



Closing the World's Investment Gap

Initial Design Report

Team Members

Ameer Shaikh

Amr Mahmud

Daniel Kecman

Stefan Momic

Contents

Problem Statement.....	4
Project Scope	5
Survey of Existing Solutions	5
Survey of Existing Solutions – Deeper Dive	6
Project Framework.....	8
Stakeholder Analysis	8
Step 1: Identify all Stakeholders	9
Step 2: Prioritize Stakeholders.....	9
Step 3: Understand your Stakeholders.....	10
Functions	11
Objectives and Constraints.....	12
Objectives	12
Constraints.....	12
Design Decisions	13
Business Logic.....	13
Asset Universe Selection.....	13
Source of Financial Data	16
Portfolio Generation Strategy	17
Parameter Estimation.....	19
Considerations for Robustness	20
Portfolio Validation and Analytics	20
Front-End.....	21
User Interface and Wireframe Sketches.....	21
Gathering User Input(s) and Feedback.....	36
Display of Computed Data	37
Decision on Technologies Used	39
Back-End.....	42
Choice of Database / Data Source for Raw Data	42
Storage of Financial Data (Both Raw and Computed Data).....	42
Setup of Data Access Layer for Data (Both Raw and Computed Data)	43
Cleaning Data & Dealing with Missing Data	44

Hosting.....	45
Flow Chart and Decisions on Technologies Used	46
Database Selection	47
Analysis of Service Environment.....	48
Client Side (Front-End)	48
Server Side (Back-End)	48
Project Plan and Timeline	48
References	51

Problem Statement

Money matters – love it or hate it, money has become a 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 we do not need to convince you about the importance of money, as we can appreciate that everyone had some concerns about money at one point or another. Rather, we are here to help you make money, so you can live the life you want!

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 of us having jobs to satisfy our 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 you to further enhance your wealth as it provides you with 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 to invest your money include, but are not limited to the following:

1. Grow your money – return on your investments allows your money to build, creating additional wealth over time
2. Save for retirement – you may be working now, but we all know you cannot work forever. Saving money for retirement, and better yet, investing your retirement savings will allow you to live off these funds after you stop working
3. Earn higher returns – if you want to grow your money, you need to put it in a place that offers a higher rate of return than the interest rates provided by your savings account(s). With many investment vehicles (more on these later!) offering opportunities to earn higher rates of return, investing your money provides the opportunity to achieve these goals
4. Reach 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 your money will help you on that path

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 you invest in your future and make your money work for you!!

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. Luckily, we have decided to focus on you – the average member of the population!

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 you through our quick questionnaires, so we can create a personalized and diversified portfolio of assets at a risk tolerance you are comfortable with while meeting your long-term investment goals. From there, we manage your money for you – this includes automatically rebalancing your portfolio to account for any market fluctuations and reinvesting your investment proceeds (e.g. dividends) to ensure that your money doesn't take any breaks while working for you. We understand that investment goals change overtime and we offer dynamic portfolio modifications to address your evolving user preferences and inputs.

Survey of Existing Solutions

Since the invention of the Internet we have greatly changed the way we 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 investors 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 seemingly new firms 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 a robust and diverse portfolio without requiring massive capital involvement (in contrast, attaining a diverse 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 feel like the portfolio generated is truly unique to their needs and created with their input

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 those robo-advisors a success. More than this, Alpha Factory aims to achieve better returns than the aforementioned robo-advisors with careful consideration of their asset universe, parameter estimation and portfolio generation methodologies.

Survey of Existing Solutions – Deeper Dive

Prior to discussing Alpha Factory’s design and methodology we provide a deeper dive into each previously mentioned competitor. Alpha Factory ultimately used these competitors as case studies in creating their own design decisions. Note that this section can be skipped without much impact on subsequent sections.

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 approach¹, 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

¹ 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]

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 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 at \$80 regardless of 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.

Project Framework

To aid with making the appropriate design decisions, Alpha Factory has completed a deep exploration of their project framework. Being conscious of their stakeholders' desires, objectives and constraints enabled them to ensure the best robo-advisor was created.

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 will ultimately allow for 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 is 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 the 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 have significant influence and/or will be the end users of our product. Ultimately, determination of the key stakeholders frames the objectives and constraints of the design which in turn has a great influence on the design decision 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 will likely 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.

