow your wealth, you need to put it in a place that offers a higher rate of return than the interest rates provided by savings accounts. With many investment vehicles (more on these later) offering opportunities to earn higher rates of return, investing money provides the opportunity to achieve these goals
4. Reaching various financial goals – whether these goals involve buying a home/car, putting your children through college, paying off loans and/or starting a business, having an additional source of income by investing money helps achieve these goals

As you can see, there are many different investment goals with as many, if not more, investment strategies to achieve them. We understand that this can be overwhelming for the average person, but that does not mean that investing should only be available to the few people who understand it. Everyone needs money, and everyone has some financial goals – Alpha Factory is here to help everyday people invest in their future and make their money work for them!!

Project Scope

With the plethora of different investment goals and strategies present, Alpha Factory simply cannot generate portfolios to accommodate them all. As such, we have decided to focus in on a range of investors to optimally satisfy their needs.

With a range of different options available for investors Alpha Factory strived to target the average member of the population (i.e. those with limited knowledge about investing/finance and/or skepticism of investing). With a range of options available for more informed members of the population, including investing independently and mutual funds to name a few, we believed that the average population was greatly underserved and provided the greatest opportunity for us to make an impact that matters.

Specifically, Alpha Factory aims to use our platform and proprietary portfolio generation techniques to help the average person with limited knowledge of the financial markets and/or limited time to manage their own asset portfolios/investments. Specifically, we aim to get to know them through our quick questionnaires, so we can create a personalized and diversified portfolio of assets at a risk tolerance they are comfortable with while meeting their long-term investment goals. From there, we manage their money for them – this includes automatically rebalancing their portfolio to account for any market fluctuations and reinvesting their investment proceeds (e.g. dividends) to ensure that their money doesn't take any breaks while working for them. We understand that investment goals change overtime and we offer dynamic portfolio modifications to address users' evolving preferences and inputs.

Survey of Existing Solutions

Since the invention of the Internet people have greatly changed the way they work, learn and interact with each other – it is time we change the way we invest! In fact, this change is already well underway with several other robo-advisors already out there. Alpha Factory has learned a great deal from our competitors that have been first to market, and we aim to use these lessons learned to not only outperform them but outperform the market as well.

In surveying the existing solutions, we decided to limit our scope to a select number of robo-advisors based in Canada and the United States. Specifically, we decided to further explore Nest Wealth, BMO Smartfolio, Wealthsimple, Questrade Portfolio IQ and WealthBar as they were ranked the Best Canadian Robo-Advisors in 2018. [2] Moving across the border, we decided to explore Betterment, Personal Capital, Schwab Intelligent Portfolios, SigFig and Wealthfront as they were ranked the Best Robo-Advisors in 2018 by Investopedia. [3]

With changing consumer preferences, and an increasingly technological world, robo-advisors have surged in popularity. Robo-advisors ultimately allow everyday people to set up a customized, diversified portfolio and provide access to a series of wealth management services that previously only seemed to be available for the ultra-wealthy and financially literate. [4] With a growing selection of robo-advisors with new firms seemingly entering the market on a daily basis and veteran robo-advisors (like those already mentioned) expanding their offerings at a rapid pace, it has become increasingly difficult to pick the best robo-advisor.

In reality, with varying investment goals and preferences, the best robo-advisor is a subjective matter highly dependent on the respective financial goals of each individual investor. However, after a comprehensive survey of the top 5 rated robo-advisors in both respective countries, it was noticed that the top-rated robo-advisors shared the following common features: [5]

- Low initial investment – namely the ability to generate robust and diversified portfolios without requiring massive capital involvement (in contrast, attaining a diversified portfolio of assets traditionally required a great amount of initial capital)
- Low fees – ultimately allowing the investors to retain most of the money their money earns
- Popular investment options – namely having a smaller universe of assets allowing the investors to be familiar with and understand the assets they invested their money in
- Comprehensive portfolio management features – especially those that allow the investors to understand exactly how their portfolio is doing at any given moment in time

While creating Alpha Factory we were conscious about all four of these features and made all our design decisions to ensure that our users obtain all the features that made these robo-advisors a success. More than this, Alpha Factory aimed to achieve better returns than the aforementioned robo-advisors with careful consideration of their asset universe, parameter estimation and portfolio generation techniques.

A deeper dive into each of the previously mentioned competitors is provided in Appendix A.

Project Framework

To aid with making the appropriate design decisions, Alpha Factory conducted a deep exploration of their project framework. Being conscious of their stakeholders' desires, objectives and constraints enabled them to ensure they created the best robo-advisory possible for their particular target audience.

Stakeholder Analysis

First, Alpha Factory conducted a deep analysis of their stakeholders. Doing so, enabled Alpha Factory to better define their project and ensure the support of their stakeholders along the way – this ultimately allowed Alpha Factory to deliver a higher quality final product. In addition, conducting the stakeholder analysis allowed us to understand the stakeholders better, thereby ensuring that the final product was well received by all relevant groups.

The stakeholder analysis conducted is comprised of 3 major steps, each of which is further described in the subsections below.

Step 1: Identify all Stakeholders

As a first step, Alpha Factory explored and identified all the parties who are affected by their work, who have influence or power over it, or have an interest in its successful conclusion. The stakeholders, along with the categories they belong in are presented in Table 1 below.

Stakeholders Affected	Stakeholders with Influence	Stakeholders with Interest
Potential new investors / customers	Supervisors – Prof. Roy Kwon and Hassan Anis	Potential new investors / customers
Existing robo-advisors (i.e. competitors)	Design Team – Ameer Shaikh, Amr Mahmoud, Daniel Kecman, Stefan Momic	Existing robo-advisors (i.e. competitors)
Existing robo-advisor users	Canadian Engineering Accreditation Board	Existing robo-advisor users
Traditional Investment Firms (e.g. Financial Advisors, Mutual Funds, etc.)		Traditional Investment Firms (e.g. Financial Advisors, Mutual Funds, etc.)
Traditional investors		Traditional investors

Table 1: List of key stakeholders

Step 2: Prioritize Stakeholders

To ensure successful adoption of Alpha Factory we decided to prioritize on key stakeholders – that is, those we believe had significant influence and/or will be the end users of our product. Ultimately, determination of the key stakeholders framed the objectives and constraints of the design which in turn had a great influence on the design decisions made. In prioritizing the key stakeholders, we divided them in three groups – those we need to keep satisfied (i.e. the most important), those we need to keep informed (i.e. medium importance) and those we should monitor and consider but requires minimum effort. The prioritization of the key stakeholders is provided in Table 2 below.

Keep Satisfied	Keep Informed	Monitor
Potential new investors / customers	Canadian Engineering Accreditation Board (CEAB)	Existing robo-advisors (i.e. competitors)
Existing robo-advisor users		Traditional Investment Firms (e.g. Financial Advisors, Mutual Funds, etc.)
Supervisors – Prof. Roy Kwon and Hassan Anis		Traditional investors
Design Team – Ameer Shaikh, Amr Mahmoud, Daniel Kecman, Stefan Momic		

Table 2: Prioritization of key stakeholders

As can be seen from Table 2 above, the most important stakeholders are future users which includes any new investors and existing robo-advisor users (who are more likely to switch to a better robo-advisor).

This is closely followed by the Supervisors and Design Team who have significant influence over the project.

To ensure the Capstone Project is legitimate, the Canadian Engineering Accreditation Board needs to be informed and the Design Team must ensure that all CEAB requirements are met.

Finally, the Design Team is wise to monitor their competitors and base select design decisions in a way to achieve their desired competitive advantages. Additionally, Alpha Factory should monitor Traditional Investment Firms to see how they react to the threats from new competitors and their customers/clients with the hope of luring them over eventually. It is worth noting that traditional investors are less of a focus than existing robo-advisor users as they are less likely to switch to robo-advisors than investors who already use robo-advisors.

Step 3: Understand your Stakeholders

Lastly, to help frame the desired product functions, constraints and objectives, Alpha Factory surveyed their key stakeholders to understand their goals and objectives. Each stakeholder along with their unique goals are provided in Table 3 below.

Key Stakeholder	Goals and Objectives
Potential new investors / customers	The main goal of new investors includes: • Easy adoption – make investing easily understandable even with minimal financial literacy • Low initial investment, low fees, popular investment options and comprehensive portfolio features • Better than average returns making investing in Alpha Factory worth their while
Existing robo-advisor users	The implicit goals of existing robo-advisor users relate to the ability to get a better product/platform than the one they currently use. This includes: • Achieving better performance/higher returns than their current robo-advisor • Obtain a friendlier user interface • Attain a platform with a lower minimum balance, lower fees, more popular investment options and/or more comprehensive portfolio features
Supervisors	Ensure that the tool developed integrates both the mathematical, statistical and financial modelling techniques learned throughout the Engineering Mathematics, Statistics and Finance (EMSF) major along with the relevant computing technologies.

Key Stakeholder	Goals and Objectives
Design Team	The Design Team aims to: • Reinforce the mathematical, statistical and financial modelling techniques learned by creating a practical design product • Create a great user-friendly product that can be adopted and sold to potential consumers • Achieve the best portfolio performance/returns among the 2018-19 EMSF graduating class as evidenced by their portfolio validation and back-testing • Learn and implement additional financial modelling techniques that have not been covered as part of the EMSF major, especially those that would aid in increasing portfolio performance
CEAB	Ensure key Engineering Accreditation requirements are met. Namely: • Ensure design integrates mathematics, basic sciences, engineering sciences and complementary studies to develop a product that meets specific needs • Ensure the project is creative and governed by the discipline standards

Table 3: Stakeholder goals

Functions

To meet the goals of all our stakeholders Alpha Factory decided to have several key functions. First, we wanted to ensure that we had an easy user-friendly interface that offered support and explained the essential investment strategies we used in an easy to understand way. Second, we wanted to ensure that clients/customers were able to effortlessly provide inputs and feedback at any time – we understood that financial goals change overtime and we wanted to ensure that users can repeat our questionnaires and/or change their views accordingly. Ultimately, this ensured that their portfolio is always right for their specific needs. Finally, and most importantly, we wanted to use all the user-inputs and responses to our questionnaires to create customized portfolios for our users that allowed them to meet their investment goals using a risk-tolerance they are comfortable with. Specifically, given the return goals of our clients, over a specific time horizon, Alpha Factory generates a portfolio that has a projected return greater than our equal to that amount while minimizing the risk and, most importantly, keeping it within our customers' risk-tolerance levels.

Beyond simply creating the portfolio, we wanted to provide our clients with the analysis of the risk and return profile of their portfolio in real time and in an easy to understand, user-friendly way over various time horizons by back-testing over real data. Specifically, we provide portfolio analysis by computing and displaying various portfolio performance metrics such as alpha and Sharpe Ratio and provide a visual of how their portfolio has performed over a selected time period.

Finally, as a way to promote our platform we provide visitors the ability to input a portfolio of assets they would hold if they had to invest alone and/or the portfolio they currently hold. After doing so we provide them their portfolio performance metrics along with a portfolio that dominates the one they provided with the hope of encouraging the user to switch to our platform as a way of realizing better performance.

Objectives and Constraints

Having identified goals for each of the key stakeholders, the objectives and constraints were easily identified. We have provided further discussion on each objective and constraint below.

Objectives

1. Achieve the best portfolio performance – both the Design Team and potential Alpha Factory users greatly benefit for achieving the highest returns possible. This includes having the optimal asset universe, portfolio generation strategies and parameter estimation techniques.
2. Achieve the most user-friendly interface and design – with first impressions mattering a lot, especially in the digital world, a good user-friendly design goes a long way in how the final product is received among stakeholders. Simply put, good design and everyone wins!

Constraints

Along with the objectives presented above, the Design Team was faced with the following constraints when selecting and making Design Decisions.

1. User Preference: low initial investment – the Design Team must select the asset universe, portfolio generation strategies and parameter estimation to ensure that clients/customers can obtain a diversified portfolio without requiring a large initial investment
2. User Preference: low fees – conscious with user preferences for lower fees the Design Team must consider strategies resulting in lower management time and/or fees (e.g. less rebalancing, outsourcing certain costs, etc.)
3. User Preference: popular options/easy to understand assets – with users preferring to understand/know the assets they are investing in; Alpha Factory was constrained in selecting the asset universe. Specifically, a large asset universe with complex securities would be too overwhelming for our intended users.
4. User Preference: comprehensive portfolio features – Alpha Factory was constrained with the need to consider user inputs and create customized portfolios for each users' unique financial situation and goals. Furthermore, Alpha Factory needed to display the performance of the portfolios using graphs and metrics that are easy to understand by the average member of the population in real time.

5. Time – with less than 3 months allowed to complete the project the Design Team experienced significant time constraints, limiting exploration/implementation of more complex financial models

Project Demonstration

The official launch, demonstration and presentation of Alpha Factory occurred on December 3rd, 2018 at the Myhal Centre for Engineering Innovation & Entrepreneurship at the University of Toronto.

Prior to dividing into the key design details and functionality of our web application, we encourage you to visit our platform at http://alphafactory.herokuapp.com/about . Doing so will help contextualize the ensuing discussion below. Please feel free to rigorously test all the functionalities of the web application. Note that all the website functionalities can be accessed free of charge (i.e. please do not hesitate to create an account to test the entire functionality of our web application).

We are looking forward to hearing your thoughts and feedback.

Alternatives and Key Design Decisions

While making Alpha Factory several alternatives were explored for each part of our design with various design decisions made to best fit our objectives and constraints. The design decisions were segmented into three major categories – Business Logic, Front-End design and Back-End design. A deeper dive into each set of key design decisions is explored in the sections below.

Business Logic

Business Logic design decisions constitute those regarding the models and methodologies used for portfolio generation. Key design decisions made include the selection of the asset universe, selecting the source(s) of financial data, portfolio generation methodologies, parameter estimation techniques, considerations for robustness, considerations for investor preferences and risk-tolerance and the portfolio validation and analytics provided. Each key design decision is further explored in the subsections below.

Asset Universe Selection

Alpha Factory used a top-down approach for selecting the asset universe. Namely, we considered investing in individual securities (e.g. individual stocks, bonds, futures, options, etc.) or aggregate asset classes (e.g. exchange traded funds (ETFs), mutual funds and/or other grouped securities).

After careful consideration of the pros and cons of each approach (refer to Table 4 below), along with an in-depth survey of existing solutions we determined that an asset universe consisting of aggregate asset classes was the best alternative.

Option	Pros	Cons
Individual Securities	Provides ultimate flexibility in allocation decisions (i.e. allows you to choose exactly what you want)	• Increased complexity for broad asset class exposure (i.e. difficult to get diversity across asset classes such as equities, fixed income and commodities) • Massive database required to store entire asset universe • Increased portfolio generation complexity including frequent rebalancing and other execution/logistical issues
Aggregate Asset Classes	• Less internal rebalancing and ability to outsource costs to aggregate asset providers • Easier to obtain exposure to a wider array of asset classes • Guaranteed diversification (main problem with most individual security portfolio generation strategies such as the MVO is that the generated portfolios are unintuitive and highly concentrated) • Few assets required (and hence less cumbersome database requirements)	Limited data (i.e. historical data does not go as far back for aggregate asset classes as it would for individual securities)

Table 4: Pros and Cons of Asset Universe Selection

Looking at Table 4 above it became apparent that aggregate asset classes was the right choice. This view was further compounded when our key design features were considered. Namely, Alpha Factory strived to make investing accessible for everyone – as such we wanted to ensure that our customers can get highly diversified portfolios with low fees and low initial investment, while still being able to understand the assets in which they are invested. Using aggregate asset classes was the obvious choice as the fewer assets required to ensure diversification wouldn't be overwhelming to follow/monitor for our customers (in contrast to following an asset universe of hundreds of different securities) and the logistical benefits of having less rebalancing, smaller databases and guaranteed diversification made it easier to lower fees and the required initial investment balance.

Having selected to use aggregate asset classes, we next turned our attention to selecting the specific asset classes. With Alpha Factory's goal of creating portfolios that perform well in all financial environments, we decided to base our asset selection methodology on the all-weather framework which was popularized by Bridgewater Associates' All-Weather Fund. [25] Specifically, we wanted to ensure that the portfolios

we created performed well in all combinations of rising and falling economic growth and inflation. To achieve this performance and diversification we needed to select a diverse range of global asset classes as shown in Figure 1 below.

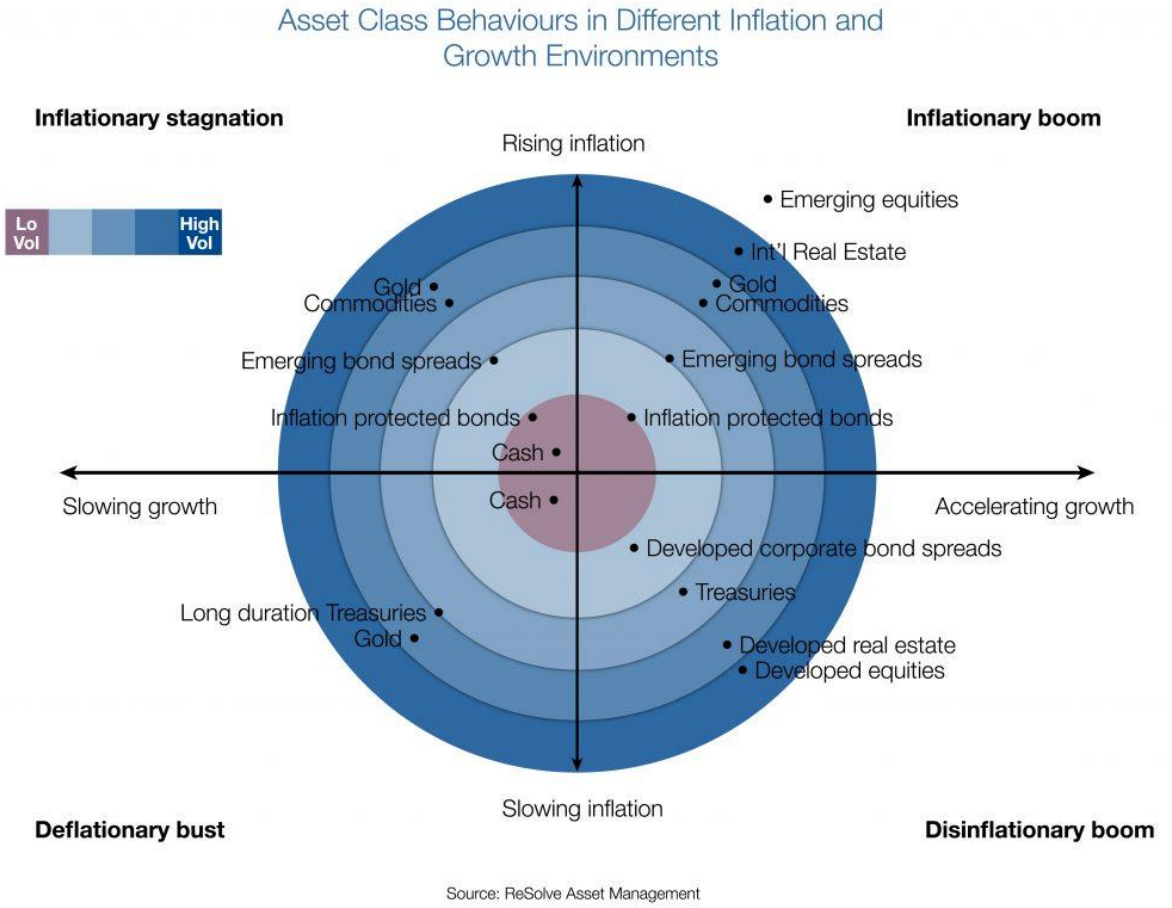


Figure 1: Asset class behaviours in various inflation and growth environments

Beyond achieving broad diversification across various asset classes, using the all-weather framework provided additional benefits. Namely, having assets with minimal overlapping exposure guaranteed that our portfolio construction did not result in excessive weighting in assets with the same type of underlying exposure. [26]

To gain exposure to the asset classes listed in Figure 1 above, ETFs were selected as the investment vehicle of choice as they cover the entire investment universe and are the easiest to invest in logistically. For example, in order to gain commodity exposure, we would require the use of futures and the constant rolling of contracts making the investment process significantly more complex. In contrast, we can easily gain exposure to commodities by investing in commodity ETFs. Other aggregate asset class investment vehicles such as mutual funds were not selected due to their higher costs and lack of readily available financial data making parameter estimation and investment decision making difficult.

The selection of the specific ETFs required an extensive process with several iterations and significant back-testing. The ETF selection process was largely based on the following criteria:

1. Similarity to index proxies (measured by tracking error against indices) with closer similarity being preferred
2. Low fees (measured by the Management Expense Ratio (MER)) with lower fees being preferred
3. Liquidity (measured by spread and Average Daily Volume (ADV)) with a minimum threshold being required (i.e. as long as we're trading something without a massive spread, we should be fine – want to avoid trading extremely rare and illiquid ETFs)
4. Data availability (measured by the length of historical data available) with longer time horizons being preferred

In selecting our ETFs criteria 2 and 3 were treated as thresholds with a certain amount being tolerable and as long as the ETFs met these thresholds, they would remain in the consideration set. On the other hand, criteria 1 and 4 were ranked with more similar ETFs / those with more data available being preferred. While we initially intended to treat criteria 1 with higher priority, we noticed that the two criteria were greatly correlated (i.e. if an ETF was more similar to its index proxy then more data was inherently available as the index can be used to proxy ETF returns). As such, no such distinction needed to be made as the ultimate design decisions would have stayed the same irrespective of which criteria was determined to have greater priority.

Furthermore, it is noted that the criteria above are similar to the criteria used by Nest Wealth. Specifically, they state that they "select what [they] believe are the best ETFs available in each asset class, making sure they work well together. [They] look for ETFs that are low cost, liquid, and have a low tracking error to their underlying index." [27]

Beyond pure asset class exposure, Alpha Factory included variations in the form of factor tilted exposures. Factor investing styles have been proven to produce excess risk-adjusted returns and therefore were worth incorporating into our portfolios. [28]. As an example, this was done by considering exposure in asset groups such as US Equity Value and US Equity Momentum instead of seeking US equity exposure alone. The criteria for including factors is that they have proven to be consistent across asset classes, geography and time, cannot be explained by other factors and are accessible through low cost options.

Using the criteria defined above, the following 19 ETFs (see Table 5) were selected to gain exposure to select asset classes. The specific asset classes were chosen to provide diversity across both geographic regions and asset types enabling an "all-weather" portfolio construction. All ETFs that were selected are USD denominated funds. This was a conscious decision as consistency was important to avoid currency exchange concerns and Canadian funds did not have sufficient diversity in their offerings.

Asset Class	Sub-Asset Class	ETF(s)
Equities	US	MTUM - iShares Edge MSCI USA Momentum Factor ETF VLUE - iShares Edge MSCI USA Value Factor ETF QUAL - iShares Edge MSCI USA Quality Factor ETF SIZE - iShares Edge MSCI USA Size Factor ETF USMV - iShares Edge MSCI Min Vol USA ETF
	Developed Markets (ex-US)	IVLU - iShares Edge MSCI Intl Value Factor ETF IMTM - iShares Edge MSCI Intl Momentum Factor ETF IQLT - iShares Edge MSCI Intl Quality Factor ETF
	Emerging Markets	EMGF - iShares Edge MSCI Multifactor Emerging Markets ETF
	EM & DM, ex-US	ACWV - iShares Edge MSCI Min Vol Global ETF

Asset Class	Sub-Asset Class	ETF(s)
Fixed Income	Long Term US	SPTL - SPDR Portfolio Long Term Treasury ETF
	US Intermediate	AGG - iShares Core U.S. Aggregate Bond ETF
	Emerging Market	EMB - iShares J.P. Morgan USD Emerging Markets Bond ETF
	Inflation-Protected	TIP - iShares TIPS Bond ETF
	Money-Market	SHV - iShares Short Treasury Bond ETF
	High Yield	HYG - iShares iBoxx $ High Yield Corporate Bond ETF
Real Estate	US	SCHH - Schwab U.S. REIT ETF
Commodities		DBC - Invesco DB Commodity Tracking Fund
Gold		GLD - SPDR Gold Shares ETF

Table 5: Current Investable Universe

As evident in the table our asset universe includes a wide diversity of ETFs that target a variety of asset classes and are expected to perform well in all economic environments. Additionally, this can be clearly seen by classifying our selected ETFs based on the economic environments they perform well in as shown in Table 6 below.

Economic Environments	Growth	Inflation
Rising	MTUM, VLUE, QUAL, SIZE, USMV, IVLU, IMTM, IQLT, EMGF, ACWV, DBC, EMB, HYG, SCHH	GLD, TIP, DBC, EMB, SCHH
Falling	GLD, TIP, SPTL, AGG	MTUM, VLUE, QUAL, SIZE, USMV, IVLU, IMTM, IQLT, EMGF, ACWV, SPTL, AGG

Table 6: Classification of our asset universe

We ultimately decided to only incorporate factor-tilted exposures for equity exposures as they were the only asset class with ETFs that met our criteria of having both adequate history and liquidity. The equity factors included are value, momentum, size, low volatility and quality for each of US, international developed and emerging markets. There are however some exceptions and differences in the implementation. Specifically, the size factor was not included for international developed markets due to the lack of sufficient liquidity in relevant ETFs. The low volatility factor for both international developed and emerging markets was accessed through a combined ETF as both separately were not accessible. Finally, the size, value, quality and momentum factors for emerging markets were accessed through a single ETF as individually they were not available in ETFs that met our criteria.

Source of Financial Data

In selecting the source of financial data, Alpha Factory considered using Bloomberg Terminal or other APIs such as Yahoo Finance and Quandl. After careful consideration of the pros and cons of each source (refer to Table 7 below), we determined that getting financial data from Bloomberg Terminal was the best option.

Option	Pros	Cons
Bloomberg Terminal	Accurate historical prices for both ETFs and indices	Inability to fetch data in real time (would require a manual process to upload financial data to our database)
Other APIs (e.g. Yahoo Finance, Quandl, etc.)	Ability to fetch data in real time using API enabled data sources	Hard to access historical prices of indices and ETFs

Table 7: Pros and Cons of Financial Data Sources

With Bloomberg Terminal lacking real time data fetching abilities and other APIs, such as Yahoo Finance lacking access to index prices and Quandl lacking free historical ETF prices it became evident that neither source of financial data is ideal. However, Alpha Factory opted to use Bloomberg Terminal at the expense of manual data fetching because of their index pricing. Since most ETFs have very limited historical data available, having access to the index proxies was very important for longer back testing capabilities. For this reason, ensuring the selected ETFs were similar to the index proxies was a key criterion in the asset selection above – specifically, knowing that a very limited amount of historical data was available for each particular ETF, we wanted to ensure that we selected ETFs that accurately tracked the indices, so those indices can be used to fill missing data.

It is important to note that the major downside of using Bloomberg Terminal is the inability to fetch data in real time. While this is an inconvenience for the time being, it is assumed that we will subscribe to the service once we release our platform publicly.

Incorporating indices required preprocessing of the data in the from of evaluating and appropriately modifying the index return time series before combining them with their respective ETF return time series. Simply using the returns on the index as a perfect proxy for the returns of an ETF before going live has multiple issues. Given that any ETF has some tracking error with respect to its index, it was important to use the available information we had to modify index time series so that they are better representative of the ETF in question. To do this we used overlapping return series of each index and ETF to determine the tracking error and modified our index returns accordingly. More specifically, the average difference in daily returns between each index and ETF was subtracted from the index returns prior to the existence of the ETF in order produce the back-filled historical returns. By doing this, we capture explicit differences like management fees and implicit differences like management style. With the help of the process, our entire asset universe has over 20 years worth of data ranging from the start of 1999 until today that is available for back-testing.

Portfolio Generation Strategy

Across the vast landscape of portfolio generation techniques there are three main characteristics of assets; namely their return, risk and co-movement that are the main determinants of “optimal” portfolios. Additionally, an investor’s attributes, primarily in the form of risk tolerance, will factor into the optimal portfolio allocation. However, risk tolerance does not inform the actual selection of the portfolio generation strategy but rather slightly modifies the specific portfolio generated. Therefore, we examined different portfolio generation techniques based on their property’s agnostic of individual investor preferences in this section as any technique can be tailored to produce portfolios for various risk

tolerances with tactical uses of leverage and additional constraints (considerations of investor performance are further discussed in a subsequent section below). In our exploration of strategies, we operate under the reasonable assumption that every investor seeks to achieve the maximum return for some given level of risk (mean-variance utility).

Portfolio generation techniques can be broadly classified based on the assumptions and estimates made on asset characteristics. In some strategies, parameters are not estimated (not direct inputs) yet in these cases it implies strong assumptions on relevant parameters and their relationships, or lack thereof, with others. For example, an inverse volatility approach only requires the estimation of each assets' risk (in the form of volatility) but assumes that correlations and return per units' risk are consistent among assets to produce a mean-variance optimal portfolio.

The most basic of portfolio allocation strategies is the equal weight portfolio which, as implied, simply allocates all assets equally. It acts as our base portfolio and should be improved on by any other method employed. Inherent to this portfolio generation technique is the assumption that we cannot estimate, nor do we have any views on asset returns, risk or co-movement. Additionally, this strategy is entirely dependent on the selected asset universe. It is reasonable to assume that we can predict to better than random certainty some asset characteristics which leads us to explore other portfolio generation techniques.

Portfolio Optimization techniques can be classified into one of the following groups:

1) Naive Risk Parity:

- Requires estimates of a risk parameter for each asset while assuming no accurate estimates can be made on asset co-movement or returns.

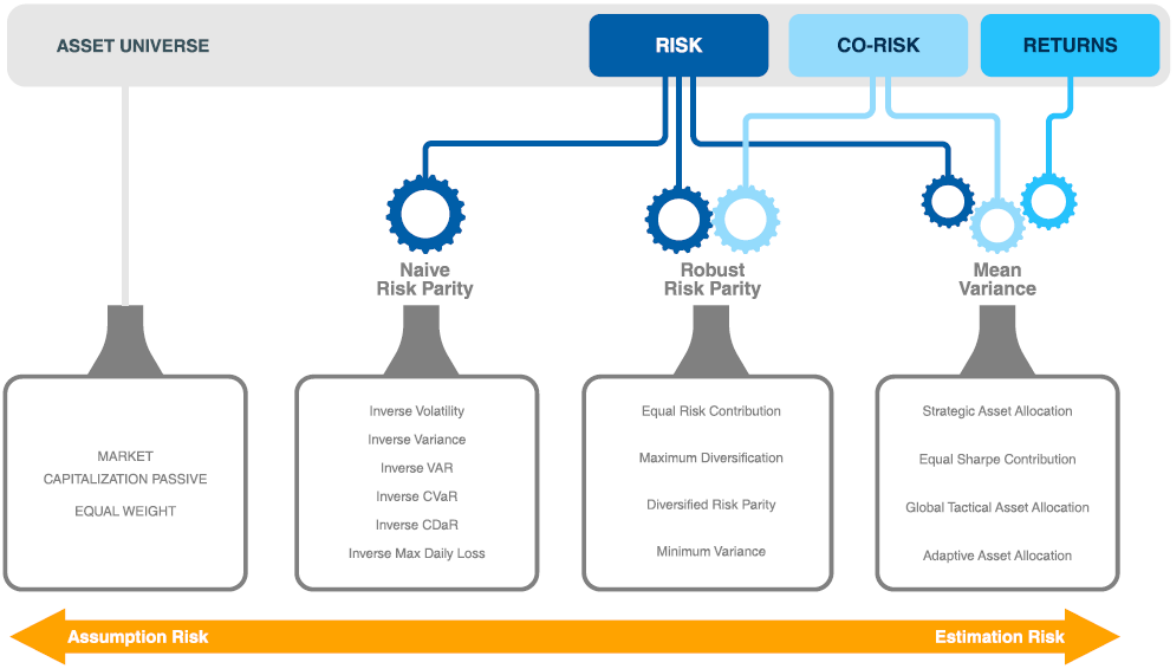
2) Robust Risk Parity

- Estimates both risk and co-movement parameters but assumes that returns cannot be accurately estimated.

3) Mean Variance:

- Estimates all parameters (risk, co-movement and returns) and therefore is very susceptible to estimation risk.

Figure 2 illustrates how various portfolio optimization techniques can be classified based on estimation of asset parameters. [29]



Source: ReSolve Asset Management. For illustrative purposes only.

Figure 2: The Portfolio Optimization Machine

To settle on our portfolio optimization technique, we first narrowed down our selection onto a group, based on which parameters we believe we could estimate. The so-called Mean Variance category was ruled out based on well-researched and documented evidence that estimating returns is an extremely difficult task. [30, 31] The naïve approach of estimating returns using historical data requires an extremely long history to be significant and is shown to be inconsistent. Factor models are often seen as a viable alternative for return estimation however, evidence supporting their use remains uncompelling. The simplest factor model in the form of the single factor CAPM (Capital Asset Pricing Model) has been shown to be empirically invalid. [32] Even other more extensive factor models (i.e. Fama-French 3-Factor Model) that demonstrate more explanatory power display poor predictive capabilities. [33] Important to note is that all factor models also create the need for additional parameter estimates in the form of factor risk premia returns and factor loadings. These introduce supplementary sources of errors and require further robustness considerations – an unnecessary complication with unproven value add.

Another approach which is popular among practitioners is the Black-Litterman model. This model uses investors' views on asset returns in combination with the implied returns from market capitalizations to generate optimal weights. [34] This method suits larger financial institutions who would like to express active views and have the proper resources and personnel to justify doing so. However, here at Alpha Factory, we cannot rationally claim to have a better than average knowledge on future asset returns nor do we target investors who are qualified to do so. Correspondingly in the absence of views, the Black-Litterman model is simply the market-capitalization weights and provides no added benefits for us.

Naïve risk parity approaches, although suspect to less estimation error, make excessive simplifying assumptions. These techniques assume that no accurate estimation can be made with regards to how

assets move together. It is evident though history that assets perform differently in certain economic conditions and this fact is vital to the foundation of the all-weather portfolio. As we believe this relationship can be estimated with better than random precision, we abandoned naïve risk parity approaches in favour of robust risk parity techniques.

A robust risk parity approach involves estimating the risk of all assets and the co-movement while conceding that very few can accurately predict future returns. Two variations of this approach include Maximum Diversification and Minimum Variance. Maximum Diversification assumes that returns are proportional to total asset risk and hence optimizes for the largest ratio of weighted average volatility of the portfolio components to its overall portfolio volatility. Minimum Variance minimizes the overall portfolio volatility and assumes that returns are independent of risk. A drawback to both these methods is that each can often produce very concentrated portfolios; Maximum Diversification will focus weighting in high volatility but low market covariance assets where as Minimum Variance will heavily concentrate in the lowest volatility assets. [35] The Equal Risk Contribution (ERC) method is a third approach which offers a compromise that ensures good diversification. It optimizes allocations to ensure that all assets contribute the same volatility to the overall portfolio while being optimal when assets have equivalent marginal Sharpe ratios (equally improve the Sharpe of the portfolio per unit risk allocated to the asset). [36, 37] The ERC approach is well suited to the all-weather approach and given that we are comfortable with its implicit assumptions and required estimations it was selected as our portfolio generation strategy.

Portfolio generation techniques all focus on diversification as a risk management technique, however it has been shown that the use of market-timing techniques in the form of a trend-following system can significantly improve portfolio return profiles. [38] At Alpha Factory we employ a trend-following overlay system on our portfolios as an added form of risk protection. Our strategy emulates a system shown to have empirical merit in *A Quantitative Approach to Tactical Asset Allocation* [39], improving risk-adjusted returns and notably drawdowns. We believe that reducing drawdowns are important as they are immediately felt by investors and can be a major driver in them recalling capital. Reducing drawdowns was not explicitly a focus in the optimization process as drawdowns are difficult to estimate and can severely hinder returns. At a high level, the system works by allocating each asset to its optimal allocation (as determined by our portfolio optimization) when trending up[1] and rotating to cash when trending down[2]. This results in a portfolio that normally contains portions of cash. We also explored potential improvements on the trend following system with variations including scaled positioning allocations and re-allocation to trending assets as opposed to cash. Overall, through a simple timing technique, we believe that we can outperform competitors with similar offerings as they rely solely on diversification as a risk management technique.

Before simply going with the portfolio generation selection above, Alpha Factory developed a Python based back-testing framework to validate research conclusions and empirically inform any other portfolio generation decisions. The back-testing framework enabled the historical simulation of a given portfolio's returns producing a portfolio tearsheet that captures a portfolio's performance and risk characteristics and was used to evaluate and compare various portfolio generation techniques. See (Appendix: Tearsheet

[1] Trending up is quantified by an asset price above its 200 Day SMA

[2] Trending down is quantified by an asset's price below its 200 Day SMA

1) for an example tearsheet generated for an equal weight portfolio rebalanced monthly and refer to Table 9 for details on the various components.

Having settled on a robust risk parity approach we confirmed our selection through our own empirical validation. To do so we first backtested an equal weight portfolio of all our assets in the current investable universe. The results for this portfolio with monthly rebalancing can be seen in Appendix: Tearsheet 1. After doing so we performed a similar backtest with a static risk parity weighting scheme in the form of equal risk contribution with results shown in Appendix: Tearsheet 2. As was evident and expected, the risk parity portfolio is superior to the equal weight portfolio on a risk-adjusted return basis (measured by Sharpe and Sortino ratios). Important to note is that this risk parity approach is a naive implementation as it uses all available data to generate volatility and covariance parameters that input into the generation of weights. This subjects the portfolio to a look ahead bias and therefore is not an accurate representation of real-time performance. To address this concern a dynamic risk-parity portfolio was also backtested. This dynamic portfolio using a rolling 200 day look back period to calculate parameters and update weights on an on-going basis. As with other portfolios, all rebalancing is done monthly. Although the results Appendix: Tearsheet 3 maintain an improvement of risk-adjusted returns against the equal weight portfolio, the dynamic risk parity approach did not improve on the static risk parity. Interestingly, this initial result was contrary to conclusions from research claiming that dynamic approaches outperform static ones. However, after further testing with a more complete asset universe and longer data history we found that the performance of either methodology was close to indistinguishable. Comparing the static weighting scheme in Appendix: Tearsheet 4 and the dynamic one in Appendix: Tearsheet 5 we can see that the dynamic portfolio outperforms on a risk-adjusted basis (Sharpe ratio). Although the outperformance is not that significant the result in itself is. Given that the static approach is subject to look-ahead bias it is important that we can achieve similar or better results using look-back methods (i.e. dynamic) that is not subject to this bias.

An essential component of any risk-parity portfolio is leverage which can be used to scale up the returns of a portfolio to desired levels. In the case of risk-parity portfolios the importance of leverage is greater than that with other allocation strategies as risk parity portfolios tend to produce portfolios with lower expected annual returns. With traditional portfolios generation strategies (i.e. MVO) concentration risk is usually taken on to achieve higher returns whereas for risk parity portfolios leverage risk is taken on to reach higher returns. Implementing leverage within a portfolio introduces costs for borrowing funds and therefore can impact a portfolios performance. In order to verify that leverage would be acceptable we back tested portfolios with various amounts of leverage. Using the reasonable assumption of leverage costing 0.5% plus the risk-free rate (3-month US T-bill) we saw that the impacts of increasing leverage: no leverage (Appendix: Tearsheet 6), 2 times leverage (Appendix: Tearsheet 7), 3 times leverage (Appendix: Tearsheet 8), did not severely affect portfolios performance (Sharpe).

Risk parity is a misnomer as it implies equal risk across all assets, however this need not be the case. We can individually assign risk targets to every asset in our universe enabling us to be more selective in how we create our “all-weather” portfolio. Without doing so; blindly assigning equal risk to each asset class severely exposes to asset selection bias. Our asset specific risk allocation was based on the economic intuition of allocating risk so that adequate amounts were subject to every type of economic environment. The following table details our risk allocation for each ETF and how the risk rolls up the larger asset classes.

Asset Class	Risk Allocation	Sub-Asset Class	Risk Allocation	Ticker	Risk Allocation
Equities	50.00%	US	20.00%	MTUM	4.00%
				VLUE	4.00%
				QUAL	4.00%
				SIZE	4.00%
				USMV	4.00%
		Developed Markets (ex-US)	12.00%	IVLU	4.00%
				IMTM	4.00%
				IQLT	4.00%
		Emerging Markets	10.00%	EMGF	10.00%
		Global (EM + DM, ex-US)	8.00%	ACWV	8.00%
Fixed Income	30.00%	Long Term US Bonds	7.00%	SPTL	7.00%
		US Intermediate Bonds	7.00%	AGG	7.00%
		Emerging Market Bonds	5.00%	EMB	5.00%
		TIPS	6.00%	TIP	6.00%
		High Yield Bonds	5.00%	HYG	5.00%
		Money-Market	N/A	SHV	N/A
Real Estate	8.00%	US	8.00%	SCHH	8.00%
Commodities	8.00%	Commodities	8.00%	DBC	8.00%
Gold	4.00%	Gold	4.00%	GLD	4.00%
Total	**100.00%**		**100.00%**		**100.00%**

Table 8: Risk Allocation Breakdown

The specific risk allocation decisions were made using a top-down approach; meaning we started by selecting an appropriate allocation for each asset class, then for each sub-asset class and finally each ETF. Starting from a desired risk allocation of 50% for equities we chose our geographical exposures based on market sizes resulting in a higher allocation in US equities, followed by international developed markets (ex-US) and lastly emerging markets. Within each sub asset class, the risk was split approximately equally across the specific ETFs. With the remaining 50% risk the majority was allocated to fixed income and the remaining 20% was split across Real Estate and Commodities and Gold. Within Fixed Income risk was relatively evenly distributed with a slightly larger emphasis being placed on the less risky assets like long term US bonds, US Intermediate Bonds and TIPS.

As seen in the table above, Money-Market (SHV) does not have any pre-set risk allocation. This is due to the fact the it is essentially a risk-free asset and Is allocated to in the place of assets that are deemed to be trending unfavourably. Its allocation is therefore a by product of other portfolio allocation decisions and ends up averaging approximately a 24% capital weight of our Alpha Factory portfolios.

We also verified our choice to include a trend following overlay to all our portfolios as an added measure of risk protection. This concept was tested on both equal weight and risk parity portfolios. The specific trend-following approach employed is as follows: if an asset is above its 200-day simple moving average then it deemed to be favourably trending and the target allocation is made, if not, the allocation is made to cash instead. As evident in the results (Appendix: Tearsheet 9 for equal weight and Appendix: Tearsheet 10 for risk parity and Appendix: Tearsheet 11 for dynamic risk parity) the trend following overlays help

reduce overall risk in terms of volatility, improve risk-adjusted returns and reduce drawdowns all with respect to their non-trend following counterparts. Our initial results generated are also conservative as when not allocating to an asset deemed to be trending unfavourably cash is held and assumed to yield 0%. When moving to a more realistic implementation, allocations will instead rotate into something like a money market instrument with positive yield. This will result in greater returns at the expense of almost no additional risk resulting in better overall risk-adjusted returns. Other alternatives like reallocation to positively trending asset classes was initially considered however later discarded since it could at times negate the diversification benefits of the portfolio.

Another consideration included in our back tests are the effects of transaction costs. Transaction costs were incorporated with the assumption of a 4-bps cost for every single transaction proportional to the weight traded. Spread was used as a measure for transaction costs and informed our cost estimate which was determined from the weighted average of ETF spreads (refer to Appendix E for calculation details). Using the average monthly turnover, we verified that transaction costs had minimal impact on our portfolio decisions. Importantly the transaction costs were not enough to detract us from implementing a trend-following overlay (Appendix: Tearsheet 12) which had higher transaction costs with its increased turnover compared to its non-trend-following counterpart (Appendix: Tearsheet 13)

Component	Description/Details
Cumulative Return Chart	Displays the cumulative historical return over back-tested period.
Underwater/Drawdown Chart	Displays cumulative drawdowns experienced by the portfolio.
Positioning Chart	Displays allocation percentage to each asset and how it changes over the duration of portfolio. Weights are affected by the daily returns of each asset and are reset to their target based on the rebalancing frequency.
Overview	Details main characteristics of the portfolio: • *Start Date:* first day of returns • *End Date:* last day of returns • *Rebalanced:* how often weights are rebalanced to their target • *Trend Following:* if a trend following overlay is used will display the trend indicator otherwise will show N/A • *Weighting:* indicates portfolio weighting methodology
Statistics #1	Performance and risk metrics of the portfolio: • *Total Return*: cumulative return of portfolio from start to end date • *CAGR*: Compound Annual Growth Rate • *Annual Volatility:* annualized value based on daily standard deviation of returns

Component	Description/Details
	• *Sharpe*: Annualized Excess Return*/Annual Volatility • *Max Drawdown*: Largest peak to trough loss • *Sortino:* Annualized excess Returns*/annualized volatility of negative returns
Statistics #2	Additional performance and risk metrics: • *VaR:* 99th percentile 1-day value at risk using historical approach • *CVaR*: 99th percentile 1-day conditional value at risk using historical approach • *Beta*: portfolio's Beta with respect to the S&P 500 • *Alpha:* expected return above that predicted by CAPM (S&P 500 as market factor) • *R-Squared:* proportion of variability in portfolio returns explained by the CAPM (S&P 500 as market factor) • *Treynor:* Annualized Excess Return*/Beta
Performance Attribution	Displays the contribution of each asset to the overall portfolio's return in each year.
Yearly Returns Chart	Shows the total return of the portfolio for each year.

Table 9: Components of a Portfolio Tearsheet

*Excess returns are measured with respect to 3-month US-T-bill yield

Parameter Estimation

The relevant parameters requiring estimation for our selected risk-parity portfolio optimization approach are risk and co-movement of assets. Risk is measured by volatility of returns as it the simplest to estimate and representative of alternative measures. Other types of risk measures like VaR, CVaR and maximum drawdown may be more representative of an investors preference to preserve capital but require additional assumptions with regards to return distributions introducing unnecessary complexity. The following statement adeptly shows how volatility can be an all-encompassing risk measure: "consider that, when a systematic strategy is regularly rebalanced, a large expected mean relative to volatility strongly implies a smaller risk of permanent loss, a smaller expected maximum drawdown, and a smaller expected shortfall." [40] As long as our asset universe does not contain assets with extremely asymmetric return distributions (i.e. options, high yield credit, etc.) volatility should be an adequate risk measure. [41] Co-movement is measured by the covariance of returns as is common practice. Both parameters are estimated using a historical lookback period, therefore avoiding any look-ahead bias and resulting in more indicative backtesting results. Additionally, parameters are estimated using a rolling lookback period thus creating dynamic allocations in accordance with shifting covariances and volatilities. This dynamic approach has shown to noticeably improve risk-adjusted performance when compared to a simple static approach. [42]

Considerations for Robustness, Rebalancing & Re-Optimization

To address any uncertainty in our parameter estimations, multiple computations and comparisons were made. By analyzing how varying parameter estimation factors affect results we can identify whether results are simply driven by data mining/overfitting of the parameters.

The first parameter in question was the rebalancing frequency; how often weights would be readjusted to their targets. Our decision to use monthly rebalancing was based on both quantitative evidence and logistical feasibility. Daily (Appendix: Tearsheet 14), Weekly (Appendix: Tearsheet 15), Monthly (Appendix: Tearsheet 2) and Yearly (Appendix: Tearsheet 16) rebalancing frequencies were tested on an equal weight portfolio. Although there was a clear trend of decreasing performance as the rebalancing frequency was reduced, the differences were minimal. Daily and weekly rebalancing were ruled out due to significant turnover that would lead to higher transaction costs and yearly demonstrated noticeable deviations from target weights. As such Monthly rebalancing was selected providing a compromise between deviation and turnover. Although consideration of a threshold rebalancing system was made, it was not selected as it introduced unnecessary complexity in the form of additional parameters and would be more involved for individual investors to implement on their own.

The second parameter examined for robustness was the rolling lookback period; i.e. the amount of data that would input into the generation of dynamic risk parity weights. Using a 200-day lookback period (Appendix: Tearsheet 17) as a starting point, comparisons were made with both 150 (Appendix: Tearsheet 18) and 250-day (Appendix: Tearsheet 19) lookback periods. Although there was stability above the 200-day look back period using a shorter lookback period did result in a noticeable decrease in performance. This was inline with our expectations that a long enough look-back period is required to produce accurate estimates for variance and covariance.

The third parameter analyzed was the trend following indicator. We decided to use the 200-day simple moving average in line with *A Quantitative Approach to Tactical Asset Allocation* [12] which also verifies the robustness of the selection by comparing it to other lookback periods. Our own testing validates their findings with 100-day SMA (Appendix: Tearsheet 20), 150-day SMA (Appendix: Tearsheet 21) and 200-day SMA (Appendix: Tearsheet 11) all having similar results. As we increased the look back period performance started to suffer (300-day SMA Appendix: Tearsheet 22) but this was expected given the nature of a trend following system. With too long of a look back the system will react too slowly to changes in an assets performance and will most likely be late for both entry and exit.

Lastly the robustness of the asset universe was also tested to ensure the selection of our universe is not the main driver of performance. Comparing the performance of our un-levered risk parity portfolio (Appendix: Tearsheet 23) with slight variations in the asset universe show minimal difference (Appendix: Tearsheet 24, Appendix: Tearsheet 25).

Considerations for Investor Preferences and Risk-Tolerance

As noted in the Portfolio Generation Strategy section above, an investor's attributes, particularly their investment goals and risk tolerance, factor into the optimal portfolio allocation. However, since risk

tolerance does not inform the selection of the portfolio generation strategy, merely requiring slight modifications to generate the unique portfolio, it was largely ignored during our portfolio generation strategy selection. Specifically, in selecting our portfolio generation techniques we examined them on their property's agnostic of investor preferences as any technique can be tailored to produce portfolios for various risk tolerances with tactical uses of leverage and additional constraints.

Turning our attention to determining investor preferences, we considered the return goals and risk-tolerance for each of our investors as discussed in the subsections below.

Return Goals

Everyone is different and has different investment goals. Often times investors do not even know what their investment goals are, and if they do, it's hard to put them in numbers/words. Alpha Factory tried to help with that. Specifically, we try to match return goals to life goals by considering what our clients are saving for. With life goals usually easier to define, we find this approach to be more intuitive and user-friendly. As this is one of our main design principles, we opted to use this approach in favour of more continuous spectrum of risk inputs.

Since we are a passive investment fund, we build our portfolios for longer time horizons (5+ years). As such our return goals are largely catered to the following broad categories of life goals:

- Saving to purchase a car/home (**short-term**: ~5 to 10-year investment horizon)
- Creating an education fund for your children (**medium term**: ~10 to 20-year investment horizon)
- Saving for retirement (**long term**: 15+ year investment horizon)

In addition, we consider age and initial investment amount by requiring the user to input all of these fields when creating an account. Initially we also tried to incorporate a specific return goal in terms of percentages, as inputted by our users, but we decided against this for two reasons. First, as we opted for a risk parity portfolio generation technique it was very difficult to ensure that our portfolios met the specific return goals provided. Second, after surveying 49 randomly selected University of Toronto students, we found that most of them struggled with putting a specific return goal – namely, with limited exposure to the stock market few even knew what constituted good returns / what returns traditional investment firms even produced. Given that we aim to keep the user experience as simple and intuitive as possible, observing that providing a specific number to individual investment goals is much harder to do and seeing that incorporating return goals is very difficult given our risk parity approach we ultimately decided to omit this from our consideration.

As such, we ultimately decided to limit our return goal inputs to one of the aforementioned life goals and a specific time horizon in years which we then considered as inputs for both our portfolio recommendation and risk-tolerance level assessment as discussed further below.

Risk-Tolerance

Finally, our unique portfolio generation techniques depend on the risk-tolerance level of our investors. Rather than overfitting with a continuous spectrum of return goals and risk constraints, Alpha Factory aimed to create five distinct portfolios to accommodate the different risk-tolerance levels of our customers. This is a trend that was noticed in our survey of existing solutions with Wealthsimple only having three distinct portfolio classes (called Conservative, Balanced and Growth). We find this strategy to better fit our business mantra as it would allow us to build and manage five distinct portfolios and pool our investor's money to the management of each. Ultimately, this will allow us to keep our fees low and require a low initial investment balance. With that said, we understand that the portfolios are not truly unique for every single investor, but we believe that is a compromise they are willing to make to ensure the costs are kept low – at the end of the day, money matters and lower costs mean more money is contributed to their life goals. More that this, we believe that having five distinct portfolios will still ensure that their views are met.

We divided our five distinct portfolios as shown in the Table 10 below. To determine the risk-tolerance of our investors we require them to fill out a questionnaire consisting of several questions. To keep the process as simple as possible we decided to limit our questionnaire to 15 questions containing only two options for each question. We decided to limit the number of questions to 15 because we figured that doing so will allow our users to breeze through the questions since they will only have to consider two options. As an added benefit, this questionnaire method will also limit the extent of outlier data and greatly simplify the logic for determining the risk tolerance of our users, without losing much on accuracy (we conducted independent surveys to test this claim – refer to the results of the Usability Testing Protocol for confirmation).

Specifically, our questionnaire consists of a series of scenarios with one option being risk-adverse and the other being risk-seeking in a random order (i.e. the risk-adverse option will not always be on the left in an attempt to reduce bias). Depending on the number of risk-adverse selections made and time horizon of the investment goal, users will be grouped in their respective portfolio as shown in the Table 10 below. Note that the number of questions that are required to be classified for each risk-tolerance level will depend on the time horizon and goals of the investor (further explained below). At the end of the questionnaire the user will be assigned a risk-tolerance level and be recommended a portfolio to hold. Users will be required to consent their approval on accepting the recommended portfolio based on their risk-tolerance and will be given the option to override this recommendation by selecting their own portfolio. For example, a medium-term investor who selected 6 risk-averse options will be classified as having a Neutral risk-tolerance level and be recommended the Balanced portfolio but will be given an option to select another portfolio instead.

Risk -Tolerance		Extremely Risk-Averse	Risk-Averse	Neutral	Risk-Seeking	Extremely Risk-Seeking
Portfolio		**Preservation**	**Conservative**	**Balanced**	**Adventurous**	**Aggressive**
# of Risk-Averse selections	Short-term	12 to 15	9 to 11	5 to 8	2 to 4	0 to 1
	Medium-term	13 to 15	10 to 12	6 to 9	3 to 5	0 to 2
	Long-term	14 to 15	11 to 13	8 to 10	4 to 7	0 to 3
Leverage		No Leverage	1.5x	2x	2.5x	3.5x

Table 10: Risk-Tolerance Level Determination Logic

Note that although the users will get the same 15 questions, the number of selections required to be classified as each risk-tolerance level varies by their time horizon and investment goals. The thought process behind the number of selections required is hopefully fairly intuitive: with a shorter time horizon, individuals are usually more risk-averse – as such, you are required to be more risk-seeking to be considered "extremely risk-seeking." On the other hand, with a longer time horizon, you can be considered "extremely risk-seeking" even if you wouldn't be considered "extremely risk-seeking" in a short-term setting as the longer-term investment horizon affords investors the luxury and comfort of more fluctuations. The numbers in the table above were selected with the aim of recommending portfolios according to the logic displayed below.

Currently, our risk-tolerance level logic only uses the time horizon to base its user risk-tolerance level assessment. The next step for this would be to include the specific return goal in some way. For example, it is entirely possible for someone to be saving for retirement with a 10-year investment horizon, while someone else can be saving for a house with a 10-year investment horizon. Given the varying natures of their return goals, the two members should have different risk tolerance levels. Specifically, given that the individual saving for retirement in 10 years will be out of an income 10 years from now, it is likely that s/he will be more risk averse hoping to avoid losses more than generating crazy gains. On the other hand, the individual saving for a house in 10 years time is more likely to accept pushing back their time horizon (i.e. if they waited 10 years, they can wait another one or two years before purchasing a home). As such, it is expected that this individual would be more willing to take on some more risk knowing that s/he has some flexibility with their time horizon with the hope of saving more money towards their dream home.