Lastly, 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. 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, the 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	<p>The main goal of new investors includes:</p> <ul style="list-style-type: none"> • 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	<p>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:</p> <ul style="list-style-type: none"> • 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	<p>Ensure that the tool developed integrates both the mathematical, statistical and financial modelling techniques learned throughout the EMSF major along with the relevant computing technologies.</p>

Key Stakeholder	Goals and Objectives
Design Team	<p>The Design Team aims to:</p> <ul style="list-style-type: none"> • 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 backtesting • 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	<p>Ensure key Engineering Accreditation requirements are met. Namely:</p> <ul style="list-style-type: none"> • 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 stakeholder Alpha Factory was developed to have several key functions. First, we wanted to ensure that we had an easy user-friendly interface that offered the 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 understand that financial goals change overtime and we wanted to ensure that users can repeat our questionnaires and/or change their views accordingly. Ultimately, this ensures 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 the questionnaires to create customized portfolios just as unique as our customers. 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 (and any others they might want to input) over various horizons by backtesting over real data. Moreover, we provided portfolio analysis by computing and displaying various portfolio performance metrics such as alpha and Sharpe Ratio and given any portfolio we can find a portfolio that dominates the one provided considering a specific performance measure.

Objectives and Constraints

Having identified the goals of the key stakeholders the objectives and constraints were easily identified. We have provided a 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 the users.
4. User Preference: comprehensive portfolio features – Alpha Factory was constrained with the need to consider user inputs and create customized portfolios for each of their user's unique financial situation and goals
5. Time – with the project deadline less than 3 months away the Design Team experienced significant time constraints, limiting exploration/implementation of more complex financial models

Design Decisions

Several design decisions were made while developing Alpha Factory. 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 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 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 way to go.

Option	Pros	Cons
Individual Securities	<ul style="list-style-type: none">Provides ultimate flexibility in allocation decisions (i.e. allows you to choose exactly what you want)	<ul style="list-style-type: none">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 universeIncreased portfolio generation complexity including frequent rebalancing and other execution/logistical issues

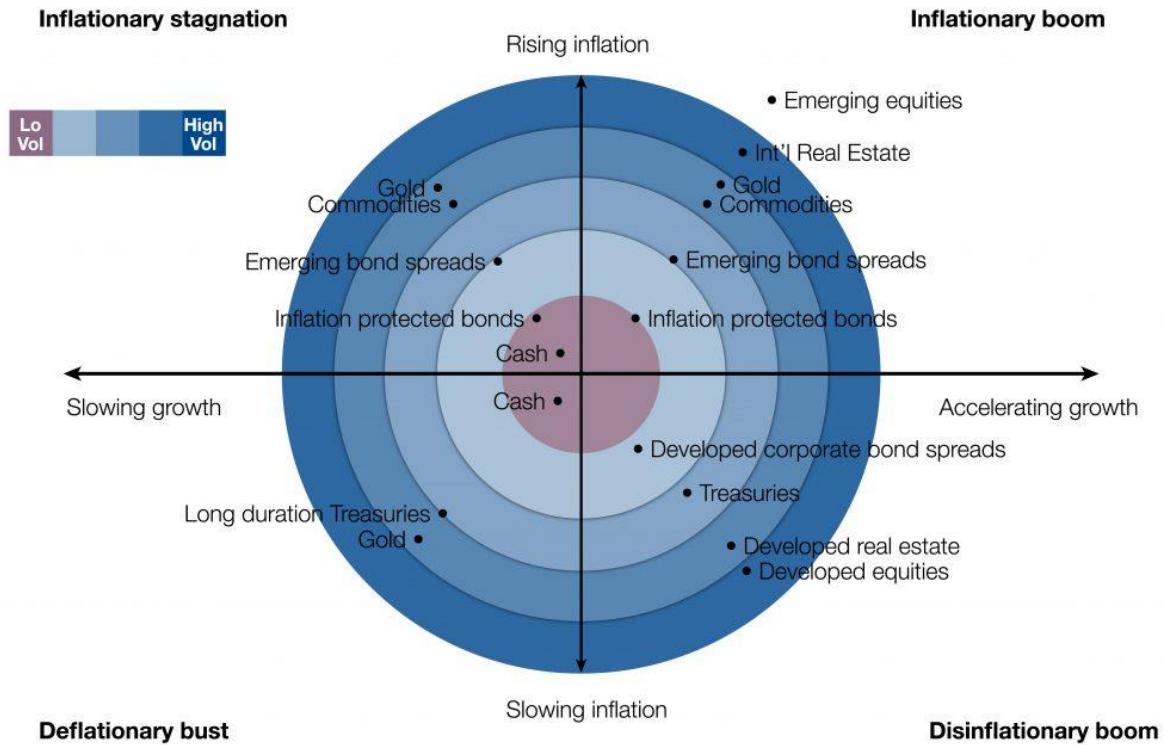
Option	Pros	Cons
Aggregate Asset Classes	<ul style="list-style-type: none"> • Less internal rebalancing and ability to outsource costs to aggregate asset providers • Easier to obtain exposure to a wider array of asset classes • Guaranteed diversification (main problem with most individual security portfolio 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) 	<ul style="list-style-type: none"> • 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 strives 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 diversified portfolio 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.