The Preservation/Conservative portfolios are best for:

1. Users with a short-term goal (~5 years) like buying a home or saving for vacation and are only comfortable with small fluctuations
2. Retired users who are comfortable with small fluctuations
3. Anyone else who's really risk adverse

The Balanced portfolio is best for:

1. Users with a medium-term goal (~5 to 15 years) and are comfortable with some fluctuations
2. Users with longer term goals (15+ years) and are willing to take some risks to ensure funds grow over time but aren't comfortable with very large fluctuations
3. First time investors who haven't had experience with investing or account fluctuations before

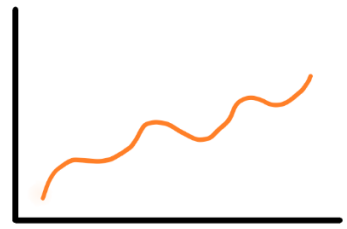
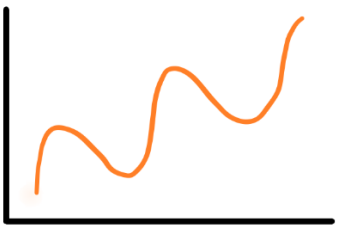
Finally, the Adventurous/Aggressive portfolios are best for:

1. Users with long-term goals (15+ years) who are comfortable with seeing very large fluctuations
2. Users with stable sources of income and are comfortable with larger fluctuations
3. Users investing a small portion of assets and are okay to "gamble" more
4. Users hoping to grow their money more aggressively for a medium-term goal but accepting the fact that a market dip might require them to push back on the time horizon – goals with flexible time horizons like buying a house can fit this bill

The five portfolios available to our investors are in essence the same optimal portfolio adjusted for various risk tolerances. This is done with various applications of leverage to our base portfolio. Our five portfolios are unlevered, 1.5 times leverage, 2 times leverage, 2.5 times leverage and 3.5x times leverage. Depending on any user's risk tolerance we can place them into the portfolio that produces the best return for their assessed level of risk tolerance. For details on each of our portfolios refer to the relevant appendix: Appendix: Tearsheet 26 for Preservation, Appendix: Tearsheet 27 for Conservation, Appendix: Tearsheet 28) for Balanced, Appendix: Tearsheet 29 for Adventurous and Appendix: Tearsheet 30 for Aggressive.

Questions

Finally, provided in Table 11 below are the questions we will ask our users. Please be reminded that both the questions and options will occur in random order in attempt to reduce survey bias.

	Question	Risk-Adverse Option	Risk-Seeking Option
1	Rather:	Gain $20 for sure	20% chance of gaining $100, 80% chance of no gain
2	Rather:	No gain	50% chance of getting $100, 50% chance of losing $100
3	Rather	Lose $20 for sure	20% chance of losing $100, 80% chance of no loss
4	Which portfolio history would you prefer:		
5	Poker tournament preferred purse:	Prizes for final table	Winner takes all!
6	Bigger fear:	Portfolio performance that is consistently less than industry benchmarks	Missed investment opportunity that could have yielded higher returns
7	Bigger fear:	Loss of principal over any period of 1 year or less	Rate of inflation that exceeds rate of return long-term
8	Bigger fear:	Portfolio performance insufficient to meet financial goals	Portfolio performance that is consistently less than industry benchmarks
9	Your focus when monitoring portfolio(s):	Recent results of overall portfolio	Long-term performance of overall portfolio
10	Your focus when monitoring portfolio(s):	Investments doing poorly	Investments doing very well

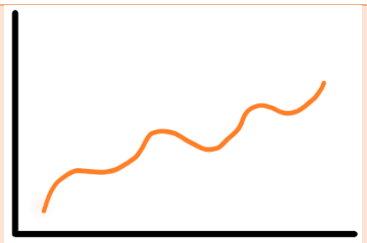
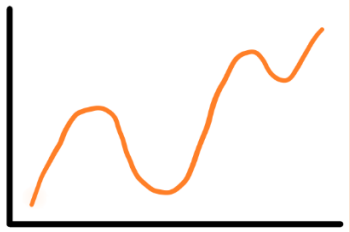
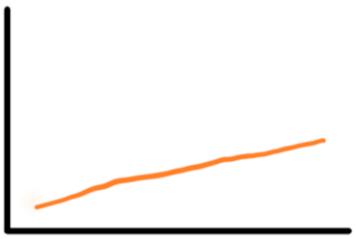
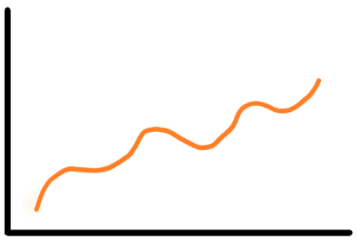
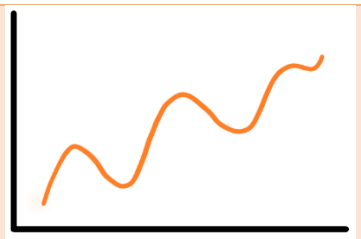
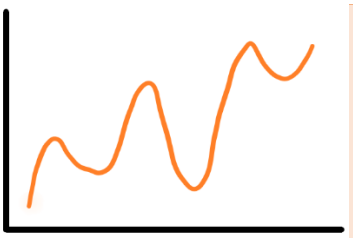
	Question	Risk-Adverse Option	Risk-Seeking Option
11	Your focus when monitoring portfolio(s):	Investments doing very well	Recent results of overall portfolio
12	Value of portfolio drops 20%, what do you do:	Move money elsewhere immediately	Consult with advisor to ensure asset allocation/portfolio is correct
13	Range of portfolio returns:		
14	Range of portfolio returns:		
15	Range of portfolio returns:		

Table 11: Risk-Tolerance Determination Questionnaire

Portfolio Validation and Analytics

Alpha Factory tested and validated it's generated portfolios by comparing them to several other portfolios. First, we compared our generated portfolios to an equal weighted portfolio to ensure that our portfolio does in fact outperform the equal weighted portfolio. If our portfolio is not beating the equal weighted portfolio than our estimates/methodology clearly is not doing anything helpful. Second, we compared our portfolios to standard asset allocation mixes, such as the popular 60/40 and 70/30 equity/fixed income asset allocations. Finally, we compared our portfolios to a portfolio generated using a standard Mean-Variance Optimization technique using our selected asset universe. Alpha Factory may also compare its generated portfolios to common indices such as the S&P 500 as deemed appropriate. Comparing our generated portfolios with several select portfolios will hopefully validate our methodologies and justify their use.

In addition to the standard tests we ran, our robo-advisor platform also allows users to input their own test portfolios and compare the performance of their inputted portfolio to those that we generate over various time horizons.

To further validate the generated portfolios, Alpha Factory conducted a regime analysis by testing their portfolios over different market periods. Having selected the use of an all-weather framework, Alpha Factory's portfolios should theoretically perform well in all market conditions. As such, we tested our portfolios by backtesting them over different periods (e.g. inflationary booms, inflationary stagnations, financial crises, etc.) and compared their performance over those periods with the performance of the aforementioned portfolios during the same timeframe.

Finally, Alpha Factory calculated a suite of portfolio performance metrics for each of its portfolios to assess their performance. Metrics that were considered include average returns, average volatility, maximum drawdown, the Sharpe Ratio and alpha with respect to some factor models.

The results of the tests are not further discussed in the report as our web application enables those interested to run their own tests and compare the performance on their own.

Front-End

Front-End design decisions constitute those regarding the user interface of our robo-advisor. Key design decisions made include the user interface and wireframe, the gathering of user inputs, display of computed data and decisions on the technologies used. Each key design decision is further explored in the subsections below.

Finalized User Flow Diagram

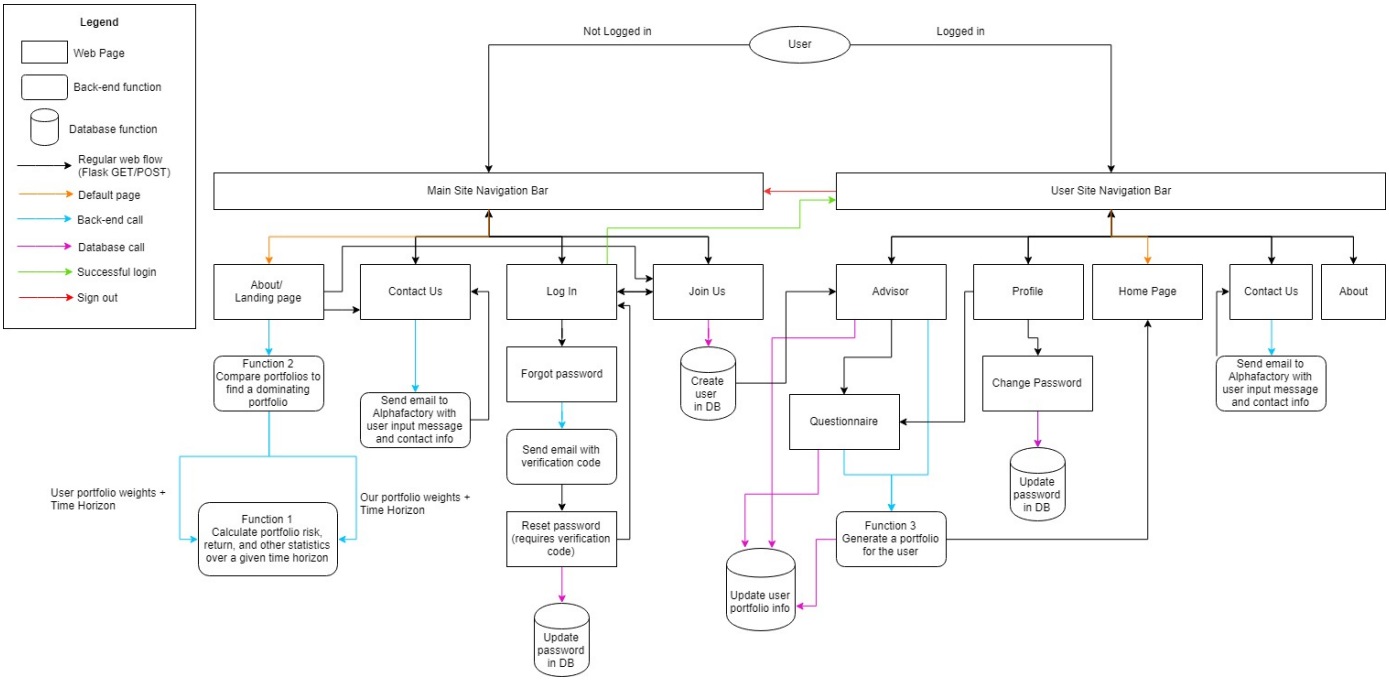


Figure 3: User Flow Diagram (full-scale diagram can be found here)

The above user flow was finalized and referenced from the original user flow found in Appendix C. Several changes were made in order to optimize user experience while using our platform which will be spoken about in detail through our alternative designs section. Our previous high-level BPMN diagram, summarizing the high-level overview of the site is provided in the subsequent section.

BPMN Diagrams and Descriptions

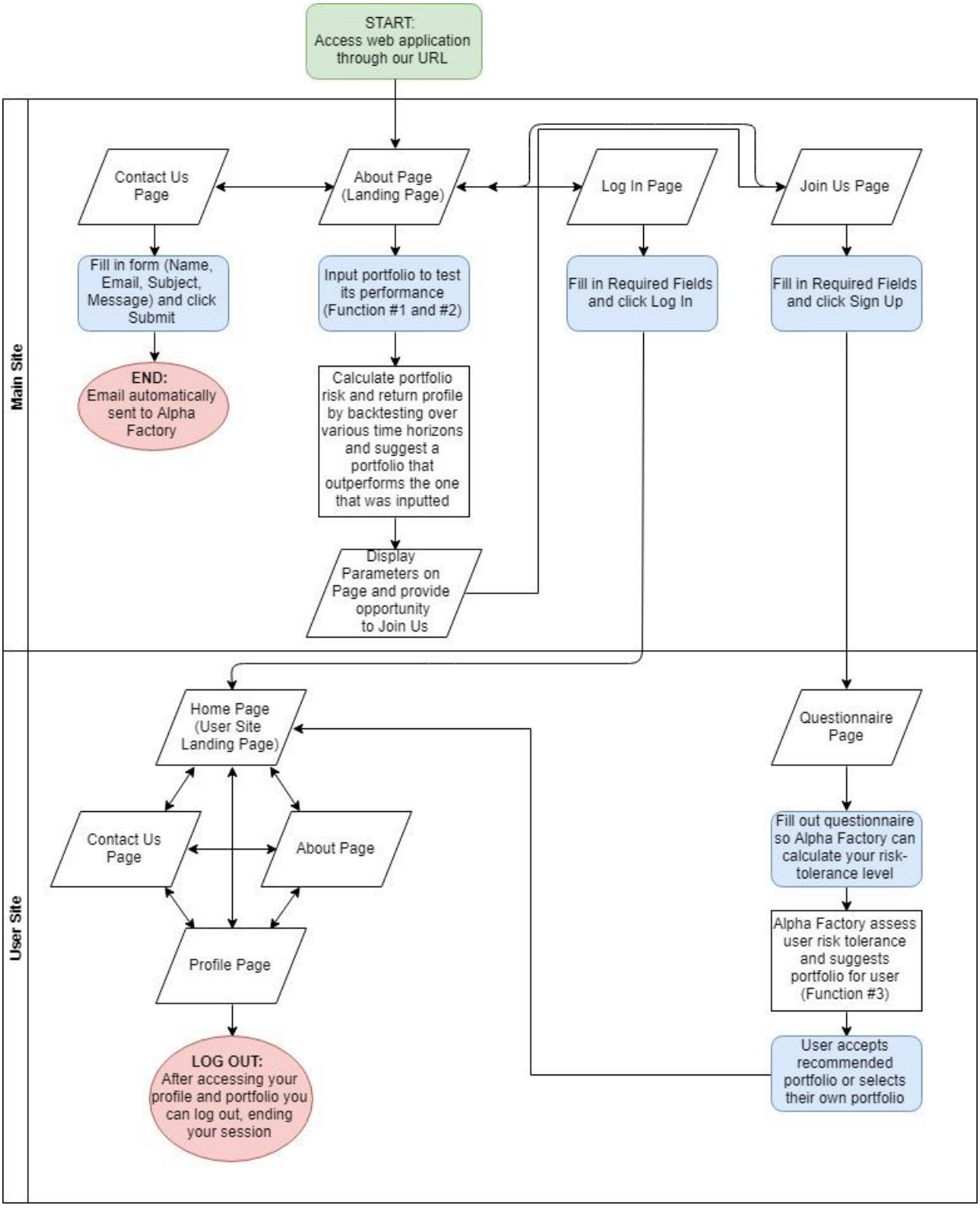


Figure 4: High-Level BPMN Diagram

A detailed Business Process Model of the user flow has already been provided in Figure 3. Here we provide a high-level diagram as we believe it is most beneficial to our stakeholders as it enables them to obtain a better understanding of our web application flow without worrying about all the back-end nuances and additional features that are not central for the flow of the key product details. Further, with the web application being live at http://alphafactory.herokuapp.com/about we encourage readers to check out our application and view the various pages/features as doing so will provide users with a better understanding of our robo-advisor than written text ever could.

Figure 4 above contains our high-level BPMN diagram. For starters, our diagram is split into two main swimlanes – the main site and the user site to distinguish between the two distinct platforms of our website.

As evident from our diagram users can access our web application through its URL where they are taken to the About Page. From the About Page they can access the other pages using the menu bar on top of the page. These pages include Contact Us, Log In and Join Us pages. Staying on the About Page, visitors can read more about our goals, objectives, asset universe and portfolio generation strategy. They can also input their own portfolios by specifying the quantity of each asset they would like to hold. After submitting their portfolio, we calculate the risk and return profile of their portfolio and generate a portfolio that outperforms the portfolio they just submitted, fulfilling functions #1 and #2 of the project requirements. Using this as a marketing technique for our platform we ask the user if they want to join us and conveniently provide a hyperlink that redirects the visitor to our Join Us page.

The Contact Us page contains a form that the users can fill out. Once completed, the users can submit their form resulting in an e-mail being automatically sent to the Alpha Factory email account.

The Log In page asks for the user to input their e-mail and password and if it's inputted correctly grants them access to the user site. This page also contains fields such as "Sign Up" for users that have not registered yet and "Forgot Username / Password" for those users who forgot their credentials, but these nuances have been omitted from the high-level BPMN diagram for simplicity.

The Join Us page, whether accessed through the menu bar or output of function #1/2, provides the user with several fields to fill out. After filling out the initial fields, the user is taken to the user site to complete their registration. Here they are required to go through our detailed questionnaire which determines their return goals and risk tolerance level. From here, Alpha Factory recommends a portfolio for the user to hold based on their return goals and risk tolerance level, fulfilling the third, and final project requirement. Next the user either accepts the recommended portfolio or overrides it with their own choice. Having selected the portfolio, the user has completed registration and is granted access to the main landing page of the user site.

The Home Page of the user site, which is either accessed immediately after logging in or completing the questionnaire, provides the user with information about their portfolio and its performance. From the Home Page the user can access the About Page (which contains the same information as the About Page from the main site) and the Contact Page (which only includes the Subject and Message parts of the form to fill out, as the Name and Email are already known from the user's login credentials). The user can also access their Profile Page where they can see their risk tolerance level and repeat the questionnaire / request to hold another portfolio. Having completed the session, the user can log out where there are taken back to the landing page of the main site (About Page).

New to the user site is the Advisor page which provides the user access to three major functions, as seen in Figure 5. First, it allows the user to select another portfolio, by clicking change your portfolio. After selecting this option, the user is taken to a page showing them the portfolio they already hold, along with the other 4 options and asks them to select which portfolio they would like to hold prior to continuing. Next, we allow users to repeat the risk-tolerance level questionnaire. We understand that risk-tolerance levels and investment goals change over time (e.g. new jobs, starting families, planning for retirement, etc. can all elicit these changes) and feel it is necessary to allow users to understand how their risk-tolerance levels are changing. After selecting this option, the user is taken to our risk and return assessment question as completed when they initially joined our platform (currently we only have one set of 15 questions, but our next steps would be to include several varying sets of questions such that users are not getting the same questions with each repetition of the questionnaire). Finally, we provide users with the option to schedule a meeting with one of our advisors. By selecting this option, an automatically generated email is sent to our Gmail account and the user is notified that we will reach out to them shortly.

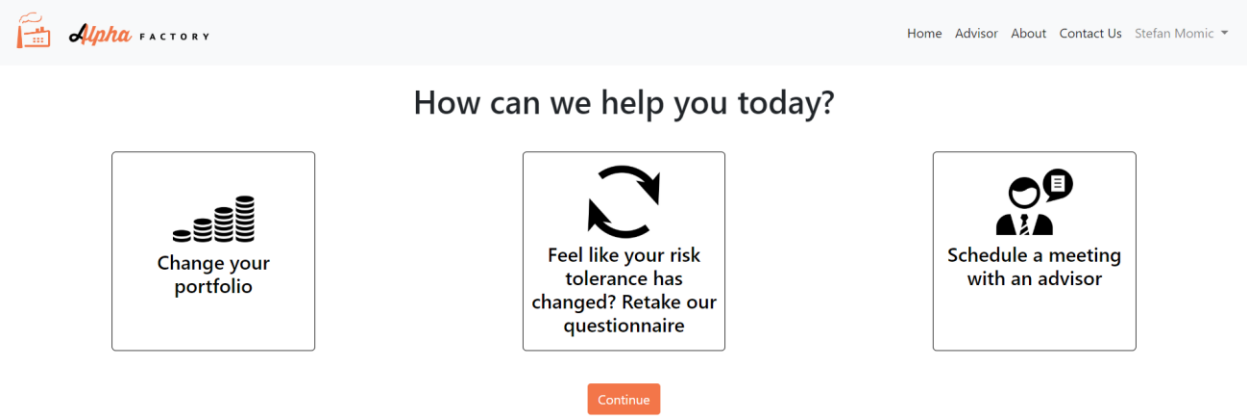


Figure 5: Advisor Page

Finalized User Interface and Wireframes

This section aims to display the finalized wireframes of the main pages required for our robo-advisory web application. Additionally, pages that further supplement the user experience and descriptions of their use can be found in Appendix B. Finally, the original flow of the website can be found in Appendix C where the number of changes that were made through user feedback and project revision are apparent.

While developing this subsystem we focused on the user experience and look/feel of the finalized product. In doing so, we made sure that our design met the project requirements by delivering on the three basic functions required. Specifically, as per the project requirements, our robo-advisor was required to have the following three basic functions:

1. Given a portfolio of assets (i.e. specification of the securities and its weights), the wizard will analyze the risk and return profile over various horizons by backtesting over real data
2. Given a portfolio to be held for some time horizon, the investment wizard will seek to find a portfolio of assets that dominates it (e.g. has a better Sharpe-Ratio using a reasonably large class

of assets to draw from (the user can set the number of assets k to include in the portfolio from a universe of n assets where n>>k))

3. Given a return goal over some time horizon, generate a portfolio that has a projected return of at least the goal amount, but with minimum risk (e.g. portfolio variance, min VaR or CVaR over the time horizon)

With the Design Team's goal to create a user-friendly product that can be adopted and sold to potential consumers, we decided to focus on function #3 as the main function of our website. That is, we decided that the crux of our platform will be creating customized portfolios for our clients based on their return goals and risk-tolerance levels. We also decided to incorporate functions #1 and #2 as promotional materials for our robo-advisor platform. Specifically, we framed function #2 as a way for users to input their portfolios and be shown how much better they could be doing by joining Alpha Factory. A detailed walkthrough of our web design, along with the design decisions that were made is provided below.

First, we decided to split our web application into two main parts – the part that is available to the public (referred to as the "main site") and the part that becomes available after logging in (referred to as the "user site"). You can identify the main site and user site by the menu bar located on the top of the page – please refer to Figures 6 and 7 below.

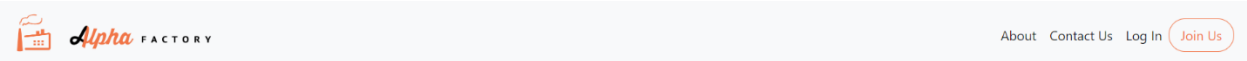


Figure 6: Main site menu bar

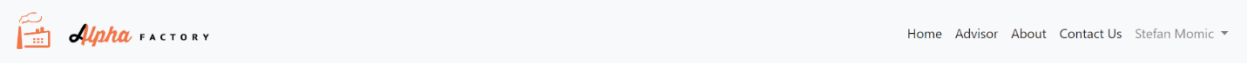


Figure 7: User site menu bar

As seen from Figure 6, visitors have access to various pages from the main site including the About page where we detail our portfolio generation strategies, asset universe selection and several other key design decisions; the Contact Us page where we allow visitors to email us any questions; and Log In / Join Us pages that allow visitors to register to our platform and access the user site.

Our main site focuses on completing functions #1 and #2 from the requirements. We decided to provide these functions to all visitors as a way of promoting and validating our platform with the end goal of encouraging more members to join our service after seeing how well it performs. After logging into our service, users get access to function #3.

When accessing our URL, visitors are taken to our main site landing page, as shown in Figure 8 below. This landing page is simply a part of our About page where the visitors are provided a brief overview of our objectives and goals.

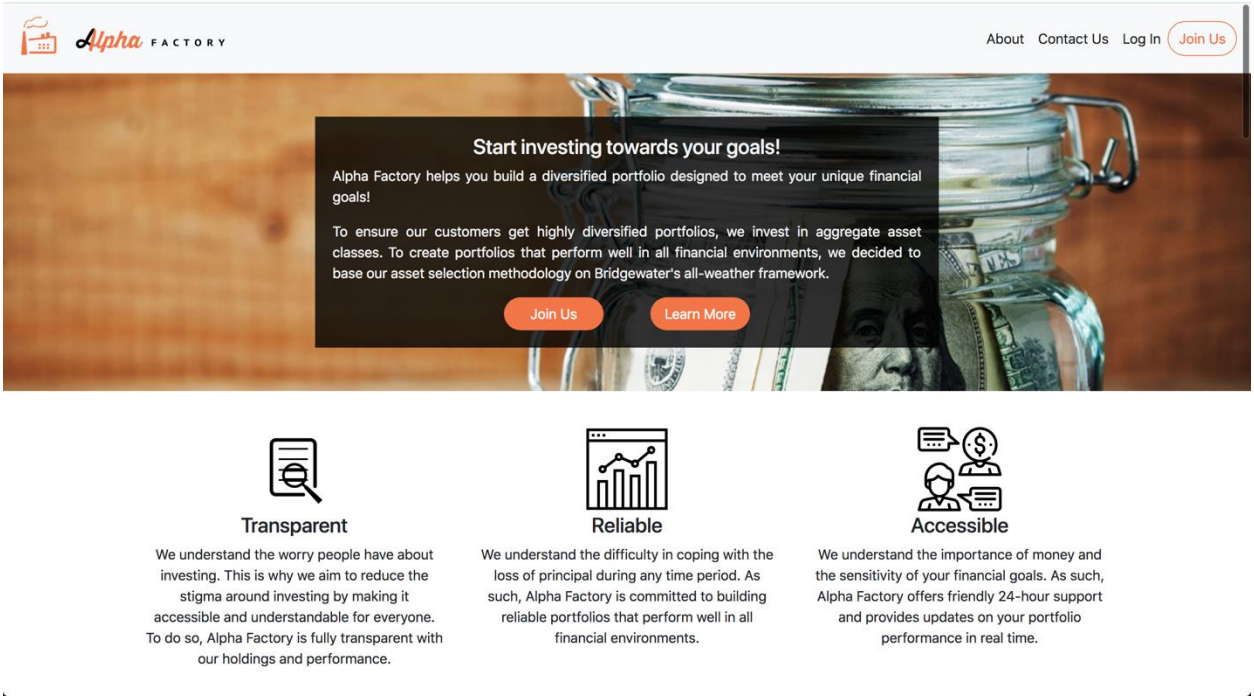


Figure 8: About Landing Page

Scrolling along this About page, we provide visitors the ability to access functions #1 and #2 (as defined in the project requirements). Specifically, they are given the list of securities in our asset universe and the ability to enter the amount of dollars they plan to invest in each asset they choose to hold, as shown in Figures 9 and 10 below. To help make this process easier we decided to give customers a simplified and detailed view (Figures 9 and 10 respectively).

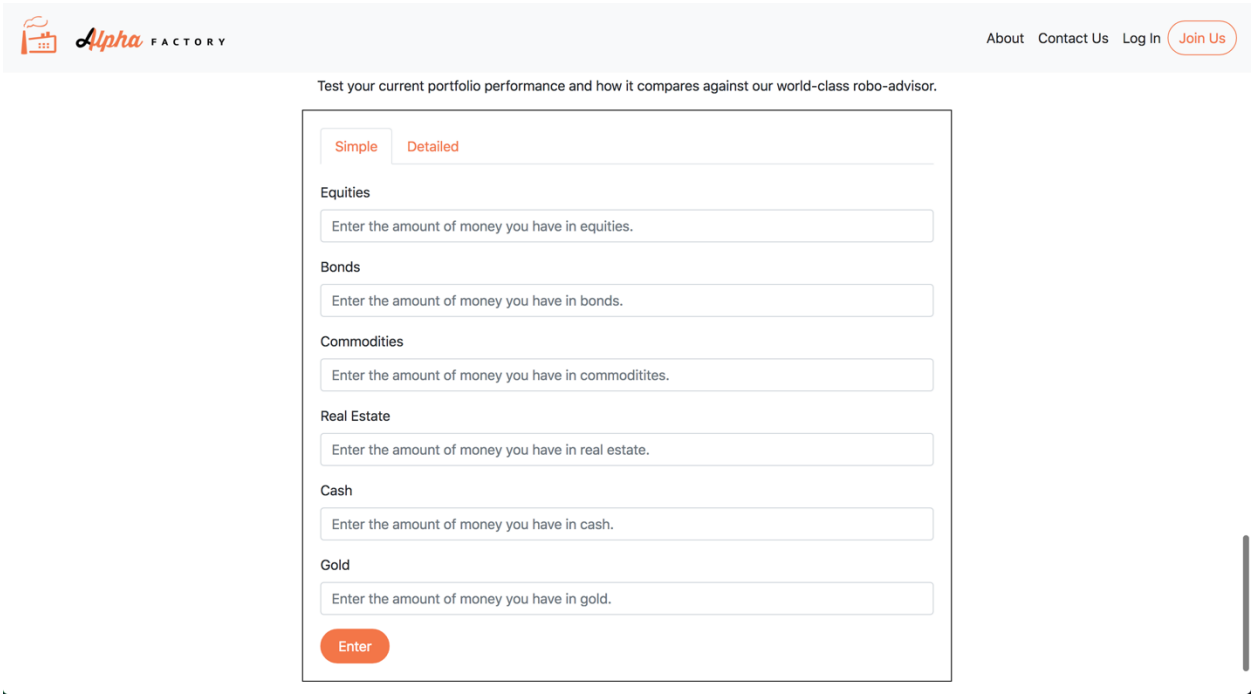


Figure 9: Simple inputs for Function #2

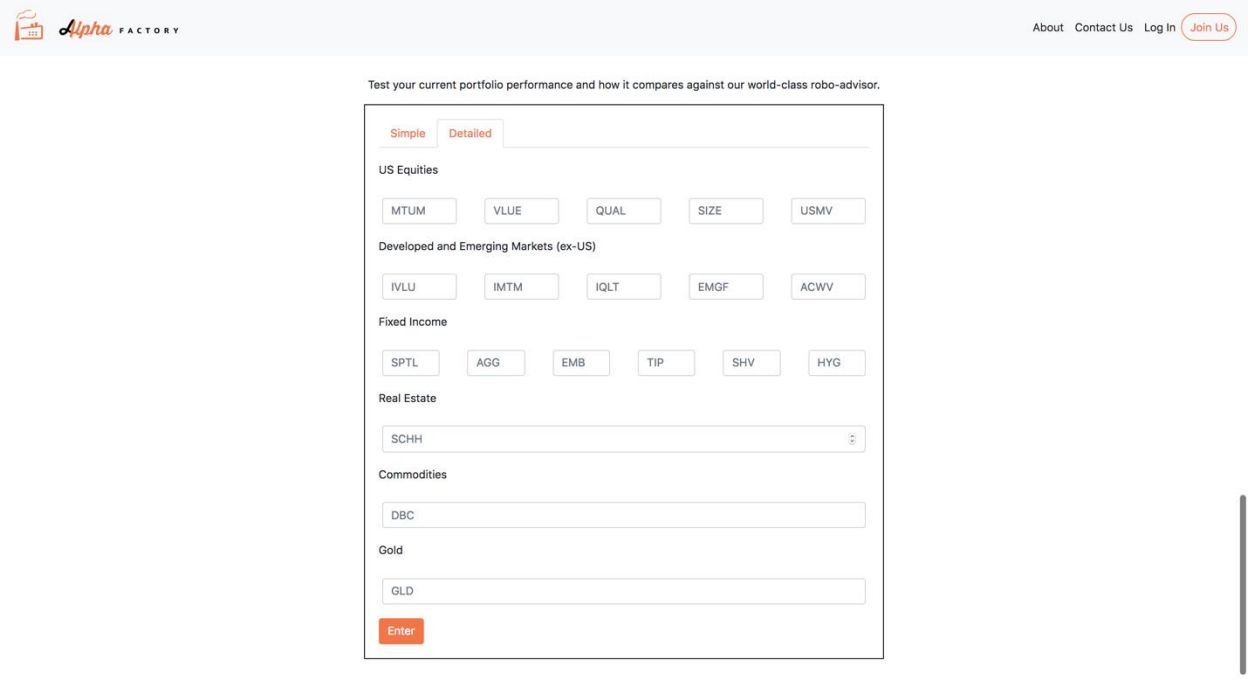


Figure 10: Detailed inputs for Function #2

In the simplified view, users are only asked to allocate their money among the various asset classes we invest in, while in the detailed view they are able to allocate money in the specific ETFs in our asset universe. This design was put in place as a result of us wanting to tailor fit our approach to the individualized investor. However, as mentioned previously, our asset universe is composed of ETFs. As such, users will only be able to input portfolios of our ETFs and we will create better portfolios using those same assets. As we are primarily targeting users who do not have enough capital and/or sophistication to invest in individual stocks, we did not feel the need to consider allowing our users the ability to input portfolios of individual securities. We understand that while many people aren't advanced in the subject of investing, there will still be experienced customers looking to give our portfolio a chance. With the detailed input feature in place, they can be rest assured that their portfolio has been thoroughly considered in its entirety when compared to our portfolios.

It is noted that we choose to allow visitors to input the dollar amount they wish to invest in each asset class as we find dollar amounts to be more intuitive for our users – we will then compute the weights invested in each asset class while doing the calculations and computations. After entering the amount of money they plan to invest in each asset class and clicking "Enter", our robo-advisor analyzes the risk and return profile of the portfolio they entered over various time horizons by backtesting over real data (fulfilling function #1) and finds a portfolio of assets that dominates the one they just entered (fulfilling function #2). The users are then displayed the results in the same area, as shown in Figure 11 below. They are also given the option to click the colored rectangular legend in order to filter out either graph (further method displaying function 1). Ultimately, by doing this, we demonstrate how much better they could be doing by using our platform (assuming the portfolio they entered is the one they currently hold or would hold if they had to invest on their own) and encourage them to join our platform using the button prompt seen in Figure 11. They are also given the ability to view more stats if they are not yet fully convinced.

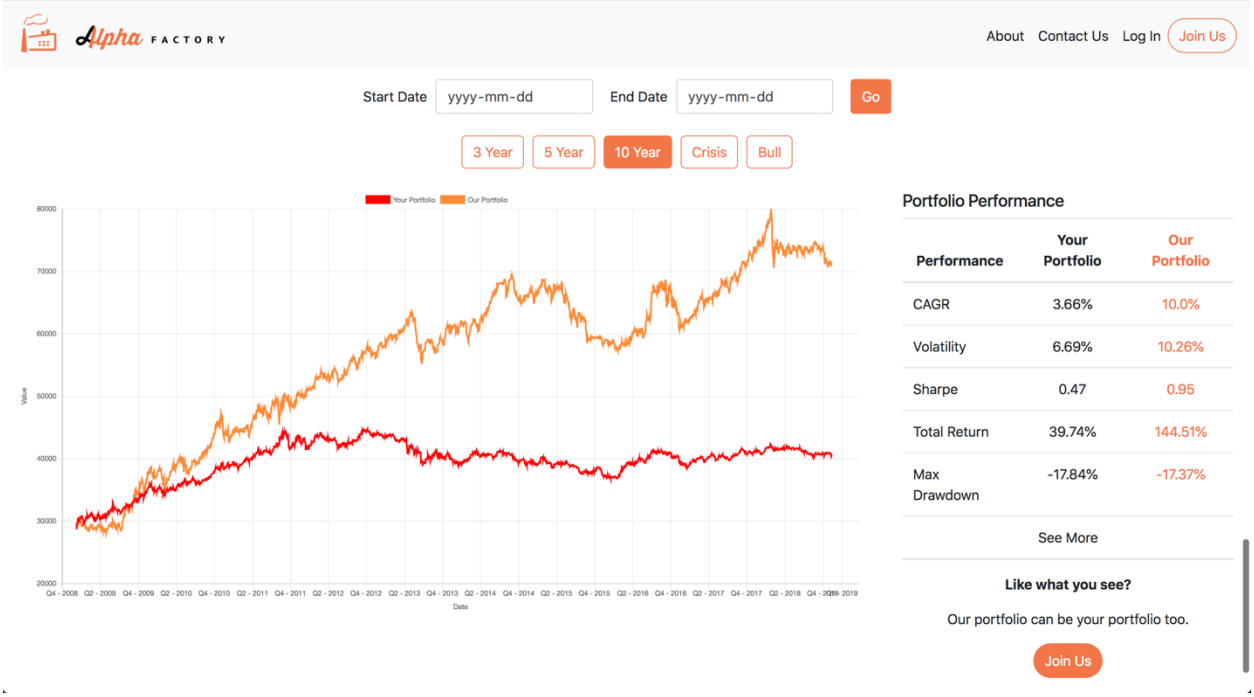


Figure 11: Outputs of Function #2 (and #1) for a selected time horizon (5 years in this case)

Having demonstrated functions #1 and #2 from the main site, our user site focuses on providing function #3 (while also incorporating function #1). After a user joins our platform and creates an account by filling out the required sign-up fields (shown in Figure 12 below), they are taken to a questionnaire aimed at assessing their return goals and risk-tolerance level over a specified time horizon, as shown in Figures 13 and 14 below.

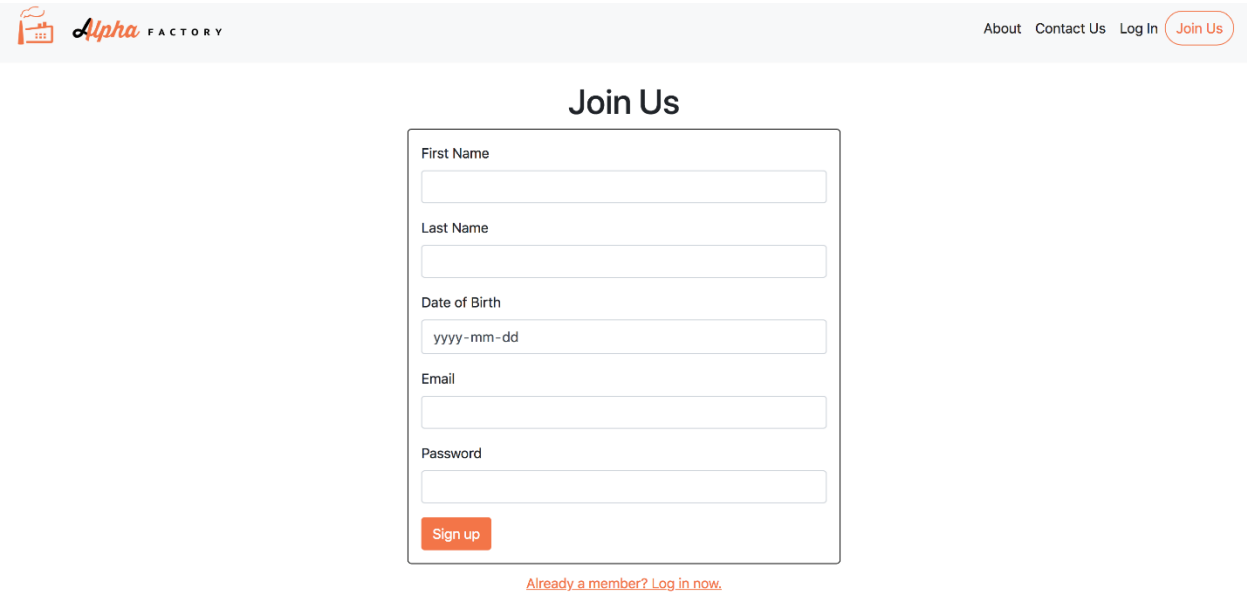


Figure 12: New user Sign Up Page

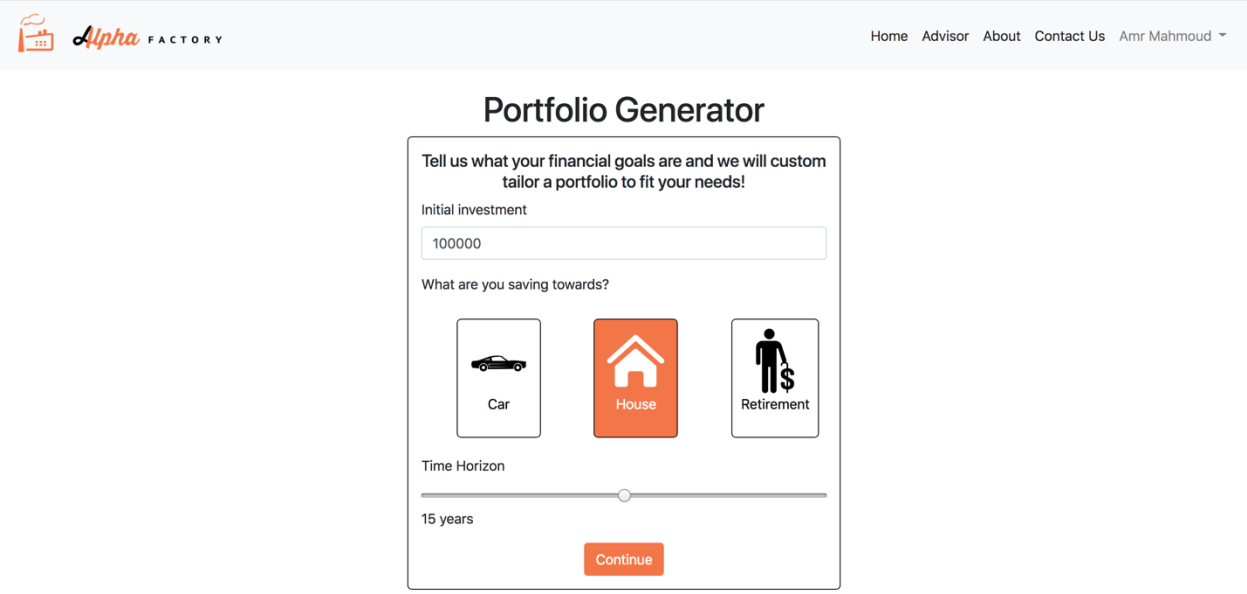


Figure 13: Advisor Page with Function #3 Inputs

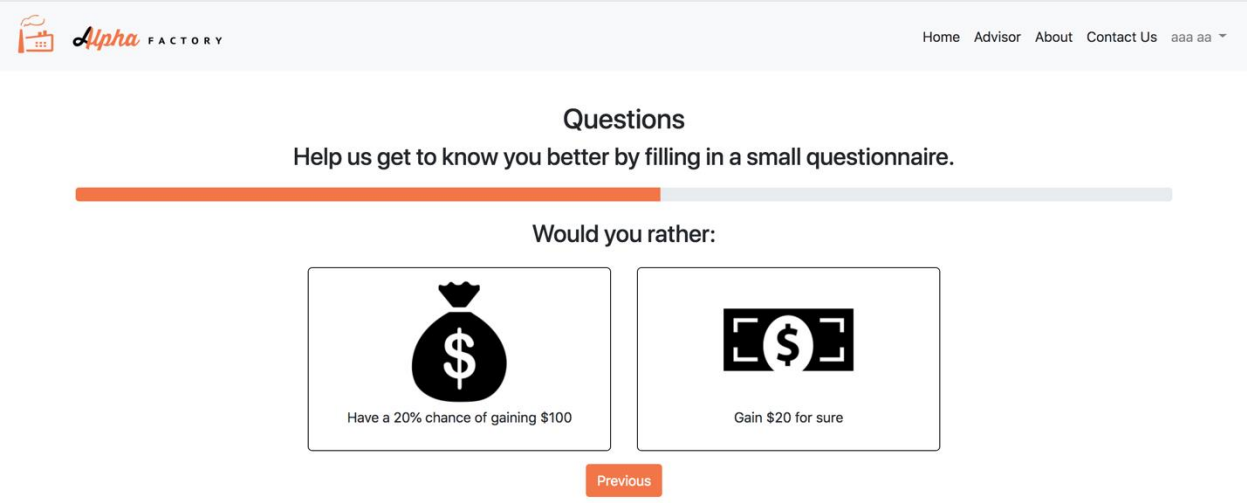


Figure 14: Risk-Tolerance Assessment Questionnaire

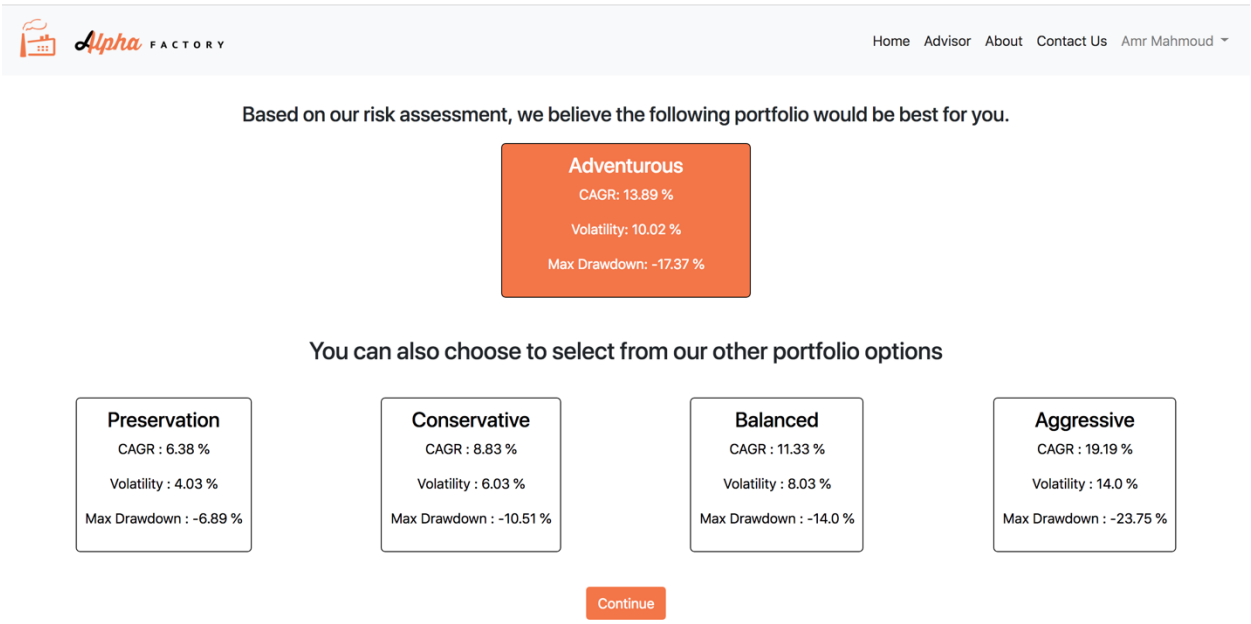
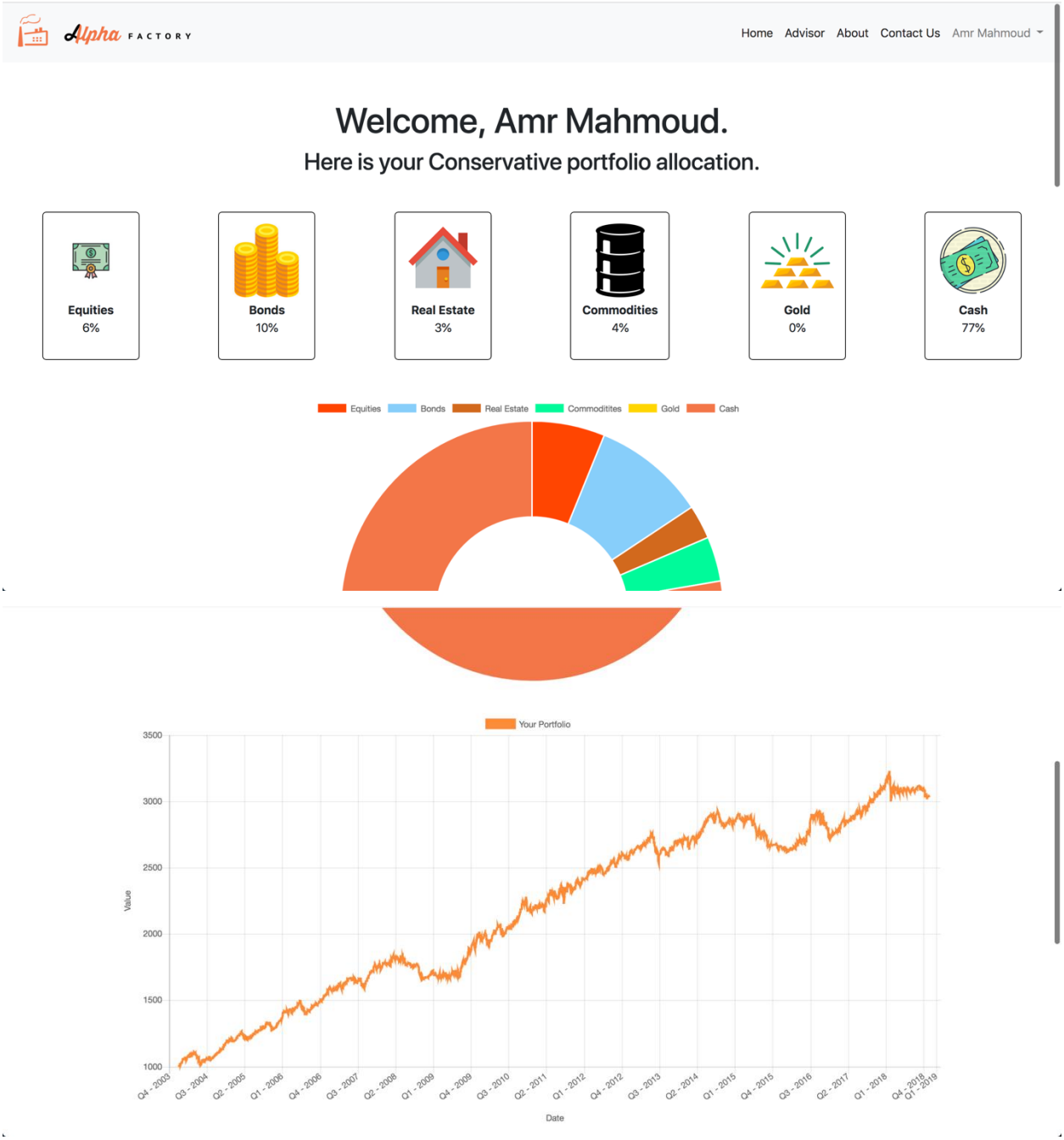


Figure 15: Portfolio Selection (final step of portfolio creation process)

Using the information provided from the questions above we suggest the optimal portfolio for the user, fulfilling function #3. We also demonstrate function #1 during this stage as well by analyzing the risk and return profile of the portfolio we created over various horizons by backtesting over real data. The user is then given the option to select between the recommended portfolio or our other portfolio options through Figure 15. The selected portfolio along with its performance is displayed on the Home page of the user site, as shown in Figure 16 (split into 3 images as length of page is constraining the full view). The home page has been set to a default 15-year horizon for display and demo purposes but would implement the unique user portfolio holding time period instead in actual production. We also provide more in-depth statistics and analysis on the user selected portfolio through the tearsheet pdf which they are able to download from the 'Download the full report here' link.


Alpha FACTORY
Home
Advisor
About
Contact Us
Amr Mahmoud
Welcome, Amr Mahmoud.
Here is your Conservative portfolio allocation.
Equities
6%
Bonds
10%
Real Estate
3%
Commodities
4%
Gold
0%
Cash
77%
Equities
Bonds
Real Estate
Commoditites
Gold
Cash
Your Portfolio
Value
Date

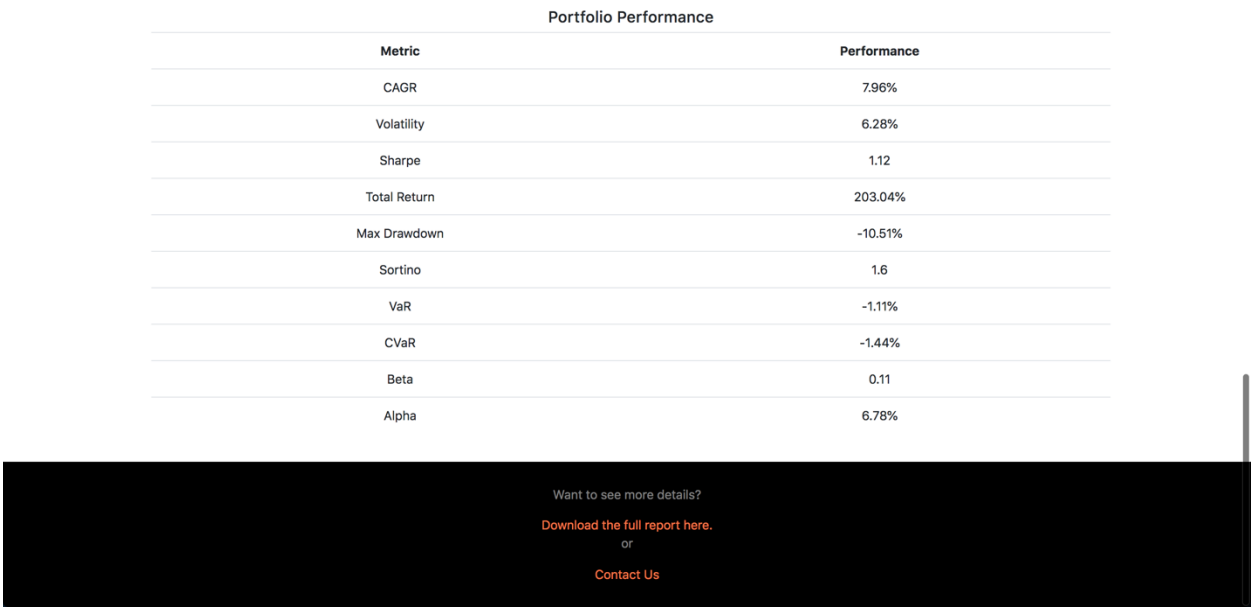


Figure 16: Home page with information and statistics on portfolio

From the advisor options page found in Figure 17, users are able to select between three options; change portfolio type, change risk tolerance (and create new portfolio), or schedule a meeting with an advisor in person.

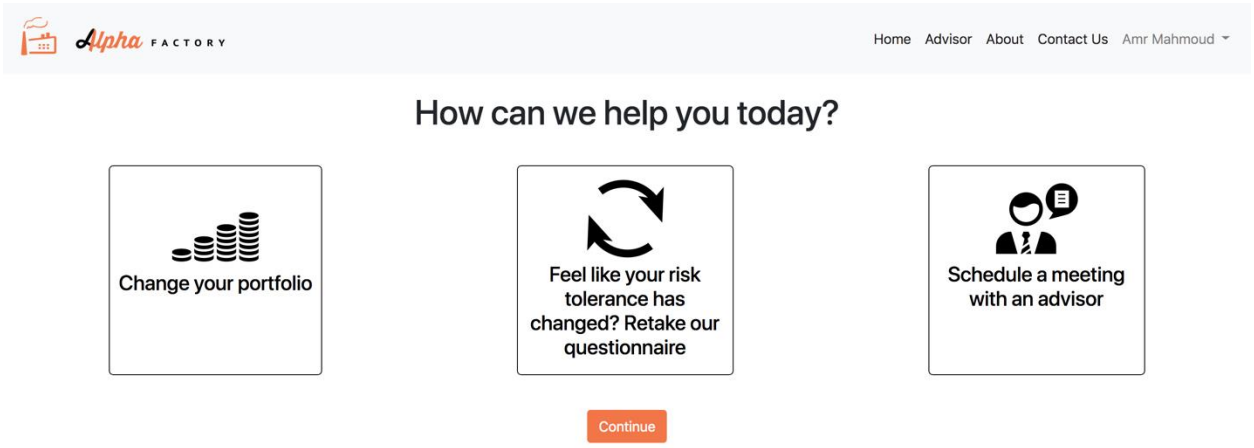


Figure 17: Advisor options page with three choices to select from

The advisor options page is meant for completeness as a method to allow users to manage their portfolio currently by either changing the type or creating a new one. Selecting the first option (left) will lead the user to the selection page in Figure 15 and give them the ability to select a new portfolio type. The second option will lock the users into a new portfolio generation process where they will begin again from the advisor page found in Figure 13 until they either complete the process or log out. The third option will directly send us an email to our registered Gmail account requesting for a meeting with the user and will display the necessary user info. The user is then taken to the completed contact us page informing them

that we have received the message and we will be in touch shortly. This completes the full circle of portfolio creation, editing, and ignores deletion as that would require an account deletion which the user can request manually through the contact us method.