Asset Class Behaviours in Different Inflation and Growth Environments



Source: ReSolve Asset Management

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 guarantees that portfolio construction does 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 specific ETF selection is still a work in progress and will require several iterations and significant backtesting. However, the ETF selection process will be 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

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] Alpha Factory has not determined the exact number of ETFs to use in their asset universe but is likely going to use anywhere from 8 to 15 ETFs (estimated by considering the asset universe size of the existing market solutions).

Beyond pure asset class exposure, Alpha Factory considered variations in the form of factor tilted exposures. Factor investing styles have been proven to produce excess risk-adjusted returns and therefore are worth incorporating into our portfolios. [28]. As an example, this is done by considering exposure in asset groups such as US Equity Value and US Equity Momentum instead of seeking US equity exposure. 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.

Source of Financial Data

In selecting the source of financial data, Alpha Factory considered using Bloomberg Terminal or Yahoo Finance. After careful consideration of the pros and cons of each source (refer to Table 5 below), we determined that getting financial data from Bloomberg Terminal was the best option.

Option	Pros	Cons
Bloomberg Terminal	<ul style="list-style-type: none"> Accurate historical prices for both ETFs and indices 	<ul style="list-style-type: none"> Inability to fetch data in real time (would require a manual process to upload financial data to our database)
Yahoo Finance	<ul style="list-style-type: none"> Ability to fetch data in real time using API enabled data sources 	<ul style="list-style-type: none"> Hard to access historical prices of indices

Table 5: Pros and Cons of Financial Data Sources

With Bloomberg Terminal lacking real time data fetching abilities and Yahoo Finance lacking access to index 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 creating parameter estimates. 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.

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 examine different portfolio generation techniques based on their properties agnostic of individual investor preferences as any technique can be tailored to produce portfolios for various risk tolerances with tactical uses of leverage and additional constraints. 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).

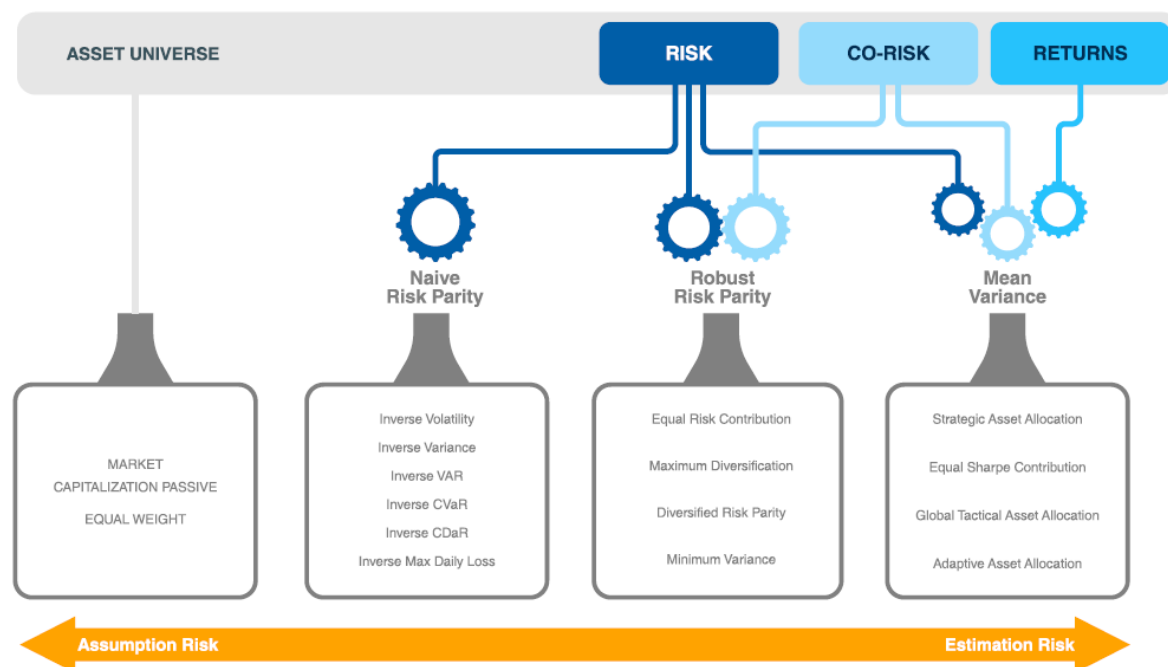
Portfolio generation techniques can be broadly classified based on the assumptions and estimates made on asset characteristics. In some strategies, parameters are not estimated (not direct inputs) yet in these cases it implies strong assumptions on relevant parameters and their relationships, or lack 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 unconvincing. 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.