Please refer to Appendix B for wireframes of additional pages that have not been mentioned here (e.g. Contact Us, Profile, Log In, etc.). These pages have been omitted from the main body of the report as they are not central to the business model and flow, but have been included in the web application design as they are integral for the user experience nonetheless. Additionally, note that we focused the discussion of our wireframes to the business flow and user experience as we discuss the detailed design decisions in subsequent subsections. For example, we simply show the risk-tolerance assessment questionnaire here as we justify the choice of only having two options and the design features (e.g. being automatically taken to the next question after selecting the answer, inclusion of a progress bar at the top, etc.) in other sections of the report.

Usability Testing Protocol

Alpha Factory also developed an extensive usability testing protocol to ensure that their robo-advisor platform was properly tested prior to its official release. The purpose of the usability testing protocol was to document what they were going to do, how they will conduct the tests, the metrics they will capture, what scenarios will be used and the number of participants to surveyed. The elements of the testing protocol include the scope, purpose, participants, sessions, scenarios and metrics, with each element being further discussed below.

Scope

The scope of our usability test includes testing both the web application for bugs and the general user experience and sentiment. As such, we conducted extensive debugging of our web application along with a survey of participants to gauge their feel towards the web application. Specifically, we wanted to ensure that our web application was easy and intuitive to use, the risk questionnaire is sufficiently accurate, the user is likely to use our platform again, and that there are few, if any, web application bugs/glitches.

Purpose

The purpose of our testing protocol is to ensure that users can navigate the web application and gauge how they feel about the web application. We were ultimately striving for a very simple and user-friendly design, hence making most of our design decisions with that in mind (e.g. we wanted to ensure that our risk questionnaire was short and easy to fill out ultimately deciding to limit the number of questions to 15 and only allowing two options for each question). At the same time, we wanted to make sure that our web application produced accurate results and correctly identified users return goals and risk tolerance levels.

Participants

Alpha Factory tested it's application by surveying a random sample of 25 University of Toronto Engineering Finance Association (UTEFA) members. UTEFA members were selected as the participants as they are largely composed of undergraduate engineering students who are interested in finance. As such, it was believed that their feedback will be extremely valuable for future iterations and improvements as they can comment on both the computer programming aspect, through their engineering background, and the investment aspect, through their background/interest in finance.

Alpha Factory also surveyed a random sample of 24 students in Sidney Smith hoping to obtain feedback from students who have an unknown level of financial and computer literacy. With the belief that these survey participants are representative of the general population, we mainly focused on whether or not these students find our platform easy and intuitive to use.

Sessions

Survey sessions were conducted within a 14-minute time limit. This decision was made for two reasons. First, we wanted to be respectful and conscious of our participants' time and wanted to ensure our surveys intrude as little as possible. Second, the ease of use design factor was one we greatly valued and focused on while making our design decisions. As such, we believe that our users should be able to set up an account in 10 minutes or less! To ensure this is the case, we limited survey sessions to 14 minutes, with a maximum of 12 minutes allocated to participants playing around with our demo. Doing so ultimately allowed us to gauge the web application usability and engagement.

Scenarios

Apart from the unique scenarios our randomly selected survey participants generate, the full list of all-encompassing functionality testing scenarios can be found in Appendix D. It is also important to note that while the participants were being observed during their survey, their particular flow through the website was not recorded at every interaction. This was because the first few participants found no difficulty in navigating the site and testing the main functionalities of the web app and we therefore deemed the study of unique user flow unnecessary. Instead, we decided to narrow our focus down to the pages/functions that participants did get stuck on or weren't completely satisfied with.

Metrics

As mentioned previously, our testing protocol aims to test three main things.

First, we hope to ensure that the web application is functional without any bugs. As such monitored our tests for any bugs that arise during our surveys and/or scenario test cases and worked to rectify any and all issues prior to the December 3rd deadline.

Second, we wanted to make sure that our web application is user-friendly and intuitive/easy to use. To do this, we looked at the length of time it took our users to set up. If a large majority of the users (more than 80%) are able to set up their account and obtain a customized portfolio in under 10 minutes, we can conclude that we met our ease of use target relatively well. We also surveyed session participants asking for their thoughts on the user experience and general feel of the web application. Questions asked included:

- Ranking their user experience on a scale of 1 – 10 with 10 being preferred
- Ranking the ease of understanding questions on a scale of 1 – 10 with 10 being preferred
- Ranking the intrusiveness of the questions on a scale of 1 – 10 with 1 being preferred
- Ranking the likelihood of using the service on a scale of 1 – 10 with 10 being preferred

By cumulating all of the survey results we will conclude on the general user sentiment of our web application and attempted to improve on the user feel based on the feedback received.

Finally, we hoped to test the accuracy of our risk-tolerance questionnaire and portfolio recommendation strategy. To do so, we proxied how accurate our risk-tolerance level assessment is by seeing how accurately it predicts the surveyed individuals' self-identified risk-tolerance level. To do this, we asked participants to select a risk-tolerance level from one of the five discrete options prior to filling out the risk questionnaire. We then gauged the accuracy of our risk questionnaire by determining whether or not our platform was able to accurately predict our user's self-identified risk tolerance level.

Note, that we will continuously gauge the accuracy of our risk questionnaire even after releasing the platform. To do this, we will assess the percentage of users who accepted the recommended portfolio. For example, if a user is identified to be risk-neutral they will be recommended to hold our Balanced portfolio. If this user accepts the Balanced portfolio, we will interpret this as accurately predicting their risk-tolerance level. However, if the user chooses to invest in another portfolio, we will interpret this as not being an accurate prediction of the user's risk-tolerance level and store their risk-tolerance level and portfolio of choice in our database. If we observe a trend of a select score favouring a particular portfolio, we will modify our risk-tolerance level assessment logic to start classifying users accordingly.

Usability Testing Protocol – Surveyed Results

Alpha Factory surveyed participants from November 14th to 28th. Surveys were conducted in person by having selected participants fill out our Google Sheet on our computer. All of the tests were administered by Stefan Momic and he was available to clarify any questions they had with the Google Survey, but provided no help by answering any questions relating to the Alpha Factory platform. Stefan also timed the time it took for users to set up their accounts on the platform, limiting tests to 12 minutes.

The link to the Google Sheet is provided below:
https://docs.google.com/forms/d/e/1FAIpQLSe2EqssvdTkOBIUz579WXVo7KipsuSZsabCLB-3t87Xp0FRng/viewform?usp=sf_link

Survey Procedure

After engaging prospected participants, Stefan ensured that the participants filled out their perceived risk tolerance level as seen in Figure 18 below.

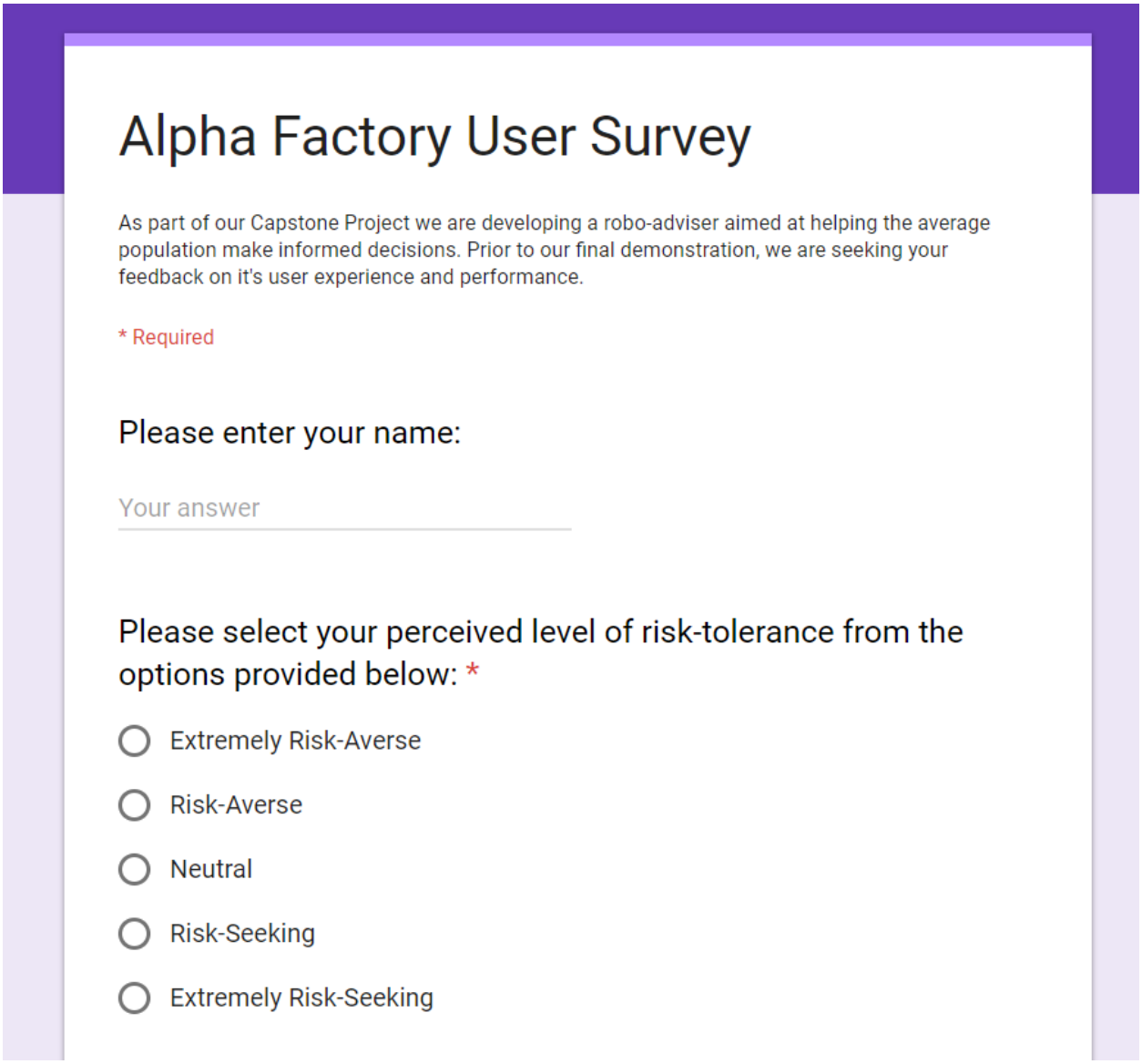


Figure 18: Preliminary survey questions

Having completed the preliminary questions, participants were next prompted to play around with our web application and encouraged to create an account by filling out the questionnaire, as show in Figure 19. Stefan observed their interaction with the platform, noting any bugs or frustrations the users seemed to have. During this time, he used the timer on his phone to time the amount of time it took users to get set-up but provided no other assistance in navigating the users through the platform.

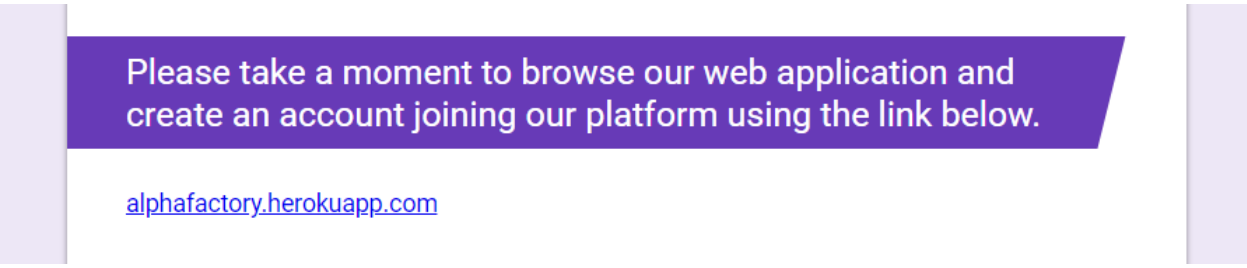


Figure 19: Prompt to visit our web application

After successfully completing the risk questionnaire and obtaining their risk-tolerance level and portfolio, the timer stopped, and the results were noted. Given that none of the participants took more than 12 minutes to register, Stefan never had to cut off the test prematurely (which would signify ultimate failure in the goal of a quick and easy to use platform).

Next, with the help of Stefan, the results of the of the test were noted. Specifically, the time it took and whether or not the platform accurately identified the risk-tolerance of the participant was noted as shown in Figure 20.

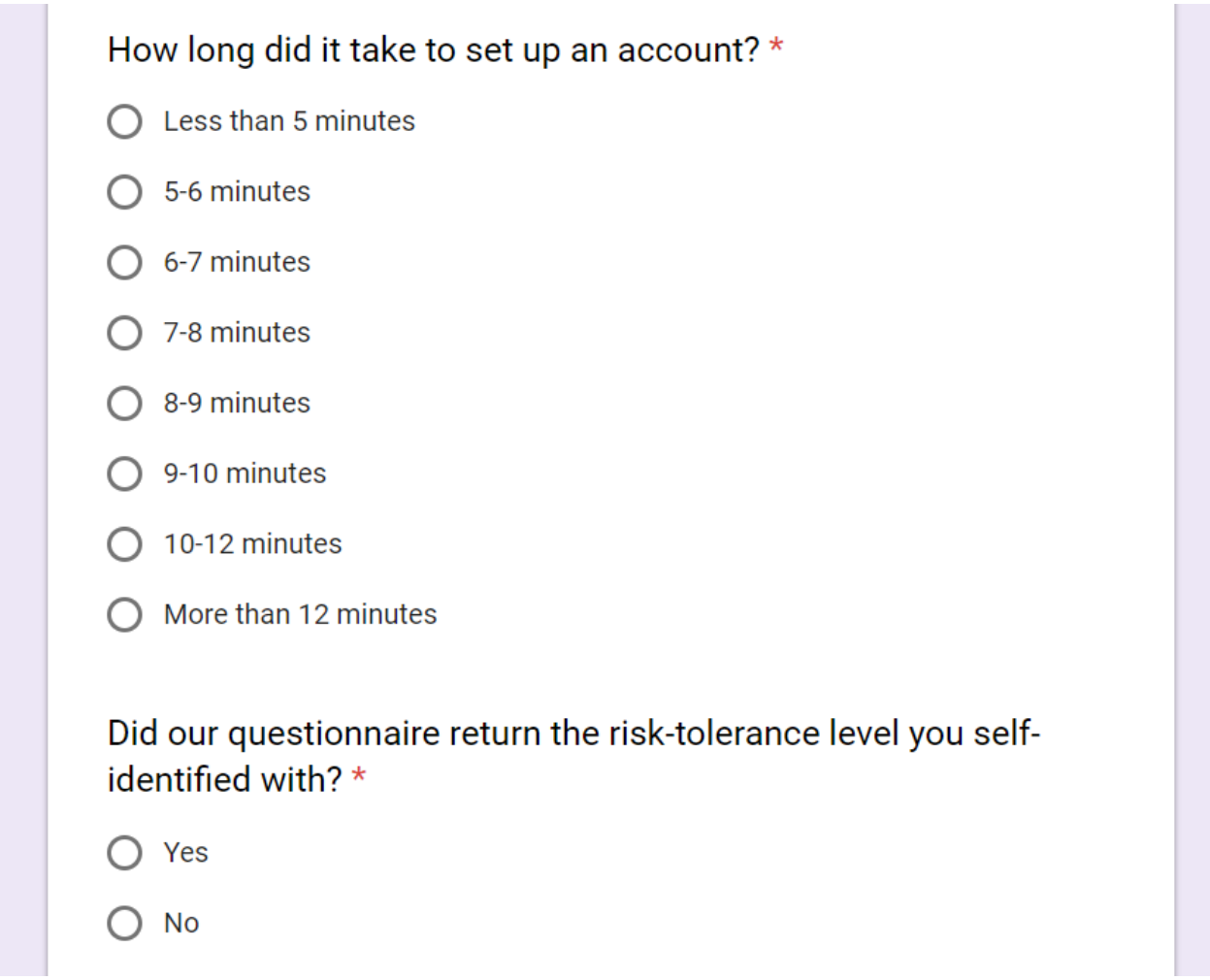


Figure 20: Results of the application test

After inputting the results of the application, users were provided with a set of 4 questions to assess their reaction to the platform, as shown in Figure 21. Upon responding to the questions below, the participants were prompted to submit their responses by clicking the “Submit” button.

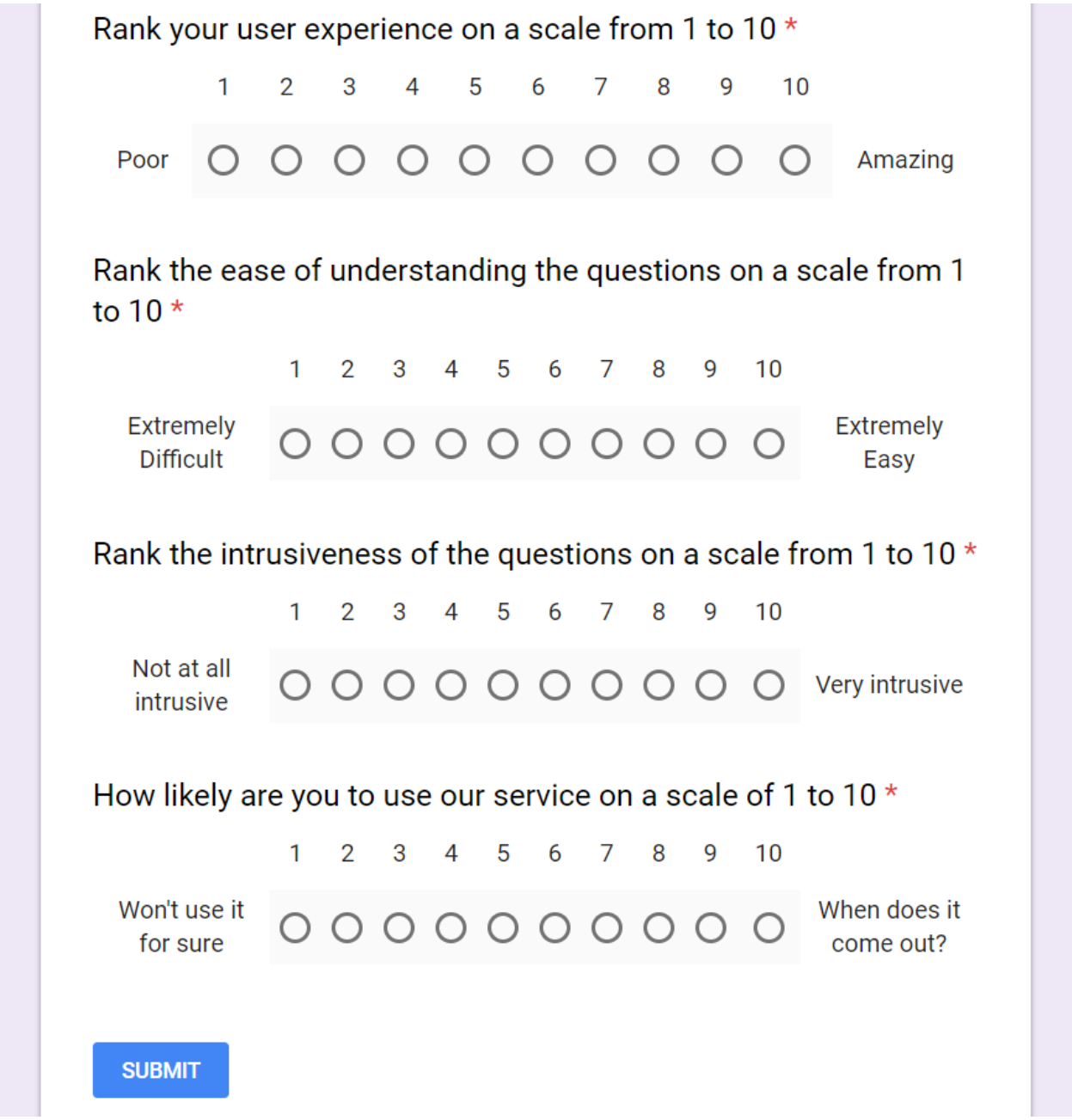


Figure 21: User-experience survey

Finally, Stefan thanked the users for participating and asked for any feedback and/or suggestions they would like to see improvement on. While few participants provided any feedback, the few recommendations that were made were noted and relayed back to the rest of the team who then tried incorporating the feedback that was received.

Survey Results

As mentioned previously, our testing protocol aims to test three main things:

1. Ensure the functionality of the web application (indicated by users being able to effortlessly breeze through the demonstration and having a good user experience while doing so)
2. Ensure that the web application is user-friendly (indicated by positive participant feedback and reviews)
3. Ensure the risk-tolerance questionnaire returned accurate results (indicated by participants obtaining the risk-tolerance level they self-identify with)

Ultimately, based on the 49 participants we surveyed, the results that came back were generally better than expected. For starters, as seen in Figure 22, the vast majority of the participants (almost 90%) managed to complete the demo in under 10 minutes, which was the amount of time we were striving for.

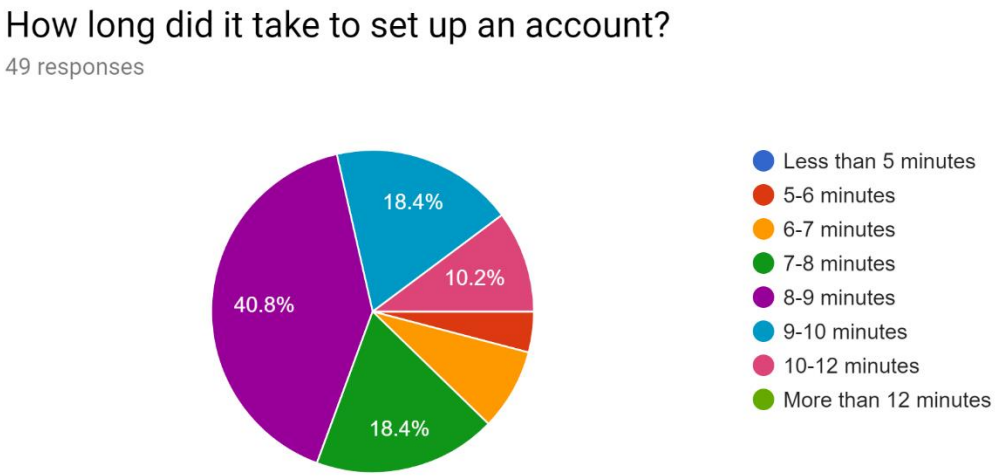


Figure 22: Responses to time question

With only 5 participants (or 10.2%) taking slightly longer than 10 minutes, we feel like our design decisions were validated as participants were able to breeze through the questions as intended.

With such a simple questionnaire, the accuracy of it was something that was of great worry and importance for us going in – what good is such a simple questionnaire if none of the participants feel at easy with the conclusions it made? Ultimately, this proved to be a needless worry as our risk-tolerance questionnaire seemed to produce the risk-tolerance level our participants self-identified with. Looking at Figure 23 below, it was evident that our questionnaire returned the risk-tolerance level our participants self-identified with close to 70% of the time. Further, it was observed that in the 15 cases where it failed to return the risk-tolerance level initially entered it was never wrong by more than one point. More than half of the "failed" cases resulted in identifying individuals who self-identified as neutral but were assessed to be either risk-averse or risk-seeking. The rest of the instances were of individuals who self-identified as "extremely risk-averse" but ended up getting "risk-averse" instead. Ultimately, given that our risk-

tolerance level questionnaire was never off by more than one, we are even more confident in the selection of our questionnaire method and back-end logic.

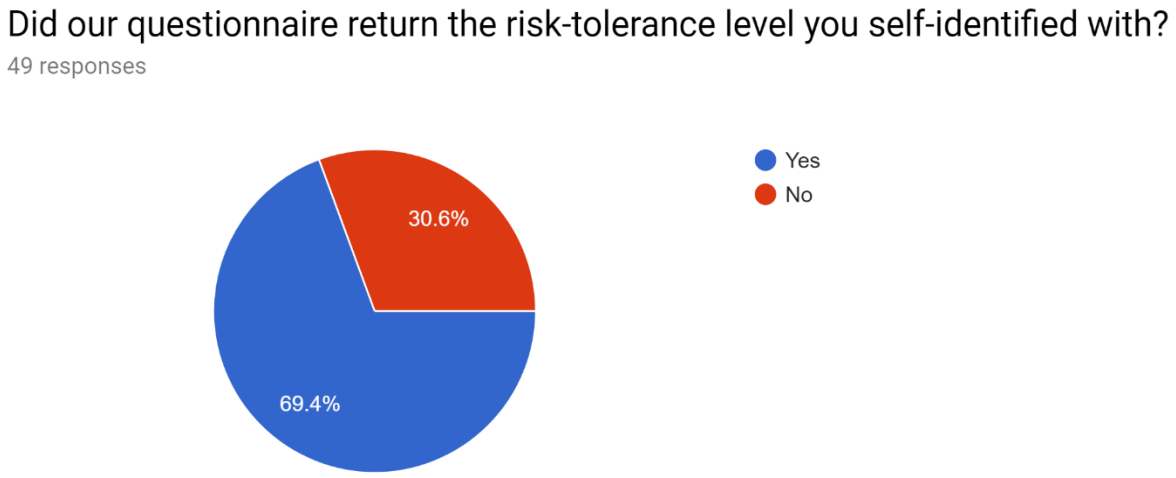


Figure 23: Risk-tolerance level questionnaire success rate

Finally, the survey of user-experience went very well, which was expected given that most users were able to breeze through the self-guided demonstration in under 10 minutes and obtained accurate results that were in-line with their self-identified views. Looking at the results of the 4 questions in Figures 24 to 27, it was noted that we were never on the opposite end of the spectrum.

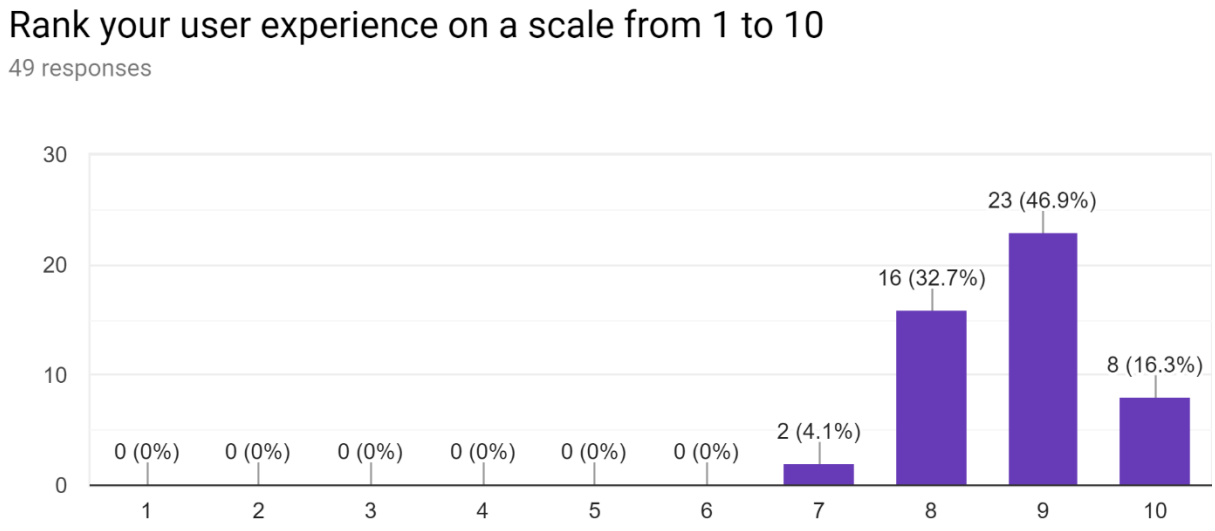


Figure 24: User experience ranking

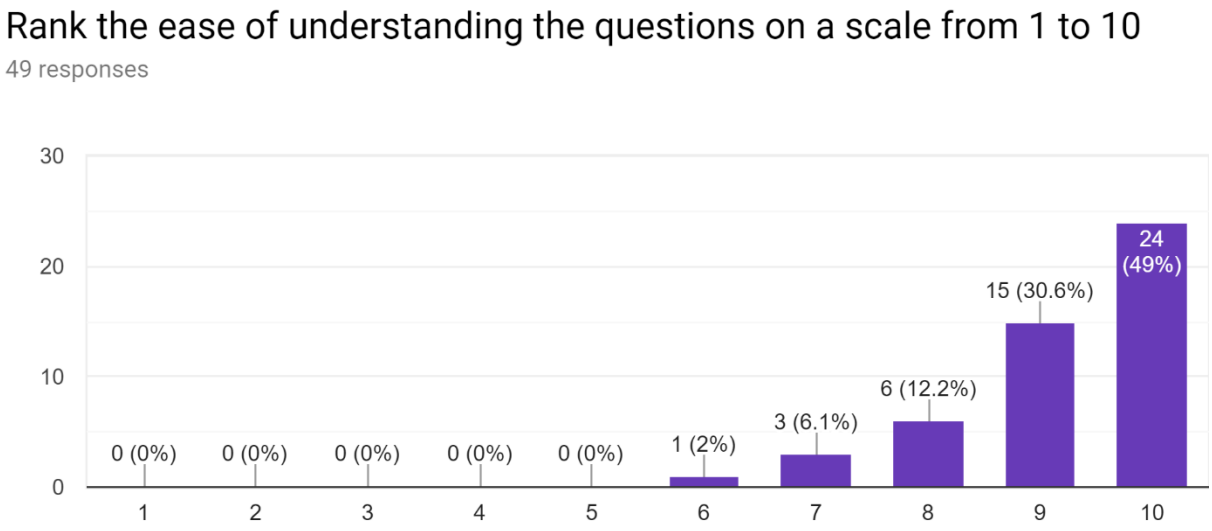


Figure 25: Question ranking

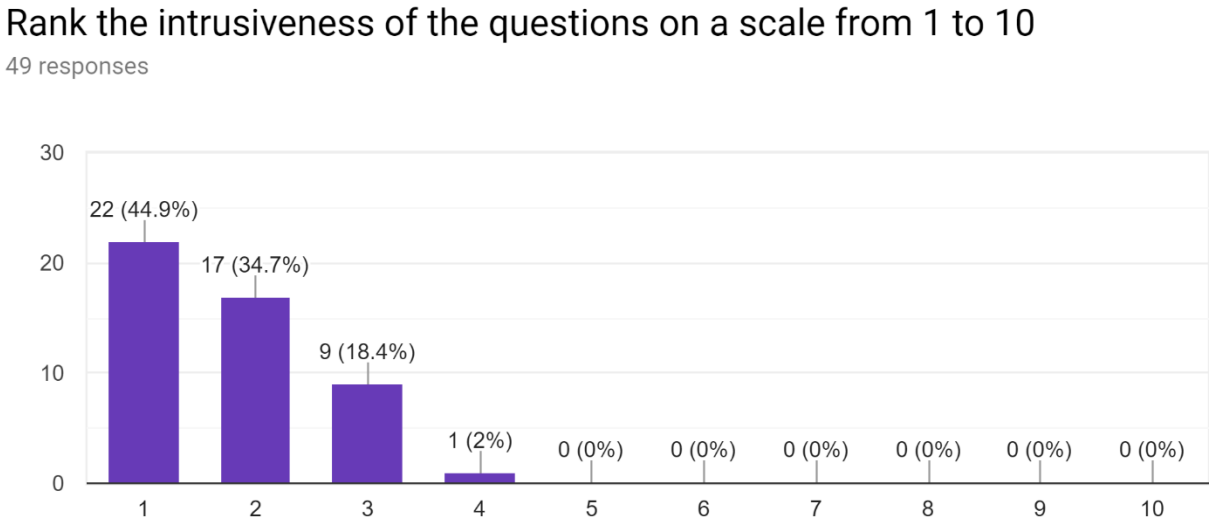


Figure 26: Intrusiveness ranking

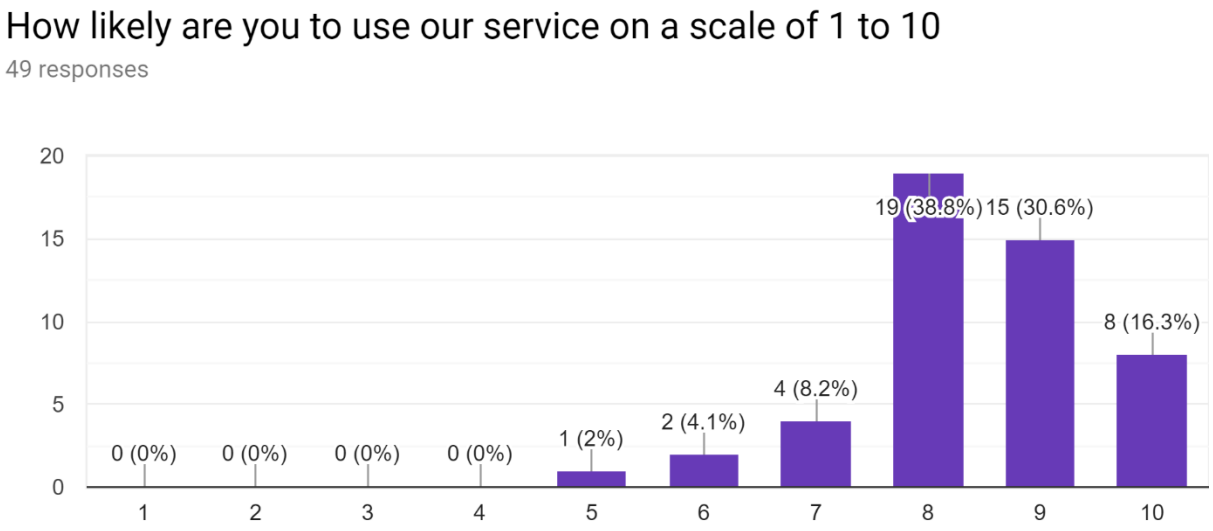


Figure 27: Likelihood to use our service ranking

Looking at Figure 24, it was noticed that more than 95% of participants rated their user-experience as being an 8 or greater, with no participants ranking our service less than 7. Moving to Figure 25, it was observed that almost half of our surveyed participants rated the questions as extremely easy to understand, giving it the highest possible rating available. It was noted that the user giving the lowest rating (still gave it a 6), was also the user that took the longest to complete the demonstration. Judging by the fact that this user also had the worst English out of the surveyed population it was quite likely that the language barrier resulted in the question difficulty more than the questions themselves. Offering our platform in different languages is something we can definitely consider as our next steps, especially if we feel certain ethnic demographics are a major unserved market for our platform. Looking at Figure 26, it was observed that majority of our surveyed participants (around 45%) thought our questions were not at all intrusive and ultimately selected the lowest possible rating. This was something we took as a great sign as we were able to accurately identify user risk-tolerance levels despite not asking them very personal questions. We believe if the questions were more personal and sensitive, users might not give an honest answer, but rather default to providing the answer they believe is the right answer. By asking questions that weren't intrusive we believe we were able to elicit honest responses, which ultimately led to more accurate results. Finally, looking at Figure 27 it was noted that more than 85% of our surveyed participants rated the likelihood of them using our platform as an 8 or higher. With no one responding with a rating below 5 as well, we obtained the validation we were looking for and are extremely satisfied with the user response.

Front End Technologies Implemented

In the following sections, we discuss the technologies that we decided to use for the final website which don't include, but still reference, the justifications from previous design reports. To find the original section regarding design decisions and their corresponding justifications, refer to the Technologies and Design Decisions section of Appendix C.

Gathering User Input(s) and Feedback

For gathering data, a standard form method was used on all pages with the exception of the questionnaire. Most forms were generated with client-side validation built into them to ensure that initial data is feasible and sufficient. A lack of data or invalid data will result in the user being unable to proceed or submit their input. Further data validation and confirmation will occur in the back-end where the process will also be running on valid data.

For the forms, a standard template was used in combination with the Bootstrap form component. This component was selected in particular due to its simplistic implementation, beautiful design, seamless feel and excellent user experience. It is extremely well documented, and the implementation is sufficiently simple such that we can use it in all our form-requiring pages. A powerful library that we leveraged in order to accomplish validation and form generation is WTForms. WTForms provides easy to use form templating and validation while integrating seamlessly with Bootstrap to generate beautiful and simple forms.

Display of Computed Data

For displaying the computed data, the popular JavaScript library, Chart.js, was selected. The decision to use Chart.js over alternatives such as Bokeh and Dash has paid off. The charts that Chart.js provides are more than sufficient for the needs of transmitting information to the customer in a clean, simple, and effective manner. While the original concern was that graph customization and interactivity is minimal, after actually implementing the library into our website, the results were clear and apparent that Chart.js was the right selection to get the job done. It was seamless to integrate with the other technologies we choose like Jinja and Flask and even easier to implement while still providing a large number of charting and customization options across the board. Examples of charts created using the Chart.js library can be seen directly from the about page show earlier in Figures 9 and/or 13. We quickly found that a particularly useful feature of this library is its ability to plot time-series. It effortlessly plots portfolio performance and makes the timescale extremely intuitive and adaptive, as seen by its ability to differentiate the time horizon and switch between years, months, or even quarters.

Technologies Used

Alpha Factory had to make decisions regarding which styling technologies and templating / front-end development technologies to use. Those specific decisions are further explored in the subsections below with the results of their implementations and the related concerns either being addressed or dismissed.

Styling Technologies

The usage of Bootstrap is, quite possibly, the most vital and fundamental technology design decision we made throughout the report. This is because Bootstrap has a vast number of usages ranging from forms, buttons, texts, navigation bars, and so much more. Set up of Bootstrap was as simple as including 4 lines of code into our base html file and using it limitlessly throughout the rest of the site. Examples and usage of Bootstrap is apparent all throughout the site as every button, form, and even the navigation bar is almost entirely comprised of Bootstrap components. It wasn't long before we found that Bootstrap was an extremely vital aspect of our project and we had to take extra measure to ensure it works across several platforms and even different browsers. In fact, to show the importance of this library to our website feel and functionality, we have included an image of what the homepage would look like if Bootstrap failed to load or was removed.

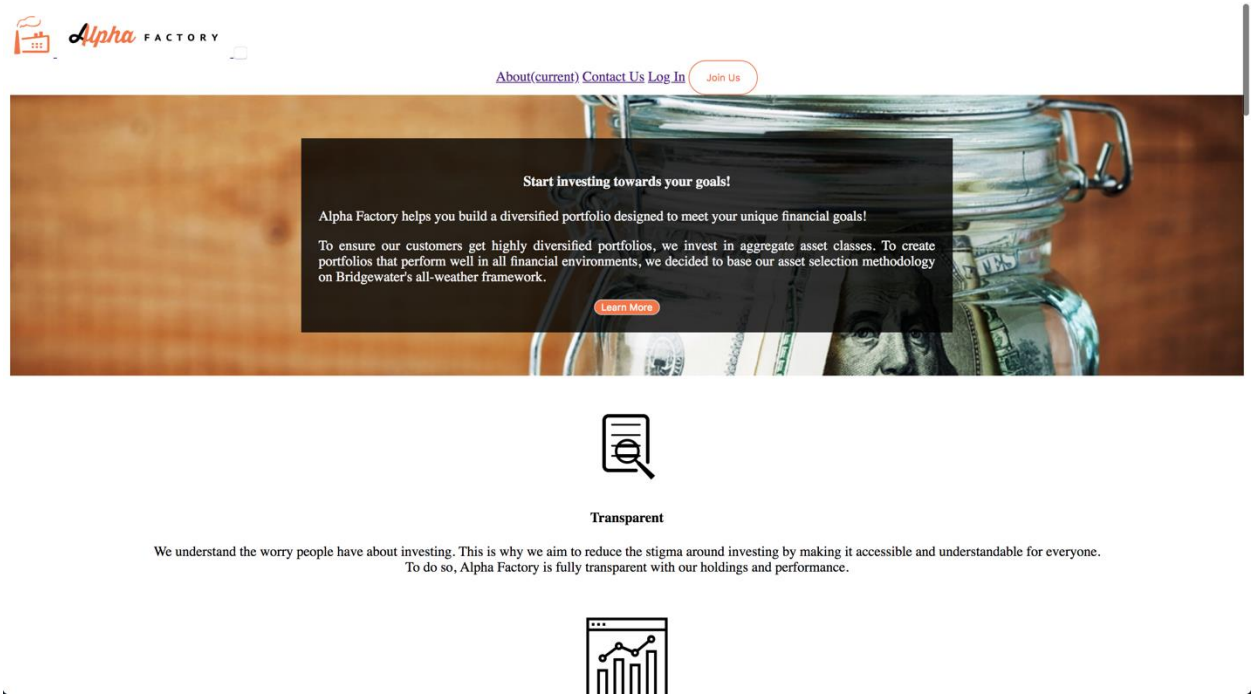


Figure 28: Alpha Factory without Bootstrap

As is apparent from Figure 28, the implications of such an event would be a nightmare for both Alpha Factory and all of its users. The great thing about Bootstrap is that it is so popular and widely adapted that the documentation for the library is impressive even by the greatest of standards. There were many instances where example implementations of functionalities extremely similar to what we required or

fixes to bugs that we encountered were found within one Google search. The only drawdown experienced while using Bootstrap was that sometimes we had to override the styling for some elements in order to get the desired effects. However, even with this minor drawback, Bootstrap was an absolutely delightful styling library that we would use again in any of our future work.

Templating vs Front-End Development Technologies

The use of Jinja2 as the templating technology of choice for Alpha Factory was a pleasantly surprising experience. While the learning curve wasn't steep, it required a change of mindset and continuously active thought to fully leverage Jinja2 into our platform. The results from using Jinja2 against the risks were well worth it in the grand scheme of the project. The worry, or risk, with using Jinja2 was primarily from the limited usages and integration of the technology as well as the difficulty of picking up a templating library and trying to learn it from scratch. We were quickly reassured otherwise as soon as we got our first page up and running. It was extremely satisfying to be able to extend different HTML files and "blocks" of code. Making the entire site very modular allowed us to reuse hundreds, if not thousands, of lines of codes while maintaining full functionality and design/UI. The most useful part of Jinja2 by far was the ability to code IF statements and FOR loops directly into the HTML file. The FOR loops allowed us to dynamically generate HTML code while IF statements allowed us to leverage Jinja to accomplish tasks that would normally require a tedious amount of JavaScript, Python, and HTML coding. Another thing that we didn't appreciate at the time was Jinjas' server-side rendering ability. Server-side rendering makes the site extremely quick to respond and navigate throughout. It also works great for primarily static pages with sensitive and important information such as our website. Although we didn't get to see much of this use, search engines better index on server-side rendered websites than client-side rendered websites, making it easier for potential customers to find and learn more about us. While it was sometimes challenging to find example implementations or fixes to bugs found while using Jinja, the results of actually leveraging the technology in our favour reduced the workload for the front-end member by at least 20% and could not have been a better decision overall.

Alternative Designs and Considerations

Aside from the numerous and heavily apparent differences from the original design plans for the website found in Appendix C, we have dedicated an entire section discussing the alternative designs and considerations particularly for the front-end as it is an extremely important and sensitive matter when it comes to the end user. It is important to note that we have only included the original wire frames for the pages that were heavily changed the original report as to conserve space on this report and also to highlight the key alterations and design iterations that were performed. Since we are building our entire project around the consumer, we ended up having to make a number of key design decisions when it comes to user experience that were influenced by both direct user feedback and thorough investigations done by the members of the Alpha Factory Design Team.

Implementation of Function 1 and 2

The greatest thing we had to address throughout the project development stage was deciding how and where to incorporate the required functionality of the robo-advisor/investment wizard as defined by the requirements of the project. While it was mentioned in prior sections of the report, the required functionality of the advisor/wizard is reiterated here for the purposes of justifying our implementation.

1. Given a portfolio of assets (i.e. specification of the securities and its weights), the wizard will analyze the risk and return profile over various horizons by backtesting over real data
2. Given a portfolio to be held for some time horizon, the investment wizard will seek to find a portfolio of assets that dominates it (e.g. has a better Sharpe-Ratio using a reasonably large class of assets to draw from (the user can set the number of assets k to include in the portfolio from a universe of n assets where n>>k))
3. Given a return goal over some time horizon, generate a portfolio that has a projected return of at least the goal amount, but with minimum risk (e.g. portfolio variance, min VaR or CVaR over the time horizon)

Initially we didn't know how to implement and integrate these functions, aside from function 3, into our robo-advisor. We originally decided to keep the functionality separate and provide logged in users with full access to all 3 methods. It was only after several design iterations that the idea of merging functions together was drafted. It originally started with the idea of calling function 1 directly after function 3 in order to calculate portfolio returns and display them to users. We then expanded on this idea to include function 2 into the equation by making it so that the comparison of both portfolios is clearly visible to the user. Finally, we branched this idea out further and decided to cleverly come up with a strategy that leverages these functionalities to pull potential customers to joining our platform by offering them a portfolio that can outperform their current portfolio. This method of implementation makes the most use of the functionalities required by the project while maintaining the desired effect that the robo-advisor is looking to achieve. This also took care of the need for providing individual access to function 2 for users that have already joined our platform. The idea behind this is that if users already have a portfolio with us, they should have no need to call function 2 with our portfolios since it would only return the same portfolio and function 1 is already, in essence, integrated directly into the home page.

Hosting Alpha Factory

The decision of hosting is important enough to be its own section but will be considered under the Alternative Designs and Considerations as it required a detailed comparison with other existing web-app hosting services as well as a strong justification for the selected service. Deciding on a hosting service that met all our needs was a particularly challenging task since a lot of great services out there are competitive for this type of business. With this in mind, we chose to focus on our particular values to narrow down the choice of hosting application to only 1 selection. We looked for the following traits that are indicative of our priorities and design values:

1. A free hosting platform (top priority)
2. Easy to implement with our chosen technologies (Flask, Python, etc.) and well documented
3. Hosting service with computing and allocation of power that is sufficient for the scope of the project
4. The extent of services provided beyond website hosting (least priority)

With those values in mind, we set out to find the best hosting platform for us. The following 4 cloud hosting services were selected for further review and compared below. [47] As a side note, while Amazon Web Services met and even exceeded all of the requirements above, the actual implementation was non-trivial. AWS has a very large number of products that they offer, and even if we didn't get lost after that, the actual integration of the service with our web-app resulted in several hours of frustration, bugs, and, eventually, a dead end. As a result, while AWS is not mentioned below, it has been noted that AWS was a first choice and was also the first to be removed from consideration.

Service	Pro	Con
Heroku	• Simple, easy and well-documented implementation (with Flask apps) • Developer focused user-experience • Free plugins available for DB integration and other functionalities	• Limited number of monthly hours (1000 hours per month) • Free sites 'sleep' during inactivity (require 10-20 seconds to reboot)
Google Cloud	• Highly respected and well-documented hosting service • Advanced functionalities • Extensive extra capabilities and customization available	• Lack of support for non-static web apps (Flask) • Excessive functionality that will not be used after project deadlines
GitHub Pages	• Already using GitHub to store everything related to project • Easy integration with existing codebase location	• Limiting in its capabilities (not very good with Flask) • Primarily used for static apps
Microsoft Azure	• Reliable and powerful hosting service • Extensive extra capabilities and customization available	Excessive functionality that will not be used after project deadlines

Table 12: Hosting service comparison

With all of these options laid out and the pros and cons weighed in, we decided to go with Heroku as our choice of website hosting platform. We chose Heroku primarily because it was completely free, widely accepted, well documented, sufficient for our project requirements, and already uses technologies that we use (Git). Google Cloud and Microsoft Azure were extremely excessive for the 'hobby' project we were making and requires credit card information that we weren't willing to provide. GitHub Pages had a major drawback in that it wasn't able to deal with non-static Flask applications very well. On the other hand, there exists several online examples of non-static Flask applications hosted through Heroku and even more bug documentation/fixes. After having performed the demo as well as after numerous series of live testing, Heroku lived up to its reputation as an excellent hobby-hosting site. Although initial load times were long since they turn off the website when not in use, the overall performance of the Heroku platform was extremely satisfactory with almost no complaints. The deployment process was ridiculously simple, in fact, any one of the team members could do it since it only required us to push to a git repository, which we have been doing since the beginning of the project. Finally, the main point of concern was addressed

with the website load times while running the robo-advisor functions which ultimately pushed us to improve our service and get those functions down to lighting fast response times. Overall, Heroku was a phenomenal choice for a hosting platform and we intend to keep our site hosted on https://alphafactory.herokuapp.com/ for the foreseeable future.

Questionnaire

During the Initial Design Report, it was mentioned that we had taken a look at some existing form-based questionnaires and completely ruled them out of the possibilities because they lacked a sense of care for the user experience. Form-based questionnaires are boring, unintuitive, and seemingly endless to the user that is willing to give our platform a chance. We believe strongly in user experience and a big part of that is a user's first impression of our website through our Sign-Up processes. If they dislike any aspect of the sign up, they will either give up on our platform or, worse yet, discourage others from joining us as well. That is why we made the initial decision to directly rule out having a traditional form-based questionnaire, instead opting for a more simplified approach to determining user risk-tolerance. To gain inspiration on how to tackle this problem, we hit the web to do some researching and quickly turned up several alternative results. The problem with our search initially is that form-based solutions were so popular and heavily implemented that we had to extrapolate our own designs based on the surveys that we actually enjoyed. What we took away from our research and cumulated opinions can be summarized in the following points below:

1. When questions were displayed one at a time to overcome the crowding of questions gave us a chance to focus on the currently displayed question and ultimately enhance experience
2. When we were given a simplified form of input such as a selection between less than 3 options or a slider as a numerical input
3. When there were pictures and other helpful tools that made it easier and more intuitive for us to fully understand and answer the questions

With these results in mind, we began to design a questionnaire that embodies these design values and came to the current design found in our website. This design is centered around a questionnaire that displays questions one at a time with only 2 options to select from that have an image associated with it to provide the optimal user experience. The number of questions we chose was kept small at only 15 to make it such that users didn't feel like they were spending excessive amounts of time on the questionnaire and were even given a progress bar to track their progression through the questionnaire. Coupled with the fact that there are only two options to consider, the user is expected to breeze through the questionnaire with the highest level of user experience. The choice to only have 2 options also restricts the number of things users can enter so that the back-end doesn't have to deal with outlier data, ex: a user can't select a salary that is abnormal such as $20/month.

Although the structuring of the questions required more thought and a higher level of effort to calculate the resulting risk tolerance, the user experience is tremendously impacted in a positive manner which was the highest priority when we designed the front-end. The shift away from traditional form-based questionnaires has been directly seen and quantified through the survey results displayed in the Usability Testing Protocol section. It is clear from Figure 25 and Figure 26 that the selection of questions as well as the decisions that went into making it were a raging success, with hardly anyone below a 7/10 rating in terms of user understanding and experience.

Single Portfolio Allocation Per User

Prior to starting the final report draft, the decision to restrict the number of portfolios assigned to a single user was made by the group members. The interesting thing is that the easier part of this problem would be the actual implementation of multiple portfolios for a single user since it is only an extension of the current platform that we already have set up. The crux, however, is in the fact that we believe our users should aim for goals one at a time or try to incorporate several goals together and directly look to find a portfolio that, at a minimum, matches their goals, if not exceed them, while being within their risk tolerance. For users to split their attention between multiple portfolios that they have to individually track and monitor independently would cause such a huge detriment to the user experience that we decided we would like to avoid the problem altogether by only allowing the user to be assigned a single portfolio at a time. This resulted in the introduction of the Advisor page as a medium for the users to access Function 3 to change their currently assigned portfolio to their changing needs and requirements. The advisor page was decided to only allow the user to change their existing and active portfolio based on their financial goals, initial capital, and assigned risk tolerance instead of creating a brand-new portfolio which the user can choose to follow and monitor as well. While we have yet to see the effects of this design change reflected in the response and feedback of the users, we believe that we made the right choice and stand by the decision in favour of single portfolio allocation.

Highlighted Extra Implementations

Outside of the impressive (and unique) mailing service and loading mask features that the team has implemented; Alpha Factory has done a lot of work that can be overlooked by both the user and the readers of the report at times. In this section, we wish to shine a light on the exhaustive efforts of our team to ensure smooth user-experiences that would not normally be seen in the regular operation of the site.

One of the main things that Alpha Factory has implemented which can be overlooked at times is the restriction of user access to areas of the site depending on their status. The main example that can be seen is the separation between the ‘main site’ and the ‘user site’ which is dependent on the user’s logged in status. However, an extension of this is applied to all pages of the Alpha Factory flow and was a major consideration when programming and designing the website. An example of this is the particular redirection found when users try to access secure parts of the user site that aren’t available to them if they’re not logged in such as the /change_password route. If the user is not logged in, going to the /change_password route will actually redirect them to the /forgotpass page under the assumption that they wish to change their password but are not logged in. While these are only small considerations and implementations, the real effort can be seen when it comes to the redirection and restriction of accesses to pages when users are on specific processes. For example, if the user is currently trying to create a new portfolio, they will not be allowed to access the home, advisor options or profile page until they either complete the process or log out. On that same note, it is important to mention that until the user completes the process, the information on their account will not be updated in the database. This is a major consideration since we wished to use the same functions for new users (which require active user DB updates) and existing users looking to change the information on their profile (which requires them to

only update at the end of the process). It can be seen from the description of the problem that the solution is not trivial and required a heavy amount of consideration and design iterations from the team members in order to create the functionality required for this system.

Another feature that will be overlooked is the error handling feature for page rendering. Although there are rarely any path problems in the controlled usage of the web-app during demos, this is not always the case in production environments where users will want to access different parts of the site directly. Users may try to also access parts of the site that simply don't exist. In the case of a such a situation, Alpha Factory has put certain measures in place so as to provide the user a route back on the right path.

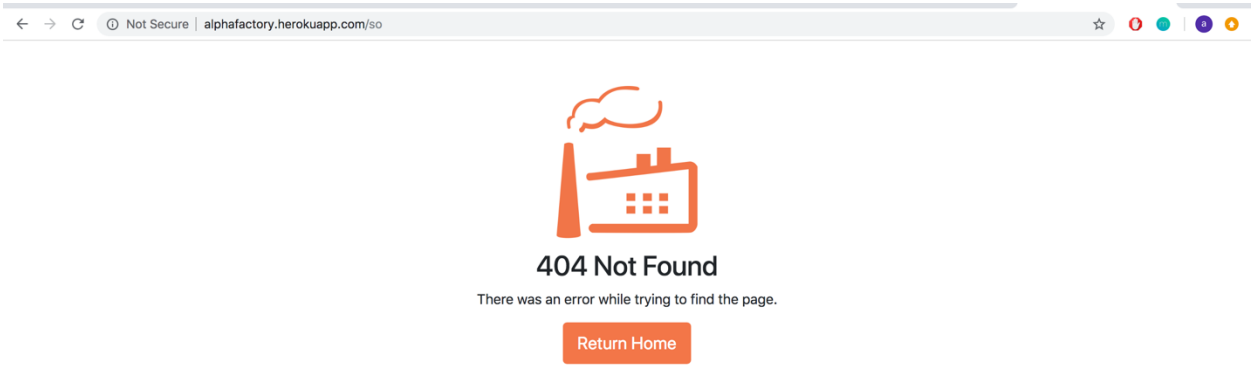


Figure 29: 404 Page not found error handling screen

The page displayed in Figure 29 is what the user is shown if they try to access routes of the site that don't exist. The 'Return Home' button provides the user a method to return back to the home page. It is easy to see how this consideration could be overlooked, hence why we added it into the spotlight section.

Back-End

The core functionalities of the back-end section of the Alpha Factory services are responsible for the decisions regarding the efficient storage, maintenance, and use of all the data that is used in other aspects of the product's design. The key design decisions that were made were around the choice of structure for a database management system, as well as the structure of all the raw and calculated values that impacted both the business logic and the user interface. Some of the key features and decisions that were made are outlined in the sections below.

Choice of Database / Data Source for Raw Data

To store its various forms of data, Alpha Factory used MongoDB, the popular database service that allows for the deployment, operation, and scaling of NoSQL databases on its servers. Mongo offers a free version of their services, which were estimated to be more than adequate for the purposes of Alpha Factory and made the choice to use MongoDB for our back-end database services quite easy. In addition to the fact

that we could use this database service for free, the other prominent reasons for deciding to use MongoDB were as follows:

1. Document oriented storage: Data in MongoDB is saved on the system in the form of JavaScript Object Notation (JSON) documents, meaning the creation and maintenance of the data is not as strict as it would be with a classic relational database management system. Each document can vary in size and format, so there is no restriction on making each dataset as useful as possible.
2. Intuitive querying: With the loss of restrictions imposed by a relational database, MongoDB uses its own querying system to search through the data efficiently. Since the queries are specific to MongoDB, they are easy to work with and leverage the other properties of the database.

The decision to use the NOSQL MongoDB over traditional SQL Relational Database Management Systems is discussed in more depth in a later section and was primarily done through a mixture of manually examining the documentation for each of the databases and viewing a table of comparators across the databases. [45] The specific choice of MongoDB over other NoSQL DBs was done in an extremely similar manner to the method mentioned above with a comparable table between NoSQL DBs, which was thoroughly examined to conclude that using MongoDB was the best option. [46] For extra details pertaining to this selection, please see Appendix F.

After having located the first set of assets that was considered for the asset universe, their data was pulled from the Bloomberg terminal as CSV files. After finalizing the selection of 19 assets that would be included in the asset universe, each individual asset's data was added into one final CSV file that contains information on all the assets in the asset universe. As the data was imported into MongoDB, the consolidated CSV file was converted to JSON documents, a widely accepted file format for MongoDB and other Document-Stored database systems. The format of a JSON document and how it is the ideal file format for use in a database management like MongoDB that stores data in documents is explained in the next section. Since we only care about the price of the asset, only the adjusted closing price and date is given for each asset, with all assets having data from 1999 to 2018.

In addition to the raw data, the computed data related to our portfolios needs to be stored on our databases as well, such as parameters for returns, volatility and drawdown, as well as the return data pertaining to these portfolios that get created through Alpha Factory's business logic and processes. While the statistics of the portfolios over all our historical data are useful to have, the return series was much more important to include because it allowed us to calculate these values for different time horizons to perform more comprehensive back-testing. Once again, the choice to use an object-oriented database, and MongoDB in particular, allows a very straightforward organization for this data. The detailed plan for how these objects is going to be set up can be seen below in the Database Schema section. The data usage for calculated values will be lighter on data usage, as there are fewer portfolios to keep data on than there were individual ETFs that went into creating them. This is also true of components like user information, as Alpha Factory is not a service in production, so creation and maintenance of user information is used more for testing the functions of the application. However, the impact the user information can have on overall data usage is described below in the 'Data Usage' section.

Storage of Financial Data (Both Raw and Computed Data)

To use the financial data from the previous section in Alpha Factory's business processes and portfolio generation, we are taking that data and storing it with our own database, which was stated in the previous section to be MongoDB. In storing the data for the different items in our asset universe, the format of each asset's data is expected to be similar, and in addition to similar formats for raw data, the time horizon for the historical data of the assets was made to be identical, with proxies used when one asset didn't have enough historical data (this is further discussed in the section labeled Cleaning Data & Dealing with Missing Data). So, the pricing information for the asset universe can be created as a single collection, with prices for all assets grouped together and indexed by date. This way, any interactions between the assets can be done seamlessly, like finding different ways to track the portfolios of different assets over various horizons for the purpose of better predicting the movement of the entire asset universe.

In addition to the raw data, the computed data that will be included is the statistics for each of the portfolios that is offered by Alpha Factory, such as parameters for returns, volatility, Sharpe Ratio and other performance metrics. Once again, the choice to use an object-oriented database, and MongoDB in particular, allows a very straightforward organization for this data into one collection with documents for each portfolio containing its relevant statistics. The plan for how these objects is going to be set up can be seen below.

For the raw data, the main objective is simply to ensure that all the data needed for estimating the parameters of each asset is readily available and can be called upon when these values need to be used or updated. The key to the organization of the data is that the functions using the raw data generally care more about matching a time horizon for multiple assets than using one individual asset's entire dataset. As such, we can organize the data so that the pricing data is indexed by date, and for each date we receive the prices for each asset enabling us to select which ETFs to use. Figure 30 shows how this might look within MongoDB's structure, with more in-depth breakdowns in the section outlining the Database Schema:

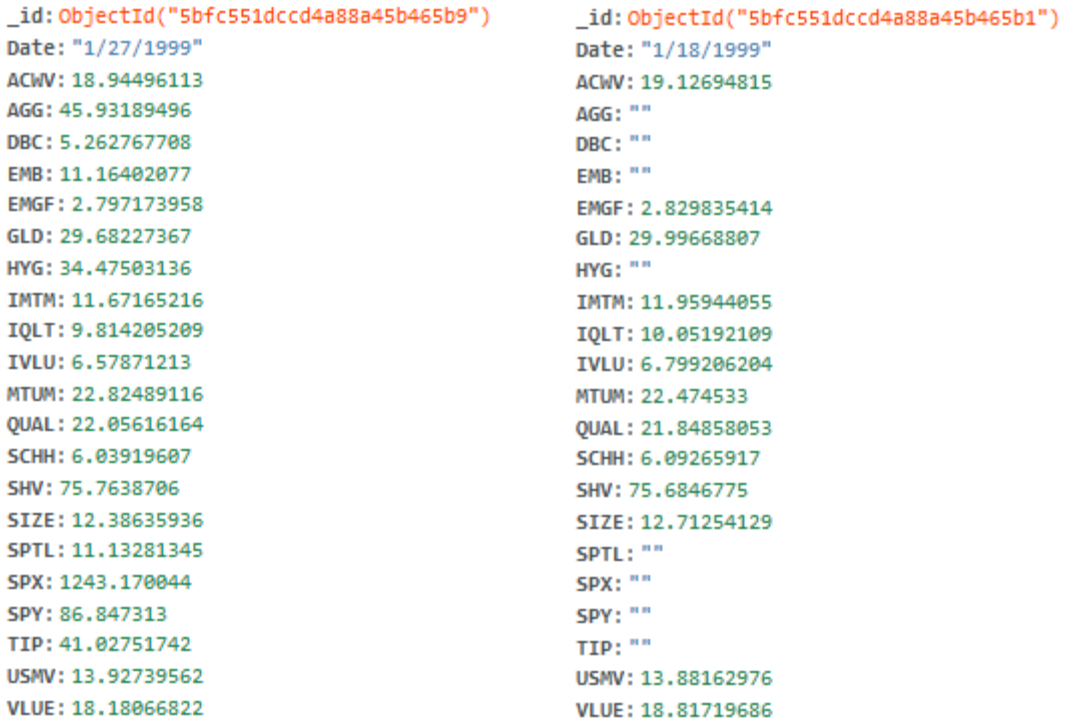


Figure 30: Layout of a single document within the collection of raw data with a MongoDB database

The calculated data for our portfolios is included in two different collections for two different purposes. One collection contains the statistics of each portfolio for the entire time horizon for which we have relevant data. The reason for including this is to give current and potential users a first look at the performance of our portfolios over that consistent time horizon of 20 years that we have pricing data for. For calculating the statistics over different time horizons, we also store the return series of each portfolio. These allow us, among other things, to optimize over a user's inputted portfolio for a specific time horizon, like the bull and crisis scenarios that are explained above. The other main functionality of keeping the return series is for trend analysis that is incorporated into the creation of our portfolios. The format of the 2 collections for calculated data is show below in Figure 31 and is further explained in the Data Schema section.

Figure 31: Layout of a document within the collections for Portfolio Returns and Portfolio Statistics

Setup of Data Access Layer for Data (Both Raw and Computed Data)

The data access layer of a database is the actual set of commands and functions that allow data to be accessed from the database and used in the business logic and other parts of Alpha Factory's processes. For relational databases, these are also known as the CRUD functions (create, read, update, delete). In object-oriented databases, the same functions need to be implemented between the data and the rest of the application, but since the data is stored as an object to begin with, the level of complexity that needs to be inserted into the creation of the data access layer is significantly reduced. The framework for the main functionality of the data access layer is outlined below:

1. Creating:

   Within each individual data and object type that is initialized for use in Alpha Factory, the object-oriented style of MongoDB means that all that is necessary to be able to add more objects into a collection is a function for initializing the parameters of the new instance of that data type.

2. Reading:

   Like SQL, which aids in getting information from relational databases based on a querying language, the method for reading certain documents and getting specific information is based on searching through your datasets by the various fields that each document contains. The results of these searches can then be used in other functions and calculations.

3. Updating:

   The two ways to update documents in MongoDB are through updating an existing document, where the fields that need to be replaced are listed as parameters, as well as the new values that these fields should take on. This can be done with one collection, or with many collections in one statement, with each collection receiving its own set of parameters. The other way to update a collection is to replace it entirely, which is equivalent to deleting an existing object and then creating a new one in its place but in one method instead of two. The choice of whether to update or replace will come down to the individual task that is required in the business logic.

4. Deleting:

   There are built-in functions in MongoDB that can delete objects and collections in a dataset. The function 'drop' and its counterparts are responsible for these tasks and are made so that dropped/deleted items are removed from memory and that storage becomes available for other data.

Cleaning Data & Dealing with Missing Data

In the process of collecting the raw data for each asset in Alpha Factory's asset universe, there is a very good chance that the data collected will not be perfect, and that there will be data missing or data that has been corrupted and is not usable in our processes for a variety of reasons. To combat these scenarios and mitigate the effects that this unusable data poses on the rest of Alpha Factory's processes, the following approach is taken for the various kinds of data issues and tasks that can come up during data collection:

1. **Missing data**: There are two overlying scenarios that can constitute as missing data. The first is the case of an asset only having pricing information for a time horizon that is shorter than what we decided to keep in our datasets. The second is the case of missing data anytime within the database. Each scenario and our method for tackling it is listed here.

   As explained, the asset universe consists entirely of ETFs. The initial problem that was discovered was that most ETFs didn't have pricing data for time horizons that were long enough to justify our using them. The solution that is currently in place is to proxy them with the indices that they track. These indices typically have pricing information that goes much further back than the ETF data. The main issue with this technique is the error that comes between the price of the index and the price of the ETF, which we tried to account for with adjustments for management fees and other possible factors. By using this technique, we managed to get 20 stable years of daily data for the asset universe, giving a large enough sample that we were comfortable optimizing and back-testing with the data we had.

   In the case of a trading day that contains pricing information for some but not all the ETFs in the asset universe, as well as the indices used to proxy the values of the ETFs. the prices of those missing ETFs are excluded from the database for that day. The result is that some documents for pricing data can be found with empty fields for some of the ETFs Causes for this inconsistency in

data can stem from regional holidays or other unknown issues from the data source. The decision to exclude the data is based on the decision to keep the original pricing data genuine, instead of finding a technique for proxying the value from other surrounding values. Another result of this and decision we made was to then exclude the portfolio returns for the days that didn't have complete data for them. This is not expected to impact calculations of portfolio metrics because each ETF has 20 years of data stored in our databases.

An alternative approach that could have been taken in addressing missing data is to average the missing values of the data from the days before and after the missing data. The main reason for not following this approach was the authenticity of the data, which was meant to be as transparent as possible. If further data collection showed that it was necessary to start filling in values for missing days, this is the approach that would be followed.

2. **Data cleaning and maintenance**: If a dataset is shown to be 'damaged' and is causing problems to other components of Alpha Factory's design, the solution in place is a manual investigation and fix/replacement of the data. Since each individual problem with a dataset could carry a different underlying error, the most important factor is correctly determining the root cause of the error, and the most accurate approach to this is through human quality assurance.

Hosting

The hosting service for MongoDB is Atlas, which comes from the creators of MongoDB and allows users to choose from the popular hosting platforms, while simultaneously providing services for MongoDB databases. With the first 512 MB free through Atlas, we decided that using Atlas' services, which includes all the benefits listed in Table 12 below that the service claims to provide, was the best way to host our database. As we work toward gathering and updating the financial data for our asset universe in production, we will take care to monitor our data usage to judge whether continuing with this service is the best way to support Alpha Factory's scaling needs.

Benefit	Explanation
Automated	The easiest way to build, launch and scale apps on MongoDB
Secured	You get access to multiple levels of security available to give you peace of mind
Scalable	Deliver massive scalability with zero downtime as you grow
Highly available	Your deployments are fault-tolerant and self-healing by default
High Performance	The performance you need for your most demanding workloads
Updated	MongoDB Atlas gives you access to the latest MongoDB features

Table 13: Benefits of hosting a MongoDB database on Atlas

Data Usage

When doing initial testing of data capacity for MongoDB to ensure that it would be suitable for the uses of Alpha Factory as a proof-of-concept prototype, the two factors that had to be considered were the space that would be occupied by the raw pricing data and the space that would be occupied by calculated values and user information. The impact of each of these is outlined below.

We start by looking at the raw pricing data. At the time of testing , the inclusion of the raw data of 32 ETFs and the SPX index from Bloomberg into MongoDB amounted to a total data usage of 17 MB, with the largest number of documents belonging to the collection of the SPX index dating back to 1950 for a total of over 17,300 data points, while some of the more recent ETFs only have information for the last decade, with approximately 1,000 data points. Using one of the collections as a sample, the number of documents and size of the collection was used to determine that the average size of one data point for one asset was approximately 0.13 KB. Thus, given the current space occupied by the data in storage, our choice of MongoDB as a database management service is more than adequate for dealing with the addition of more assets like indices that could be added to proxy the values of the ETFs for which there was less data than we would like for back-testing purposes. The final choice of asset universe of 19 ETFs ensures that we can continue to hold onto the pricing data for the ETFs without worrying about data limits.

Next, we needed to consider the inclusion of all other values used by the site. While the questionnaire data collection is not expected to grow (i.e. there will only be a constant 15 questions, even though Alpha Factory can choose to change the contents of each whenever they wish), the Users collection is expected to continue growing. Keeping in mind that MongoDB provides the first 512 MB of their services free, the concern lies in the ability for the database to outgrow the constraints of the provided free services. It was calculated that, with 7 users in the Users collection taking up approximately 1.65KB of data, each user document takes approximately 0.25 KB of data. The user history document, on the other hand, can be found to take up around 125B per user transaction. With this information, combined with the knowledge that a user must have at least 3 historical items attached to them (account created, risk tolerance assigned, portfolio generated), we can see that the minimum size one user will take up on the database is 0.625 KB. Ignoring all other DB storages, we can find the crude upper limit on the number of users Alpha Factory can have without paying for extra features from MongoDB as 512000/0.625 = 819,200 users max. For the purposes of this capstone project, there will never be a time where over 800 thousand users are needed but if Alpha Factory wishes to expand their services to the general public, the limited capabilities of the free services will likely be inadequate for production performance.

Flow Chart and Decisions on Technologies Used

The following are flowcharts that go into the processes that are engaged when interacting with the database during the overall flow of the system shown in Figure 3. They give a sense as to what happens on the back-end with the database as the user is using the web application.

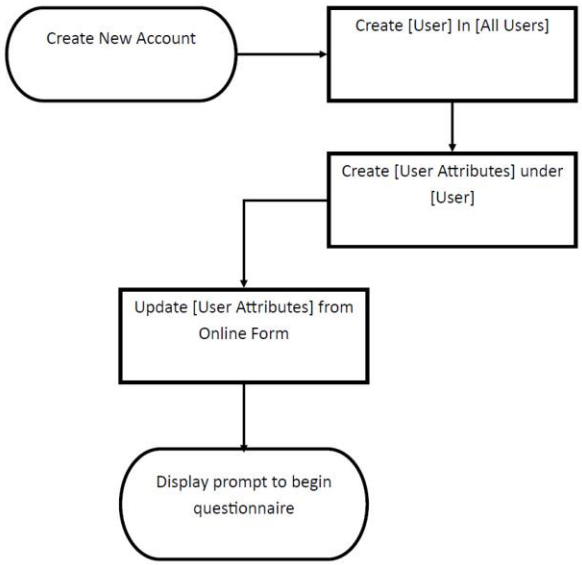


Figure 32: Flowchart for Creating a New User Account

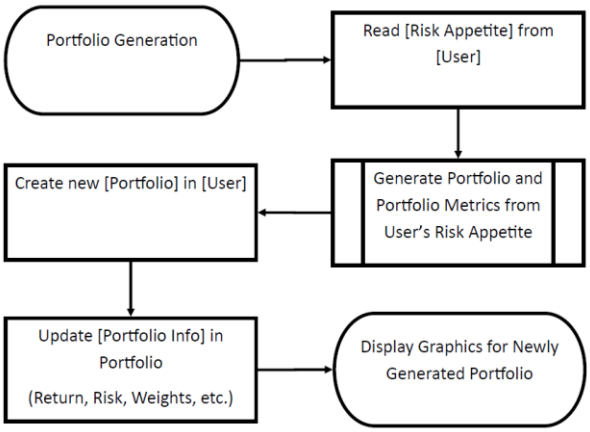


Figure 33: Flowchart for Portfolio Generation

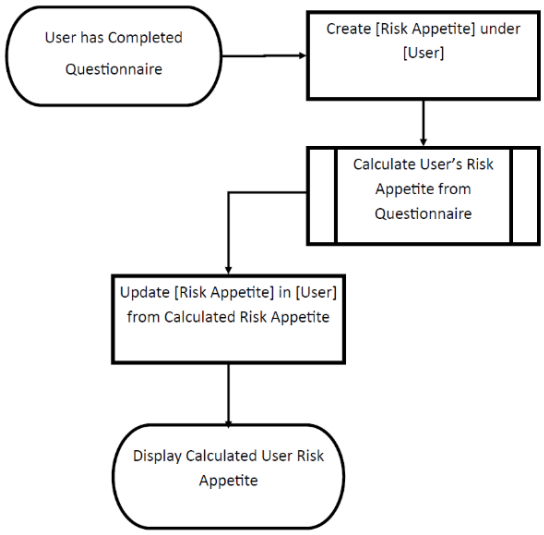


Figure 34: Flowchart for Completed Questionnaire

Database Selection and Alternative Choices/Designs

In choosing to use MongoDB, there was a decision to make regarding whether or not to use a relational database management system (RDBMS) that uses the Structured Query Language (SQL), which has historically been the most popular way to retrieve information from a database for use in other applications. While this can provide a very structured way of accessing the data, particularly across different tables and data structures, it is very difficult to adjust the setup of the database once it has been initialized because of how tables are linked to one another. Therefore, an extended period is required for initial planning to ensure that all subsequent changes to individual data structures are targeted towards improvements in efficiency more than towards altering the functionality.

The alternative to the RDBMS is the Object-Oriented Database Management System (OODBMS), which tackles many of the limitations faced by the RDBMS. The main differences between the two are the use of SQL for querying information and the format in which the data is stored. Whereas an RDBMS stores data in tables linked by primary keys, an OODBMS stores data in 'documents' as objects, which can take on various forms and are minimally restricted in terms of format. The advantages of this approach, and specifically how they apply to the application, are summarized in Table 14 below:

Advantage	Explanation
Wider variety of allowed data types	• Allows for data to be stored in an increased number of formats, allowing more flexibility to the database and its potential uses • Objects in the databases can be used as their own data types, for a more extensive set of data types available

Advantage	Explanation
More coherent modelling capability	• By organizing each data structure as an object, there can be more extensive links between various data structures to form a more realistic model for your data • The database can be an active component of the application and business model, removing the disconnect between the data and the rest of the system
Performance	• As a newer technology that was implemented with efficiency in mind, Object-Oriented Databases have been shown to provide better performance results when compared to Relational Databases

Table 14: Advantages of an Object-Oriented Database Management Systems

For a more in-depth look at the other potential candidates for database management systems and the criteria that went into choosing MongoDB as the system that we would use, please see Appendix F.

Database Schema with Descriptions

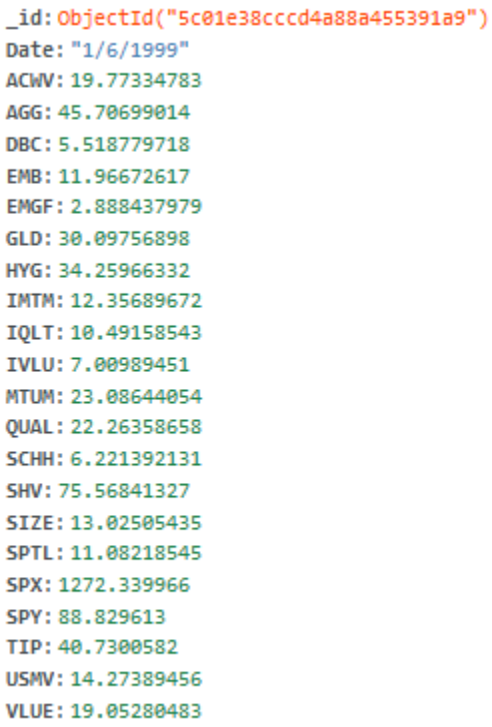
In this section, we will talk freely and in-depth about the database structure for all of the collections currently stored on the MongoDB server. Alpha Factory currently has 6 collections on the MongoDB server which encompass and account for all the storage needs required by the program to run at full capacity. Among the 6 collections, only the user collection is expected to be updated very-frequently through natural operation while the other collections are either static or manually updated through uploading of calculated/fetched data. This makes it extremely simple for us to track, monitor, calculate, and predict database storage as we will generally have linearly growing 'static/manual' data collections as well as only one dynamic and frequently updated data collection. The selection of MongoDB and the 512 MB provided by the free service really pays off here since we can easily determine how much free space, we will have left over for the user collection to grow before we incur any real costs.

As explained above, there are two components to the data that are being stored in MongoDB. The first is the raw data, which includes the pricing data for all the assets in our asset universe for the time periods that we were able to secure them for. The data is taken for each trading day, and all relevant data is transferred into MongoDB by first downloading the data, and then importing it manually, as was outlined in the Initial Design Report as one of the conditions for using the Bloomberg terminal as our chosen source of data. Thus, the resulting data format in MongoDB is given below with:

```
Raw_Prices  :  Collection
{
        Document
        {
        ID : ObjectID       "Unique ID for this document"
        Date : Date         "Trading date for all values"
```

Asset_1 : Double "The adjusted closing price of ETF Asset_1"
Asset_2 : Double "The adjusted closing price of ETF Asset_2"
Asset_3 : Double "The adjusted closing price of ETF Asset_3"
…
Asset_19 : Double “The adjusted closing price of ETF Asset_19”
}
…
}

A visual example of this as shown in the MongoDB is provided as well:



There is also an entire collection dedicated to the risk-free rate which is used in the calculation of excess returns and feeds into various portfolio statistics. The risk-free rate used is the 3-month US T-bill rate and pulled from Quandl (https://www.quandl.com/data/FRED/DTB3-3-Month-Treasury-Bill-Secondary-Market-Rate). Find the resulting data structure below:

RF : Collection

{

Document

{

ID : ObjectID “Unique ID for this document”

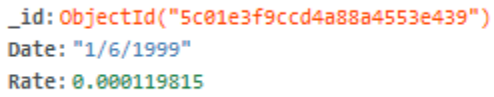
Date : Date “Date associated with this record/value”

Rate : Double “risk-free interest rate for the date recorded”

}

}

A visual example of this as shown in the MongoDB is provided as well:



The final component is the data that is created and then manipulated within Alpha Factory itself. This includes the users and generated portfolios. The data stored for the users is under 2 different collection. The first is for all the user information, which includes the data they enter when creating their profile and the data that gets assigned to them in the application, like risk tolerance and the portfolio(s) they hold. The second is to store the questions that are given to the user in the questionnaire to determine their risk tolerance. The first data collection is set up as below:

__Users : Collection

{

Document

{

ID : ObjectID "Unique ID for this document"

FirstName: String "User’s first name, provided on creation"

LastName: String "User’s last name, provided on creation"

Email : String "User’s email, provided on creation (unique across the user collections)"

Password : String “hashed (encoded) password for the user account”

DoB : Date "User’s date of birth, provided on creation"

RiskTol : String “Risk tolerance level of the user”

RiskTolNum : Int "The risk tolerance number assigned from the questionnaire (measure of risk-averseness by question answers)"

RiskProfile: String “The recommended risk profile (portfolio) of the user according to their risk tolerance and time horizon”

fillQuestions : Boolean “Flag to indicate whether the user is required to fill the questionnaire or not”

Portfolio : Object "Portfolio held by the user"

dateCreated : date “date of portfolio creation”

Horizon : Double “Investment Horizon”

Risk : String “Portfolio type selected by user”

initial : Double “Initial value invested into the portfolio”

}

…

}

A visual example of this as shown in the MongoDB is provided as well:

We also have the questions in the questionnaire:

__Questions : Collection

{

Document

{

ID : ObjectID "Unique ID for this document"

qid : Int "Chronological order of the question"

Question : String "Wording of the question"

Optiona : String "First option to answer question"

Optionb : String "Second option to answer question "

Imga : String “String storing the local path for the image associated with option a”

Imgb : String “String storing the local path for the image associated with option b”

}

…

}

A visual example of this as shown in the MongoDB is provided as well:

```
_id: ObjectId("5bcfa48b9e7cba4323261dd8")
qid: "2"
question: "Would you rather:"
optiona: "Have a 50% chance of getting $100 and 50% chance of losing $100"
optionb: "Have no gain"
imga: "./static/images/question_images/Q2a.png"
imgb: "./static/images/question_images/Q2b.png"
```


Lastly, we have the collection for the portfolios created by Alpha Factory's business logic processes, and the data for each portfolio that gets stored is its return series, with each document being occupied by the returns of each portfolio for a specified date. This is designed to be like the format of the raw data, and when the series is pulled from MongoDB, the stats of the portfolios for the time period selected are calculated by the business logic before being shown to the user.

```
Portfolio_Returns : Collection
{
        Document
        {
        ID : ObjectID       "Unique ID for this document"
        Date : Date         "Appropriate date for all values (daily)"
        Preservation : Double       "The return for the preservation portfolio"
        Conservative : Double       "The return for the conservative portfolio"
        Balanced : Double       "The return for the balanced portfolio"
        Adventurous : Double       "The return for the adventurous portfolio"
        Aggressive : Double       "The return for the aggressive portfolio"

        }
…
}
```

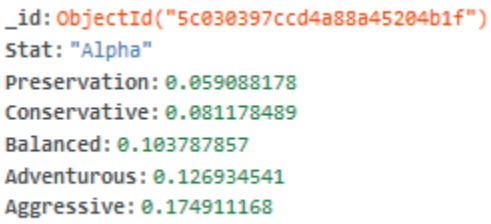
A visual example of this as shown in the MongoDB is provided as well:


```
_id: ObjectId("5c030344ccd4a88a451ff54c")
Date: "10/28/1999"
Preservation: 0.002433111
Conservative: 0.003548483
Balanced: 0.00464877
Adventurous: 0.005734166
Aggressive: 0.007861087
```

As an extension of this, to save time on recalculation for each user, Alpha Factory has decided to store the current live stats for each portfolio using a 20-year time horizon. This is then queried for to display the users on their dashboard when they access the home page. By doing this, we ultimately save significant time and recalculation costs as we don't have to recalculate each individual statistic for each unique user portfolio holding period. However, it must be noted that in production (non-demo usage), the 20-year default horizon would be removed and replaced with the stats for the unique user portfolio holding period, rendering this schema redundant. Below is the current structure of the schema outlined:

```
Portfolio_Stats  : Collection
{
Document
        {
        ID  :  ObjectID       "Unique ID for this document"
        Stat  :  String     "The name of the statistic"
        Preservation  :  Double       "The statistic value for the preservation
        portfolio"
        Conservative  :  Double       "The statistic value for the conservative
        portfolio"
        Balanced  :  Double       "The statistic value for the balanced portfolio"
        Adventurous :  Double       "The statistic value for the adventurous
        portfolio"
        Aggressive  :  Double       "The statistic value for the aggressive
        portfolio"
        }
}
```

A visual example of this as shown in the MongoDB is provided as well:



Alternative Designs and Considerations

A large part of the iterations performed on the database design were as a result of feedback or design decisions that were made in other sections which then needed to be transferred over. This section is dedicated to discussing some of the many iterations and alternative design decisions considered, with a majority of them being directly implemented and then edited, removed, or improved upon depending on the success of practical application.

The first major change that Alpha Factory was recommended came from feedback received from a vital stakeholder, Hassan Anis, after our intermediate report in which he suggested to group all the stored returns and raw data for each individual asset/ETF into one big collection. Although it seemed to be more of an organizational comment at the time, seeing as how it was extremely messy to have almost 30 collections with individual closing prices, the actual effects of implementing this change was particularly felt in the performance of Function 2. Prior to grouping the asset closing prices and returns into one collection, we were required to individually query for each asset price collection for usage in Function 2. It was quickly noticed that there were several performance issues with this design that weren't apparent before. We realized that it would take tens of seconds more in time to individually query 30+ collections when we could just group all the required information into 1-2 big collections and run only 1 query. After this realization, we took it upon ourselves to group all of the asset closing prices together into one collection with each document being an object containing the daily closing price for all of our assets. This schema can be seen in the Raw_Data collection. Another thing that we did in order to minimize the run time of function 2 while also improving database schema design was to pre-calculate the daily returns for each of our assets over the time periods we have stored in the Raw_Data collection. This saved us a significant amount of time and recalculation costs as we did not have to manually recalculate all of the returns again in the back end, hence making our functions even quicker. With the implementation of these changes, our 30+ collections with their corresponding information were condensed down to 2 collections and the run time for function 2 was reduced from ~ 37 seconds locally to ~2-5 seconds. The direct impact of these changes was very apparent and allowed our site to respond in lightning fast times which made us begin to truly appreciate the importance of quality designs and considerations across all areas of the project.

Another design decision that was proposed closer to the end of the project was the decision to remove the user history collection. The user history collection was meant to be dedicated to keeping a log of all user actions and then re-used in the front end to provide users access to their old portfolios/generations. With all the changes that were occurring in the front end, this collection was overlooked and outdated quickly, until no updates were being made to it at all. It was only during the final iteration of the project that the team sat down and discussed the use cases of this collection that we decided to render it obsolete. We decided to remove the user history collection for a couple of reasons. The first reason is that it's a dynamic collection, which means that it would grow indefinitely over time. This was concerning for the team as we only have 512 MB of free storage space and would ideally like to leave as much room for users to register as possible. The user history would also have to be updated frequently which would result in a larger database load from frequent updating. The second reason we decided to scrap this collection is that we had no real need for it anymore. Originally, we wanted to provide users the ability to view past results from portfolio generations and function 2 comparisons, however, after we removed the ability for

logged in users to access function 2 as well as restricted the number of portfolios a user can actively hold down to 1, we found that the user history collection couldn't be used for much other than risk tolerance changes. We already provided users a method to view our other offered portfolios through the advisor options page where they can change their portfolio type and because we restricted the number of portfolios they could generate and actively hold; the user history had no place left to go on the website. Overall, the decision to remove the user history collection can be summarized by the lack of need for a dynamic user actions log. We hope to revisit this idea in the future if the need arises for such a collection/functionality or if we decide to expand on our current functionalities and provide users the option to hold multiple portfolios.

Appendix A – Deeper Dive of Existing Solutions

Alpha Factory ultimately used these competitors as case studies in creating their own design decisions.

1. Betterment LLC [6, 7]
   With over $10B in assets under management (AUM), Betterment is one of the robo-advisor behemoths. What makes Betterment so appealing to inexperienced investors is that it has no minimum account balance for its basic plans, while still offering top-tier portfolio management services through its premium plans. Betterment also advertises that "you can potentially keep an additional 2.9% of your investor returns each year" because of their passive investing approach[3], minimal rebalancing and tax-efficient techniques.

   Based on their risk questionnaire Betterment provides you with a customized portfolio of low-fee stock and bond index funds.

2. Personal Capital [9, 10]
   Personal Capital aims to provide an all-in-one financial platform by allowing their clients to connect their existing bank accounts to the platform to track their spending and retirement savings on top of their portfolio's performance. With over $5B in AUM, Personal Capital largely follows modern portfolio theory, namely Mean Variance Optimization (MVO) to construct the optimal portfolio of assets by maximizing the portfolio's returns subject to customized risk constraints. As with Betterment, Personal Capital improves upon the basic MVO by employing tax loss harvesting and rebalancing.

   It is worth noting that Personal Capital charges the highest fee (i.e. 0.89%) but justifies it by providing additional wealth/financial planning tools and a dedicated team of financial advisors.

3. Schwab Intelligent Portfolios [11, 12]
   Schwab has disrupted the disruptive robo-advisor space by taking the low fees as low as they can go. That is, Schwab charges no account fees or commissions, rather earning money from Schwab ETFs and select third-party ETFs. It is worth noting that unlike the other robo-advisors discussed, Schwab does require a minimum $5,000 balance to open an account.

   As with Betterment, Schwab creates custom portfolios of ETFs based on investor responses to their questionnaires and employs automatic rebalancing and tax loss harvesting for accounts with a value greater than $50,000.

   It is worth noting that Schwab claims that the operating expenses investors pay for the ETFs in their portfolio are the same as they would have been if they invested in them on their own. As

[3] Passive investing methods seek to avoid the fees that may occur with frequent trading. With the main goal being to build wealth gradually, the primary investment mantra of passive investment strategies is that of buying securities with the intention of holding them long term. With the underlying assumption of passive investors being that the market posts positive returns overtime, passive investment strategies do not seek to profit from short-term market fluctuations as active traders would. [8]

such, the operating expenses paid varies depending on the makeup of their portfolio and ranges anywhere from 0.03% to 0.40%.

4. SigFig [13, 14]
   SigFig attempts to differentiate itself by catering to those with an existing online brokerage as it allows you to keep your existing investments with their robo-advisor creating an "intelligent, tax-efficient, diversified portfolio." Like its peers, SigFig uses a risk-assessment questionnaire for creating customized portfolios for their clients. Like Schwab, SigFig has a minimum account balance of $2,000 and manages accounts under $10,000 for free but starts charging an account fee of 0.25% for any amounts greater (note that they only charge you on the portion exceeding $10,000).

5. Wealthfront [15, 16]
   Like Betterment and Personal Capital, Wealthfront is a robo-advisor behemoth with over $7.5B in AUM and builds investors custom portfolios based on their answers to risk questionnaires. Like SigFig, Wealthfront manages your first $10,000 for free and charges 0.25% for amounts exceeding $10,000.

   As with the other behemoths, Wealthfront offers a wide range of portfolio management services and various account types considering various nuances like tax-loss harvesting. For example, under this strategy, individual stocks representing an index are purchased instead of the index ETF such that they can be sold for tax-loss harvesting.

Alpha Factory considered the design decisions of these, and several other competitors when making their own design decisions and business plan – refer to the Design Decisions section below. Decisions pertaining to management fees, initial investment balances, portfolio management features and asset selection was of particular interest during the comparison. Next, we turn our attention to the Canadian robo-advisors to further consider the asset selection based on the market that's closer to home.

Looking at the Canadian counterparts we noticed that they also strived for the same common features as their American peers (namely, low initial investment, low fees, popular options and comprehensive personalized features) in various ways. For example, Nest Wealth aims to keep fees low by capping management fees a $80 regardless at how big your account grows, with the other four advertising management fees ranging from 0.35-0.70% and comparing the hypothetical returns you'd make with those lower fees with the 2.17% average mutual fund fee. [17, 18, 19, 20, 21]

Furthermore, all five of the robo-advisors considered offer customized portfolios matched based on every individual's investment style, risk tolerance and savings goals and provide a different range of other features in an attempt of differentiation. Additionally, they all cited "overwhelming research" suggesting that passive investment outperforms active investors attempting to beat the market, with Nest Wealth summarizing it best saying, "using this as our foundation, we build your portfolio to 'be the market', rather than try to 'beat the market.'" [22]

In looking at Nest Wealth, BMO Smartfolio, Wealthsimple, Questrade Portfolio IQ and WealthBar, what stood out most is their selection of the asset universe. Namely, they all built their investment portfolios from a different number of ETFs (e.g. Nest Wealth built portfolios from seven different ETFs, Wealthsimple

considered nine different ETFs, etc.) as they are a low cost and efficient way to ensure a diversified portfolio and gain exposure to various asset classes like bonds, real estate and equity. [23, 24] As such, Alpha Factory considered exploring creating an asset universe solely comprised of ETFs and conducted a deeper analysis of asset selection with this in mind – refer to the Asset Universe Selection section below.

Appendix B – Additional Wireframes

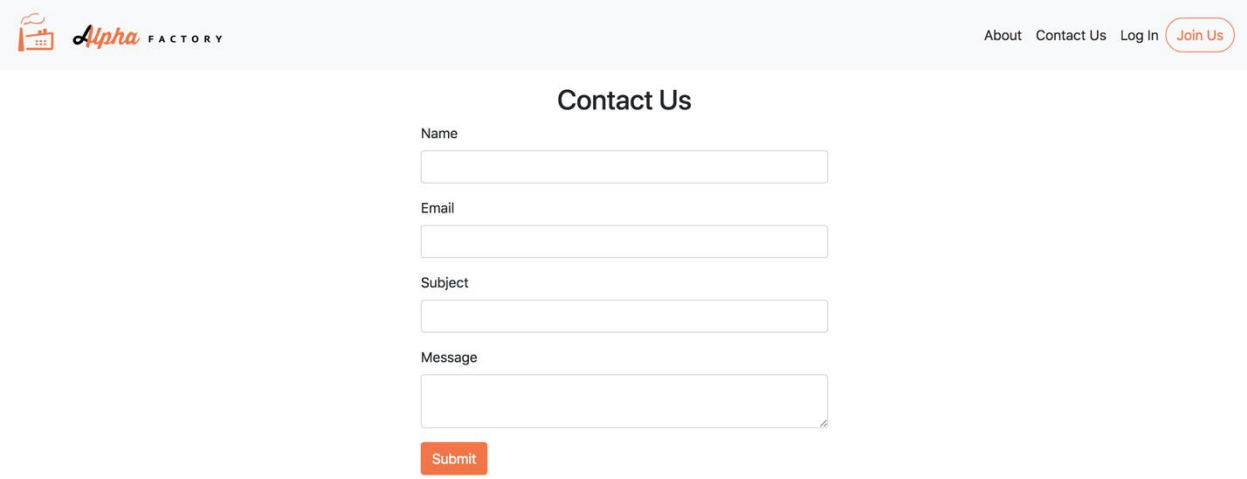


Figure B1: Contact Us Page

Accessible through the Contact Us button in the menu bar for both the main site and user site, the contact us page is available for everyone to access. By filling out the form completely, an email is automatically sent to the Alpha Factory Gmail account for our members to review. Upon completion, the user is notified that the message has been sent and that we will be in touch with them shortly.

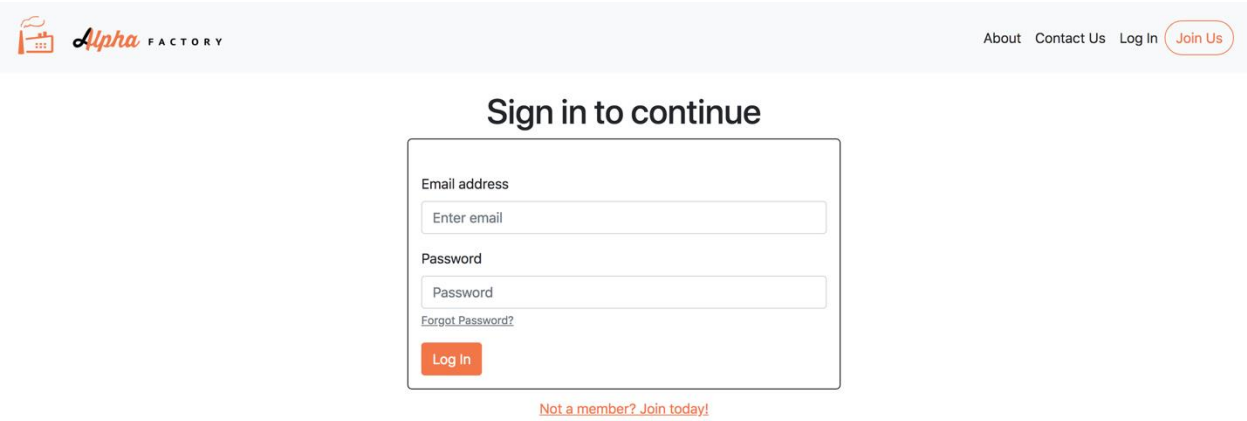


Figure B2: Sign In Page

The Log In page is available through the Log In menu link on the main site and directs the user to a login form. The user can enter their credentials and, after successful validation through the backend system, the user will be able to access the user site. If the user does not have an account, they can be directed to the Join Us page through either the 'Not a member? Join today!' link or the Join Us button in the menu bar. If the user has forgotten their password, they can select the 'Forgot Password?' link and they will be directed to the Reset Password page below.

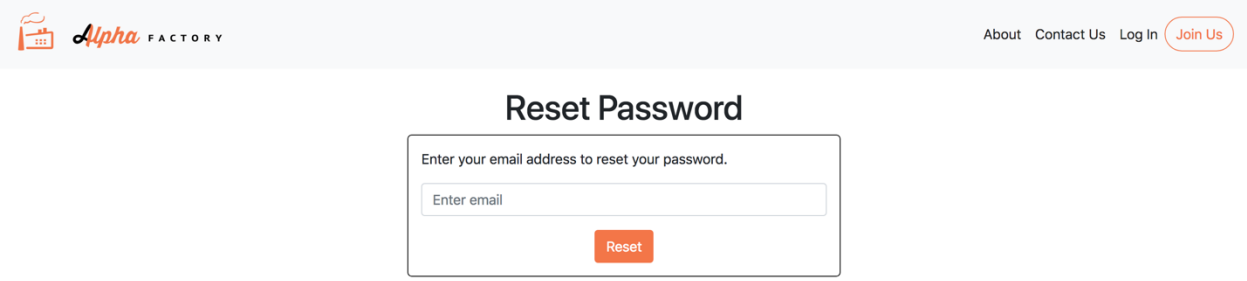


Figure B3: Reset Password Page (forgot password)

If the user wishes to reset their password, they may enter their email and a verification code will be sent to the user's email. The user will then be prompted by the form in the page below to enter the code they received from their email. They can access/navigate away to the main site through the menu bar.

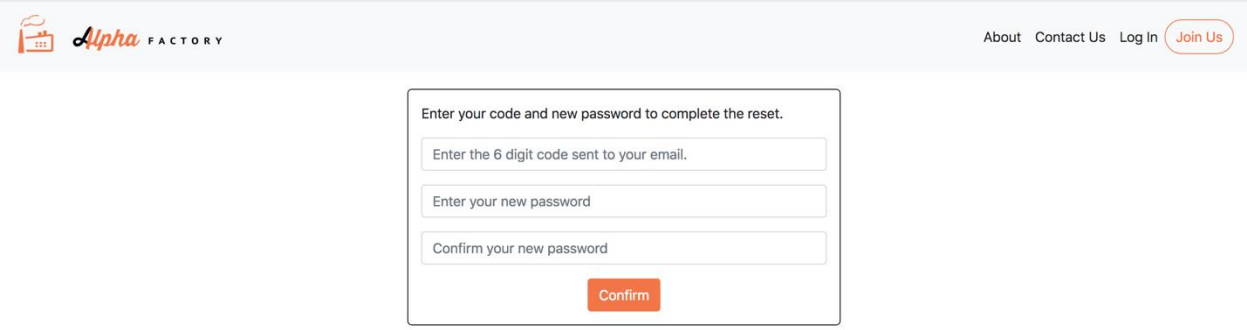


Figure B4: Reset Password Page continued

After the user has received their verification code, they can enter the code as well as a new password into the fields above. If the verification code and the confirmed/new password match up, the user's information will be updated in the backend and they will be directed to the log in page where they can sign in with the new credentials. If the user chooses to, they may access their account information through the menu bar as shown below.

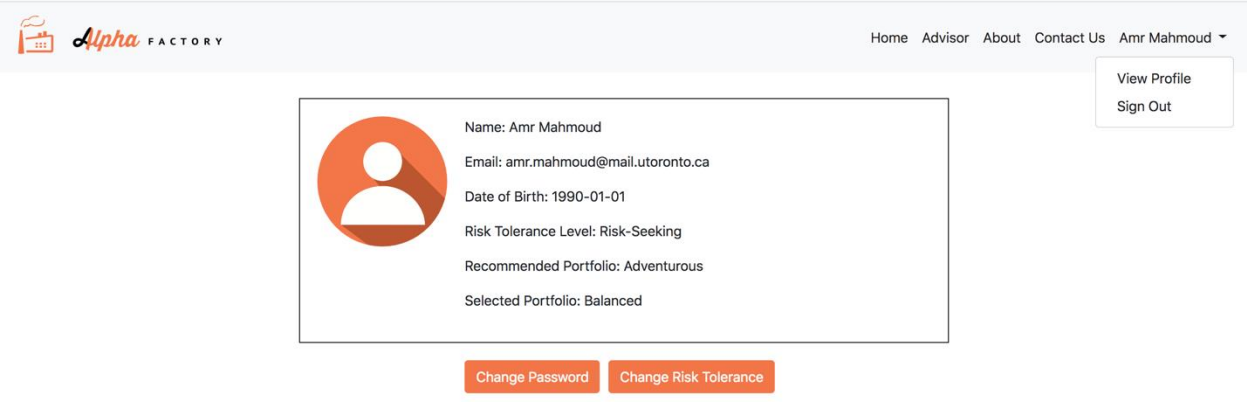


Figure B5: View User Profile Page

The user has the ability to access the information related to their account through the profile page, which can be accessed through the menu bar under the user's full name -> 'View Profile' link. On the page, the user can select to change their password and risk tolerance. In each case, the user will be prompted to confirm their decision. In the case of a user wishing to change their risk tolerance, they will be required to fill the questionnaire again. It is also possible for the user to sign out through the 'Sign Out' menu link displayed in Figure B5 above. Clicking on sign out will lead the user back to the landing page of the main site ending their session.

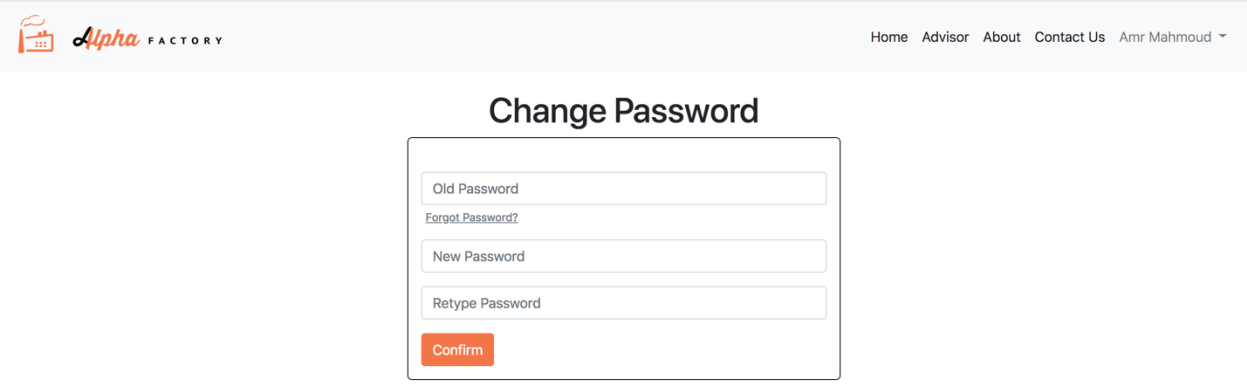


Figure B6: Change Password page

If the user selects to change their password, they will be lead to a page where they can fill out the information required in order to change their password. Once the required information is provided, the data will be sent to the back-end to update the user information in the database. If they have forgotten their password, they can click the "Forgot Password?" link which will lead them to the Reset Password page in Figure B3.

Appendix C – Original Web Flow and Technology Decisions

The user interface was designed with simplicity and user friendliness as a top priority. With that said, the design decisions and flows of the wireframes reflects these values. Knowing the importance of brand name and reputation, the logo was designed with extensive coordination and collaboration among all team members. The logo colours (orange, black and white) were taken as the colour scheme/theme throughout the website. Figure C1 below demonstrates how a user will traverse through our website.

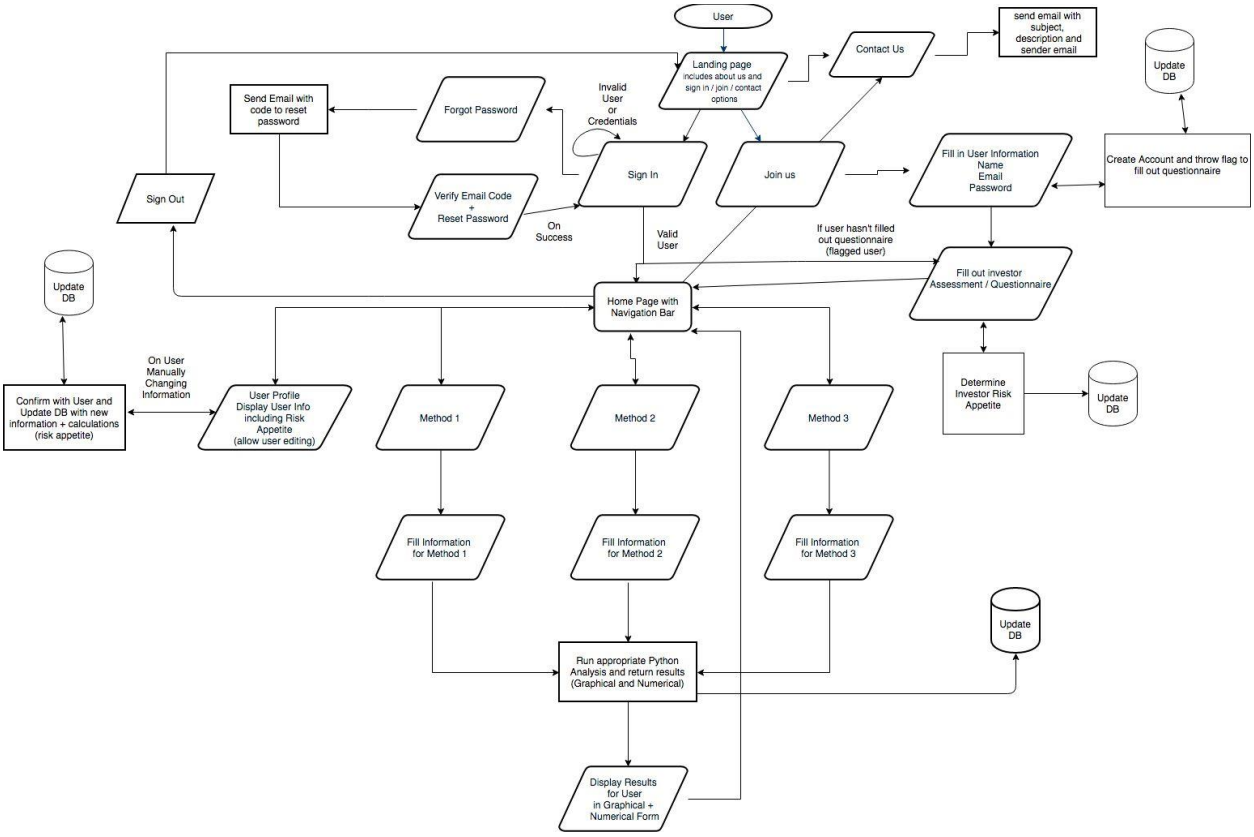


Figure C1: User Flow

The user flow was motivated by a generic user-friendly interface and references those exhibited by similar robo-advisor websites (e.g. Wealthsimple, Nest Wealth and WealthBar). The user will access the website through the hosted URL of choice and arrive at the landing page.

Landing/About Page

The landing/about page will be the first page that the user will see upon accessing the site through the URL. Upon arrival at the landing/about page, the user will be shown information about the site and our functionality as well as be given the option to sign in, join us, or contact us.

Contact Us

If contact us is selected, the user will be required to fill in information regarding what they would like to contact us about. After they have filled out the form and sent it, it will run through the back-end which will result in an email sent with the subject matter, sender, and description. The user will then be informed that their message has been sent and that we will be reviewing it shortly.

Sign In

If sign in is selected, the user will be prompted to enter their user credentials which will require their chosen email and password. The user will also be given the option to reset their password in the event that they forgot it.

If they press forgot password, the user will be led to a page where they can enter an email to send a verification code. An email will be sent to the user with a verification code required to enter on the next page. The page that follows will be a page where the user can enter the verification code and type in a new password and confirm it by re-typing the password. They will then be led back to the sign in page where they can enter their new login credentials and access the main site.

Upon successful sign in, the user will either go to the home page which displays the different functions of the robo-advisor or, if they haven't filled out the questionnaire, they will be prompted to fill it out.

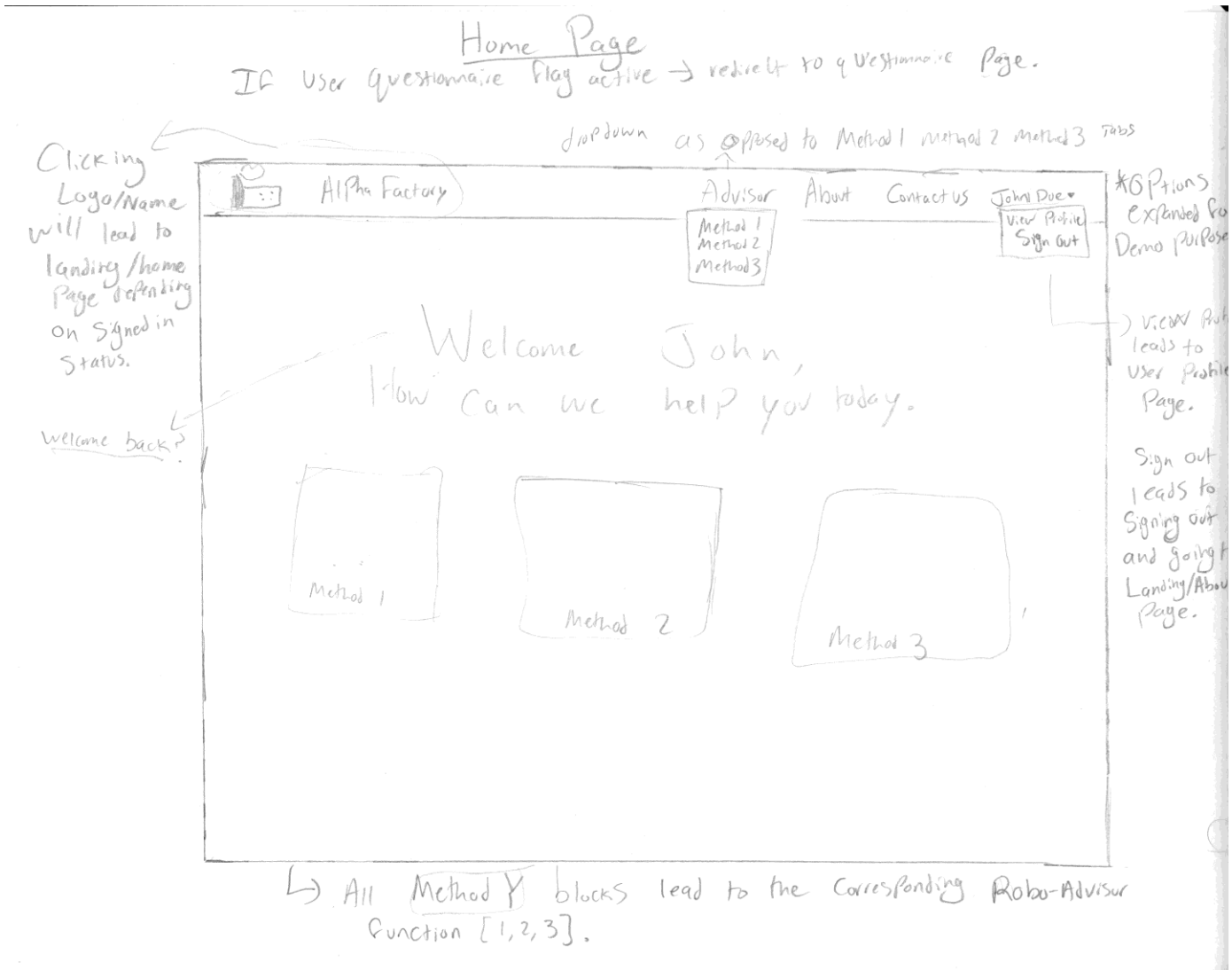
Join Us

If the user selected the join us button, they will be led to a page that requires them to fill out a form that has the following fields: First Name, Last Name, Email, Password, and Date of Birth. The fields will be checked appropriately to ensure they are valid entries and screened before they are sent for processing. Upon completion, the information the user has provided will be sent to the backend where a user object will be instantiated and updated in the database with a flag thrown requiring the user to fill out the questionnaire. An example of a valid entry screening is ensuring that all fields are complete and appropriate (e.g. for Date of Birth, we require all members to be 18 years or older).