Another approach which is popular among practitioners is the Black-Litterman model. This model uses an investor's 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 that this relationship can be estimated with better than random precision, this leads us to abandon 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 is our portfolio generation strategy of choice.

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 intimately 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 up² and rotating to cash when trending down³. This results in a portfolio that normally contains portions of cash. We will explore potential improvement 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.

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

² Trending up is quantified by an asset price above its 200 Day SMA

³ Trending down is quantified by an asset's price below its 200 Day SMA

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

To address any uncertainty in our parameter estimations, multiple computations and comparisons will be made. By analyzing how varying parameter estimation factors like time period and lookback period length affect final results we can identify whether results are simply driven by data mining/overfitting of the parameters. In terms of robustness in our asset universes, multiple iterations will be tested to ensure selection of our universe is not the main driver of performance. In all cases, stability of results across small parameter variations will be monitored for indication of adequate robustness as significant volatility in results with small variations would indicate otherwise.

Portfolio Validation and Analytics

Alpha Factory will validate it’s generated portfolios by comparing it to several other portfolios. First, we will compare 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 will compare our portfolios to standard asset allocation mixes, such as the popular 60/40 and 70/30 equity/fixed income asset allocations. Finally, we will compare 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.

To further validate the generated portfolios, Alpha Factory will conduct 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, so we will test our portfolios by backtesting them over different periods (e.g. inflationary booms, inflationary stagations,

financial crises, etc.) and compare their performance over those periods with the performance of the aforementioned portfolios during the same timeframe.

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

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.

User Interface and Wireframe Sketches

The user interface will be designed with simplicity and user friendliness as a top priority. With that said, the design decisions and flows of the wireframes will reflect 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) will be taken as the colour scheme/theme throughout the website. Figure 3 below demonstrates how a user will traverse through our website.

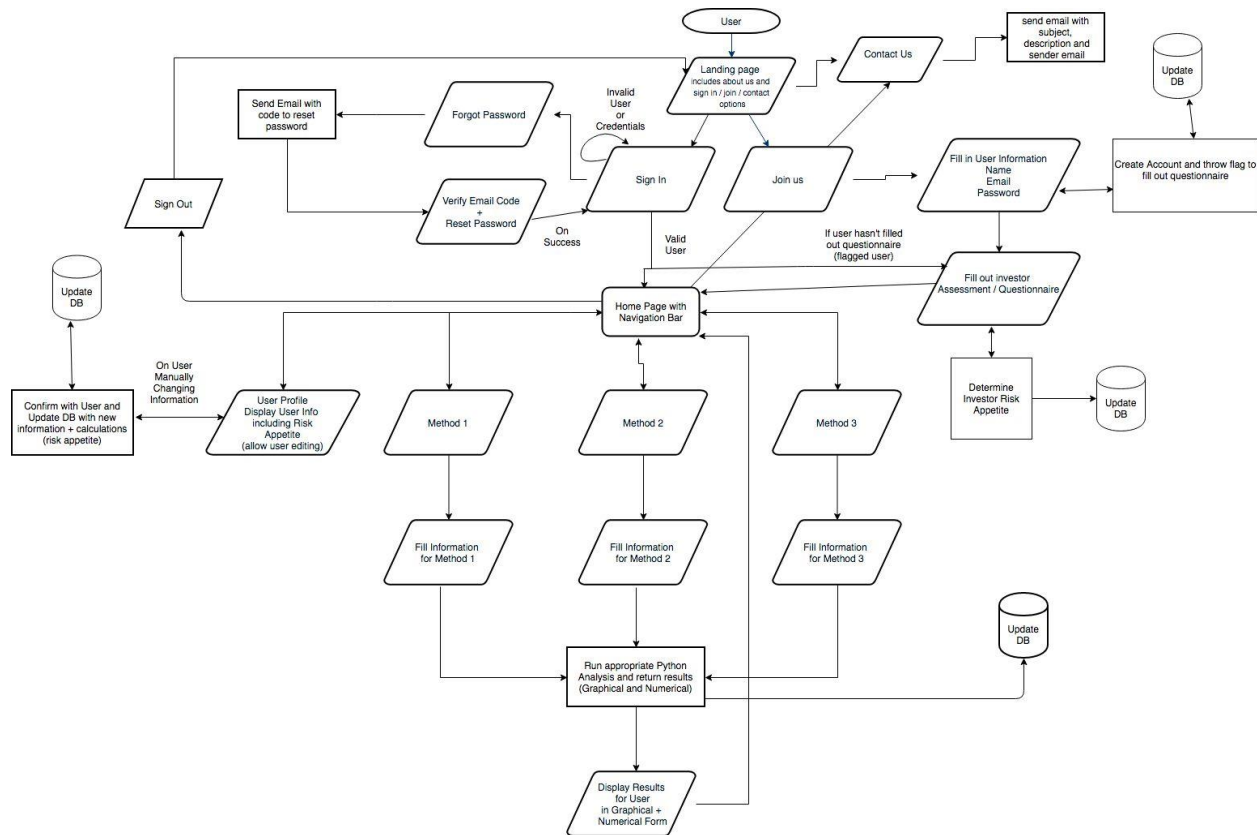
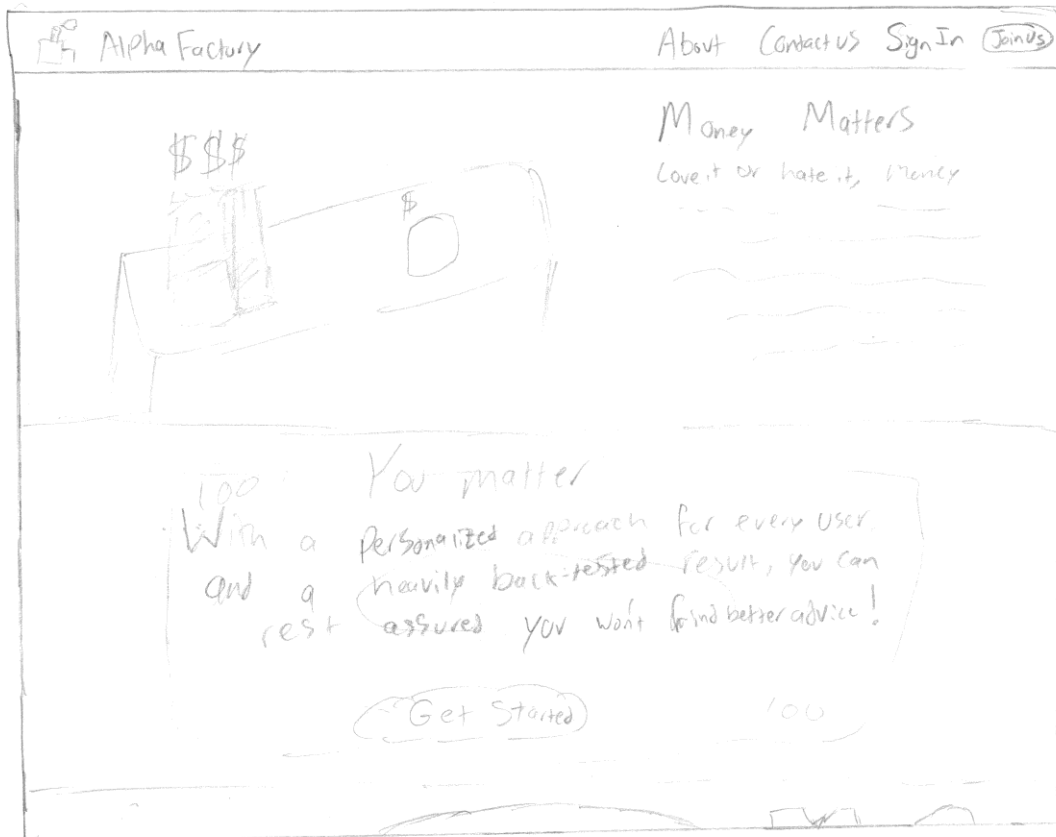