The user will then be led to a questionnaire page that asks them a series of questions to determine their risk appetite and investor mentality. The current setup for the questionnaire functions by allowing the user to select scenarios that better suit them. The Business Logic decided which questions were required

and sent it to the front end to populate the two options for every question. A progress bar tracks and displays the users' progress throughout the questionnaire and allows them to go back to change previous answers as required. On the final question, the next button turns into a finish button. The answers are recorded and sent to the backend to be integrated with the business logic to calculate and determine the users' risk tolerance. Upon successful completion, the user will be led to the home page.

Home Page



Initial designs planned to allow the user to access the 3 different functions of the robo-advisor only after logging in. It also allowed them to view their user profile, all from the navigation bar or home page screen. The home page was initially planned to be extremely simplistic with only the navigation bar and 3 square elements on the page that will lead users to the different functions of the robo-advisor. During the initial design of the web application the 3 functions of the advisor (as laid out by the capstone project requirements) were respectively named Method 1, Method 2, and Method 3. As this was very unintuitive

and provided little use apart from fulfilling the project requirements the use of these functions was later changed through countless iterations ultimately leading to the web application we presented.

Navigation Bar

Initially, the navigation bar was planned as a sticky header element located at the top of the page regardless of the user location. The user will have the option to navigate to: Home Page (through logo), Method 1, Method 2, Method 3, About/Landing page, Contact Us, and user profile page/sign out. However, the design team decided to simplify the navigation bar to only provide the user with the About Page, Contact Us and Log In / Join Us options – this was selected in an attempt to make the user experience flow as seamlessly as possibly without the distraction of unintuitive buttons.

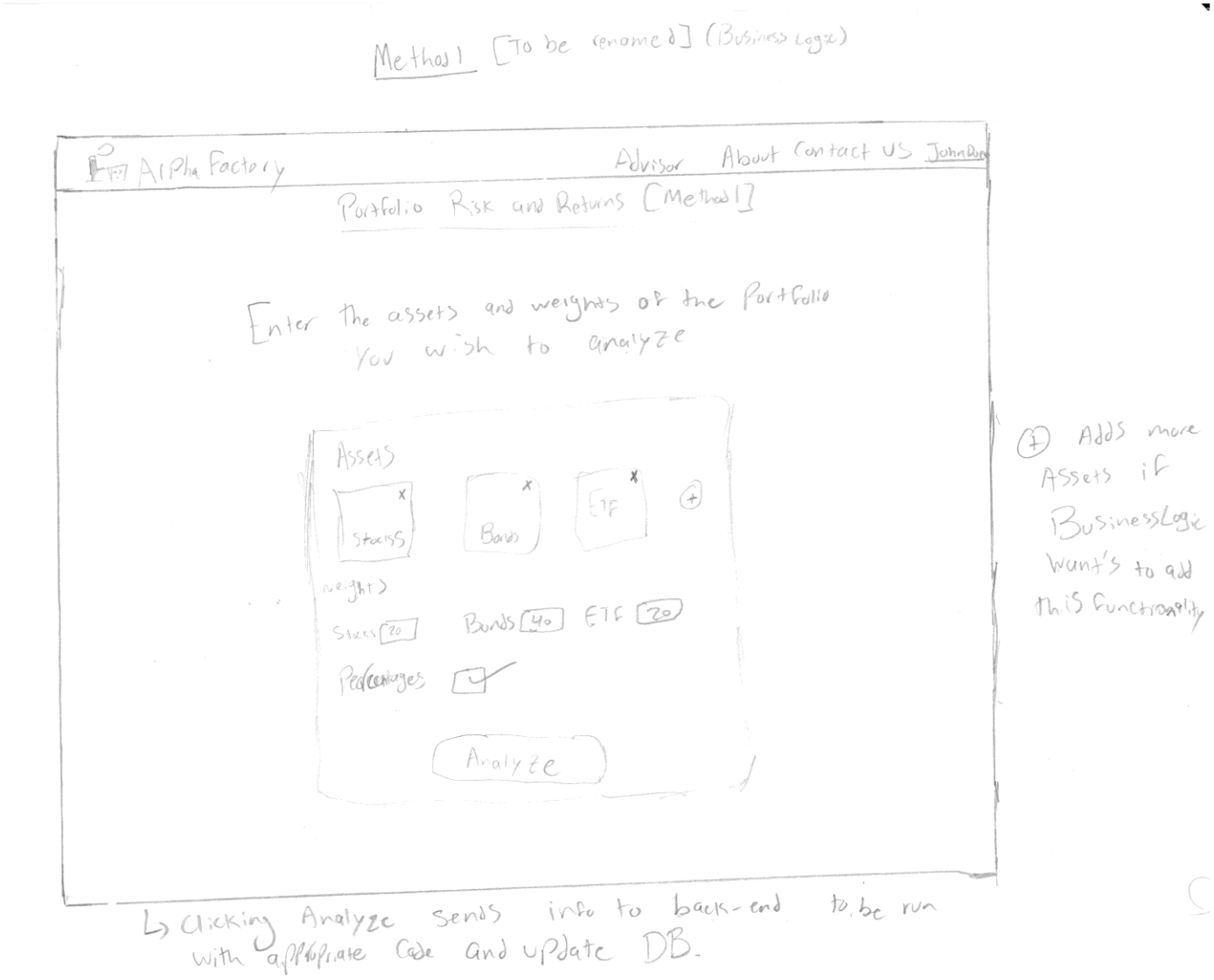
User Profile

When the user selects to view their user profile through the navigation bar, they will be directed to a page that contains the information pertaining to their account including name, date of birth, email, change password(button) and risk appetite. Initially we planned to include the history of user's risk tolerance levels to display how it evolves over time, but this idea was not implanted as it provided little value add to the demonstration and the design team did not have enough time to implement it during the tight project deadlines. This is definitely something that we'd hope to implement going forward as monitoring the trends of changing risk-tolerance levels is something that we could be mindful of in our portfolio strategies and product offerings (i.e. we'd expect individuals to become more risk-averse as they approach older ages – monitoring how risk-tolerance changes overtime on aggregate might inform investment strategies and methodologies going forward. The user must be able to edit the information (aside from name) upon clicking an edit button or pressing change password. While we initially planned for the risk appetite to be manually changed, we decided against it. Rather, we decided to allow users the ability to pick any portfolio they wanted despite being recommended a different portfolio based on their questionnaire. As such, we view risk-tolerance level as an inherent quality of the investor, while the portfolio they hold is a choice they can make (Alpha Factory merely recommends a portfolio based on their risk tolerance level, but we are happy allowing our customers to choose a portfolio for themselves). The resulting changes will be recorded and sent to the backend for validation and updating of the DB.

Sign Out

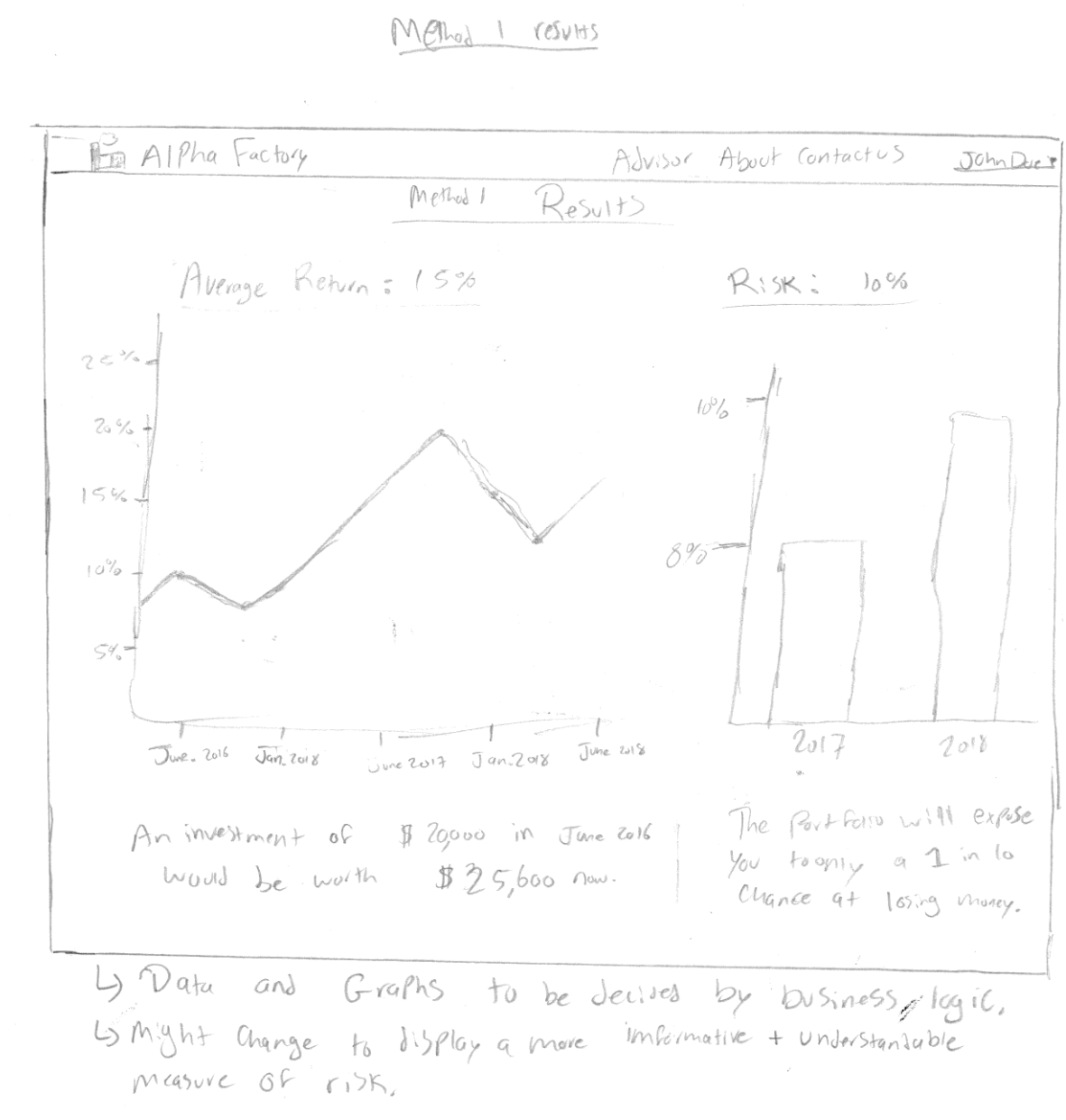
The user can choose to sign out of their account through the navigation bar under the profile section. They will be led to the landing page if they select this option.

Method 1



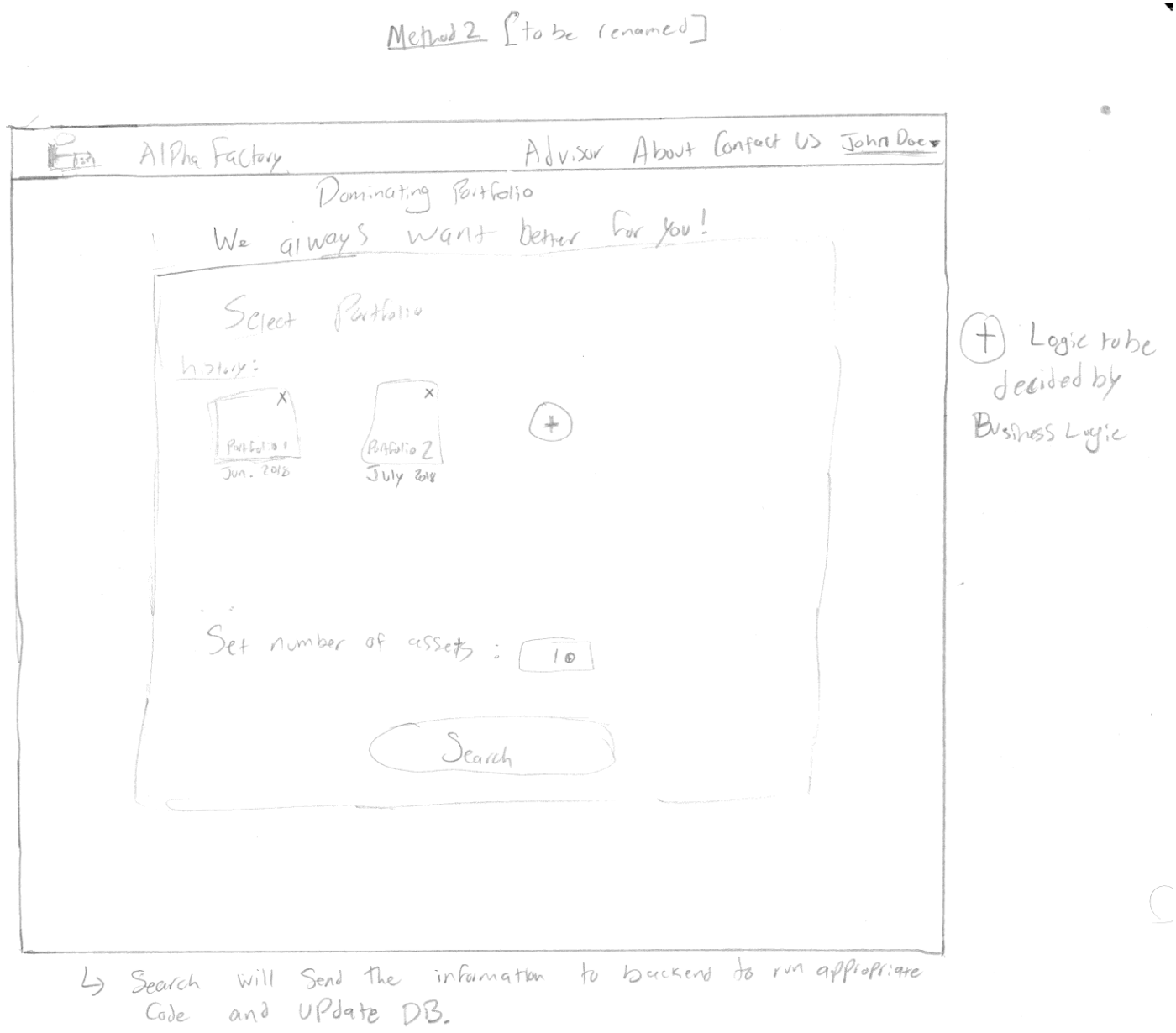
Initially, we planned that function 1 to be its own function (called Method 1 during our initial design phase). Upon selecting Method 1, it was planned for the user to be led to a page where they will have to fill out the information required to run Method 1 through the backend. Upon validation that the fields are not empty and are valid (in range), the information will be sent to the back end where it will undergo further validation and finally be run through the Python script to generate the required output. The output will also be backtested and stored in the database for future reference.

Ultimately, during various iterations this was changed and function one was run during competition of both functions 2 and 3. To simplify the user experience, we ultimately decided to simplify our web application by not providing a separate location for function 1 as it is unintuitive and adds culture to our web page. Instead, we decided to backtest portfolios that are either generated by our platform (function 3) or inputted by users on the main site (function 2).

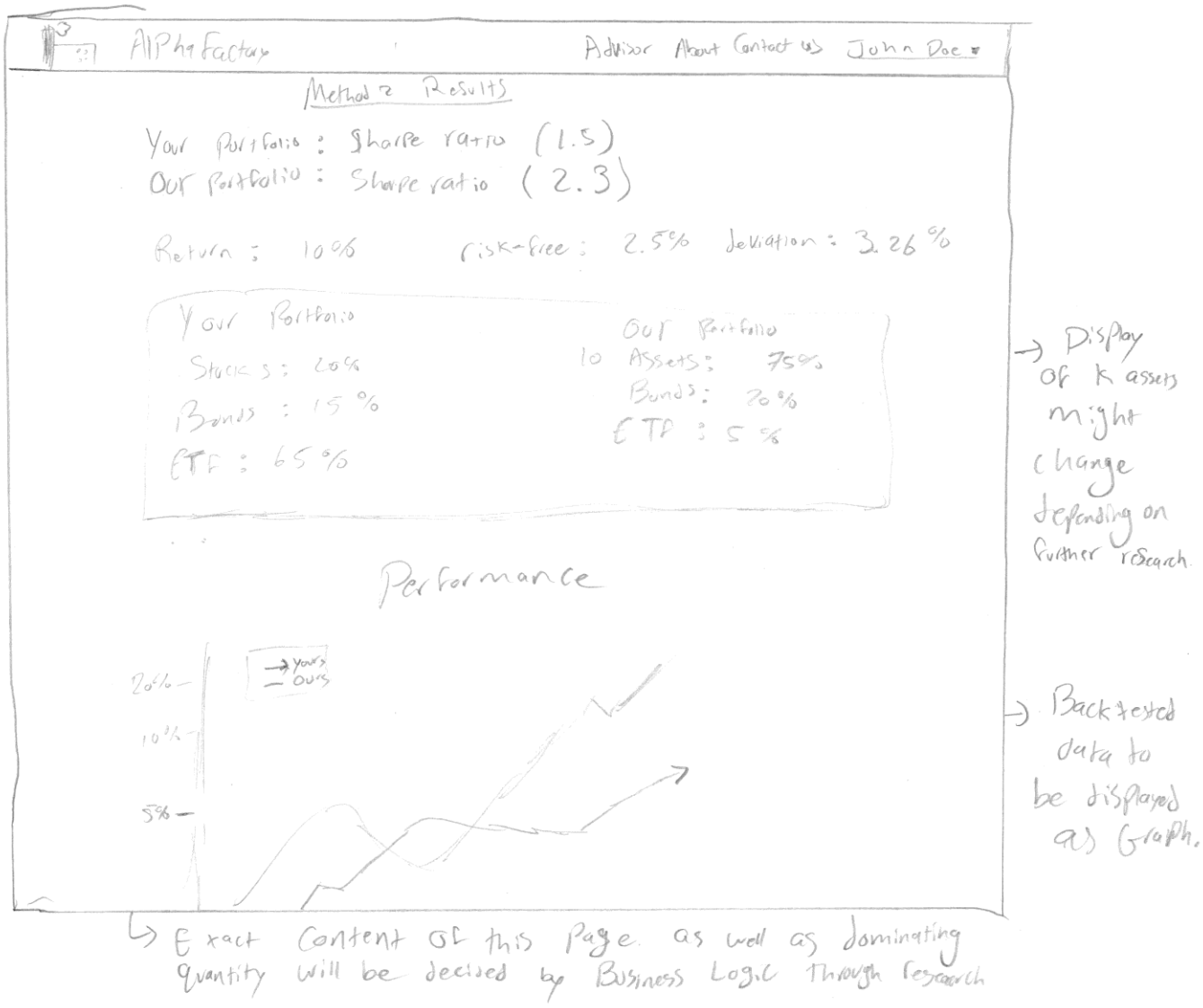


The back-end will return the results to the front-end with both numerical and graphical display information where the front end will process the information to produce informative and easy to understand graphs and results. The user can navigate away from this results page through the navigation bar.

Method 2



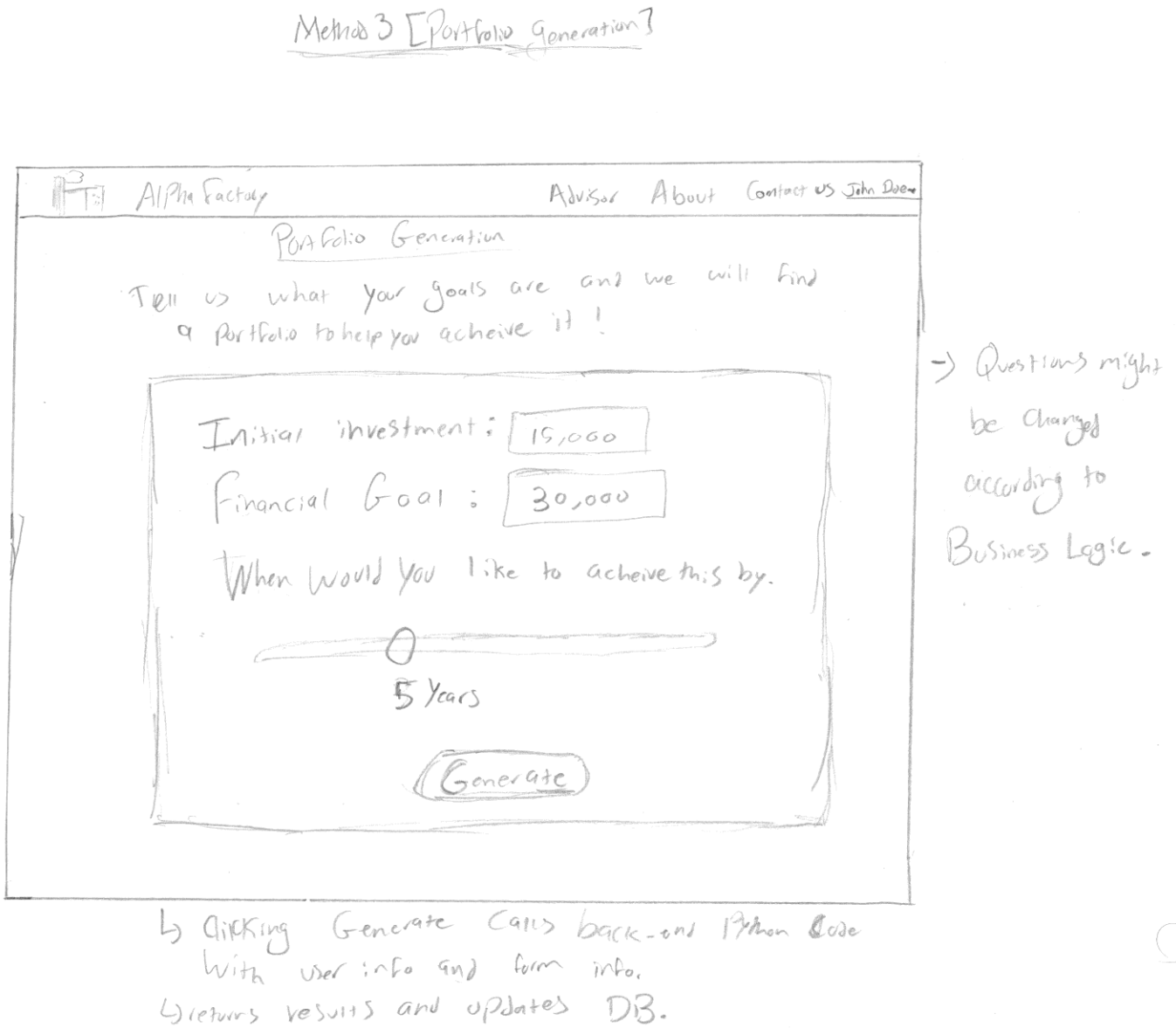
Initially we planned Method 2, to lead the user to a page where they will have to fill out the information required to run Method 2 through the backend after section. Upon validation that the fields are not empty and are valid (in range), the info will be sent to the back end where it will undergo further validation and finally be run through the Python script to generate the required output. The output will also be back tested and stored in the database for future reference.



The back-end will send back the result to the front end with both numerical and graphical display information where the front end will process the information to produce informative and easy to understand graphs and results. The user can navigate away from this results page through the navigation bar.

While this is still how function 2 largely runs, we decided to offer it as part of our About page on the main site. While the function still works by inputting quantities of assets to be held, and displaying the results that follow, we tried to simplify the process by making the function flow with our About page. Rather than acting as a separate page that you need to click into through buttons, it is one that hopefully naturally supplements our “Learn More” information provided. Ultimately, we hope that by providing this function our portfolio generation strategy/platform gains credibility, encouraging users to join our platform.

Method 3



Upon selecting Method 3, the user will be led to a page where they will have to fill out the information required to run Method 3 through the backend. Upon validation that the fields are not empty and are valid (in range), the info will be sent to the back end where it will undergo further validation and finally be run through the Python script to generate the required output. The output will also be back tested and stored in the database for future reference.

The back-end will send back the result to the front end with both numerical and graphical display information where the front end will process the information to produce informative and easy to

understand graphs and results. The user can navigate away from this results page through the navigation bar.

Technologies and Design Decisions

Gathering User Input(s) and Feedback

For gathering data, a standard form method will be used on all pages. Forms will be generated with validation built into them to ensure that initial data is feasible and sufficient. A lack of data or invalid data will result in the user being unable to proceed or submit their input. Further data validation and confirmation will occur in the back-end where the process will also be running on valid data.

For the forms, a standard template will be used in combination with the Form Bootstrap component. This component was selected in particular due to its simplistic implementation, beautiful design, seamless feel and excellent user experience. It is extremely well documented, and the implementation is sufficiently simple such that we can use it in all our form requiring pages.

Some elements required a little more consideration and thought than others. One such element was the decision to use a slider on the portfolio generating function (Method 3) for the time horizon. On sites such as Wealthsimple, they use sliders to allow the user the ability to see returns graphed over different time horizons. The reason we chose to go with this approach over tradition dropdown/input form options is because it not only limits the range of user input but it also allows the backend to set pre-determined methods for the different time horizons so that it can better support them instead of having a backend that is dynamic to the point of vagueness when it comes to time horizons. The slider is also extremely intuitive and easy to use for most users and should therefore propose no learning curve for any unexperienced members.

Another decision revolving around gathering data that was particularly impactful was the choice to make the questionnaire only a left or right option type of questionnaire as opposed to the traditional form questionnaire. Traditional form questionnaires are too plain, simple, boring, and non-modernized. Millennials today are very active and can't sit through a 10-20 question form that they have to fill in one by one. It is also particularly difficult to have to set the many different ranges, input types, and values for all the different form answer types. It is with this idea in mind that we decided to follow other approaches to questionnaires which included the method that we chose. We chose a method that makes the process seamless and even fun for the user. As long as the number of questions, n, decided by the Business Logic is not too high (15+), the user will breeze through the questions since they only have to consider 2 options. This also restricts the number of things they can enter right down to two options so that the backend doesn't have to deal with outlier data, ex: a user can't select a salary that is abnormal such as $20/month. Although the structuring of the questions will require more thought and a higher level of effort to calculate the resulting risk tolerance, the user experience is tremendously impacted in a positive manner which is of the highest priority when we designed the front-end.

Display of Computed Data

For displaying the computed data, there was a decision to be made on the method of producing and displaying graphs/other information. This decision primarily revolved around 3 different charting libraries that would make the chart displaying and implementation significantly easier. The three libraries in consideration were:

1. Bokeh
2. Plotly.js/dash
3. Chart.js

Each of the libraries had their perks and their draw backs. Table C1 below displays some of the perks and drawbacks of each of the three options.

Library	Pros	Cons
Bokeh	• Wider range of interactions than Dash • Large number of examples on library usage and integration with Flask	• More difficult to implement than Dash • Learning curve can be steep with little documentation and implementation
Plotly.js/Dash	• Very well documented • Contains a wide breadth of functions for users to filter, analyze, print, and customize when it comes to graphing • Possible to attach React components on top of the existing graph	• Charting can be complex to an unexperienced user • Integration with flask and some of the other technologies chosen in the next section is tough • Higher learning curve for implementation
Chart.js	• Extremely simple to use and populate graphs • Intuitive and simplistic graph display for users • Implementation and integration are not difficult • Easy integration with Jinja2 templating	• Limited graphs available to display • Graph customization is minimal

Table C1: Pros and Cons of Data Libraries

The final decision on the library selected for creating the graphs is based on our values for the project. The greatest thing that we value is user experience and design simplicity. With that said, Plotly.js/Dash was chosen over Bokeh even though they were extremely similar in what they can produce. The main difficulty with Bokeh is in implementation, which was directly implemented and researched to be tougher to use than Dash. [40] The comments on the article contain many developers that voiced their concern over the use of Bokeh while praising Dash. The main problem with Dash boiled down to the fact that Dash is built off the Flask framework. This causes a disconnect when trying to integrate Dash plots and Flask applications. We would either need a middleware dispatcher or we would have to route to a different server where the Dash plot was being held. Since this would cause significantly more work and would

potentially lead to more problems, the decision then came down to one of Chart.js and Bokeh. Since we must use Flask to build a RESTful web application, we had to find a corresponding library that can work with Flask and the technologies we chose to use with it (discussed in the next section). With this in mind, we decided to go with the charting library that would best fit our values of making the user experience as great as it can be. The result was that Chart.js was selected for the graphing purposes of this application. While Chart.js has minimal charting capabilities and customization, it excels in producing simple, easy to understand, and easy to implement graphing. This reflects our priority to make the user experience as friendly as possible without sacrificing functionality.

Decision on Technologies Used

Alpha Factory had to make decisions regarding which styling technologies and templating / front-end development technologies to use. Those specific decisions are further explored in the subsections below.

Styling Technologies

Library	Pros	Cons
Bootstrap	• Heavily documented • Plethora of examples, websites, and help forums • Beautiful and simple designs of just about everything we will need • Highly user friendly • Allows for easy customization	• Bootstrap 4 isn't as supported by older versions of web browsers • Not necessarily better than Pure CSS and Foundation when it comes to element design
Pure CSS	• Very simple and minimalistic • Beautiful cascade of colours and other options for the elements	• Less comprehensive than Bootstrap or Foundation • No Icons available • Licensed by Yahoo
Foundation	• Used by major sites that rely on user experience (Facebook, Yahoo, eBay) • Supports a wide breadth of old and new browser versions • Similar to Bootstrap in possibilities and usage	• Learning curve is steeper than Bootstrap without prior experience in CSS

Table C2: Styling Technologies

The decision on which frontend technologies to use was based on both research and actual testing implementation that we did. To start, we decided the styling library by using a mix of researched examples and JSFiddle. JSFiddle is a code editor that can test JS, HTML, and CSS files. It was quite simple to set up

and fiddle around with existing examples to see how some of the elements and forms would look by directly implementing them. Check out the links below to see how each of the styling elements would look.

1. Bootstrap 4: https://jsfiddle.net/5vspyL1a/
2. Pure CSS: https://jsfiddle.net/v24x79db/
3. Foundation: https://jsfiddle.net/v6yhom7L/

It is clear to see that Bootstrap dominates the competition in both simplicity, design, and implementation. For this reason, we have decided to stick with Bootstrap 4 as the primary styling library to make our job of designing clean beautiful UI very easy.

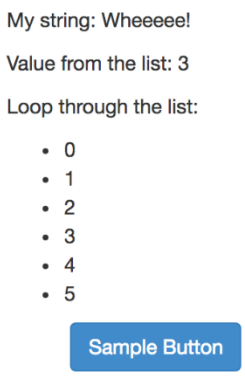
Templating vs Frontend development Technologies

Library	Pros	Cons
Angularjs	• Extremely well documented • Easy to implement and deploy • Large number of elements with different functionalities for all types of usages • Integrates well with Dash and other technologies	• Usually used for SPAs (Sing Page Applications) • Client-side rendering • Steep learning curve with no prior experience
Jinja2	• Default templating framework in use with Flask • Simple, easy to use • Allows for reusable elements (navbar, pages) • Allows for server-side rendering	• Limited integration with other frontend technologies like Angular • Harder to create a seamless user experience using only templates
React.js	• Diverse range of applications and documentation • Provides support for modular, mobile, and responsive web apps • Supported by Facebook and used by Uber, Airbnb, Twitter and more • Prior experience using React • Integrates well with Dash and other technologies	• Integration with flask can be complex at times • Newer framework with not as much support/documentation for bugs as Angular

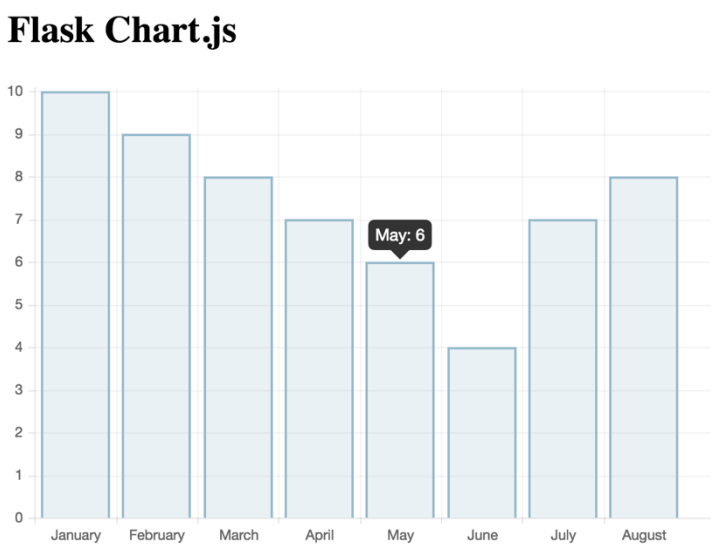
Table C3: Development Technologies

The hardest choice to make came from deciding what type of frontend technology would be used to create the HTML pages that the user will view. The battle was centered around JS related front end technologies

like React.js and Angular JS against the traditional and default templating framework of Jinja2. The primary reason we decided to go with Jinja2 is the fact that it is a server-side rendering framework. Airbnb has mentioned why server-side rendering is the better option when it comes to user experience in this article about the differences between JS libraries and templating frameworks: “First and foremost, server-side rendering is a better user experience compared to just client-side rendering. The user gets the content faster, the webpage is more accessible when JS fails or is disabled, and search engines have an easier time indexing it.” [41] For this reason, we decided to go with the default templating language, Jinja 2. Although Angular JS is a phenomenal framework that could be potentially integrated with Jinja and Flask, the difficulties it introduces, and the learning curve required is much too high a cost in comparison to the use cases and project timeline/requirements. And although the frontend member has prior experience with React, the use of React with Flask would require a significantly larger amount of coding. The reason for this discrepancy is the fact that React would require its own routing in the frontend (client side) and Flask would have to be used as a RESTful API which would also need the appropriate coding, or “wiring”, for the backend (server side). To meet the project deadlines, the idea to use React was put to the side as we wanted to focus our time into creating a great product instead of doubling our work and debugging when we can avoid it by using a different framework. We believe that user experience is top priority and therefore we have decided to build the front end entirely based on a design for user friendliness. To make sure our decision is correct, we ran an initial test on the technologies chosen above to make extremely simplistic pages. We decided to run a quick test project using Flask, Jinja2 and Chart.js. The test project folder can be found under the GitHub repository as “Test Folder”.



The first page we created was a simple page that uses Jinja2 templating to populate a dynamically created page while also using Bootstrap to style a button element. The list elements and string values were sent to the template and populated within the server. The button has no functionality but stands to show how easy it is to integrate and use Bootstrap. Overall, the resulting page took less than 5 minutes to setup without following the tutorial and 2 minutes to set up by strictly copying and pasting the code.



The second page was an extension of the home page under the route “/simple_chart”. By following some tutorials, we were able to create a Chart.js chart and populate it with static data in under 20 mins. This confirmed two things to us which gave us a strong sense of confidence in our choices. First, it allowed us to show that we can use Chart.js, Jinja2, and Flask to create simple, intuitive, and beautiful graphs that can be viewed and understood by the user with no learning curve for inexperienced users. Second, it confirmed that the technologies we chose allows for simple integration and implementation such that we can create the best site possible with as few hiccups/bugs/delays as possible all in order to meet the project deadlines while delivering an amazing project.

Appendix D – Functionality Testing Scenarios

#	Scenario	Action	Expected Result
1	New/Existing User accesses the site to browse for information and contact us for more details	1. Access the site through the URL 2. Browse the landing/About Us page 3. Explore all the menu links 4. Access the Contact Us page through the menu link 5. Fill out contact form and send email	1. User is shown landing page/home page depending on the logged in status 2. User is able to access the About Us page with the corresponding information displayed 3. User is able to freely access all parts of the main site/user site depending on their logged in status 4. User is able to access the Contact Us page and the contact Form is displayed 5. User is able to fill out form and send email successfully (receipt of corresponding email and contents must be confirmed). User is shown a confirmation of receipt screen

#	Scenario	Action	Expected Result
2	New user trying out our platform through Function #2 and deciding to join us	1. Access the site through the URL 2. Scroll to the Function #2 form. 3. Fill out form with corresponding information 4. Review results and select Join Us 5. Fill out sign up form 6. Answer risk assessment questionnaire 7. Input financial goals 8. Review generated portfolio	1. User is shown landing page 2. User is able to see Function #2 form 3. User is able to fill out correct information in form and click Enter 4. User is returned a comparison of results of the two portfolios over several different time horizons. User is able to click 'Join Us' button in the same section or menu link 5. User is able to fill out sign up form and click Sign Up 6. User is shown the correct questions of the risk questionnaire with a progress bar and back/'previous' question button. (all must be correct and working) 7. User is shown the advisor page with the ability to enter their financial goals into the form and generate a portfolio. 8. User is redirected to the Home page with portfolio information and statistics displayed

3	Existing user logs in to access their portfolio and change their risk tolerance	1. User accesses the site through the URL 2. User chooses to Log In from the menu link 3. User fills out log in form and clicks Log In button 4. User accesses the Home page with portfolio information displayed 5. User accesses the profile page through the menu link 6. User selects edit information 7. User selects to manually change risk tolerance 8. User agrees to the prompt 9. User fills out questionnaire 10. User is able to see new risk tolerance	1. User is shown landing page 2. User is able to navigate to the log in page and see the form 3. User is able to fill the form and select log in 4. Upon verified credentials, use is directed to the Home page with the correct corresponding portfolio information and statistics displayed 5. User is able to access and view profile Page and its corresponding information 6. User is able to click edit button to change their profile information 7. User is able to select to change risk tolerance and is prompted a confirmation alert informing them they must answer a questionnaire 8. User is taken to the questionnaire page and the corresponding questions are displayed 9. User is able to navigate through the questionnaire with ease 10. User is lead to Home page, on completion with the new risk tolerance (and generated portfolio) updated in the db and visible on the profile page

#	Scenario	Action	Expected Result
4	Existing user forgot their password and wants to reset it	1. User accesses the site through the URL 2. User clicks the Log In page 3. User clicks the 'Forgot Password?' link 4. User fills the reset password email form and selects 'Reset' 5. User types in code from received email and fills out the reset password form 6. User clicks 'Confirm' 7. User logs in with the new password	1. User is shown landing page 2. User is directed to the Log In page with the log in form displayed 3. User is directed to the forgot/reset password page 4. User is able to fill out form and select 'Reset'. Email is sent to user with confirmation code included 5. User is able to fill out the reset password form and validation of variables is expected to work correctly 6. User information is updated in the db and user is directed to log in page 7. User is able to log in with the new credentials and is directed to the home page

Appendix E – Calculation of Transaction Costs

ETF Ticker	ETF Name	Bid-Ask Spread (Bps)[1]	Average Weight[2]
MTUM	iShares Edge MSCI USA Momentum Factor ETF	0.03	1.6%
VLUE	iShares Edge MSCI USA Value Factor ETF	0.03	1.7%
QUAL	iShares Edge MSCI USA Quality Factor ETF	0.03	1.8%
SIZE	iShares Edge MSCI USA Size Factor ETF	0.14	2.0%
USMV	iShares Edge MSCI Min Vol USA ETF	0.02	2.2%
IVLU	iShares Edge MSCI Intl Value Factor ETF	0.23	1.4%
IMTM	iShares Edge MSCI Intl Momentum Factor ETF	0.37	1.4%
IQLT	iShares Edge MSCI Intl Quality Factor ETF	0.27	1.5%
EMGF	iShares Edge MSCI Multifactor Emerging Markets ETF	0.27	3.9%
ACWV	iShares Edge MSCI Min Vol Global ETF	0.09	4.2%
SPTL	SPDR Portfolio Long Term Treasury ETF	0.05	7.4%
AGG	iShares Core U.S. Aggregate Bond ETF	0.01	13.0%
EMB	iShares J.P. Morgan USD Emerging Markets Bond ETF	0.01	5.5%
TIP	iShares TIPS Bond ETF	0.01	12.0%
HYG	iShares iBoxx $ High Yield Corporate Bond ETF	0.01	7.6%
SHV	iShares Short Treasury Bond ETF	0.01	24.1%
SCHH	Schwab U.S. REIT ETF	0.03	2.5%
DBC	Invesco DB Commodity Tracking Fund	0.06	3.0%
GLD	SPDR Gold Shares ETF	0.01	3.4%
	Weighted Average Spread	**0.04**	

[1]Average 1 Month Bid/Ask Spread (Source: https://screener.fidelity.com/)

[2]Average weight of ETF over the total history of portfolio back test

Appendix F – Decision-Making for Database Selection

The other possible DBMS solutions that could have been used in place of MongoDB ranged from a variety of products that run exclusively on a Structured Query Language (SQL) and those that don't require SQL for manipulating data. The decision to use the NoSQL MongoDB over a traditional SQL Relational DBMS (RDBMS) was discussed in the Initial Design Report, with the main decision factors listed below:

- Document oriented storage for lenience in creation of a variety of document types under one single collection of data
- Efficient querying to be able to quickly find the relevant data for any business process and effectively search at any level of data access

To further clarify this decision to use a NoSQL DBMS, and MongoDB in particular, please refer to the tables below, which were used to compare different alternatives that were found when making the initial selection.

List of possible RDBMSs

When investigating the possibility of using a SQL-oriented database, the three options that were most closely considered were Microsoft SQL Server (MSS), MySQL, and PostgreSQL. Some of the key features for each option are summarized below:

Name	Microsoft SQL Server	MySQL	PostgreSQL
Description	Microsoft's relational DBMS	Widely used open source RDBMS	Widely used open source RDBMS
Primary database model	Relational DBMS	Relational DBMS	Relational DBMS
Secondary database models	Document store Graph DBMS Key-value store	Document store Key-value store	Document store Key-value store
DB-Engines Ranking	Score 1051.55 Rank #3 Overall #3 Relational DBMS	Score 1159.89 Rank #2 Overall #2 Relational DBMS	Score 440.24 Rank #4 Overall #4 Relational DBMS
Developer	Microsoft	Oracle	PostgreSQL Global Development Group
Initial release	1989	1995	1989
Current release	SQL Server 2017, October 2017	8.0.12, July 2018	11.0, October 2018
License	commercial	Open Source	Open Source
Cloud-based only	no	no	no
DBaaS offerings (sponsored links)		Google Cloud SQL: A fully-managed database service for the Google Cloud Platform	Google Cloud SQL: A fully-managed database service for the Google Cloud Platform
Implementation language	C++	C and C++	C
Server operating systems	Linux Windows	FreeBSD Linux OS X Solaris Windows	FreeBSD HP-UX Linux NetBSD OpenBSD OS X Solaris Unix Windows
Data scheme	yes	yes	yes
Typing	yes	yes	yes

XML support	yes	yes	yes
Secondary indexes	yes	yes	yes
SQL	yes	yes	yes

Table F1: Alternative SQL Database Solutions

After looking at the individual services, and the factors that went into the grading for each option, we agreed to trust the rankings provided by this chart, which give MySQL as the optimal choice between the three options that were considered. It is appropriate to mention that Oracle, which was the #1 ranked DBMS, was not considered for Alpha Factory because we were unsure of the range of features that was given with a free license.

List of Possible NOSQL DBMSs

The next step was to select a few contenders for NoSQL database services that would be considered given their preferable structure that doesn't rely on a query language for all functionality. The three alternatives selected were MongoDB, Amazon DynamoDB and Couchbase, which were given the top three rankings for Document-Oriented database management systems as shown in the table below:

Name	**Amazon DynamoDB**	**Couchbase**	**MongoDB**
Description	Hosted, scalable database service by Amazon with the data stored in Amazons cloud	JSON-based document store derived from CouchDB with a Memcached-compatible interface	One of the most popular document stores
Primary database model	Document store Key-value store	Document store	Document store
Secondary database models			Key-value store
DB-Engines Ranking	Score 53.81 Rank #21 Overall #2 Document stores #2 Key-value stores	Score 34.85 Rank #23 Overall #3 Document stores	Score 369.48 Rank #5 Overall #1 Document stores
Website	aws.amazon.com/-dynamodb	www.couchbase.com	www.mongodb.com
Technical documentation	aws.amazon.com/-documentation/-dynamodb	docs.couchbase.com	docs.mongodb.com/-manual
Developer	Amazon	Couchbase, Inc.	MongoDB, Inc
Initial release	2012	2011	2009
Current release		5.5.0, July 2018	4.0.3, October 2018
License	commercial	Open Source	Open Source
Cloud-based only	yes	no	no
Implementation language		C, C++, Go and Erlang	C++

Server operating systems	hosted	Linux OS X Windows	Linux OS X Solaris Windows
Data scheme	schema-free	schema-free	schema-free
Typing	yes	yes	yes
Secondary indexes	yes	yes	yes
SQL	no	SQL-like query language (N1QL)	Read-only SQL queries via the MongoDB Connector for BI
APIs and other access methods	RESTful HTTP API	Memcached protocol RESTful HTTP API	proprietary protocol using JSON

Table F2: Alternative NOSQL Database Solutions

We see that in general, the 'Document Store' database systems are newer and therefore less established and ranked lower than their RDBMS equivalents. However, they are still preferred because of the reasons listed above. When we see the performance of these options in the grand scheme of database management systems, we see that MongoDB outperforms its counterparts by a significant margin.

The final decision that was made was to compare the potential benefits and drawbacks of implementing MySQL as opposed to MongoDB. Here, we looked at the overall design objective of making the implementation of the database as seamless as possible with the rest of the project. The main factor in selecting the preferred database system was through the cohesion of each solution's API, a resource of paramount importance for reference purposes in order to ensure the proper use of the Python connector to the database. Here, MongoDB stood out with a very organized API that tackles each level of data access with a corresponding level of API documentation to give users a thorough tool for ensuring they can properly implement MongoDB's Python connector PyMongo properly.

Appendix G – Commented Code

In attempt to keep the size of the Final Report manageable, a separate ZIP file has been created containing the entire commented code. With Alpha Factory dedicated to reducing their ecological footprint, the commented code has only been submitted electronically.

Please refer to the FinalProjectCode.zip folder submitted electronically for Alpha Factory's entire code.

Appendix H – Data Used

In attempt to keep the size of the Final Report manageable, a separate ZIP file has been created containing all of the data used. With Alpha Factory dedicated to reducing their ecological footprint, the data used has only been submitted electronically.

Please refer to the AlphaFactoryData.zip folder submitted electronically for Alpha Factory's data used.

References

[34] Black Litterman Paper (from MIE377)

[3, 4, 5, 6, 9, 11, 13, 15] Blenman, Joy. "5 Robo-Advisors for 2018." *Investopedia*, Investopedia, 22 August 2018, www.investopedia.com/tech/top-robo-advisors/.

[18] "BMO SmartFolio." *BMO Financial Group*, www.bmo.com/smartfolio/?ecid=ps-SF1000SF3-CGDAD16&gclid=Cj0KCQjwl9zdBRDgARIsAL5Nyn0RhrcvR8mLzdzHHfRW1mMe2Nzh6Nh_uLN_ZjJRbvT9Q5mh1wfIPbcaAs6WEALw_wcB&gclsrc=aw.ds&dclid=CIzX0d7D790CFcLZwAodXdwGHw.

[43] "Bokeh vs Dash - Which Is the Best Dashboard Framework for Python?" *Sicara's Blog*, Sicara's Blog, 30 January 2018, blog.sicara.com/bokeh-dash-best-dashboard-framework-python-shiny-alternative-c5b576375f7f

[25] Bridgewater. "The All Wealth Story – How Bridgewater Associate created the all weather investment strategy, the foundation of the "risk parity" movement." https://www.bridgewater.com/resources/all-weather-story.pdf

[2] Broverman, Aaron. "Best Robo Advisors in Canada Comparison and Review." *GreedyRates*, GreedyRates, 5 September 2018, www.greedyrates.ca/blog/compare-top-robo-advisors-canada/.

[17] "Digital Wealth Management." *Nest Wealth*, www.nestwealth.com/\.

[36] Douthit, Philip S. "Marginal Share Ratios: or why correlation is as important as return." August 1999, https://static1.squarespace.com/static/53b462c1e4b0a4eb8afaff12/t/5997527acd39c33c614eebe5/1503089276267/Marginal+Sharpe.pdf

[32] Fama French "The Cross Section of Stock Returns" (1992) https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1540-6261.1992.tb04398.x#

[8] Floyd, David. "Passive Investing." *Investopedia*, Investopedia, 25 July 2018, www.investopedia.com/terms/p/passiveinvesting.asp.

[1] Investopedia. "Investing." *Investopedia*, Investopedia, 11 April 2018, www.investopedia.com/terms/i/investing.asp.

[29] Jagannathan, R., T. Ma. 2003. Risk reduction in large portfolios: Why imposing the wrong constraints helps. *J.F inance* 58(4) 1651–1684.

[44] Janetakis, Nick. "Server Side Templates vs REST API and Javascript Front-End." Nick Janetakis, 24 October 2017, nickjanetakis.com/blog/server-side-templates-vs-rest-api-and-javascript-front-end.

[20] "Managed Investing." *Questrade*, www.questrade.com/managed-investing?s_cid=PIQTQ093_cpc_google&se=google&gp=Questrade%2B%7C%2BPortfolio%2BIQ&kw=questrade%2Bportfolio%2Biq&gclid=Cj0KCQjwgOzdBRDlARIsAJ6_HNm_d72jElqqe10vUVIj7lQ-Y5B_HBkvok7TEJ05FzIz5csy9iHjOd0aAoQAEALw_wcB.

[28] Merton, R. C. 1980. On estimating the expected return on the market: An exploratory investigation. *J.F inancial Econom.* 8 323–361.

[21] “Online Wealth Management.” *WealthBar*, www.wealthbar.com/?utm_source=canfitsummit&utm_medium=Unbounce&utm_campaign=canfitsummit&utm_content=canfitsummit.

[10] Personal Capital. “Financial Software and Wealth Management | Personal Capital.” *Daily Capital*, Personal Capital Corporation, www.personalcapital.com/.

[22, 23, 27] “Personalized Portfolios.” *Nest Wealth*, www.nestwealth.com/personalized-portfolios/?open#our-investment-approach.

[31, 35, 40, 41] Portfolio optimization: A general framework for portfolio choice, Resolve Asset Management (https://investresolve.com/file/pdf/Portfolio-Optimization-Whitepaper.pdf)

[37, 38, 39] *Quantitative Approach to Tactical Asset Allocation (https://papers.ssrn.com/sol3/papers.cfm?abstract_id=962461)*

[42] Risk Parity Solution Brief, Resolve Asset Management (https://investresolve.com/risk-parity-solution-brief-2/)

[26] “Risk Parity Solution Brief.” *ReSolve Asset Management*, investresolve.com/risk-parity-solution-brief-2/.

[12] “Schwab Intelligent Portfolios®.” *The Role of Cash in Your Portfolio*, intelligent.schwab.com/.

[14] “SigFig - The Easiest Way to Manage & Improve Your Investments.” *Sigfig.com*, www.sigfig.com/site/#/home/am.

[33] Simin, Timothy, 2008. The poor predictive performance of asset pricing models. Journal of Financial and Quantitative Analysis 43, 355–380.

[7] “The Smart, Modern Way to Invest | Online Financial Advisor.” *Betterment*, www.betterment.com/.

[16] “Wealthfront.” *Own your finances, not the other way around,* https://www.wealthfront.com/.

[19, 24] “Wealthsimple.” *Investing on Autopilot*, www.wealthsimple.com/en-ca/.

[46] “System Properties Comparison Amazon DynamoDB vs. CouchDB vs. MongoDB”, https://db-engines.com/en/system/Microsoft+SQL+Server%3BMongoDB%3BMySQL%3BPostgreSQL

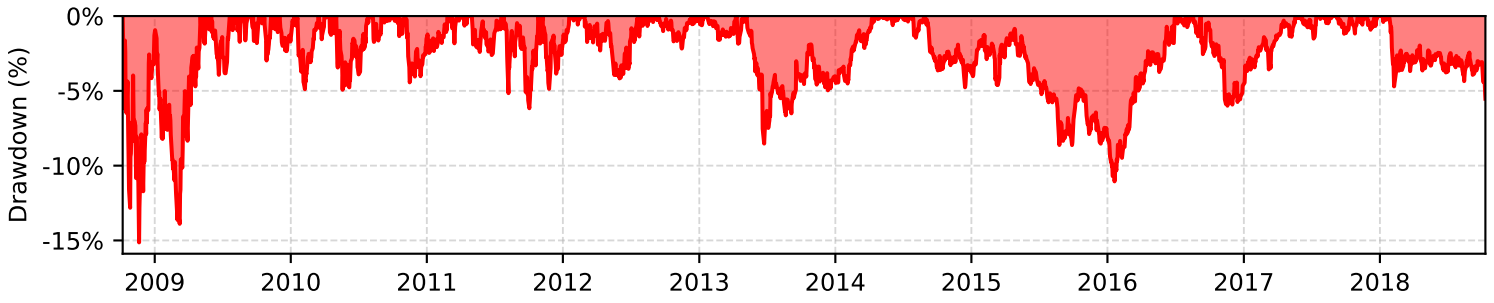
[45] “System Properties Comparison Microsoft SQL Server vs. MongoDB vs. MySQL vs. PostgreSQL”, https://db-engines.com/en/system/Microsoft+SQL+Server%3BMongoDB%3BMySQL%3BPostgreSQL

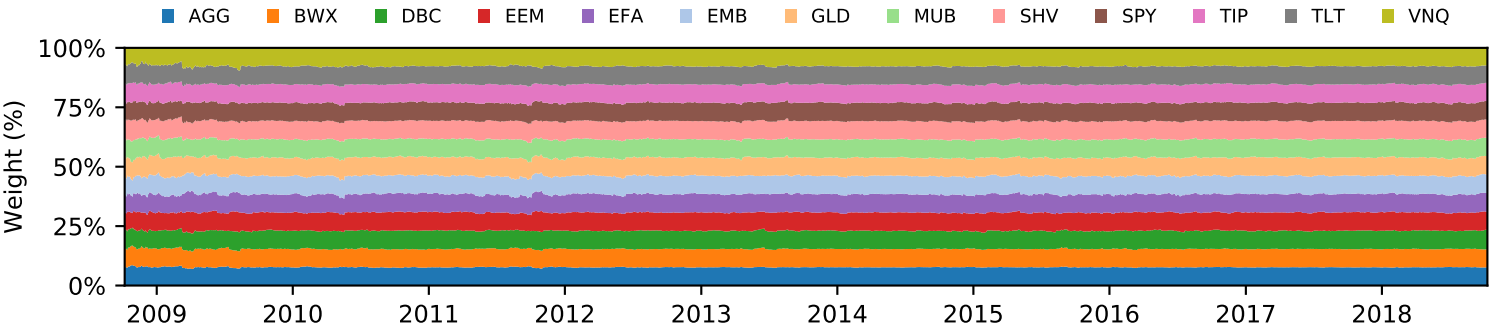
[30] https://www.aqr.com/Insights/Research/Journal-Article/Investing-With-Style

Tearsheets

Please refer to the subsequent pages for the tearsheets references throughout our report.

Portfolio Tearsheet: Equal Weight





Overview

Portfolio Code:	EW
Start Date:	2008-10-07
End Date:	2018-10-11
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	EW

Statistics

Total Return:	60.65%
CAGR:	5.26%
Annual Volatility:	8.87%
Sharpe:	0.59
Max Drawdown:	-15.13%
Sortino:	0.86

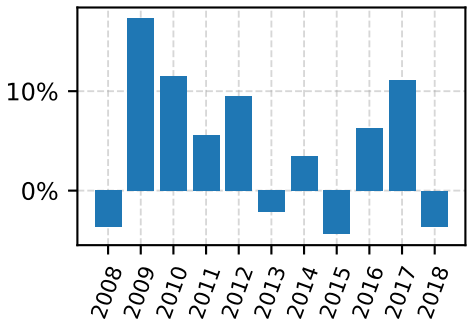
Statistics #2

$VaR_{99\%}$:	-1.63%
$CVaR_{99\%}$:	-2.32%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

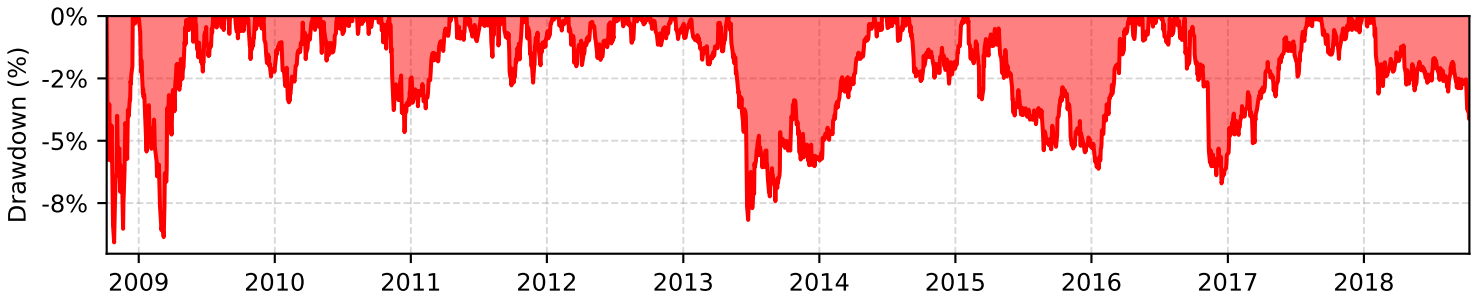
	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.5%	0.2%	0.5%	0.6%	0.3%	-0.2%	0.4%	0.0%	0.2%	0.3%	-0.2%
BWX	0.4%	0.4%	0.3%	0.3%	0.4%	-0.3%	-0.1%	-0.5%	0.1%	0.7%	-0.3%
DBC	-2.2%	1.4%	1.0%	-0.1%	0.3%	-0.6%	-2.4%	-2.3%	1.4%	0.4%	0.5%
EEM	-0.5%	4.8%	1.4%	-1.2%	1.5%	-0.1%	-0.2%	-1.2%	0.9%	2.5%	-1.3%
EFA	-0.5%	2.4%	0.8%	-0.8%	1.4%	1.6%	-0.5%	0.0%	0.1%	1.8%	-0.6%
EMB	0.5%	1.1%	0.8%	0.6%	1.2%	-0.6%	0.5%	0.1%	0.7%	0.8%	-0.5%
GLD	0.3%	1.8%	2.0%	0.9%	0.5%	-2.4%	-0.1%	-0.8%	0.7%	1.0%	-0.5%
MUB	0.3%	0.5%	0.0%	1.0%	0.4%	-0.3%	0.7%	0.2%	-0.0%	0.3%	-0.1%
SHV	0.0%	0.0%	0.0%	0.0%	0.0%	-0.0%	0.0%	0.0%	0.0%	0.1%	0.1%
SPY	-0.9%	2.2%	1.2%	0.3%	1.2%	2.2%	1.0%	0.1%	0.9%	1.5%	0.3%
TIP	-0.2%	0.7%	0.4%	1.0%	0.5%	-0.7%	0.3%	-0.1%	0.4%	0.2%	-0.1%
TLT	1.6%	-1.8%	0.7%	2.4%	0.2%	-1.1%	1.9%	-0.1%	0.1%	0.7%	-0.6%
VNQ	-2.0%	3.2%	2.1%	0.9%	1.3%	0.4%	2.1%	0.3%	0.7%	0.4%	-0.3%

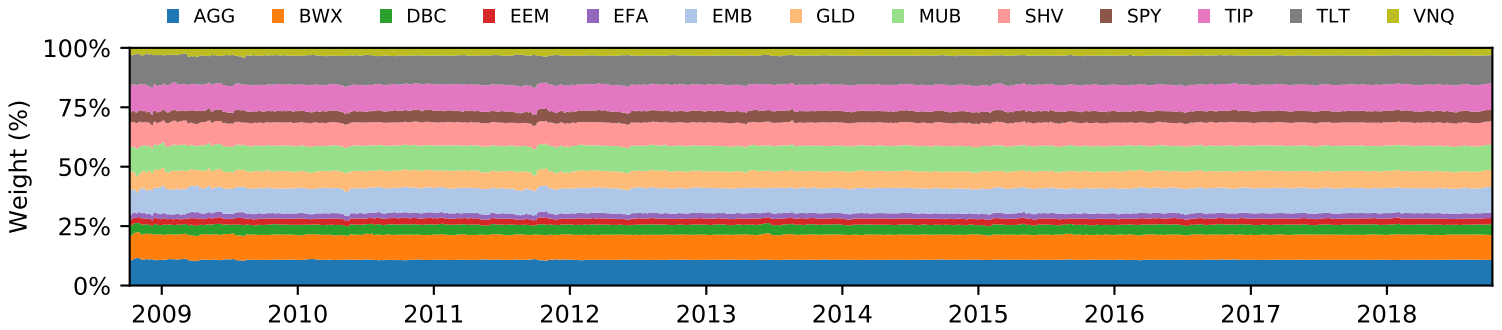
Yearly Returns (%)



Tearsheet #2

Portfolio Tearsheet: Risk Parity





Overview

Portfolio Code:	RP
Start Date:	2008-10-07
End Date:	2018-10-11
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	RP Static

Statistics

Total Return:	51.71%
CAGR:	4.41%
Annual Volatility:	5.42%
Sharpe:	0.81
Max Drawdown:	-9.08%
Sortino:	1.17

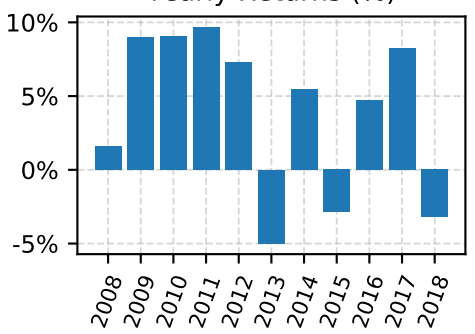
Statistics #2

$VaR_{99\%}$:	-0.92%
$CVaR_{99\%}$:	-1.31%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

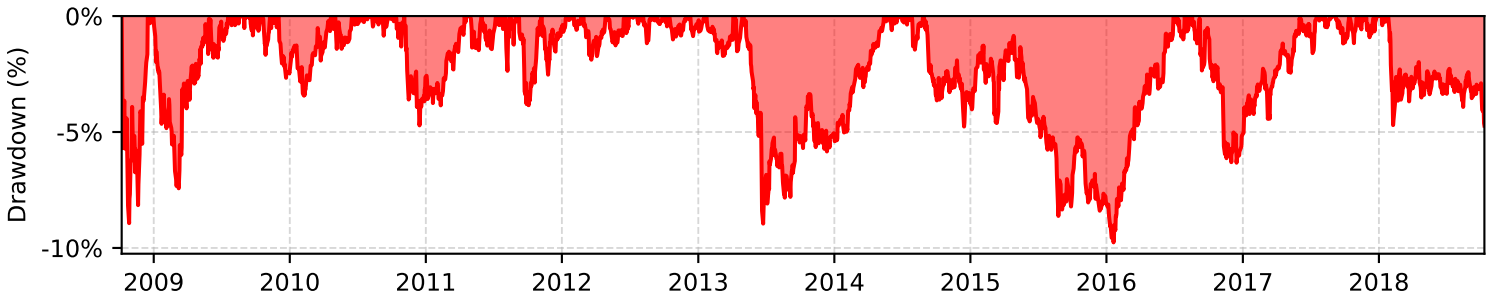
	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.7%	0.3%	0.7%	0.8%	0.4%	-0.2%	0.6%	0.0%	0.3%	0.4%	-0.2%
BWX	0.5%	0.6%	0.5%	0.4%	0.6%	-0.3%	-0.1%	-0.7%	0.1%	1.0%	-0.4%
DBC	-1.2%	0.8%	0.5%	-0.0%	0.2%	-0.3%	-1.3%	-1.3%	0.8%	0.2%	0.3%
EEM	-0.2%	1.6%	0.5%	-0.4%	0.5%	-0.0%	-0.1%	-0.4%	0.3%	0.8%	-0.4%
EFA	-0.2%	0.7%	0.2%	-0.2%	0.4%	0.5%	-0.1%	0.0%	0.0%	0.5%	-0.2%
EMB	0.7%	1.6%	1.1%	0.8%	1.6%	-0.8%	0.6%	0.1%	1.0%	1.0%	-0.6%
GLD	0.3%	1.6%	1.9%	0.8%	0.5%	-2.2%	-0.1%	-0.8%	0.6%	0.9%	-0.5%
MUB	0.3%	0.7%	0.0%	1.3%	0.5%	-0.4%	1.0%	0.3%	-0.0%	0.5%	-0.2%
SHV	0.1%	0.0%	0.0%	0.0%	0.0%	-0.0%	0.0%	0.0%	0.0%	0.1%	0.1%
SPY	-0.6%	1.4%	0.7%	0.2%	0.7%	1.4%	0.6%	0.1%	0.6%	1.0%	0.2%
TIP	-0.3%	1.0%	0.7%	1.4%	0.7%	-1.0%	0.4%	-0.2%	0.5%	0.3%	-0.2%
TLT	2.6%	-2.8%	1.1%	3.9%	0.4%	-1.7%	3.0%	-0.1%	0.2%	1.1%	-0.9%
VNQ	-0.9%	1.4%	0.9%	0.4%	0.6%	0.2%	0.9%	0.1%	0.3%	0.2%	-0.2%

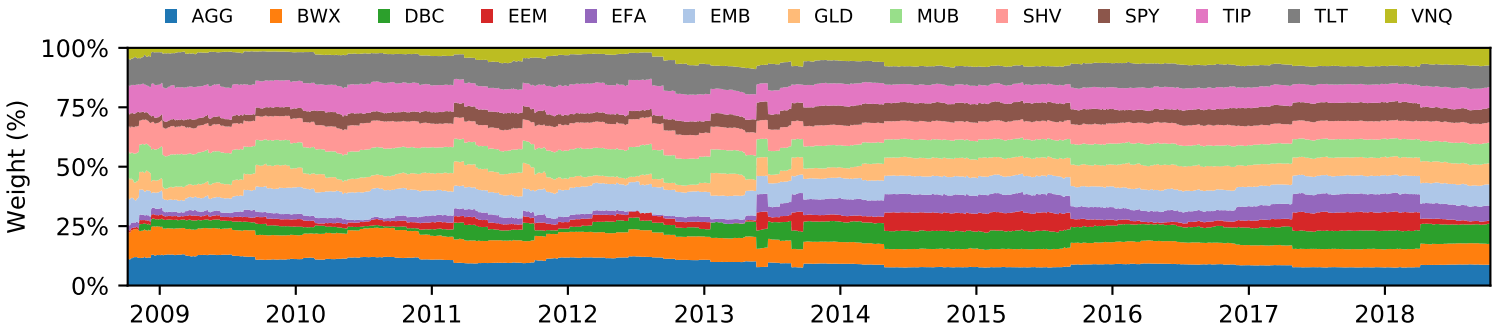
Yearly Returns (%)



Tearsheet #3

Portfolio Tearsheet: Dynamic Risk Parity





Overview

Portfolio Code:	RP
Start Date:	2008-10-07
End Date:	2018-10-11
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	RP 200

Statistics

Total Return:	44.46%
CAGR:	3.91%
Annual Volatility:	5.59%
Sharpe:	0.70
Max Drawdown:	-9.76%
Sortino:	1.00

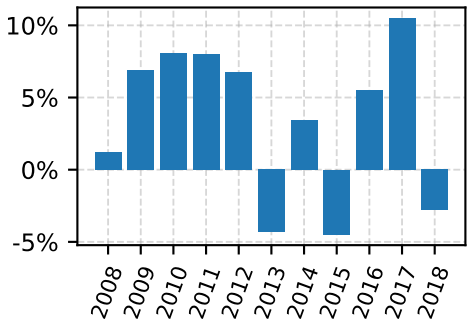
Statistics #2

$VaR_{99\%}$:	-0.99%
$CVaR_{99\%}$:	-1.36%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

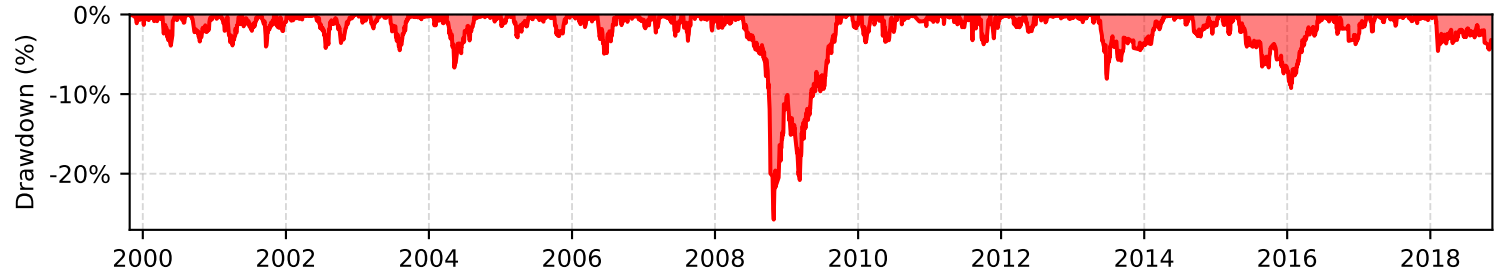
Performance Attribution

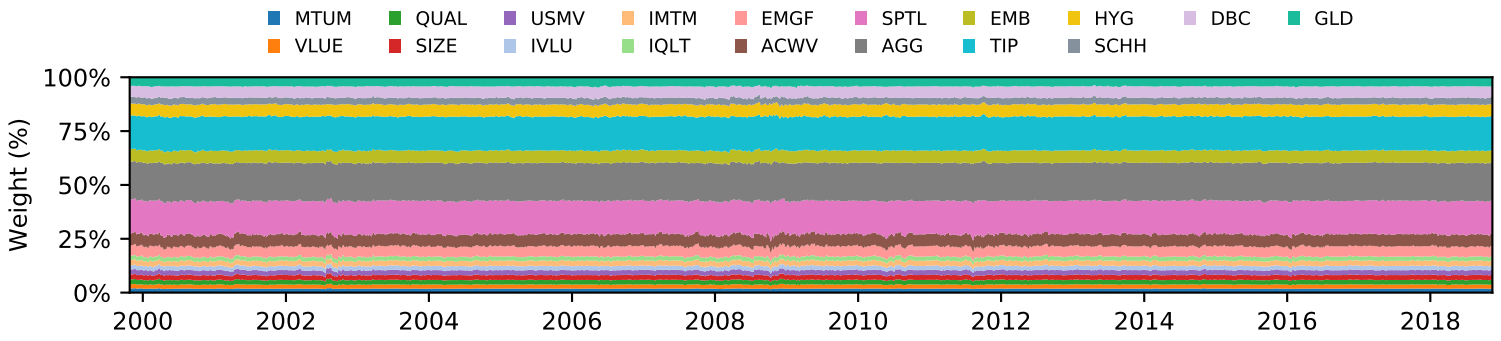
	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.9%	0.4%	0.7%	0.7%	0.4%	-0.2%	0.5%	0.0%	0.2%	0.3%	-0.2%
BWX	0.6%	0.6%	0.6%	0.4%	0.6%	-0.4%	-0.0%	-0.5%	0.1%	0.8%	-0.4%
DBC	-0.3%	0.7%	0.1%	-0.7%	-0.1%	-0.5%	-2.4%	-2.3%	1.3%	0.4%	0.6%
EEM	0.0%	1.2%	0.3%	-0.4%	0.5%	-0.3%	-0.2%	-1.1%	0.1%	1.9%	-0.2%
EFA	-0.2%	0.7%	0.1%	-0.2%	0.2%	0.6%	-0.5%	0.0%	0.1%	1.6%	-0.5%
EMB	-0.0%	1.1%	1.2%	0.8%	1.8%	-0.6%	0.5%	0.1%	0.8%	0.8%	-0.5%
GLD	-0.1%	1.8%	1.6%	1.2%	0.4%	-2.7%	-0.4%	-0.9%	0.8%	1.1%	-0.6%
MUB	0.5%	0.9%	0.0%	1.3%	0.6%	-0.2%	0.8%	0.3%	-0.0%	0.4%	-0.1%
SHV	0.1%	0.0%	0.0%	0.0%	0.0%	-0.0%	0.0%	0.0%	0.0%	0.1%	0.1%
SPY	-0.8%	1.0%	0.6%	-0.0%	0.5%	1.8%	1.0%	0.0%	0.7%	1.5%	0.2%
TIP	-0.3%	1.1%	0.7%	1.3%	0.8%	-0.8%	0.3%	-0.1%	0.4%	0.2%	-0.1%
TLT	2.5%	-3.1%	1.2%	3.3%	0.4%	-1.4%	2.1%	-0.1%	0.2%	0.7%	-0.6%
VNQ	-1.5%	0.6%	0.7%	0.2%	0.5%	0.5%	1.9%	0.1%	0.6%	0.4%	-0.4%

Yearly Returns (%)



Portfolio Tearsheet: Static Risk Parity





Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP

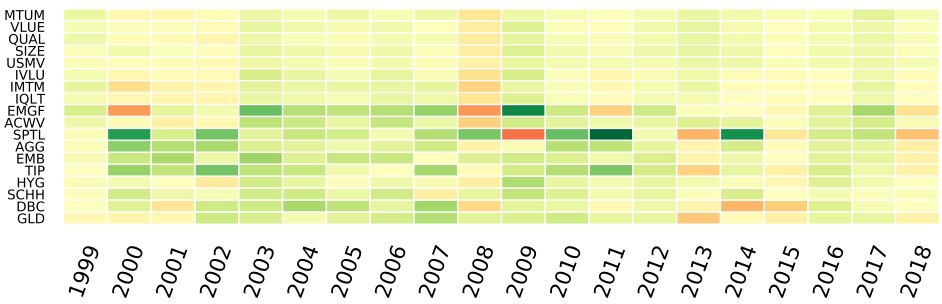
Statistics

Total Return:	270.57%
CAGR:	7.75%
Annual Volatility:	5.72%
Sharpe:	1.13
Max Drawdown:	-25.76%
Sortino:	1.64

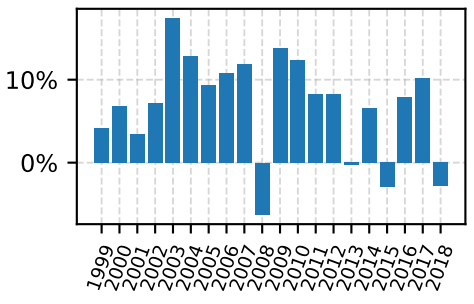
Statistics #2

$VaR_{99\%}$:	-0.92%
$CVaR_{99\%}$:	-1.44%
Beta:	0.18
Alpha:	6.08%
R-Squared:	35.62%
Treynor:	0.35

Performance Attribution

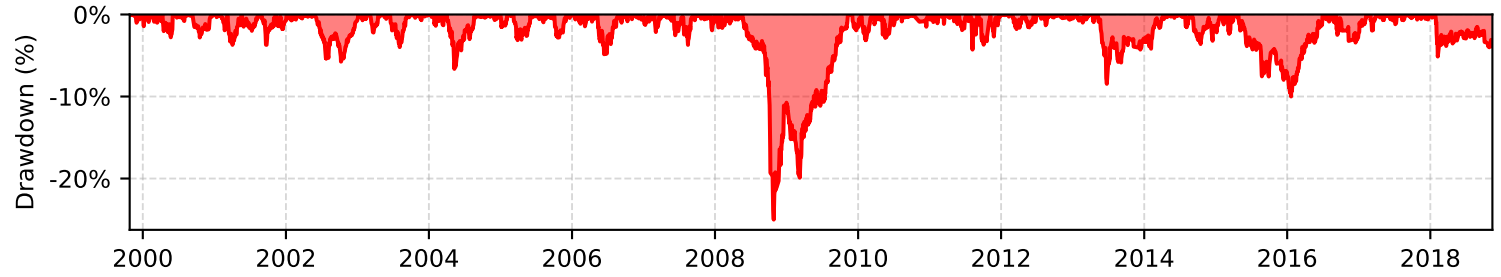


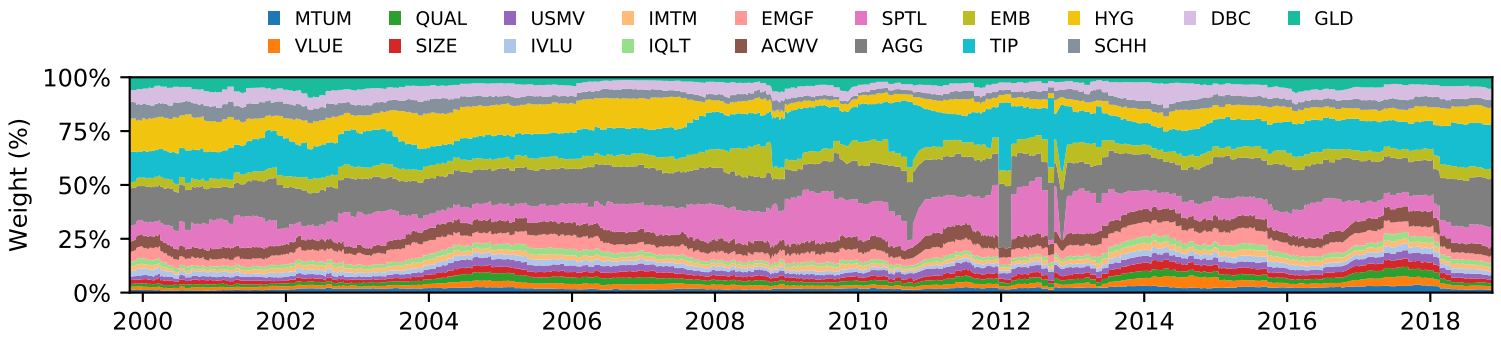
Yearly Returns (%)



Tearsheet #5

Portfolio Tearsheet: Dynamic Risk Parity





Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP 200

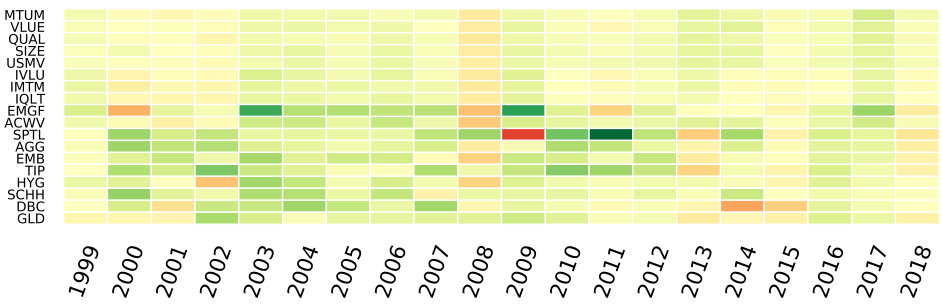
Statistics

Total Return:	264.48%
CAGR:	7.64%
Annual Volatility:	5.51%
Sharpe:	1.16
Max Drawdown:	-25.01%
Sortino:	1.66

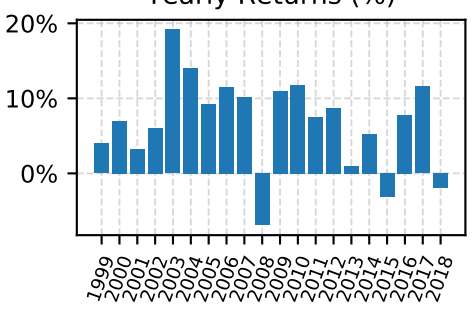
Statistics #2

$VaR_{99\%}$:	-0.88%
$CVaR_{99\%}$:	-1.43%
Beta:	0.18
Alpha:	6.03%
R-Squared:	35.36%
Treynor:	0.36

Performance Attribution

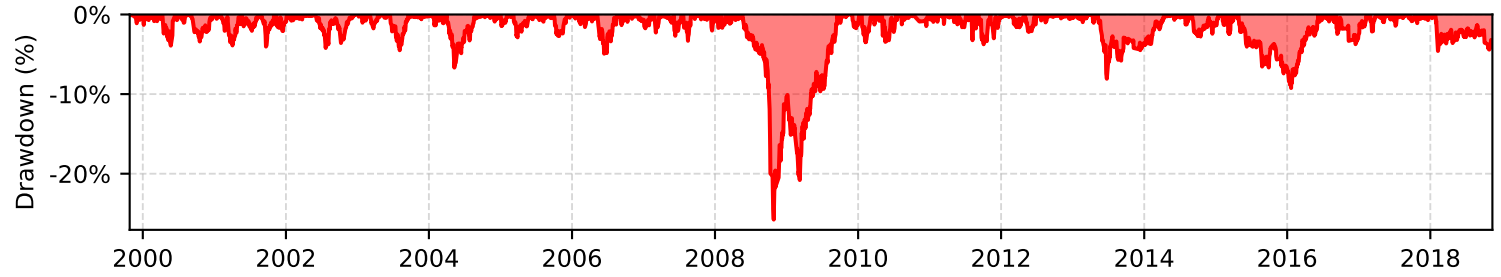


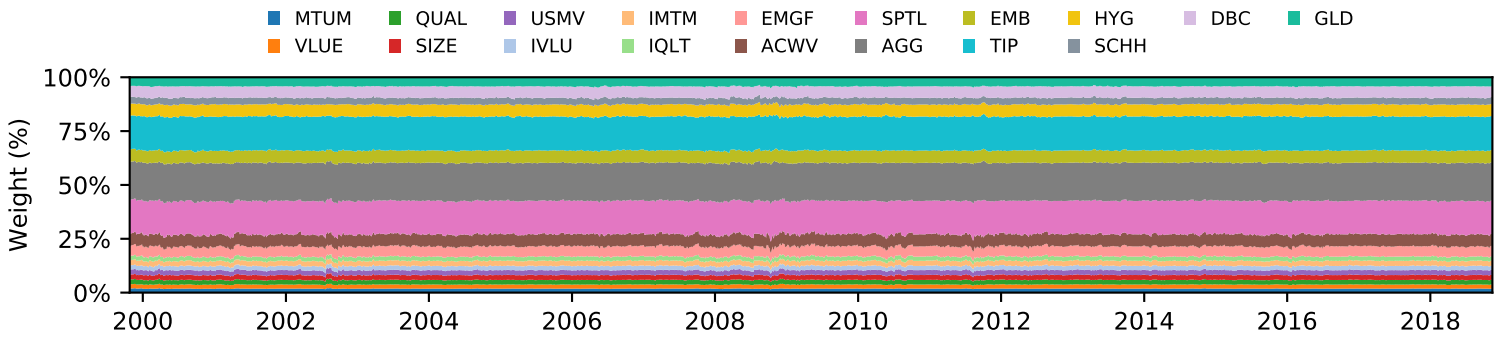
Yearly Returns (%)



Tearsheet #6

Portfolio Tearsheet: No Leverage





Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP

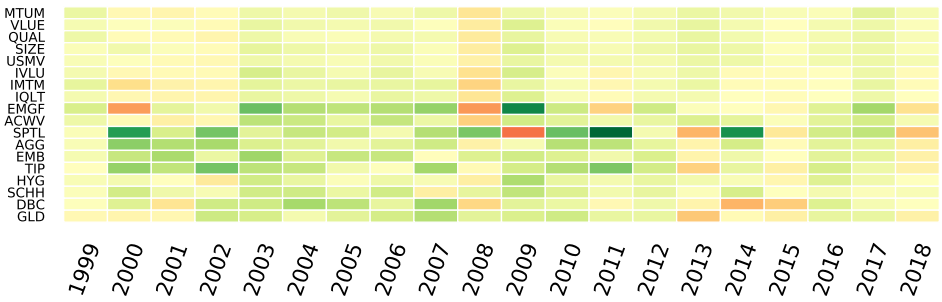
Statistics

Total Return:	270.57%
CAGR:	7.75%
Annual Volatility:	5.72%
Sharpe:	1.13
Max Drawdown:	-25.76%
Sortino:	1.64

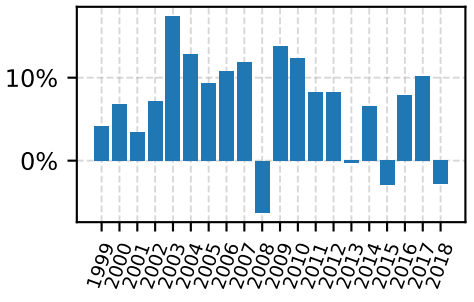
Statistics #2

$VaR_{99\%}$:	-0.92%
$CVaR_{99\%}$:	-1.44%
Beta:	0.18
Alpha:	6.08%
R-Squared:	35.62%
Treynor:	0.35

Performance Attribution

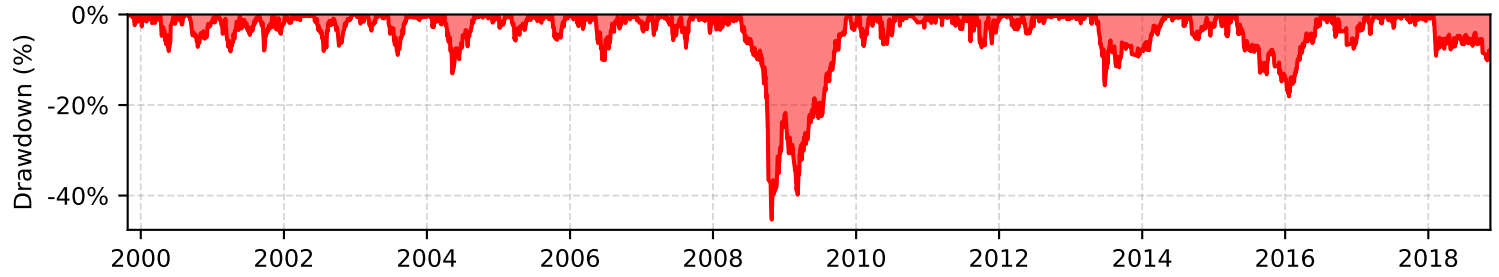


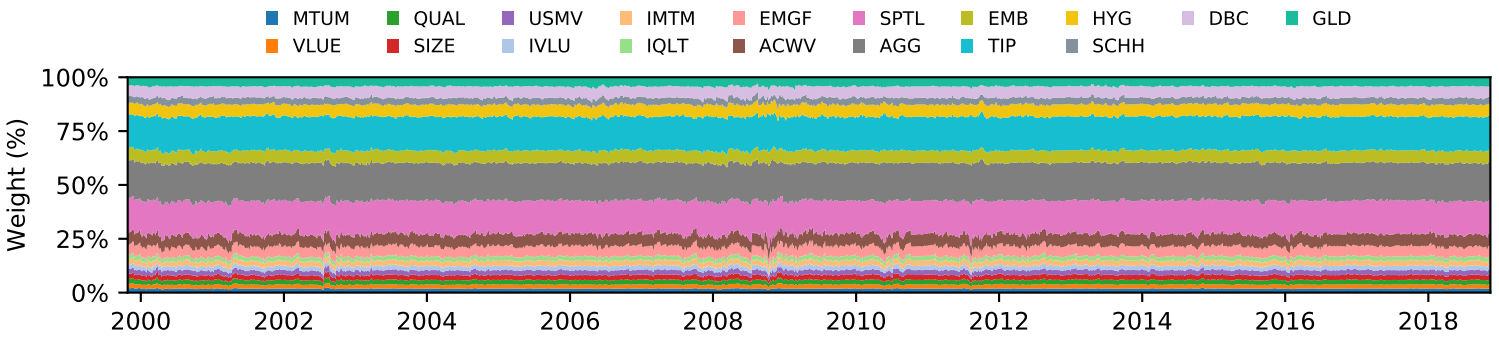
Yearly Returns (%)



Tearsheet #7

Portfolio Tearsheet: 2x Leverage





Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP

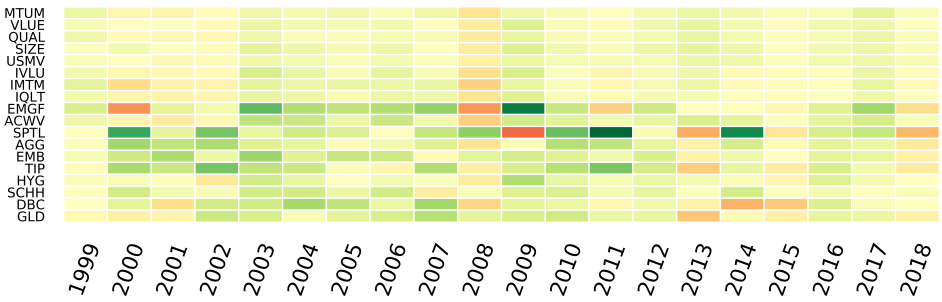
Statistics

Total Return:	817.76%
CAGR:	13.89%
Annual Volatility:	11.40%
Sharpe:	1.10
Max Drawdown:	-45.30%
Sortino:	1.58

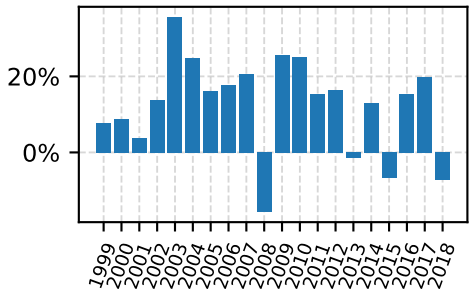
Statistics #2

$VaR_{99\%}$:	-1.83%
$CVaR_{99\%}$:	-2.88%
Beta:	0.36
Alpha:	10.44%
R-Squared:	34.83%
Treynor:	0.35

Performance Attribution

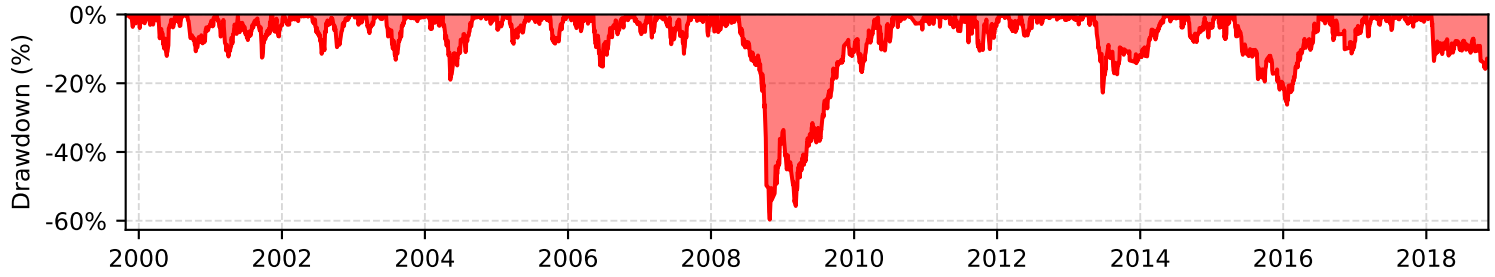


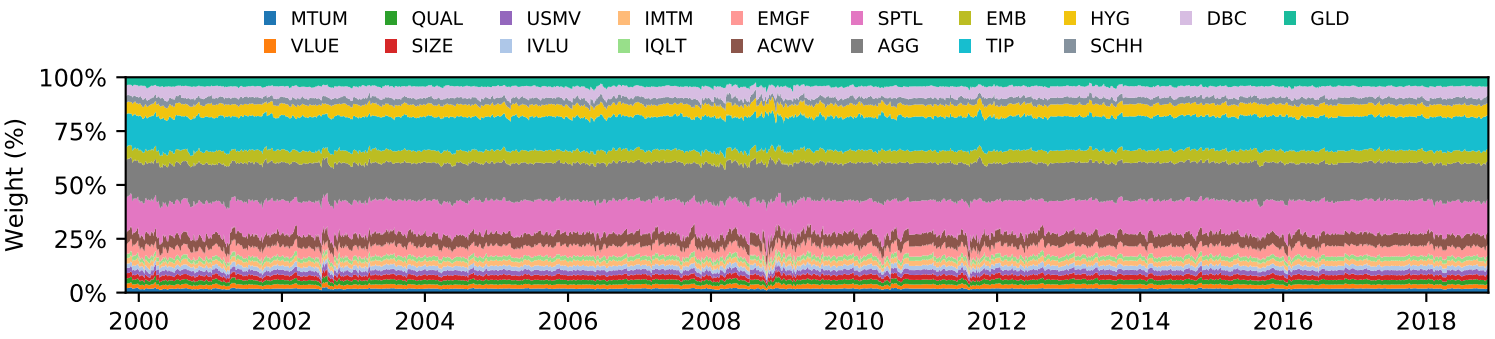
Yearly Returns (%)



Tearsheet #8

Portfolio Tearsheet: 3x Leverage





Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP

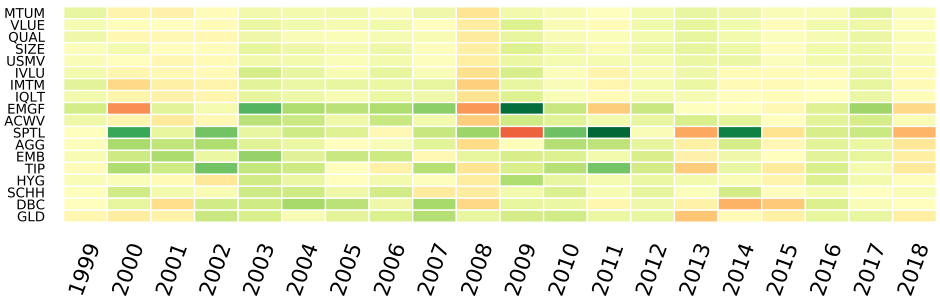
Statistics

Total Return:	1950.95%
CAGR:	20.07%
Annual Volatility:	17.03%
Sharpe:	1.10
Max Drawdown:	-59.67%
Sortino:	1.56

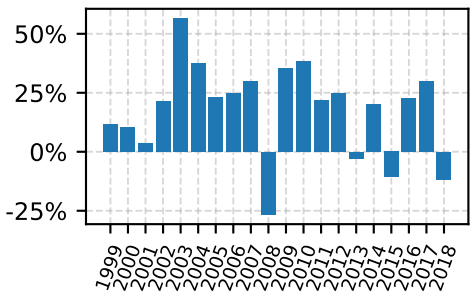
Statistics #2

$VaR_{99\%}$:	-2.71%
$CVaR_{99\%}$:	-4.29%
Beta:	0.53
Alpha:	14.75%
R-Squared:	33.84%
Treynor:	0.35

Performance Attribution

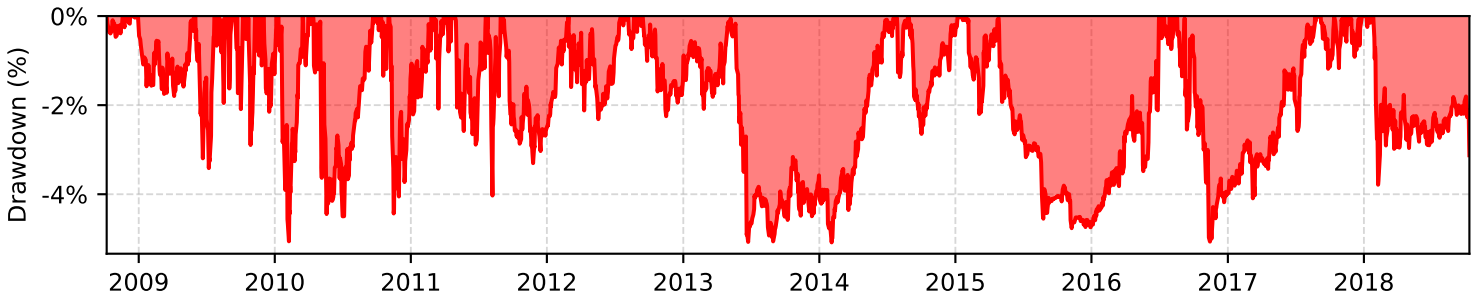


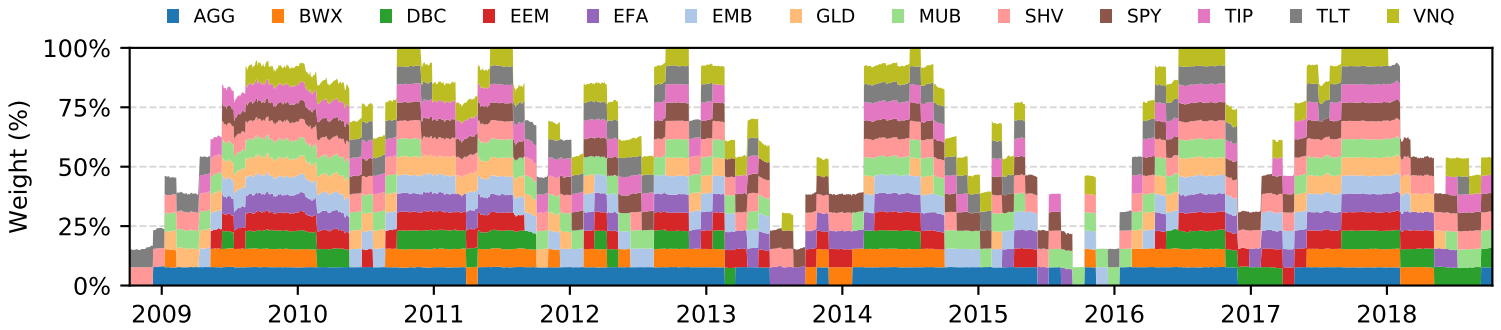
Yearly Returns (%)



Tearsheet #9

Portfolio Tearsheet: Trend Following Equal Weight





Overview

Portfolio Code:	EW_TF
Start Date:	2008-10-07
End Date:	2018-10-11
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	EW

Statistics

Total Return:	39.31%
CAGR:	3.49%
Annual Volatility:	4.80%
Sharpe:	0.73
Max Drawdown:	-5.08%
Sortino:	1.02

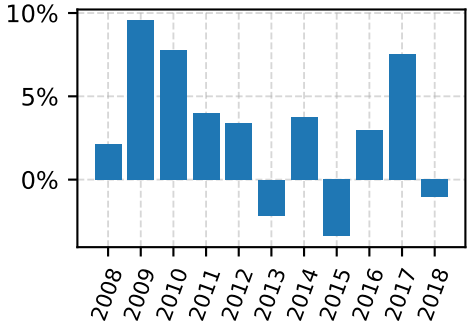
Statistics #2

$VaR_{99\%}$:	-0.98%
$CVaR_{99\%}$:	-1.21%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

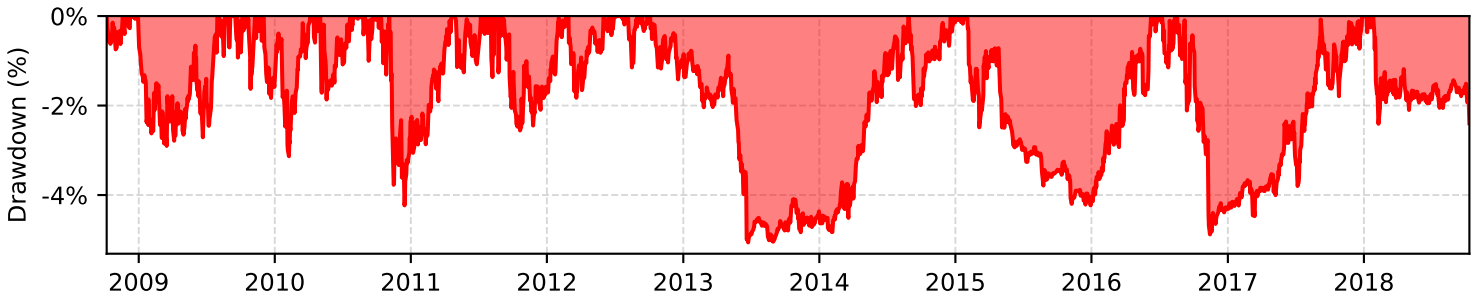
	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.4%	0.2%	0.5%	0.5%	0.3%	-0.3%	0.4%	-0.1%	0.1%	0.1%	-0.2%
BWX	0.0%	0.2%	0.1%	0.1%	-0.0%	-0.2%	0.1%	-0.3%	0.3%	0.3%	0.0%
DBC	0.0%	-0.6%	0.3%	-0.4%	-0.6%	-0.2%	-0.2%	0.0%	0.7%	0.3%	0.5%
EEM	0.0%	2.6%	0.6%	-0.7%	-0.4%	-1.6%	-0.5%	-0.3%	-0.2%	2.0%	-0.1%
EFA	0.0%	1.3%	-0.1%	0.2%	0.5%	1.6%	-0.3%	-1.0%	-0.2%	1.8%	-0.1%
EMB	0.0%	1.5%	0.8%	-0.0%	1.2%	-1.0%	0.3%	-0.2%	0.3%	0.6%	-0.2%
GLD	0.0%	1.9%	2.0%	0.9%	-0.4%	-0.3%	-0.6%	-0.4%	0.4%	0.2%	-0.0%
MUB	0.0%	0.4%	0.2%	0.7%	0.4%	-0.5%	0.5%	0.2%	0.2%	0.2%	-0.2%
SHV	0.0%	0.0%	0.0%	0.0%	0.0%	-0.0%	0.0%	-0.0%	0.0%	0.1%	0.1%
SPY	0.0%	1.4%	0.7%	-0.2%	0.7%	2.2%	1.0%	-0.3%	0.9%	1.5%	0.3%
TIP	0.0%	0.5%	0.5%	1.0%	0.5%	-0.2%	0.1%	-0.3%	0.2%	0.1%	-0.2%
TLT	1.6%	-1.5%	0.0%	2.1%	0.2%	-0.5%	1.4%	-0.3%	0.2%	0.3%	-0.5%
VNQ	0.0%	1.3%	2.0%	0.1%	1.0%	-1.1%	1.5%	-0.4%	0.1%	-0.2%	-0.4%

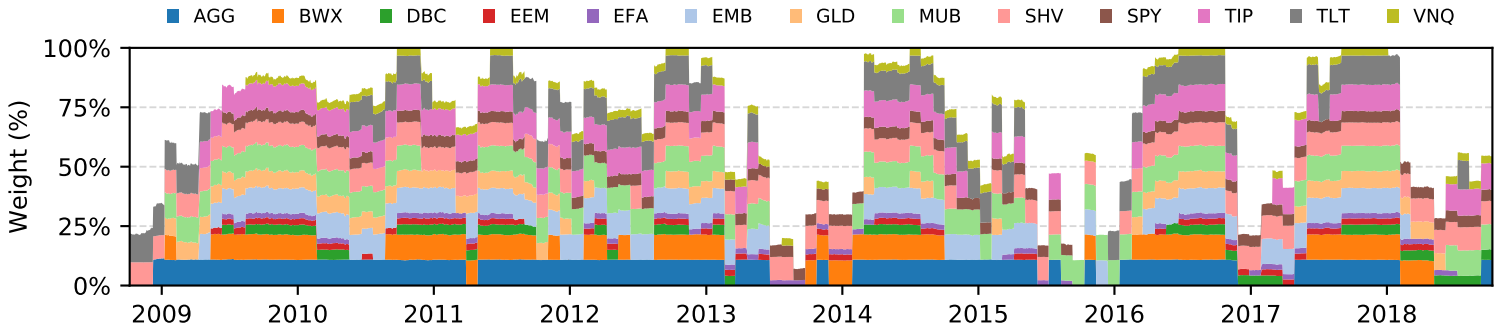
Yearly Returns (%)



Tearsheet #10

Portfolio Tearsheet: Risk Parity Trend Following





Overview

Portfolio Code:	RP
Start Date:	2008-10-07
End Date:	2018-10-11
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP Static

Statistics

Total Return:	35.56%
CAGR:	3.15%
Annual Volatility:	3.42%
Sharpe:	0.92
Max Drawdown:	-5.06%
Sortino:	1.32

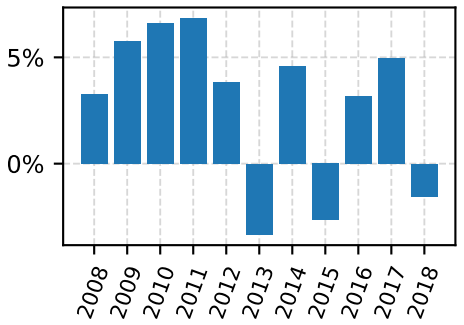
Statistics #2

$VaR_{99\%}$:	-0.64%
$CVaR_{99\%}$:	-0.82%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

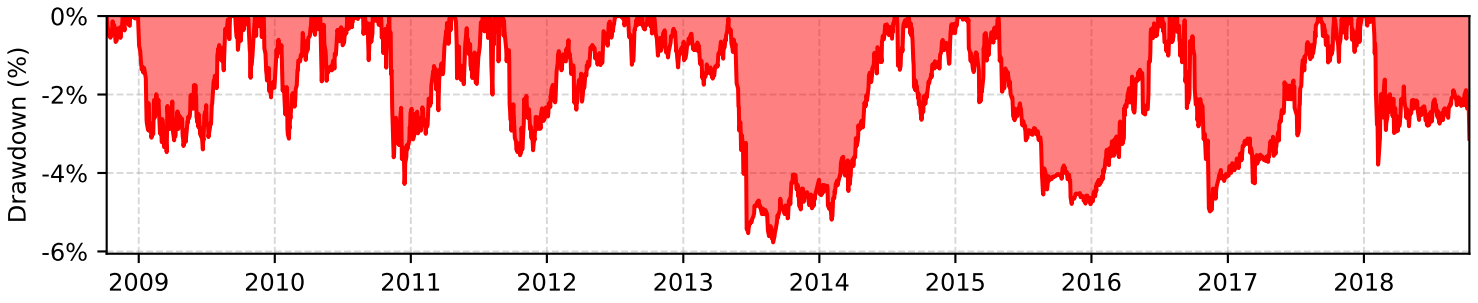
	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.6%	0.3%	0.7%	0.7%	0.4%	-0.4%	0.5%	-0.1%	0.2%	0.2%	-0.3%
BWX	0.0%	0.3%	0.2%	0.1%	-0.0%	-0.2%	0.2%	-0.4%	0.4%	0.4%	0.0%
DBC	0.0%	-0.3%	0.2%	-0.2%	-0.3%	-0.1%	-0.1%	0.0%	0.4%	0.2%	0.3%
EEM	0.0%	0.9%	0.2%	-0.2%	-0.1%	-0.6%	-0.2%	-0.1%	-0.1%	0.7%	-0.0%
EFA	0.0%	0.4%	-0.0%	0.0%	0.2%	0.5%	-0.1%	-0.3%	-0.0%	0.5%	-0.0%
EMB	0.0%	2.1%	1.1%	-0.1%	1.7%	-1.4%	0.4%	-0.3%	0.4%	0.9%	-0.3%
GLD	0.0%	1.8%	1.9%	0.9%	-0.3%	-0.3%	-0.6%	-0.4%	0.4%	0.2%	-0.0%
MUB	0.0%	0.6%	0.3%	1.0%	0.5%	-0.7%	0.7%	0.3%	0.2%	0.3%	-0.3%
SHV	0.1%	0.0%	0.0%	0.0%	0.0%	-0.0%	0.0%	-0.0%	0.0%	0.1%	0.1%
SPY	0.0%	0.9%	0.5%	-0.1%	0.5%	1.4%	0.6%	-0.2%	0.6%	1.0%	0.2%
TIP	0.0%	0.7%	0.7%	1.4%	0.7%	-0.3%	0.2%	-0.5%	0.3%	0.1%	-0.3%
TLT	2.6%	-2.4%	0.0%	3.3%	0.3%	-0.8%	2.2%	-0.5%	0.4%	0.5%	-0.8%
VNQ	0.0%	0.6%	0.9%	0.0%	0.4%	-0.5%	0.6%	-0.2%	0.0%	-0.1%	-0.2%

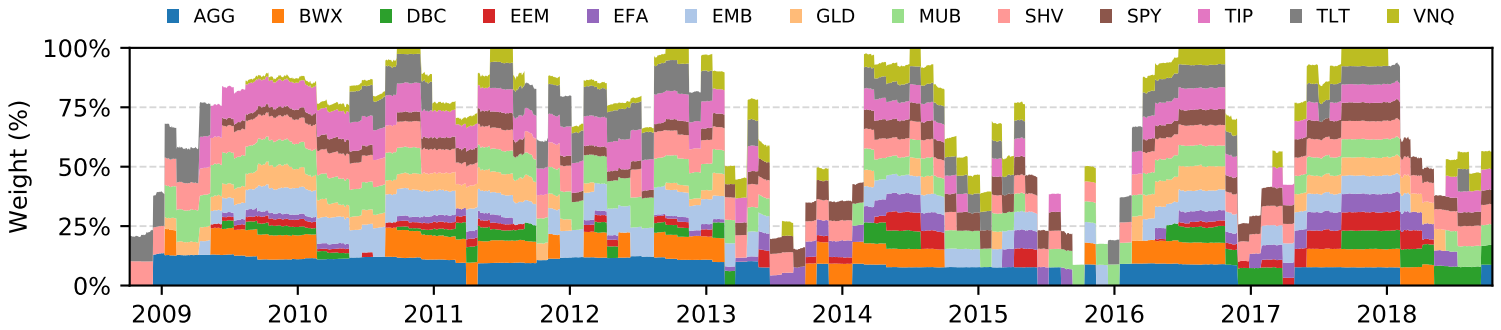
Yearly Returns (%)



Tearsheet #11

Portfolio Tearsheet: Dynamic Risk Parity Trend Following





Overview

Portfolio Code:	RP_TF
Start Date:	2008-10-07
End Date:	2018-10-11
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP 200

Statistics

Total Return:	33.91%
CAGR:	3.03%
Annual Volatility:	3.71%
Sharpe:	0.82
Max Drawdown:	-5.77%
Sortino:	1.15