Figure 3: User Flow

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 Page [About]



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

Alpha Factory

About Contact Us Sign In Join us

Contact us

Please fill out the form below
And we will be in touch
with you shortly

Email

Subject

Description

Submit

+1 888 123 4567

210 Baren Street
Toronto, ON, M5T 1T8

Alpha.factory@gmail.ca

↳ After submit, Display message:
"Thanks for contacting us, we will review your
message and be in touch with you shortly."

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

Alpha Factory

About Contact Us Sign In [Join Us](#)

Sign in to Continue

Email

password

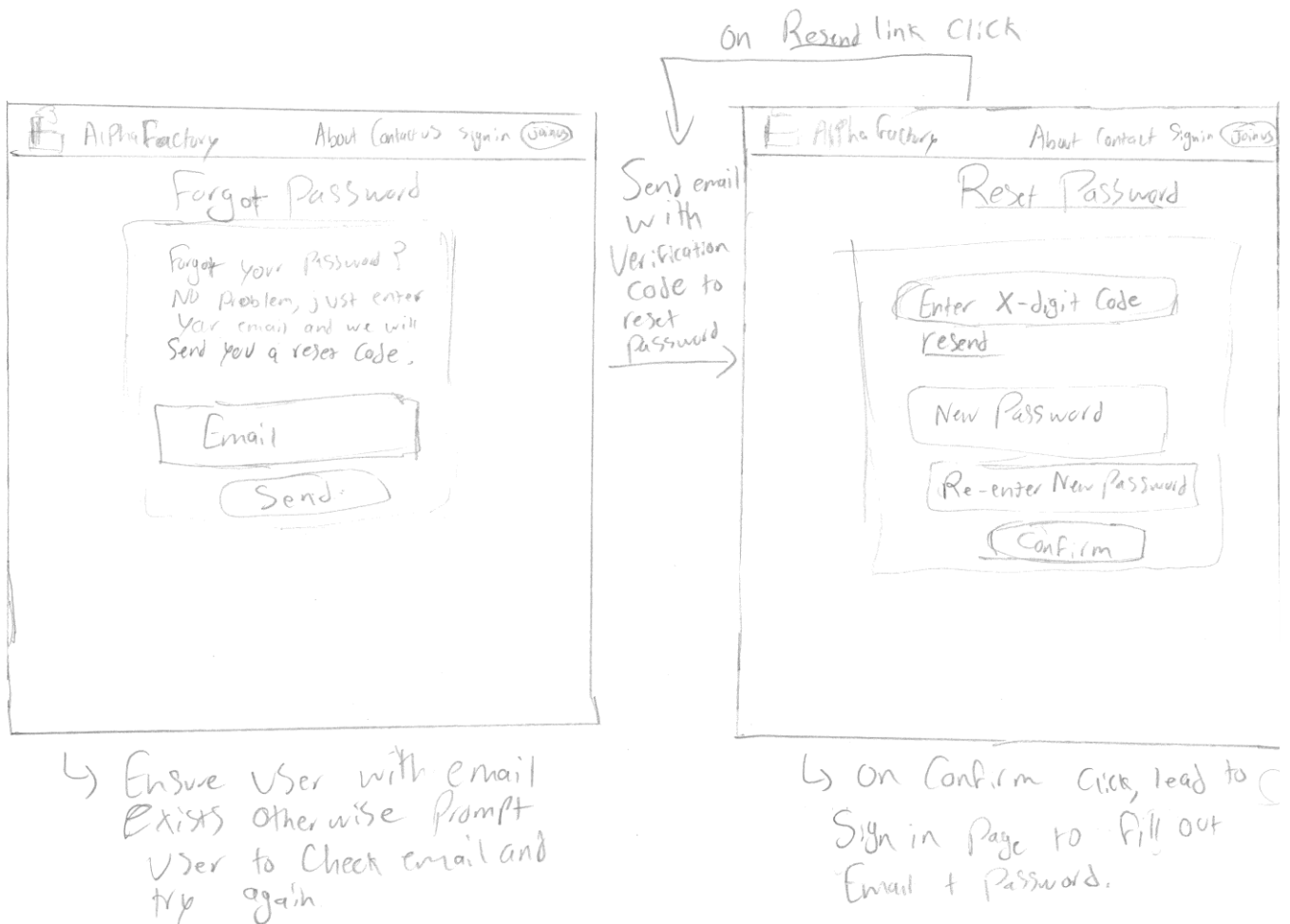
[forgot password](#)

Sign in

[Not a member? Join us.](#)

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 have forgot it.