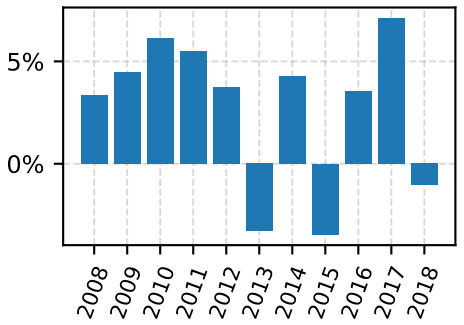
Statistics #2

$VaR_{99\%}$:	-0.71%
$CVaR_{99\%}$:	-0.90%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

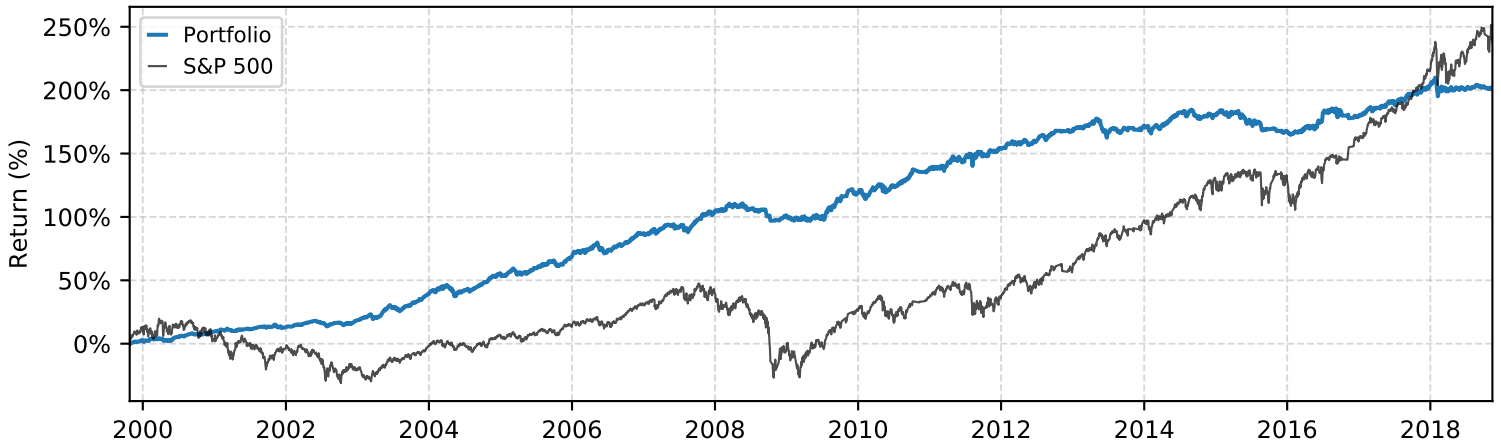
Performance Attribution

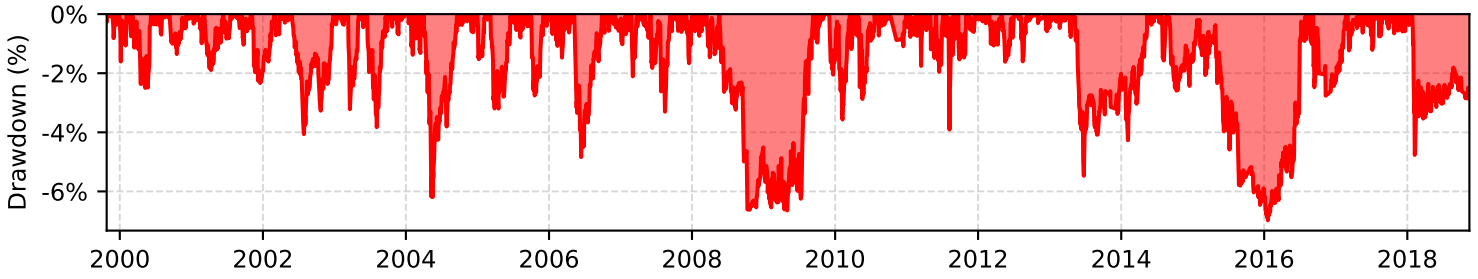
	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.7%	0.4%	0.7%	0.6%	0.4%	-0.3%	0.4%	-0.1%	0.1%	0.1%	-0.2%
BWX	0.0%	0.4%	0.2%	0.1%	-0.0%	-0.2%	0.2%	-0.3%	0.4%	0.3%	-0.0%
DBC	0.0%	-0.2%	-0.0%	-0.8%	-0.4%	-0.2%	-0.2%	0.0%	0.6%	0.3%	0.6%
EEM	0.0%	0.7%	0.1%	-0.2%	-0.1%	-1.2%	-0.1%	-0.3%	-0.0%	1.7%	0.1%
EFA	0.0%	0.4%	-0.0%	0.0%	0.1%	0.6%	-0.3%	-1.0%	-0.1%	1.6%	-0.1%
EMB	0.0%	1.4%	1.2%	-0.1%	1.8%	-1.1%	0.3%	-0.3%	0.4%	0.7%	-0.2%
GLD	0.0%	1.9%	1.6%	1.2%	-0.1%	-0.4%	-0.6%	-0.4%	0.5%	0.2%	-0.1%
MUB	0.0%	0.7%	0.3%	0.9%	0.6%	-0.5%	0.6%	0.2%	0.2%	0.2%	-0.2%
SHV	0.1%	0.0%	0.0%	0.0%	0.0%	-0.0%	-0.0%	-0.0%	0.0%	0.1%	0.1%
SPY	0.0%	0.6%	0.4%	-0.3%	0.2%	1.8%	1.0%	-0.3%	0.8%	1.5%	0.2%
TIP	0.0%	0.7%	0.7%	1.3%	0.8%	-0.3%	0.1%	-0.3%	0.3%	0.1%	-0.2%
TLT	2.6%	-2.8%	0.0%	2.9%	0.3%	-0.7%	1.4%	-0.3%	0.3%	0.3%	-0.5%
VNQ	0.0%	0.3%	0.7%	-0.1%	0.1%	-0.8%	1.4%	-0.3%	-0.0%	-0.1%	-0.4%

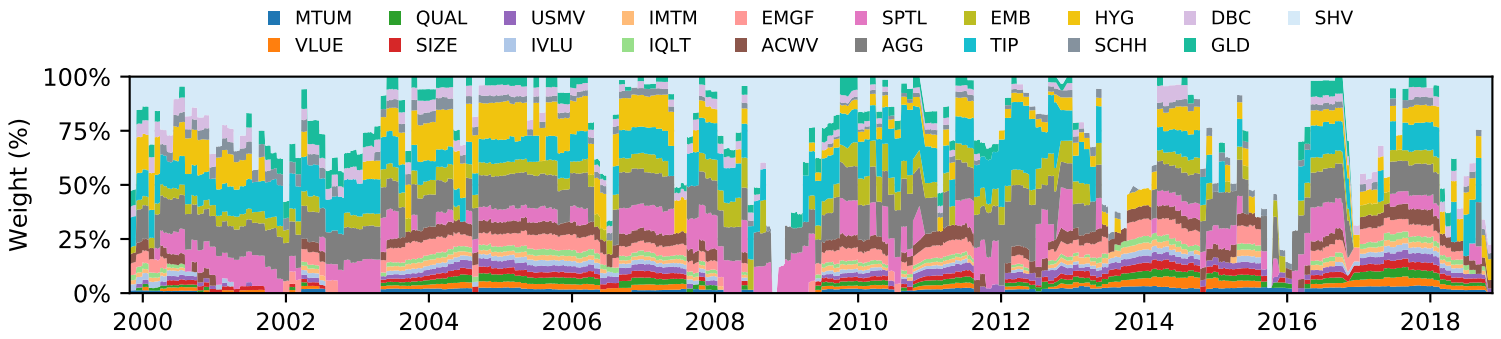
Yearly Returns (%)



Portfolio Tearsheet: Transaction Costs Trend-Following







Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP 200

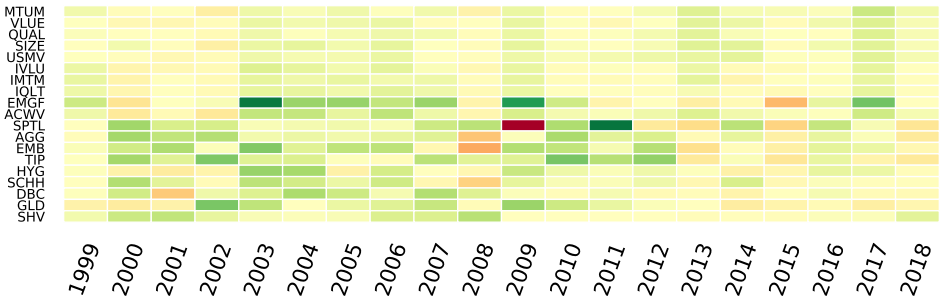
Statistics

Total Return:	201.03%
CAGR:	6.42%
Annual Volatility:	3.98%
Sharpe:	1.30
Max Drawdown:	-6.98%
Sortino:	1.88

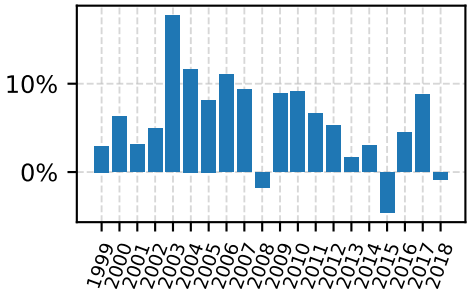
Statistics #2

$VaR_{99\%}$:	-0.68%
$CVaR_{99\%}$:	-0.91%
Beta:	0.06
Alpha:	5.87%
R-Squared:	7.95%
Treynor:	0.86

Performance Attribution

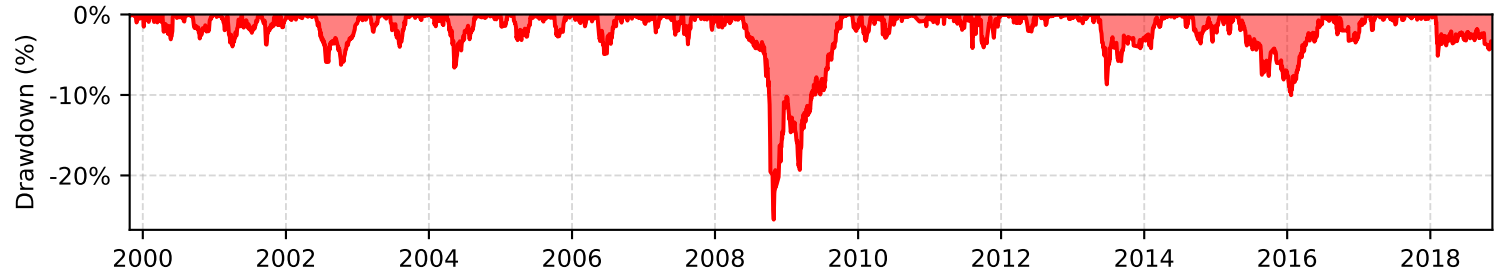


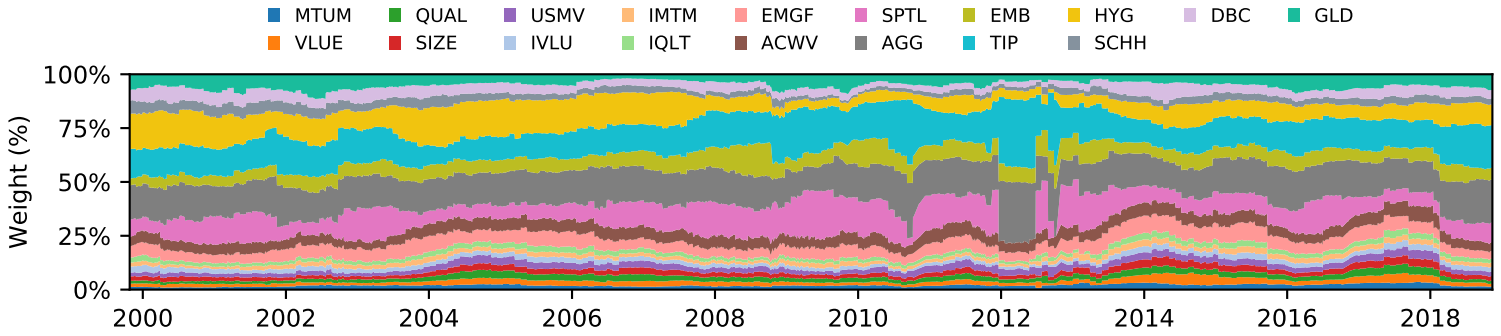
Yearly Returns (%)



Tearsheet #13

Portfolio Tearsheet: Transaction Costs No-Trend





Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	RP 200

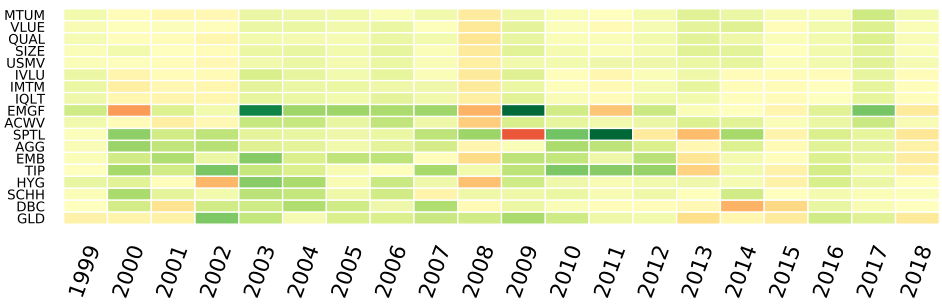
Statistics

Total Return:	277.78%
CAGR:	7.86%
Annual Volatility:	5.55%
Sharpe:	1.19
Max Drawdown:	-25.49%
Sortino:	1.71

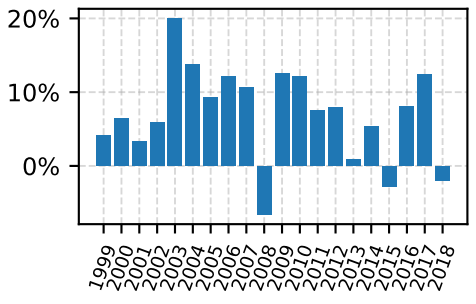
Statistics #2

$VaR_{99\%}$:	-0.90%
$CVaR_{99\%}$:	-1.44%
Beta:	0.18
Alpha:	6.24%
R-Squared:	35.43%
Treynor:	0.37

Performance Attribution

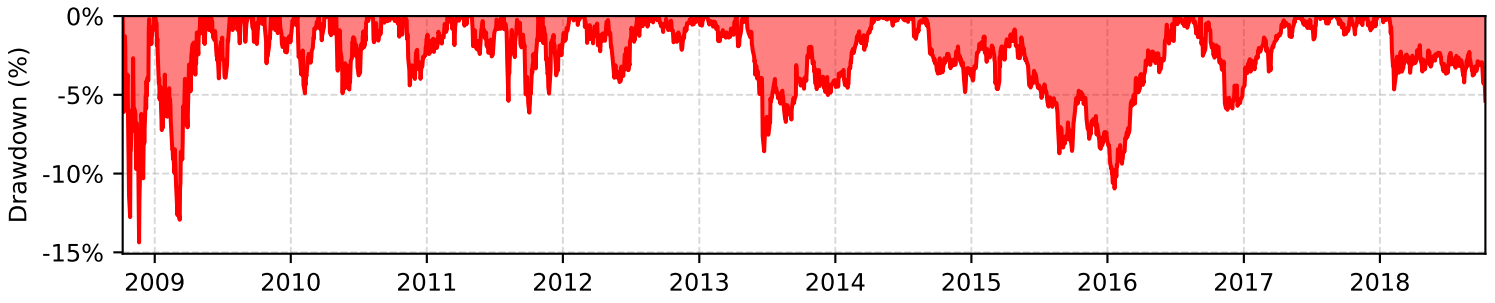


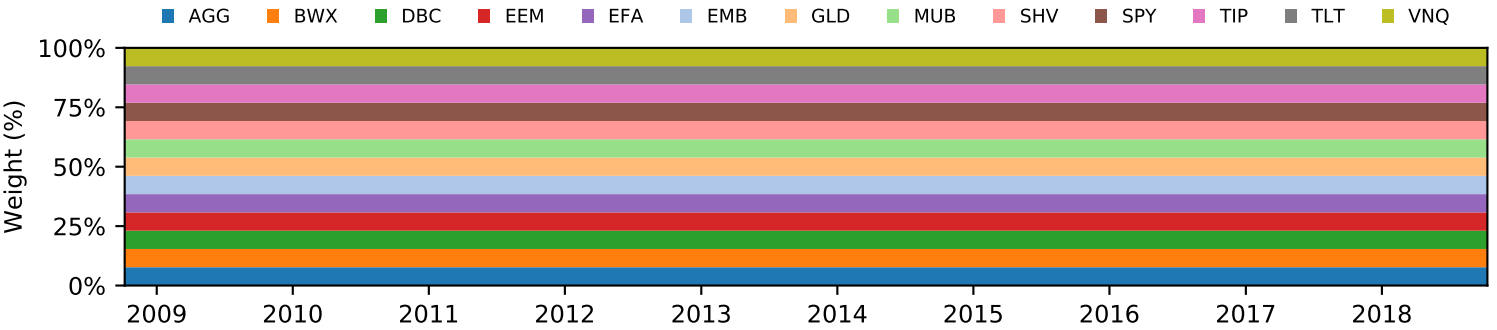
Yearly Returns (%)



Tearsheet #14

Portfolio Tearsheet: Equal Weight





Overview

Portfolio Code:	EW
Start Date:	2008-10-07
End Date:	2018-10-11
Rebalanced:	Daily
Trend Following:	N/A
Weighting:	EW

Statistics

Total Return:	67.35%
CAGR:	5.71%
Annual Volatility:	9.08%
Sharpe:	0.63
Max Drawdown:	-14.37%
Sortino:	0.92

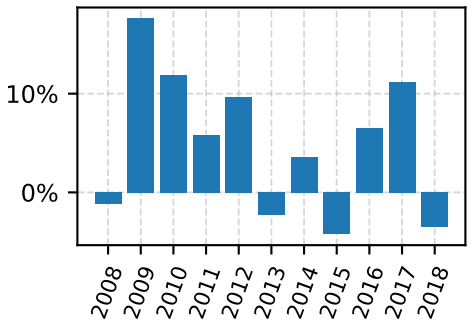
Statistics #2

$VaR_{99\%}$:	-1.68%
$CVaR_{99\%}$:	-2.39%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

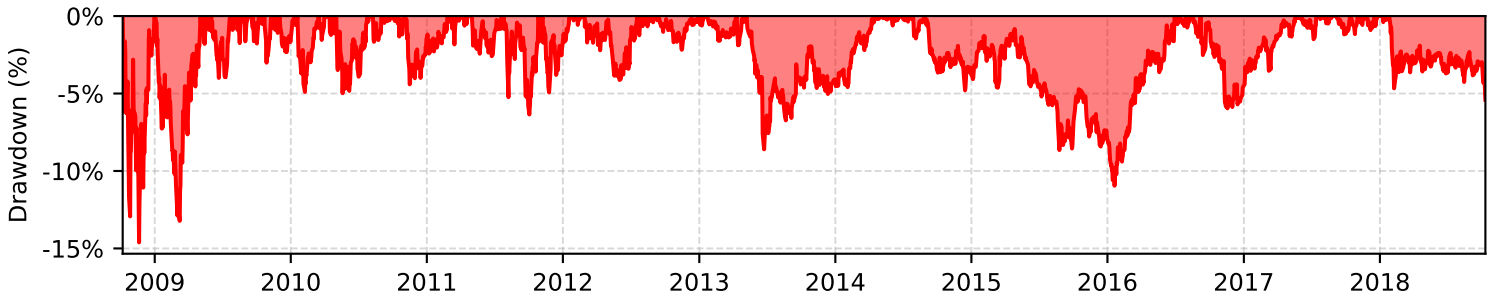
	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.6%	0.2%	0.5%	0.6%	0.3%	-0.2%	0.4%	0.0%	0.2%	0.3%	-0.2%
BWX	0.4%	0.5%	0.3%	0.3%	0.4%	-0.2%	-0.1%	-0.5%	0.1%	0.7%	-0.3%
DBC	-2.1%	1.4%	1.0%	-0.0%	0.3%	-0.6%	-2.5%	-2.3%	1.4%	0.4%	0.6%
EEM	0.3%	4.7%	1.4%	-1.2%	1.5%	-0.2%	-0.2%	-1.2%	1.0%	2.5%	-1.3%
EFA	-0.2%	2.2%	0.8%	-0.7%	1.4%	1.6%	-0.4%	0.0%	0.2%	1.8%	-0.6%
EMB	0.6%	1.2%	0.8%	0.6%	1.2%	-0.6%	0.5%	0.1%	0.7%	0.8%	-0.5%
GLD	0.3%	1.8%	2.1%	0.9%	0.6%	-2.4%	-0.1%	-0.8%	0.7%	1.0%	-0.5%
MUB	0.3%	0.5%	0.0%	1.0%	0.4%	-0.3%	0.7%	0.2%	-0.0%	0.3%	-0.1%
SHV	0.0%	0.0%	0.0%	0.0%	0.0%	-0.0%	0.0%	0.0%	0.0%	0.1%	0.1%
SPY	-0.7%	2.1%	1.2%	0.3%	1.2%	2.2%	1.0%	0.2%	0.9%	1.5%	0.3%
TIP	-0.2%	0.7%	0.5%	1.0%	0.5%	-0.7%	0.3%	-0.1%	0.4%	0.2%	-0.1%
TLT	1.6%	-1.8%	0.8%	2.4%	0.3%	-1.0%	1.9%	-0.1%	0.1%	0.7%	-0.6%
VNQ	-1.1%	3.5%	2.2%	0.9%	1.3%	0.3%	2.1%	0.3%	0.7%	0.4%	-0.4%

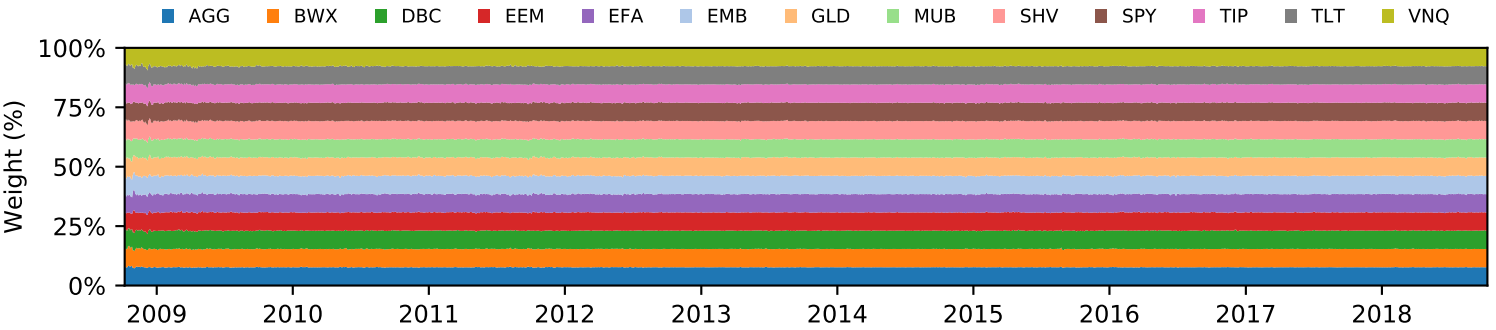
Yearly Returns (%)



Tearsheet #15

Portfolio Tearsheet: Equal Weight





Overview

Portfolio Code:	EW
Start Date:	2008-10-07
End Date:	2018-10-11
Rebalanced:	Weekly
Trend Following:	N/A
Weighting:	EW

Statistics

Total Return:	62.92%
CAGR:	5.42%
Annual Volatility:	9.01%
Sharpe:	0.60
Max Drawdown:	-14.61%
Sortino:	0.87

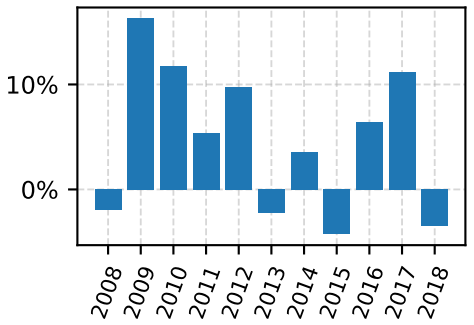
Statistics #2

$VaR_{99\%}$:	-1.68%
$CVaR_{99\%}$:	-2.38%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

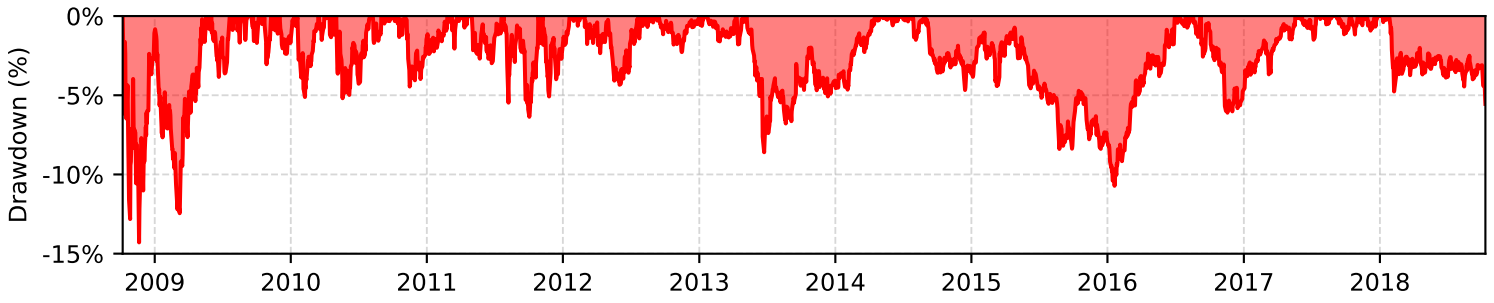
	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.5%	0.2%	0.5%	0.6%	0.3%	-0.2%	0.4%	0.0%	0.2%	0.3%	-0.2%
BWX	0.4%	0.4%	0.3%	0.3%	0.4%	-0.2%	-0.1%	-0.5%	0.1%	0.7%	-0.3%
DBC	-2.2%	1.4%	1.0%	-0.0%	0.3%	-0.6%	-2.5%	-2.3%	1.4%	0.4%	0.6%
EEM	0.1%	4.6%	1.4%	-1.3%	1.5%	-0.2%	-0.2%	-1.2%	0.9%	2.5%	-1.3%
EFA	-0.3%	2.2%	0.8%	-0.8%	1.5%	1.6%	-0.4%	0.0%	0.2%	1.8%	-0.6%
EMB	0.5%	1.2%	0.8%	0.6%	1.2%	-0.6%	0.5%	0.1%	0.7%	0.8%	-0.5%
GLD	0.3%	1.8%	2.1%	0.8%	0.6%	-2.4%	-0.1%	-0.8%	0.7%	1.0%	-0.5%
MUB	0.3%	0.5%	0.0%	1.0%	0.4%	-0.3%	0.7%	0.2%	-0.0%	0.3%	-0.1%
SHV	0.0%	0.0%	0.0%	0.0%	0.0%	-0.0%	0.0%	0.0%	0.0%	0.1%	0.1%
SPY	-0.8%	2.0%	1.2%	0.3%	1.2%	2.2%	1.0%	0.2%	0.9%	1.5%	0.3%
TIP	-0.2%	0.7%	0.5%	1.0%	0.5%	-0.7%	0.3%	-0.1%	0.4%	0.2%	-0.1%
TLT	1.6%	-1.8%	0.7%	2.4%	0.3%	-1.1%	1.9%	-0.1%	0.1%	0.7%	-0.6%
VNQ	-1.2%	2.7%	2.2%	0.8%	1.3%	0.3%	2.1%	0.3%	0.7%	0.4%	-0.4%

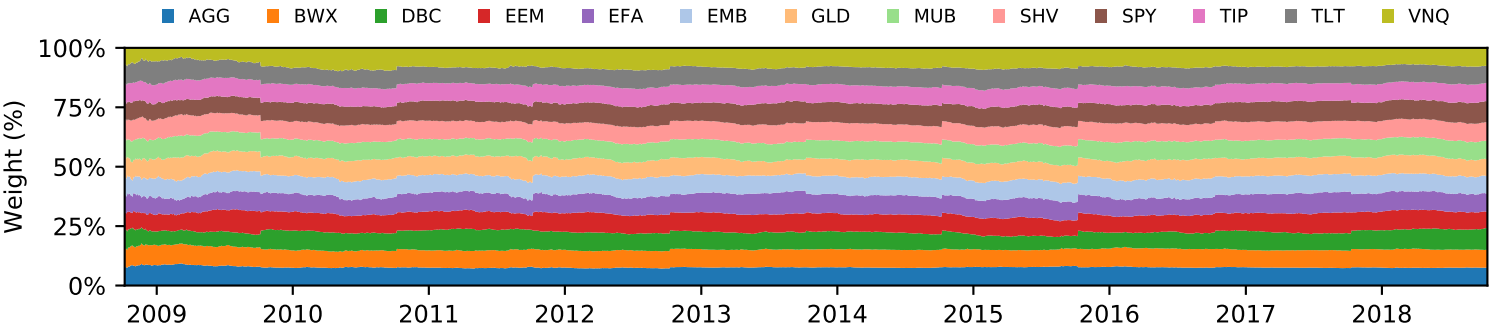
Yearly Returns (%)



Tearsheet #16

Portfolio Tearsheet: Equal Weight





Overview

Portfolio Code:	EW
Start Date:	2008-10-07
End Date:	2018-10-11
Rebalanced:	Yearly
Trend Following:	N/A
Weighting:	EW

Statistics

Total Return:	56.15%
CAGR:	4.93%
Annual Volatility:	8.51%
Sharpe:	0.58
Max Drawdown:	-14.29%
Sortino:	0.83

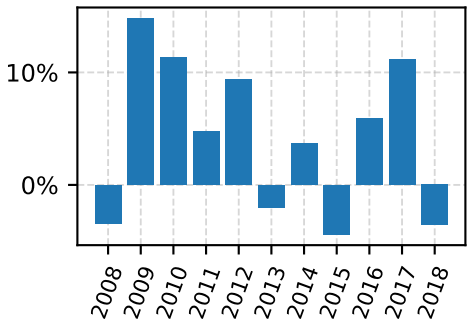
Statistics #2

$VaR_{99\%}$:	-1.48%
$CVaR_{99\%}$:	-2.19%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

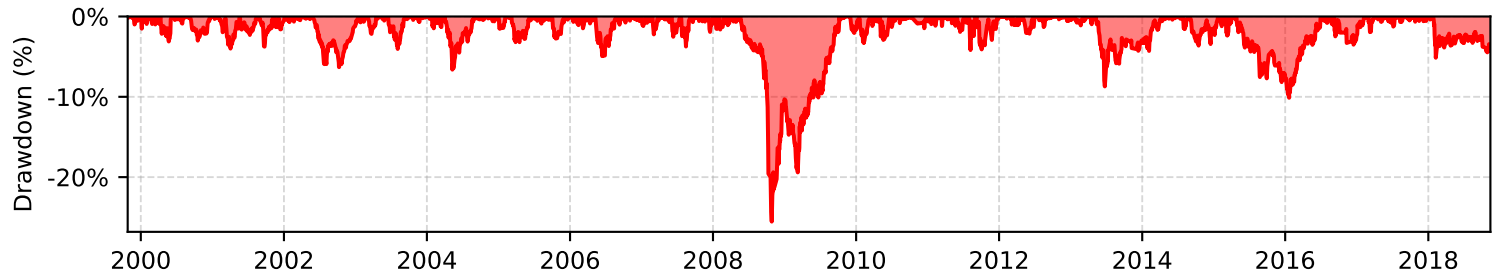
	2008	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.6%	0.2%	0.5%	0.5%	0.3%	-0.2%	0.4%	0.0%	0.2%	0.3%	-0.2%
BWX	0.4%	0.5%	0.3%	0.3%	0.4%	-0.2%	-0.1%	-0.5%	0.1%	0.7%	-0.3%
DBC	-2.2%	1.2%	0.9%	-0.1%	0.3%	-0.6%	-2.2%	-2.1%	1.2%	0.4%	0.6%
EEM	-0.6%	4.9%	1.3%	-1.3%	1.4%	-0.3%	-0.3%	-1.3%	0.8%	2.6%	-1.3%
EFA	-0.6%	1.9%	0.6%	-0.8%	1.4%	1.7%	-0.5%	-0.1%	0.1%	1.9%	-0.6%
EMB	0.5%	1.3%	0.8%	0.6%	1.2%	-0.6%	0.5%	0.1%	0.7%	0.8%	-0.5%
GLD	0.3%	1.8%	2.2%	0.8%	0.5%	-2.2%	-0.1%	-0.8%	0.7%	0.9%	-0.5%
MUB	0.3%	0.6%	0.0%	0.9%	0.4%	-0.3%	0.7%	0.2%	-0.0%	0.3%	-0.1%
SHV	0.0%	0.0%	0.0%	0.0%	0.0%	-0.0%	-0.0%	0.0%	0.0%	0.1%	0.1%
SPY	-0.9%	1.8%	1.1%	0.2%	1.3%	2.3%	1.0%	0.1%	0.9%	1.6%	0.3%
TIP	-0.2%	0.7%	0.4%	0.9%	0.5%	-0.7%	0.3%	-0.1%	0.4%	0.2%	-0.1%
TLT	1.9%	-2.2%	0.6%	2.2%	0.2%	-1.0%	1.9%	-0.1%	0.2%	0.6%	-0.6%
VNQ	-2.1%	1.8%	2.3%	0.8%	1.4%	0.2%	2.2%	0.2%	0.7%	0.4%	-0.4%

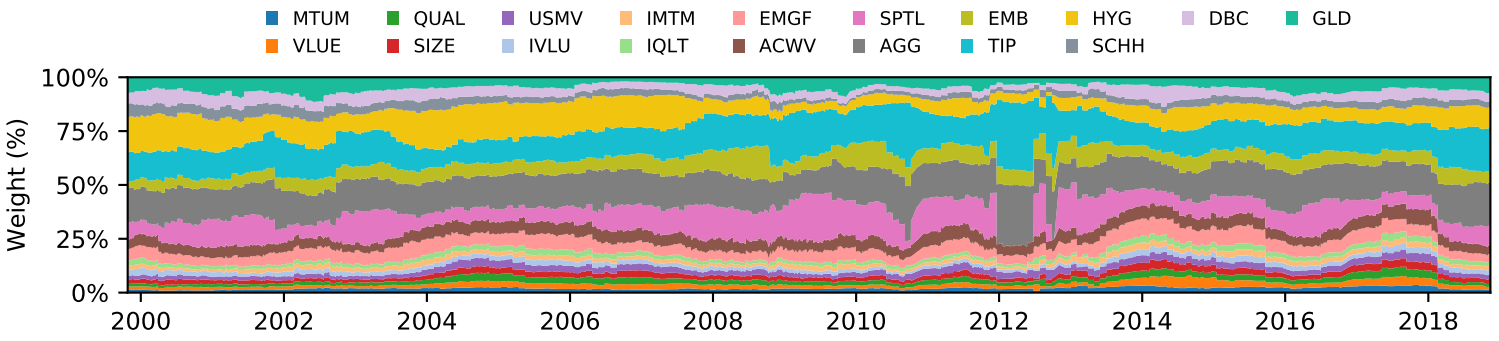
Yearly Returns (%)



Tearsheet #17

Portfolio Tearsheet: Rolling lookback 200-Day





Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	RP 200

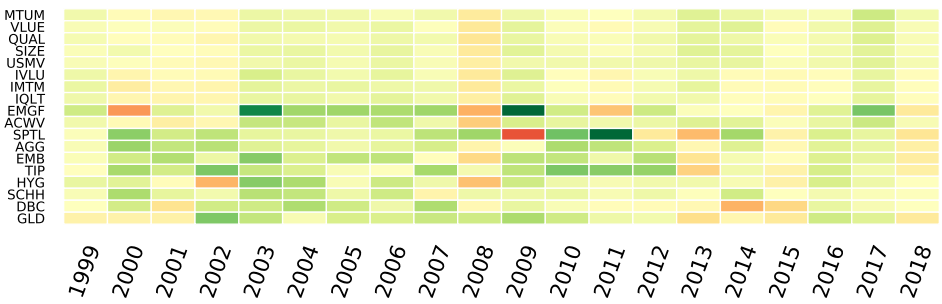
Statistics

Total Return:	266.98%
CAGR:	7.68%
Annual Volatility:	5.55%
Sharpe:	1.16
Max Drawdown:	-25.54%
Sortino:	1.66

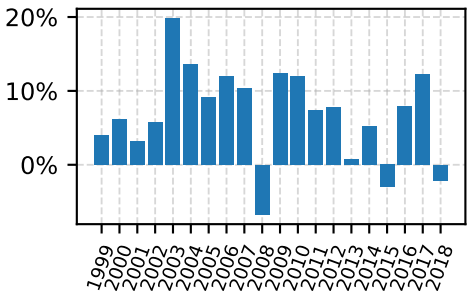
Statistics #2

$VaR_{99\%}$:	-0.90%
$CVaR_{99\%}$:	-1.44%
Beta:	0.18
Alpha:	6.07%
R-Squared:	35.43%
Treynor:	0.36

Performance Attribution

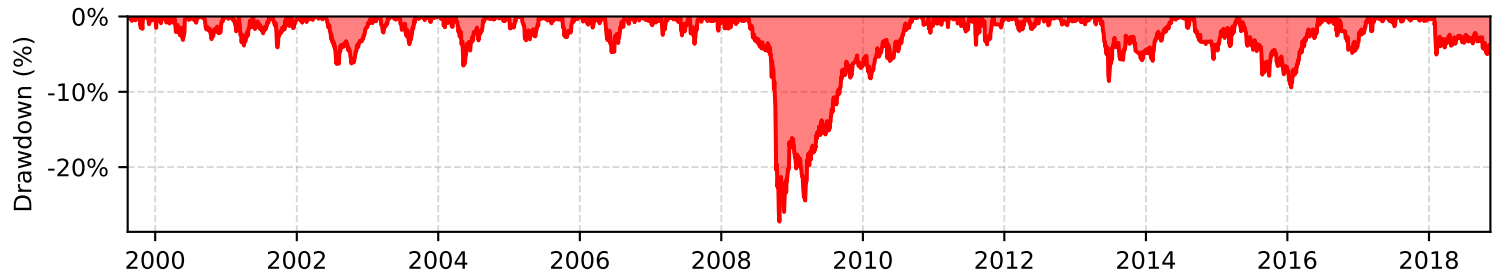


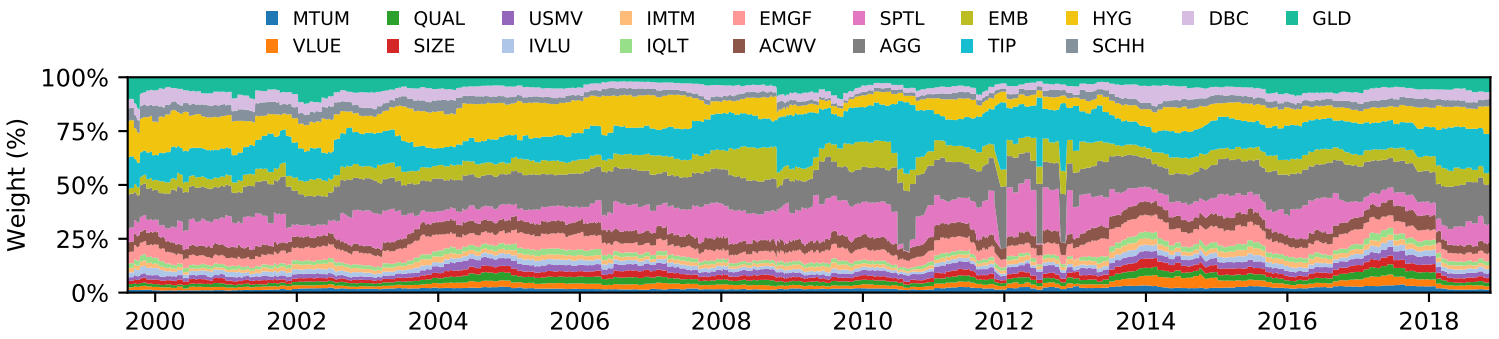
Yearly Returns (%)



Tearsheet #18

Portfolio Tearsheet: Rolling lookback 150-Day





Overview

Portfolio Code:	RP
Start Date:	1999-08-12
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	RP 150

Statistics

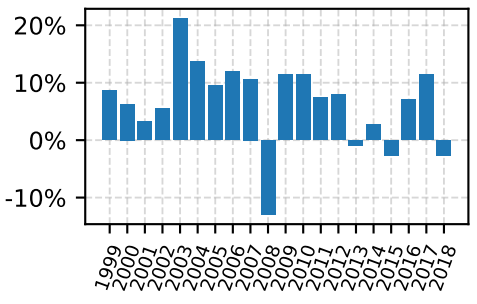
Total Return:	238.64%
CAGR:	7.07%
Annual Volatility:	5.57%
Sharpe:	1.04
Max Drawdown:	-27.23%
Sortino:	1.47

Statistics #2

$VaR_{99\%}$:	-0.91%
$CVaR_{99\%}$:	-1.48%
Beta:	0.17
Alpha:	5.86%
R-Squared:	34.25%
Treynor:	0.34

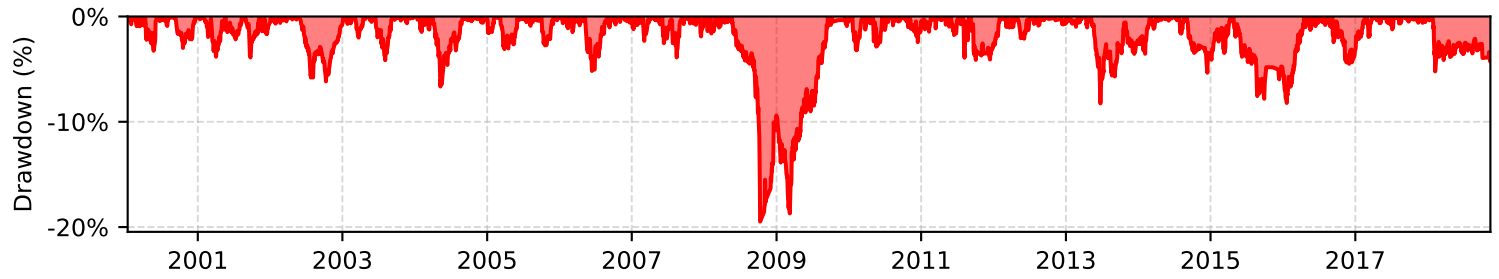
Performance Attribution

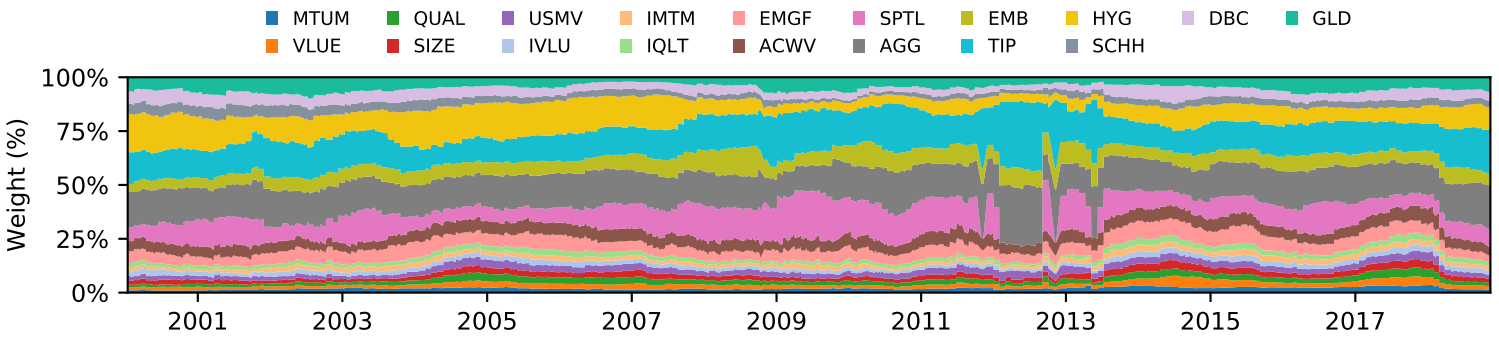
Yearly Returns (%)



Tearsheet #19

Portfolio Tearsheet: Rolling lookback 250-Day





Overview

Portfolio Code:	RP
Start Date:	2000-01-13
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	RP 250

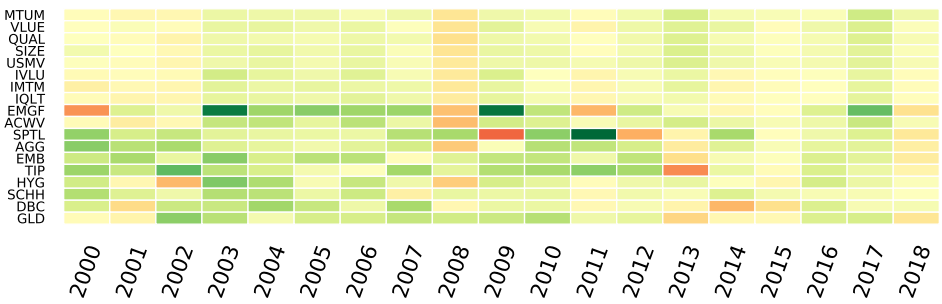
Statistics

Total Return:	246.20%
CAGR:	7.47%
Annual Volatility:	5.36%
Sharpe:	1.16
Max Drawdown:	-19.48%
Sortino:	1.68

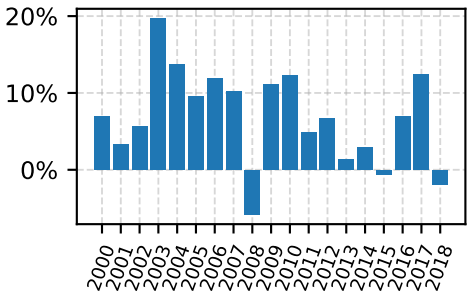
Statistics #2

$VaR_{99\%}$:	-0.89%
$CVaR_{99\%}$:	-1.32%
Beta:	0.17
Alpha:	6.14%
R-Squared:	32.74%
Treynor:	0.36

Performance Attribution

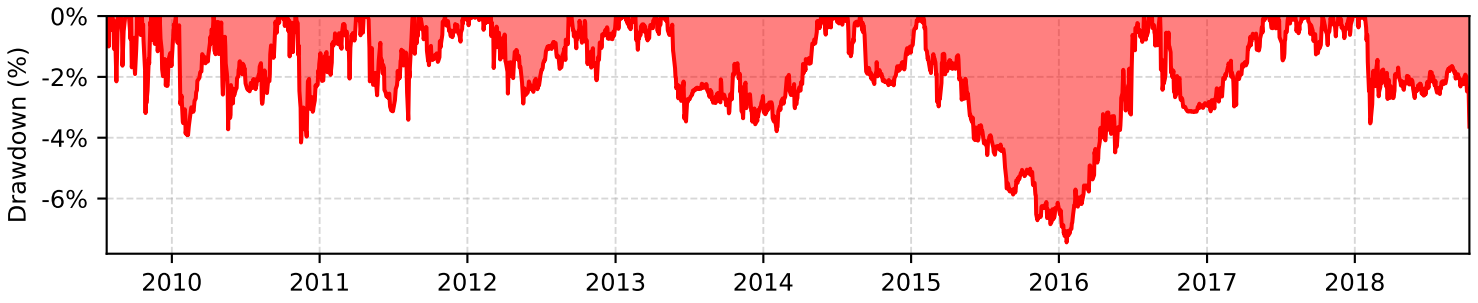


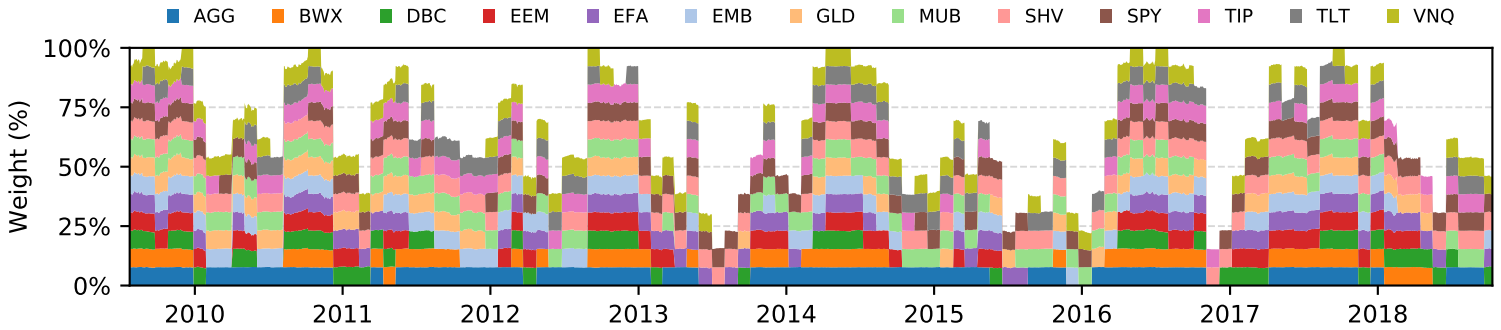
Yearly Returns (%)



Tearsheet #20

Portfolio Tearsheet: Trend Following Equal Weight





Overview

Portfolio Code:	EW_TF
Start Date:	2009-07-24
End Date:	2018-10-11
Rebalanced:	Monthly
Trend Following:	100 SMA
Weighting:	EW

Statistics

Total Return:	33.54%
CAGR:	3.30%
Annual Volatility:	4.56%
Sharpe:	0.72
Max Drawdown:	-7.44%
Sortino:	1.02

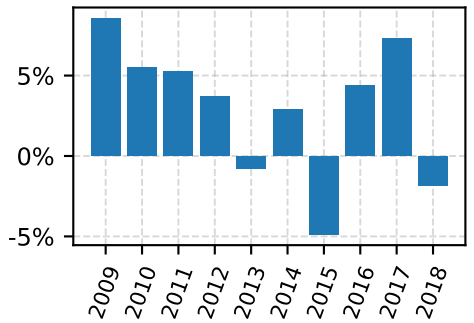
Statistics #2

$VaR_{99\%}$:	-0.93%
$CVaR_{99\%}$:	-1.11%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

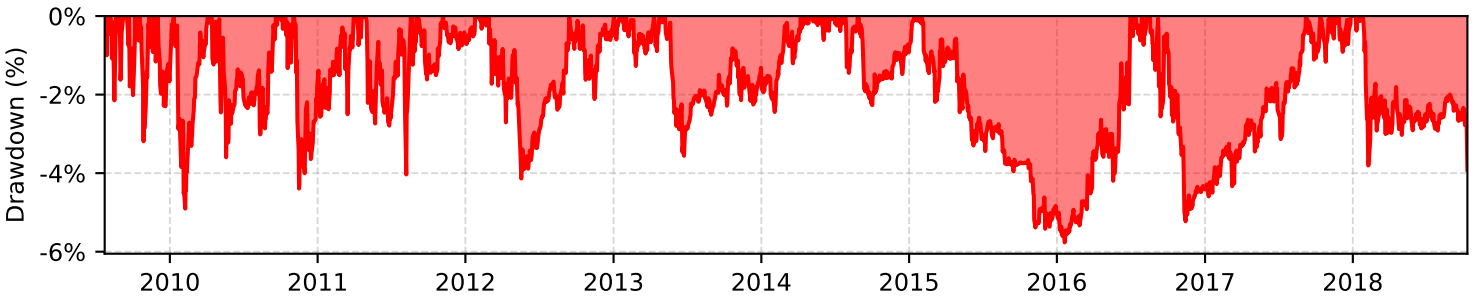
	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.2%	0.3%	0.4%	0.2%	-0.1%	0.4%	0.0%	0.3%	0.1%	-0.0%
BWX	0.3%	-0.1%	0.4%	-0.0%	-0.5%	0.1%	-0.3%	0.3%	0.4%	-0.1%
DBC	-0.2%	0.2%	0.2%	-0.5%	-0.6%	-0.1%	-0.3%	0.3%	0.7%	0.3%
EEM	1.3%	-0.3%	-0.9%	1.0%	-0.2%	-0.4%	-0.7%	0.7%	2.3%	0.1%
EFA	1.0%	0.2%	0.1%	0.3%	1.1%	-0.4%	-0.2%	-0.0%	1.6%	-0.6%
EMB	0.7%	0.8%	0.5%	0.9%	-0.7%	0.6%	-0.4%	0.8%	0.6%	-0.3%
GLD	1.2%	2.0%	0.7%	-0.4%	-0.5%	-1.1%	-1.7%	0.6%	-0.2%	-0.3%
MUB	0.3%	0.0%	0.8%	0.3%	-0.2%	0.7%	0.2%	0.3%	0.2%	-0.1%
SHV	0.0%	0.0%	0.0%	0.0%	-0.0%	-0.0%	-0.0%	0.0%	0.1%	0.1%
SPY	1.1%	0.5%	-0.2%	0.6%	2.2%	0.4%	-0.5%	-0.2%	1.5%	0.1%
TIP	0.4%	0.2%	0.8%	0.3%	-0.5%	0.1%	-0.4%	0.1%	-0.1%	-0.3%
TLT	-0.6%	0.8%	2.1%	-0.0%	-0.7%	1.5%	-0.5%	0.5%	0.3%	-0.3%
VNQ	2.8%	1.0%	0.3%	1.0%	-0.0%	1.2%	-0.1%	0.6%	-0.3%	-0.5%

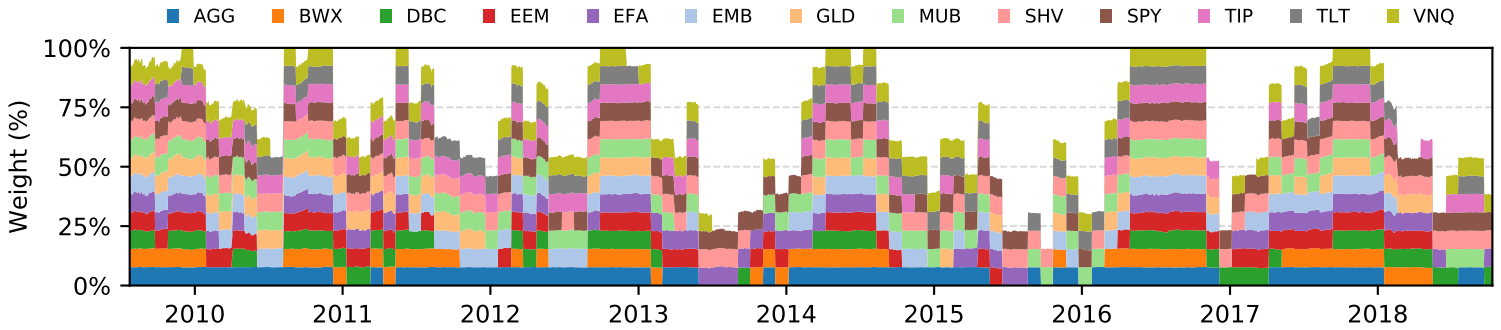
Yearly Returns (%)



Tearsheet #21

Portfolio Tearsheet: Trend Following Equal Weight





Overview

Portfolio Code:	EW_TF
Start Date:	2009-07-24
End Date:	2018-10-11
Rebalanced:	Monthly
Trend Following:	150 SMA
Weighting:	EW

Statistics

Total Return:	32.73%
CAGR:	3.24%
Annual Volatility:	4.75%
Sharpe:	0.68
Max Drawdown:	-5.76%
Sortino:	0.95

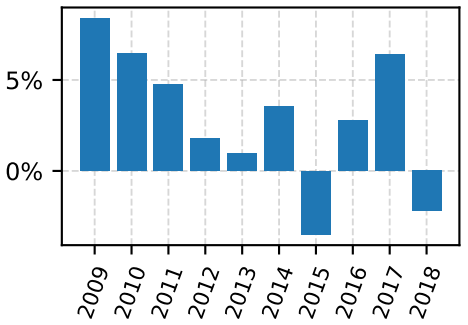
Statistics #2

$VaR_{99\%}$:	-0.96%
$CVaR_{99\%}$:	-1.17%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

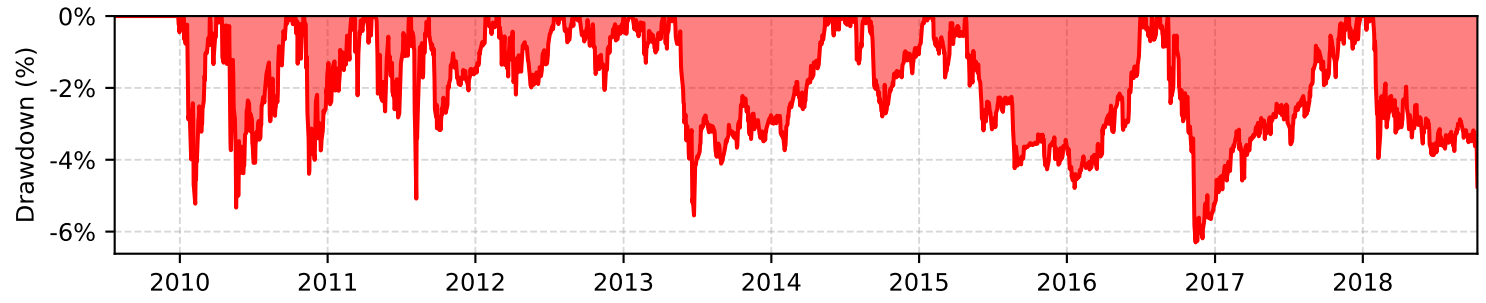
	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.2%	0.4%	0.4%	0.3%	-0.2%	0.4%	-0.1%	0.1%	0.2%	-0.1%
BWX	0.3%	0.0%	0.2%	-0.0%	-0.3%	0.2%	-0.3%	0.3%	0.5%	-0.1%
DBC	-0.3%	0.2%	0.5%	-1.0%	-0.5%	-0.3%	0.0%	0.8%	0.0%	0.3%
EEM	1.3%	0.6%	-0.9%	-0.1%	-0.3%	-0.4%	-0.4%	0.5%	2.3%	0.0%
EFA	1.0%	0.1%	0.2%	-0.1%	1.6%	0.0%	-0.2%	-0.0%	1.6%	-0.6%
EMB	0.7%	0.8%	0.3%	1.2%	-0.5%	0.2%	-0.4%	0.5%	0.6%	-0.2%
GLD	1.2%	2.0%	0.9%	-0.4%	-0.0%	-1.1%	-1.1%	-0.3%	-0.7%	-0.3%
MUB	0.3%	-0.0%	0.6%	0.4%	-0.2%	0.7%	0.1%	0.2%	0.2%	-0.1%
SHV	0.0%	0.0%	0.0%	-0.0%	-0.0%	-0.0%	-0.0%	0.0%	0.1%	0.1%
SPY	1.1%	0.5%	-0.5%	0.7%	2.2%	0.4%	-0.1%	-0.2%	1.5%	0.3%
TIP	0.4%	0.4%	0.6%	0.5%	-0.3%	0.2%	-0.4%	0.1%	-0.1%	-0.3%
TLT	-0.7%	0.4%	2.1%	-0.2%	-0.5%	1.5%	-0.8%	0.8%	0.3%	-0.7%
VNQ	2.8%	1.0%	0.4%	0.8%	-0.0%	1.8%	0.1%	-0.0%	-0.0%	-0.5%

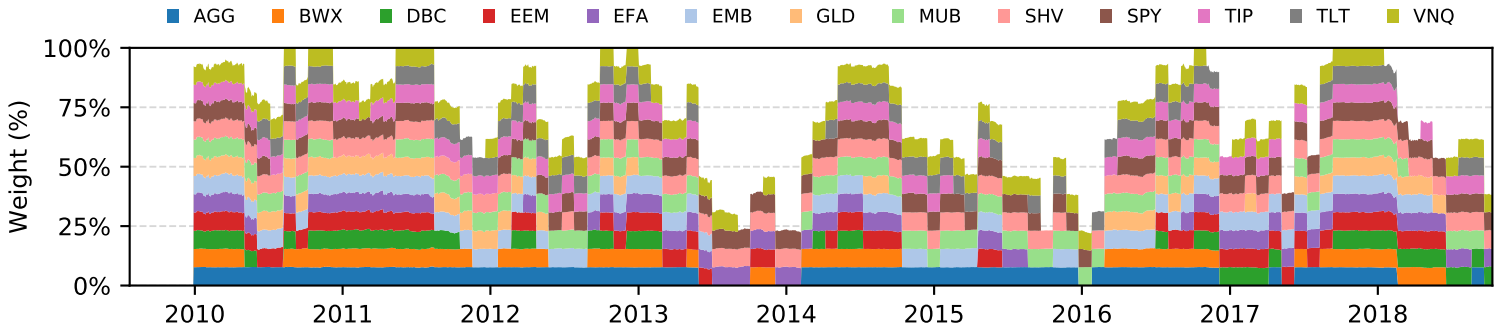
Yearly Returns (%)



Tearsheet #22

Portfolio Tearsheet: Trend Following Equal Weight





Overview

Portfolio Code:	EW_TF
Start Date:	2009-07-24
End Date:	2018-10-11
Rebalanced:	Monthly
Trend Following:	300 SMA
Weighting:	EW

Statistics

Total Return:	21.43%
CAGR:	2.24%
Annual Volatility:	4.59%
Sharpe:	0.49
Max Drawdown:	-6.30%
Sortino:	0.67

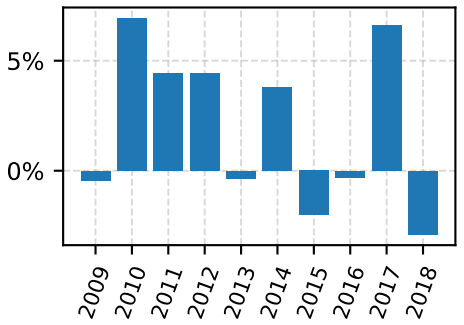
Statistics #2

$VaR_{99\%}$:	-0.92%
$CVaR_{99\%}$:	-1.22%
Placeholder:	N/A
Placeholder:	N/A
Placeholder:	N/A
Placeholder	N/A

Performance Attribution

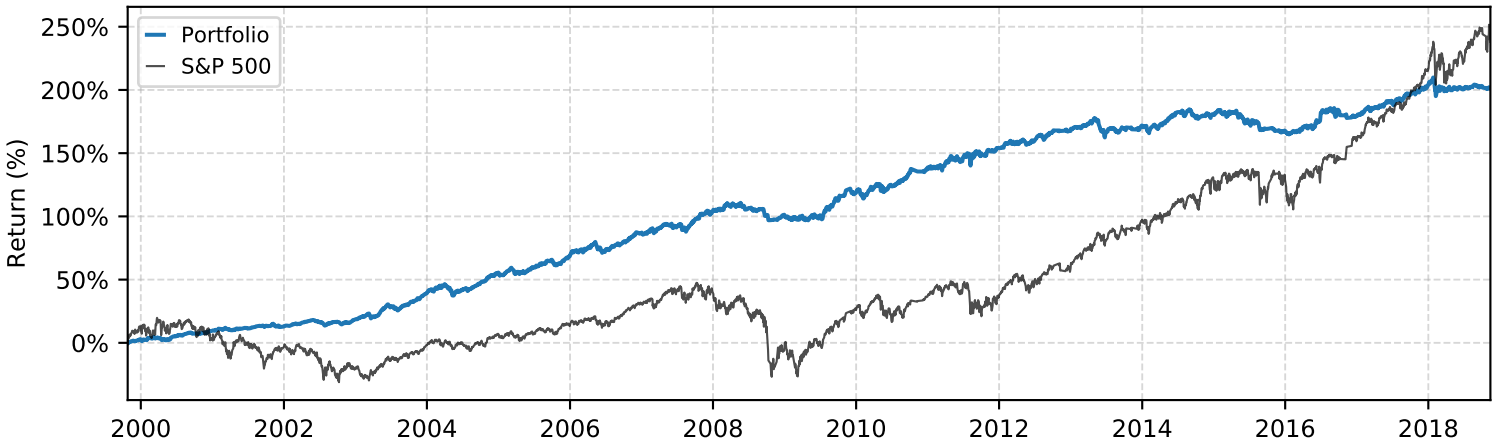
	2009	2010	2011	2012	2013	2014	2015	2016	2017	2018
AGG	0.0%	0.5%	0.6%	0.3%	-0.1%	0.3%	0.0%	0.1%	0.1%	-0.2%
BWX	-0.0%	-0.2%	0.3%	0.2%	-0.6%	0.1%	0.0%	-0.1%	-0.1%	-0.2%
DBC	-0.0%	0.1%	0.0%	-0.6%	-0.2%	-0.2%	0.0%	-0.3%	0.0%	0.5%
EEM	0.0%	1.3%	-0.8%	0.2%	-1.0%	-0.0%	-0.4%	0.2%	2.5%	-0.4%
EFA	-0.1%	-0.8%	-0.4%	0.4%	1.6%	-0.6%	-0.4%	-0.1%	1.8%	-0.8%
EMB	0.0%	0.8%	0.6%	1.2%	-0.7%	0.0%	-0.0%	0.6%	0.8%	-0.4%
GLD	-0.1%	2.0%	0.8%	-0.4%	0.0%	-0.5%	0.0%	-0.3%	-0.4%	-0.1%
MUB	0.0%	-0.0%	0.6%	0.4%	-0.0%	0.5%	0.2%	-0.1%	0.1%	-0.1%
SHV	-0.0%	0.0%	0.0%	-0.0%	-0.0%	-0.0%	-0.0%	0.0%	0.1%	0.1%
SPY	-0.1%	0.5%	-0.2%	0.8%	2.2%	1.1%	-0.1%	0.3%	1.5%	0.3%
TIP	0.0%	0.5%	0.6%	0.5%	-0.2%	-0.0%	-0.2%	0.1%	-0.0%	-0.3%
TLT	0.0%	0.0%	2.1%	0.2%	-0.7%	1.3%	-0.9%	-0.2%	0.1%	-0.7%
VNQ	-0.2%	2.2%	0.2%	1.3%	-0.5%	1.8%	-0.2%	-0.5%	-0.0%	-0.5%

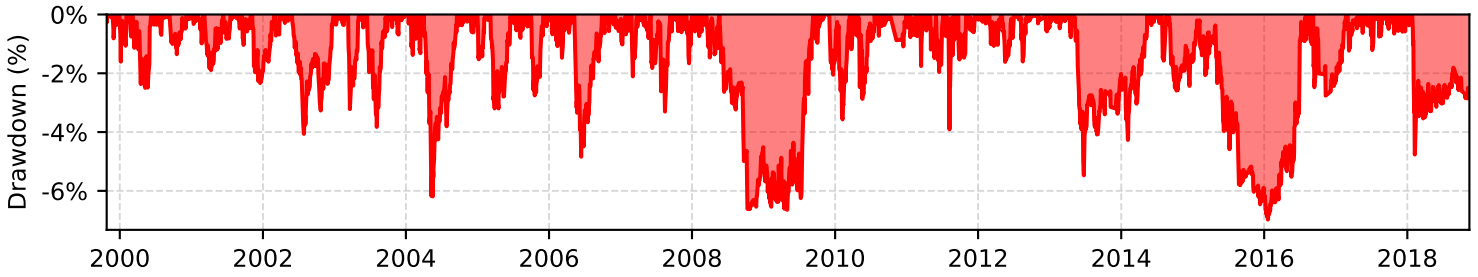
Yearly Returns (%)

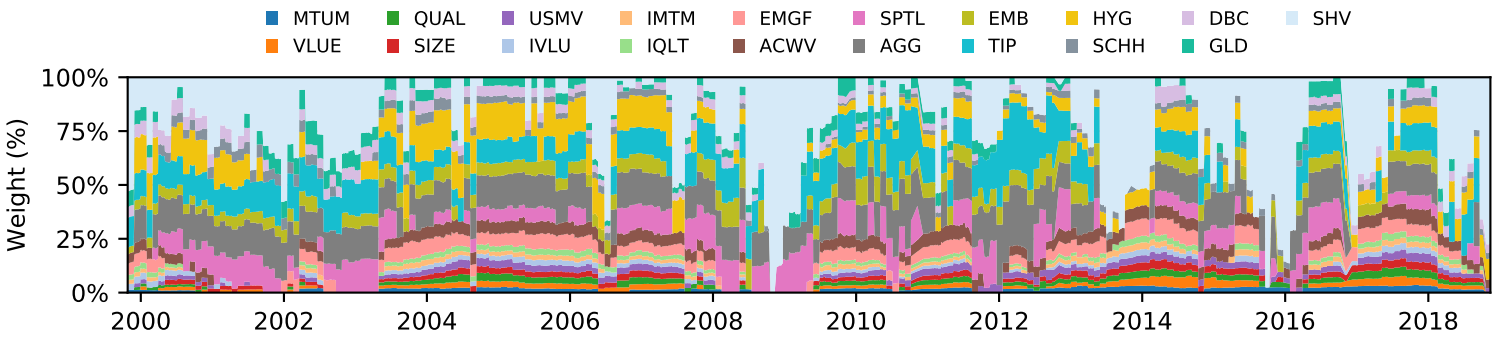


Tearsheet #23

Portfolio Tearsheet: Asset Universe 1







Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	RP 200

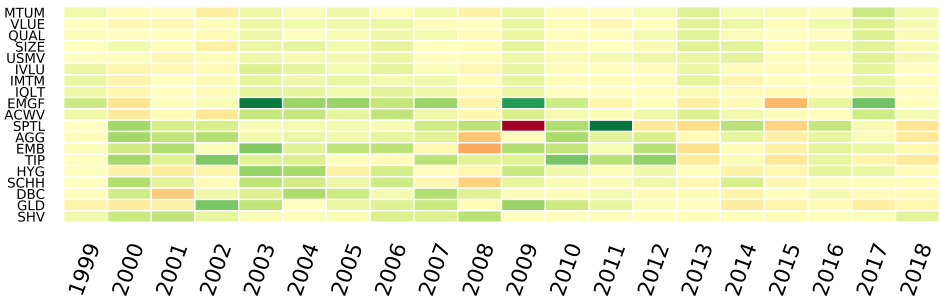
Statistics

Total Return:	201.03%
CAGR:	6.42%
Annual Volatility:	3.98%
Sharpe:	1.30
Max Drawdown:	-6.98%
Sortino:	1.88

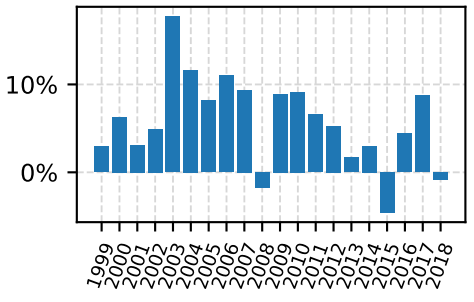
Statistics #2

$VaR_{99\%}$:	-0.68%
$CVaR_{99\%}$:	-0.91%
Beta:	0.06
Alpha:	5.87%
R-Squared:	7.95%
Treynor:	0.86

Performance Attribution

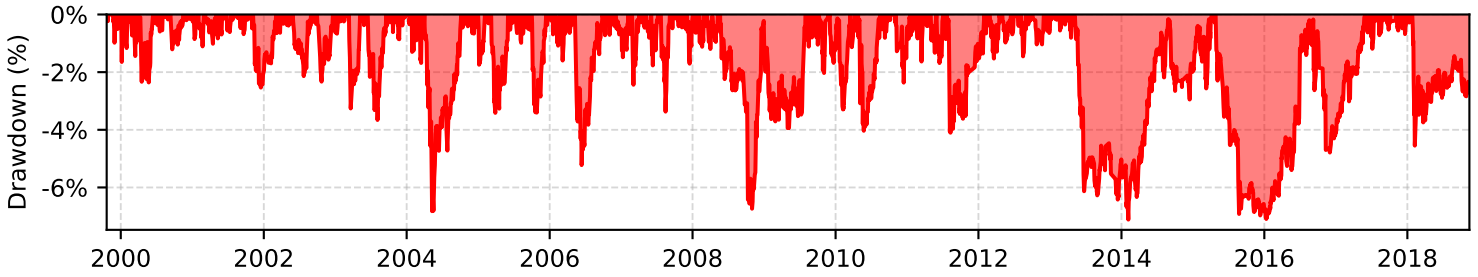


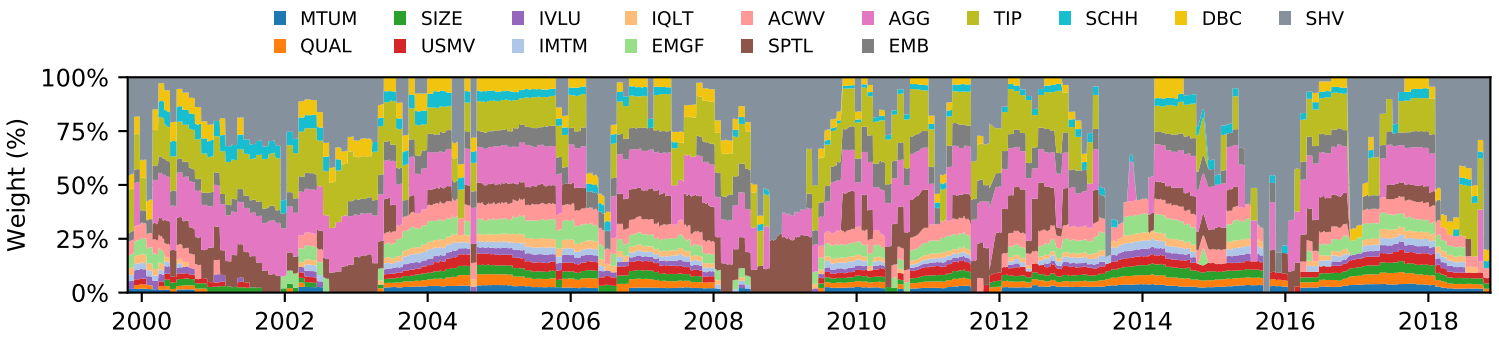
Yearly Returns (%)



Tearsheet #24

Portfolio Tearsheet: Asset Universe 2





Overview

Portfolio Code:	RP
Start Date:	1999-10-21
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	RP 200

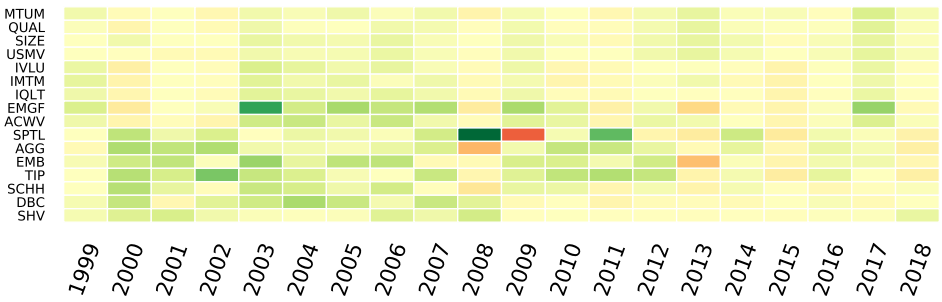
Statistics

Total Return:	223.97%
CAGR:	6.76%
Annual Volatility:	4.25%
Sharpe:	1.29
Max Drawdown:	-7.11%
Sortino:	1.86

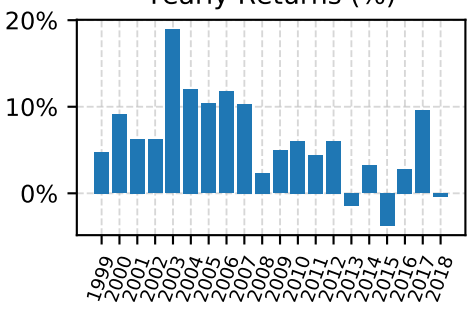
Statistics #2

$VaR_{99\%}$:	-0.75%
$CVaR_{99\%}$:	-0.98%
Beta:	0.06
Alpha:	6.38%
R-Squared:	6.19%
Treynor:	0.99

Performance Attribution

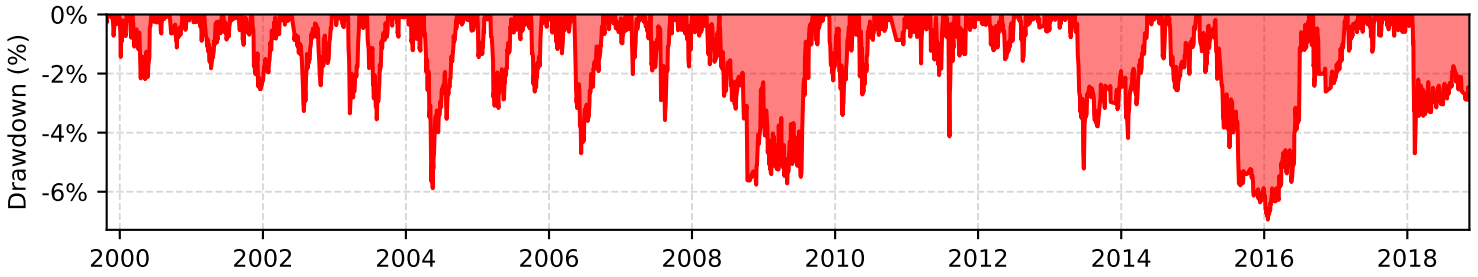


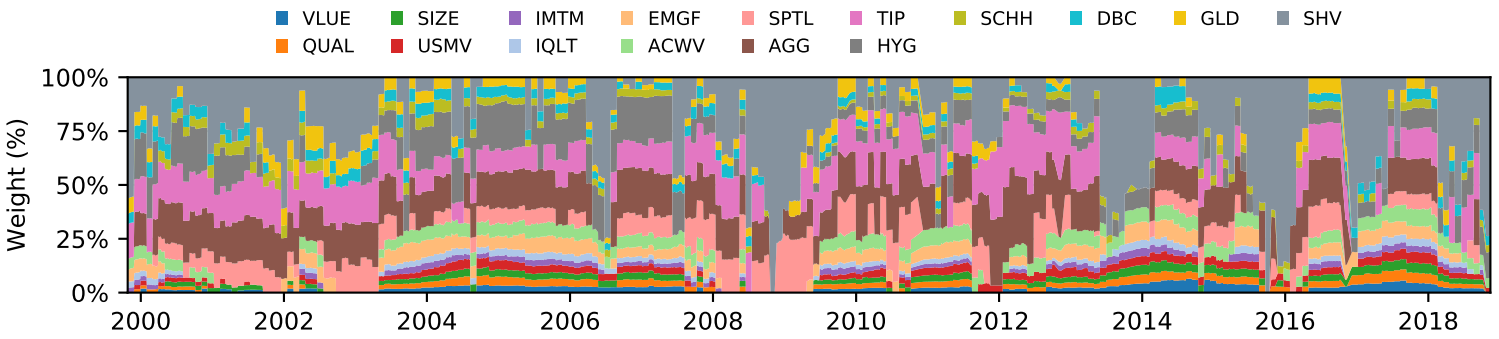
Yearly Returns (%)



Tearsheet #25

Portfolio Tearsheet: Asset Universe 3





Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	N/A
Weighting:	RP 200

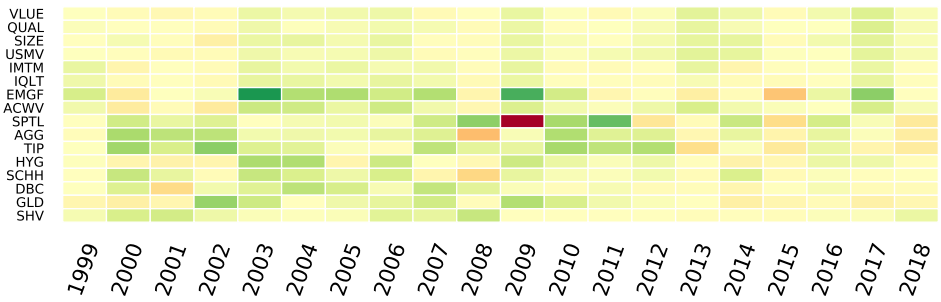
Statistics

Total Return:	179.91%
CAGR:	5.99%
Annual Volatility:	3.96%
Sharpe:	1.20
Max Drawdown:	-6.94%
Sortino:	1.74

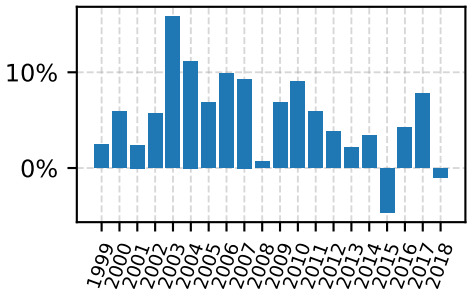
Statistics #2

$VaR_{99\%}$:	-0.68%
$CVaR_{99\%}$:	-0.89%
Beta:	0.05
Alpha:	5.52%
R-Squared:	6.20%
Treynor:	0.90

Performance Attribution

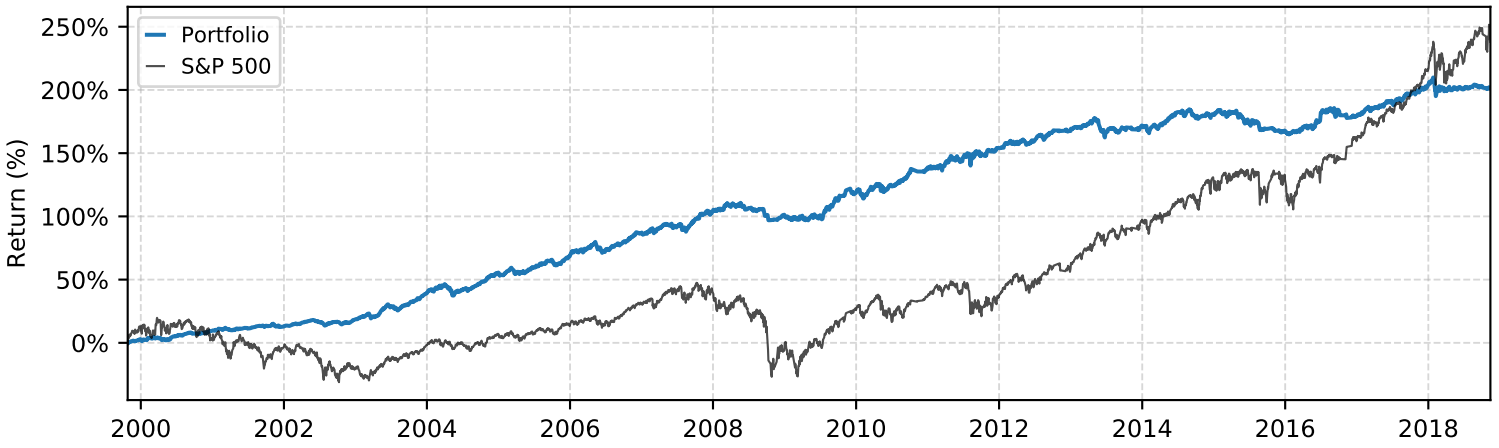


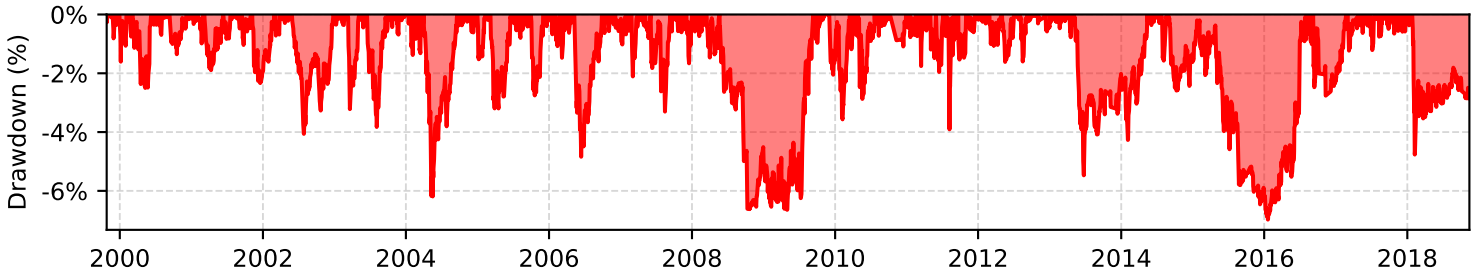
Yearly Returns (%)

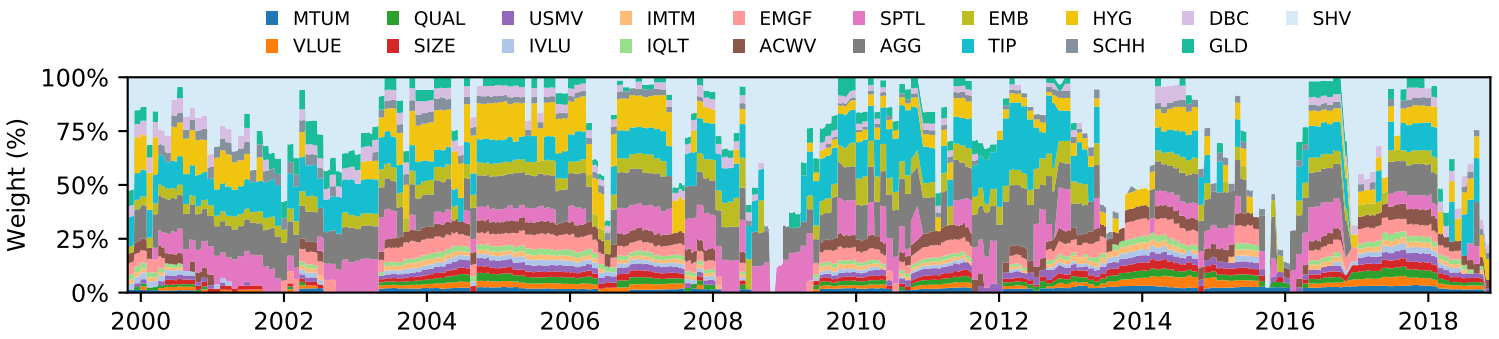


Tearsheet #26

Portfolio Tearsheet: Preservation







Overview

Portfolio Code:	RP
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP 200

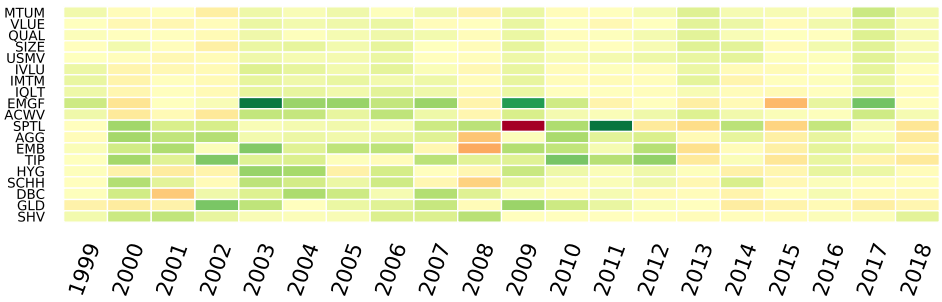
Statistics

Total Return:	201.03%
CAGR:	6.42%
Annual Volatility:	3.98%
Sharpe:	1.30
Max Drawdown:	-6.98%
Sortino:	1.88

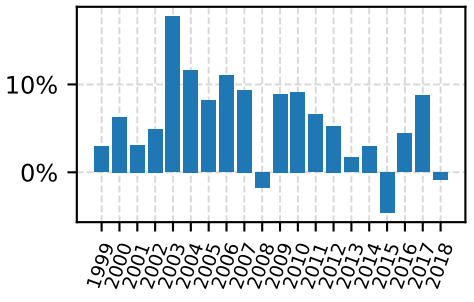
Statistics #2

$VaR_{99\%}$:	-0.68%
$CVaR_{99\%}$:	-0.91%
Beta:	0.06
Alpha:	5.87%
R-Squared:	7.95%
Treynor:	0.86

Performance Attribution

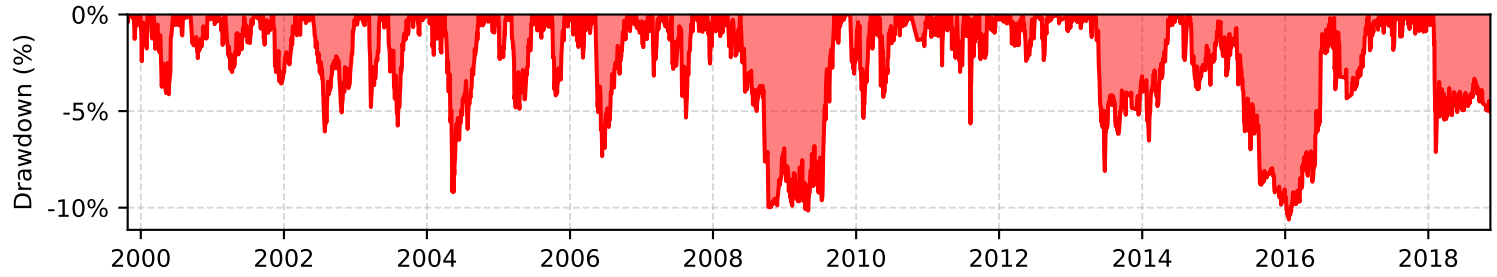


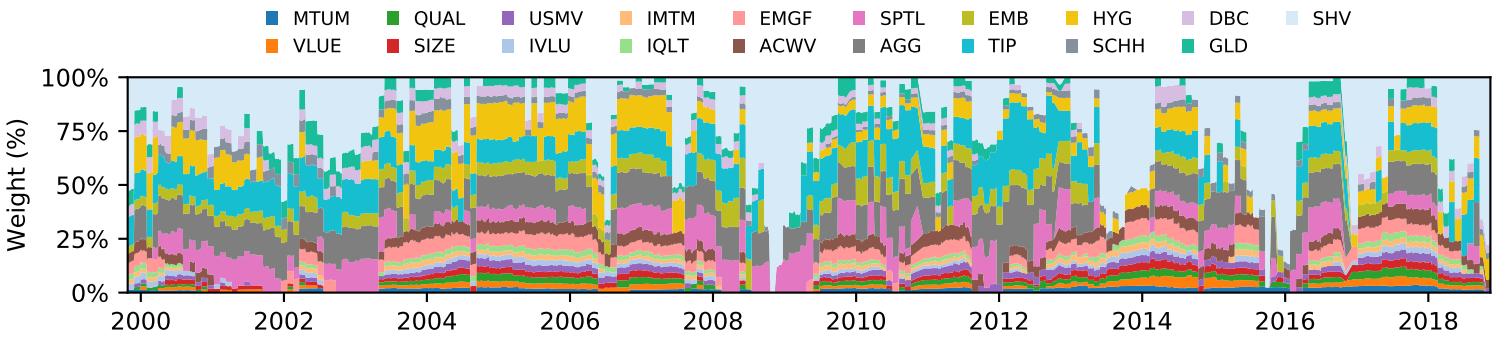
Yearly Returns (%)



Tearsheet #27

Portfolio Tearsheet: Conservative





Overview

Portfolio Code:	RP 1.5x
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP 200

Statistics

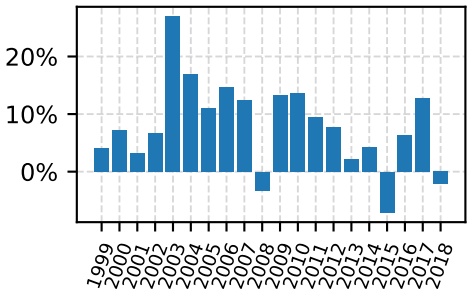
Total Return:	344.80%
CAGR:	8.87%
Annual Volatility:	5.96%
Sharpe:	1.28
Max Drawdown:	-10.62%
Sortino:	1.84

Statistics #2

$VaR_{99\%}$:	-1.03%
$CVaR_{99\%}$:	-1.36%
Beta:	0.09
Alpha:	8.04%
R-Squared:	7.78%
Treynor:	0.85

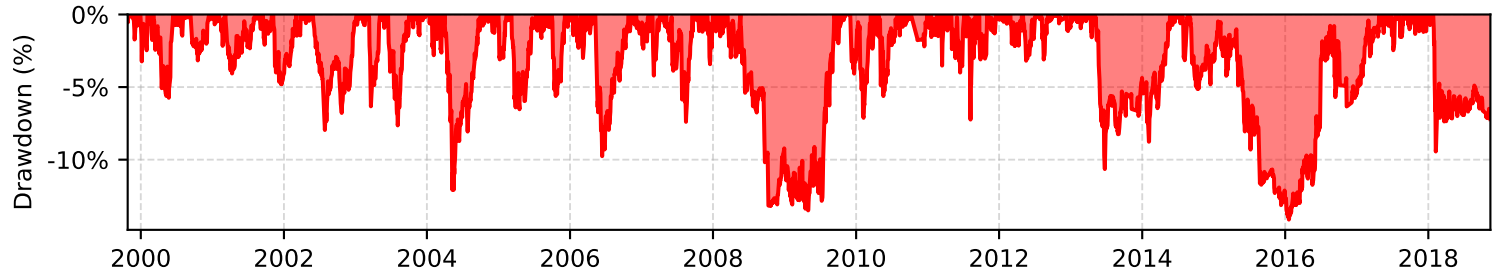
Performance Attribution

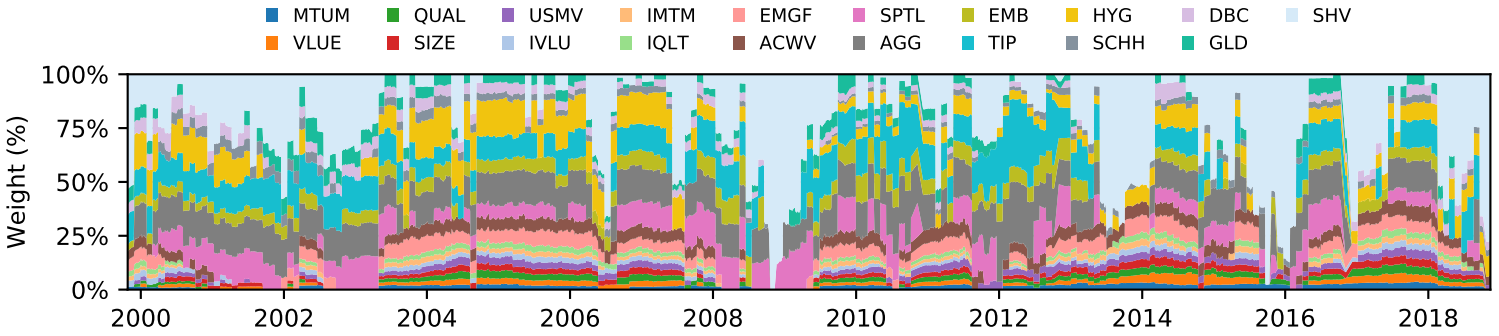
Yearly Returns (%)



Tearsheet #28

Portfolio Tearsheet: Balanced





Overview

Portfolio Code:	RP 2x
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP 200

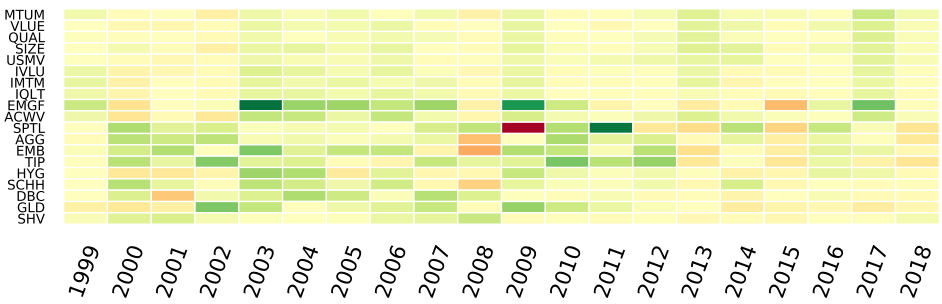
Statistics

Total Return:	552.76%
CAGR:	11.37%
Annual Volatility:	7.93%
Sharpe:	1.27
Max Drawdown:	-14.13%
Sortino:	1.83

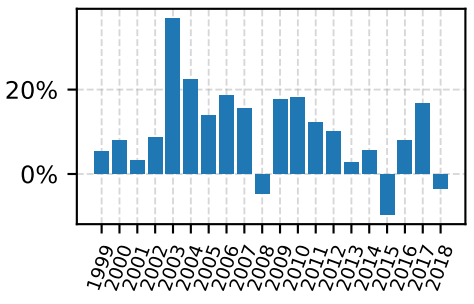
Statistics #2

$VaR_{99\%}$:	-1.37%
$CVaR_{99\%}$:	-1.80%
Beta:	0.12
Alpha:	10.26%
R-Squared:	7.61%
Treynor:	0.86

Performance Attribution

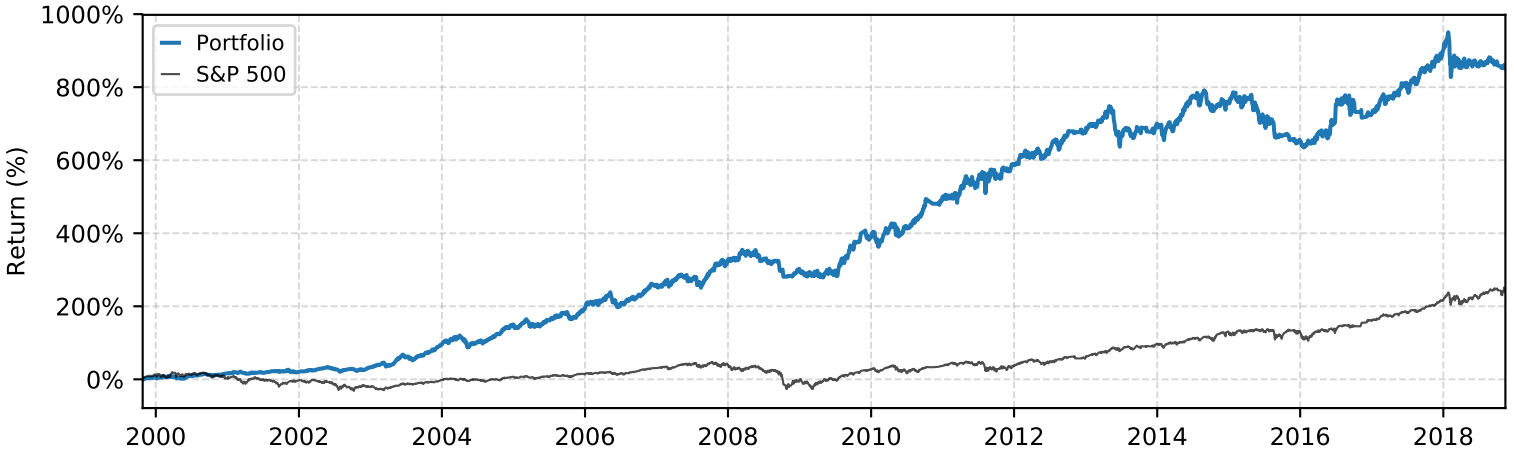


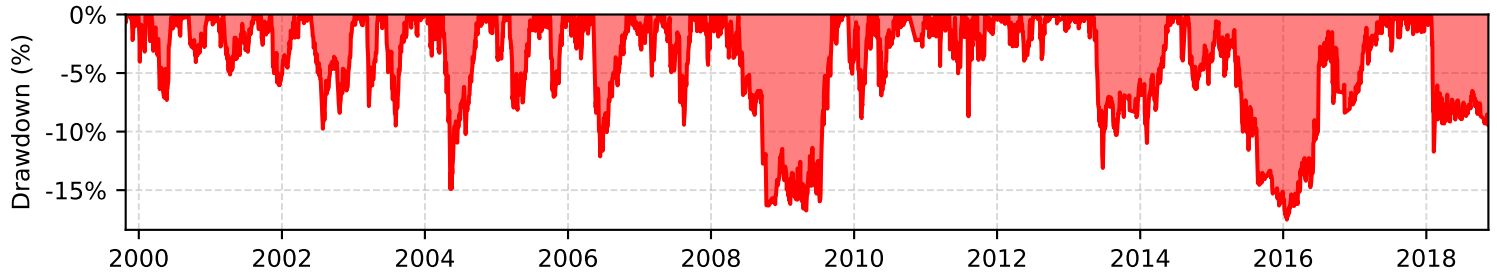
Yearly Returns (%)

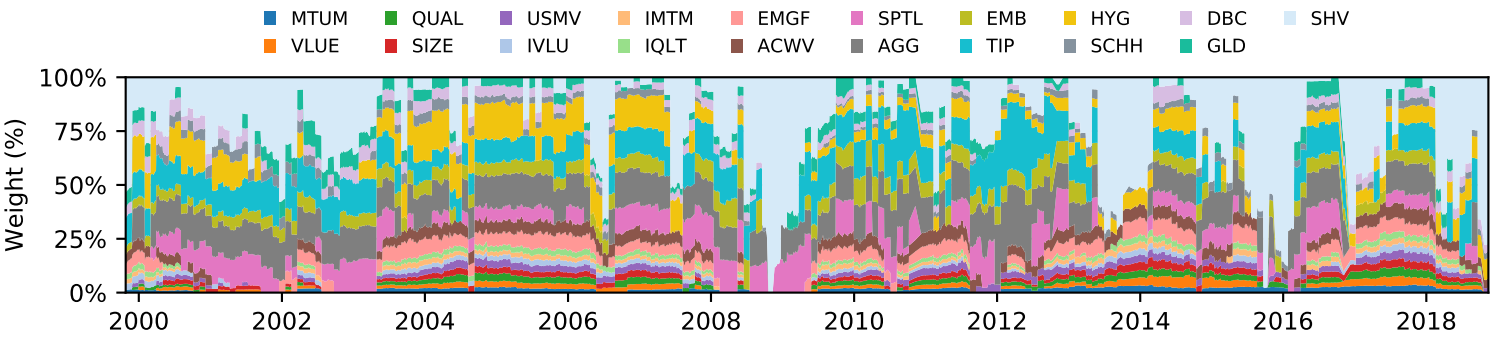


Tearsheet #29

Portfolio Tearsheet: Adventurous







Overview

Portfolio Code:	RP 2.5
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP 200

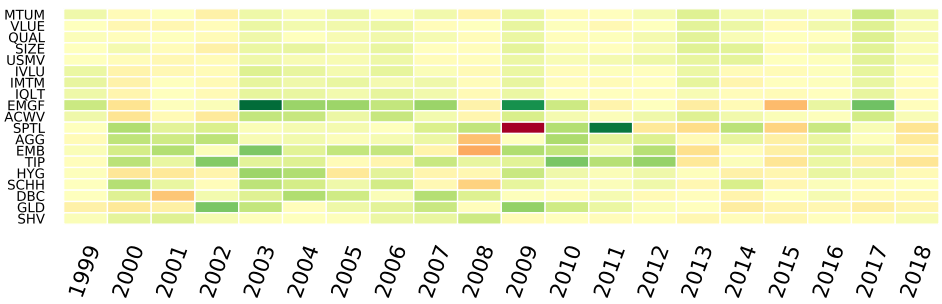
Statistics

Total Return:	851.59%
CAGR:	13.94%
Annual Volatility:	9.89%
Sharpe:	1.28
Max Drawdown:	-17.51%
Sortino:	1.83

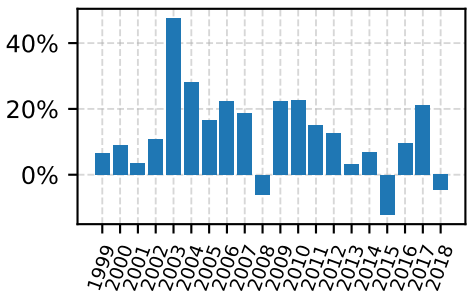
Statistics #2

$VaR_{99\%}$:	-1.74%
$CVaR_{99\%}$:	-2.24%
Beta:	0.14
Alpha:	12.54%
R-Squared:	7.43%
Treynor:	0.87

Performance Attribution

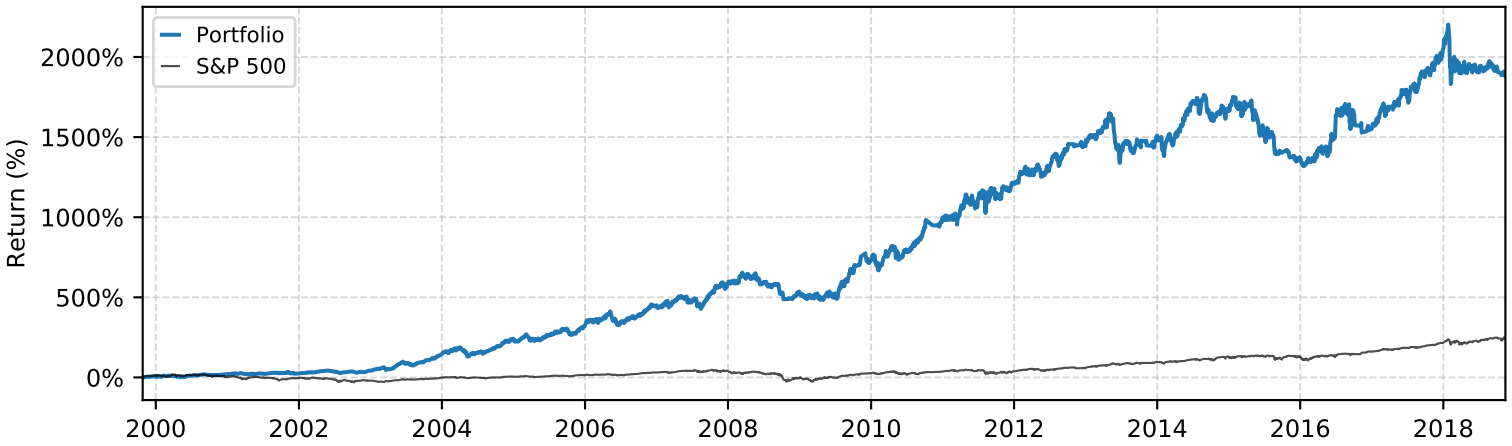


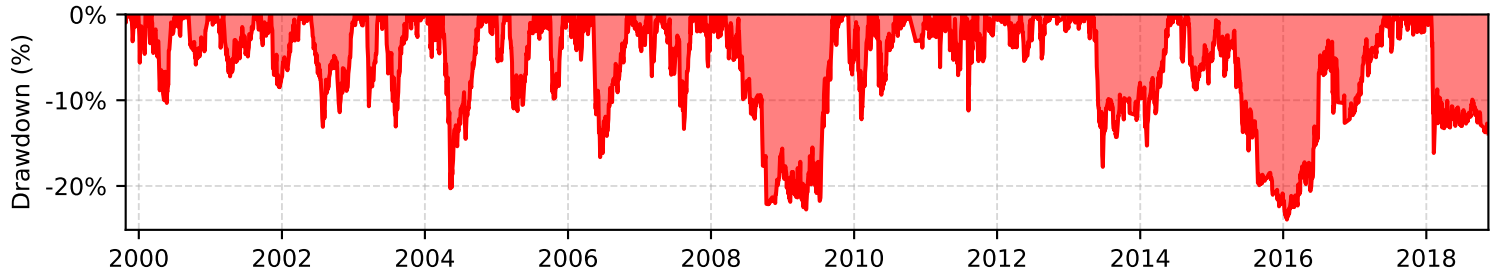
Yearly Returns (%)

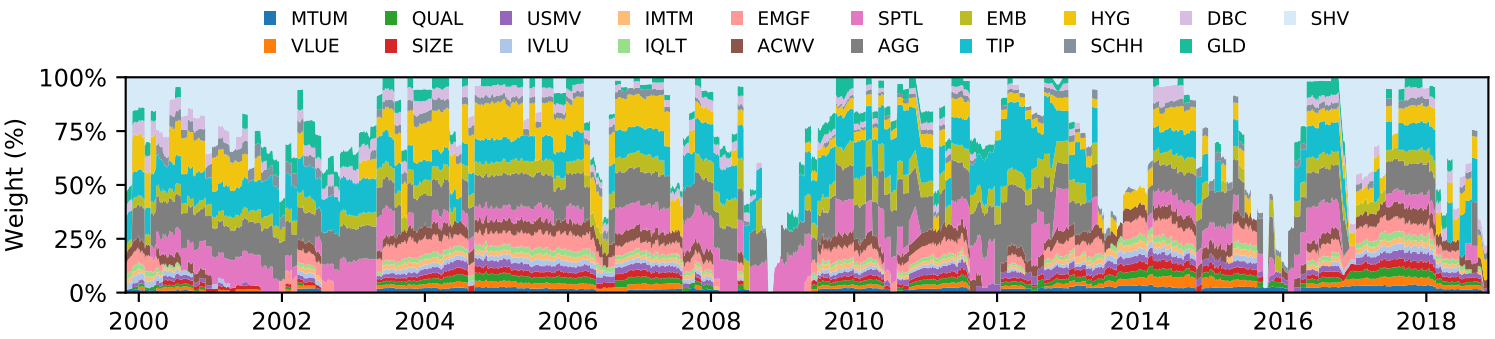


Tearsheet #30

Portfolio Tearsheet: Aggressive







Overview

Portfolio Code:	RP 3.5x
Start Date:	1999-10-26
End Date:	2018-11-14
Rebalanced:	Monthly
Trend Following:	200 SMA
Weighting:	RP 200

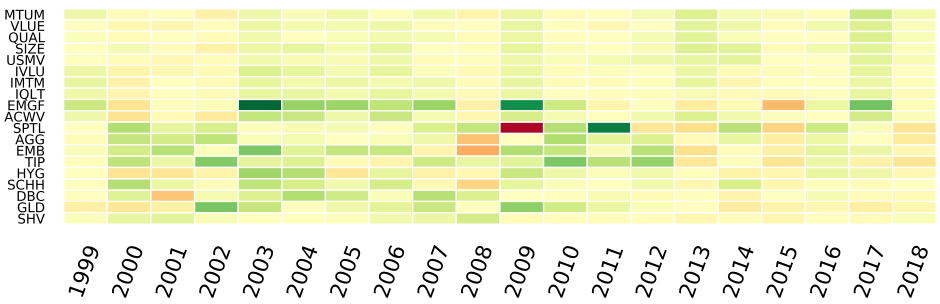
Statistics

Total Return:	1883.39%
CAGR:	19.25%
Annual Volatility:	13.80%
Sharpe:	1.29
Max Drawdown:	-23.92%
Sortino:	1.85

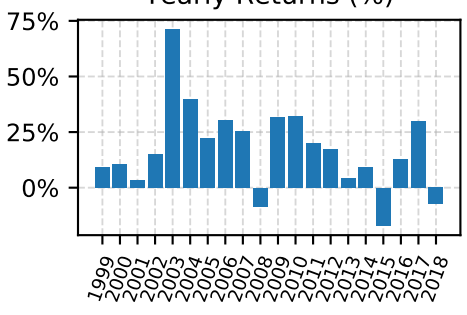
Statistics #2

$VaR_{99\%}$:	-2.42%
$CVaR_{99\%}$:	-3.11%
Beta:	0.20
Alpha:	17.26%
R-Squared:	7.06%
Treynor:	0.91

Performance Attribution



Yearly Returns (%)



Closing the World's Investment Gap