Forgot Password/Reset Password



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

Validation

↓

First + Last Name
> 2 characters
↓

Email
> 6 Characters
includes "@"
and "."

Alpha Factory
About Contact Us Sign in Join Us

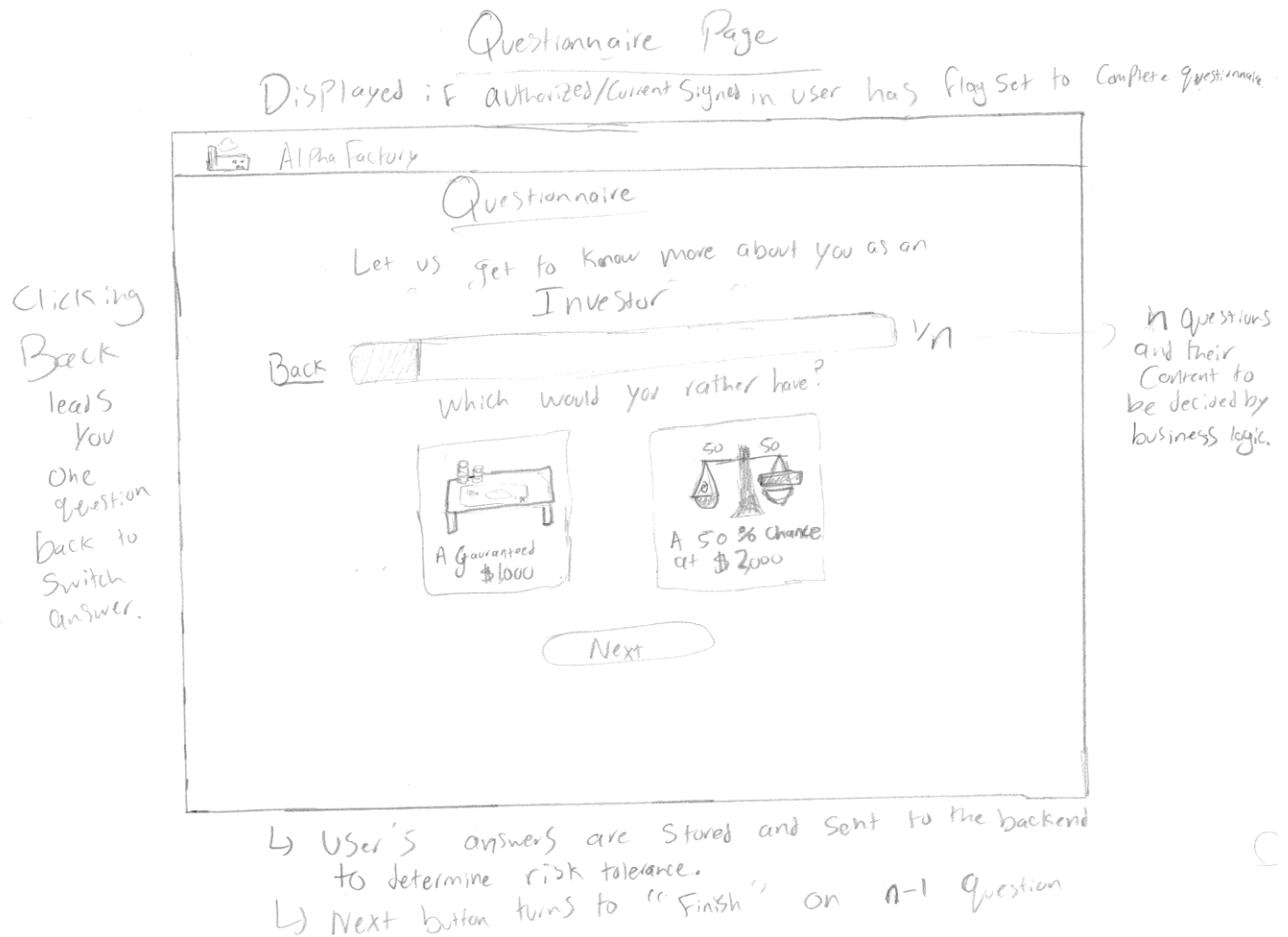
We're glad to have you aboard!

Let's get you set up.

↳ Password Validation [min. 8 digits and requires at least 1 number]

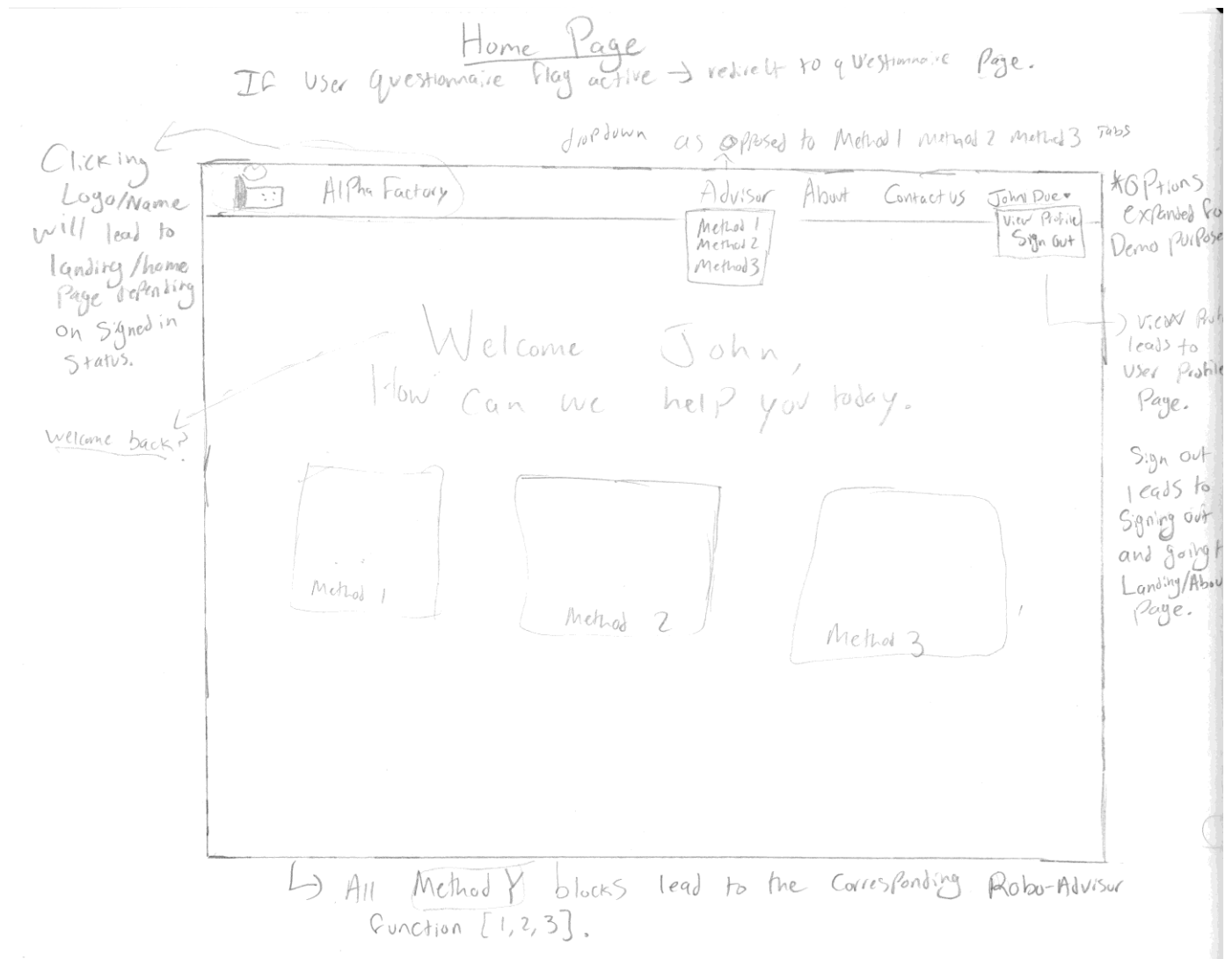
↳ on Sign up click, create Account in backend [PlusDB] and throw user flag to fill out Questionnaire

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 back-end where a user object will be instantiated and updated in the database with a flag thrown requiring the user to fill out the questionnaire.



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 will decide which questions are required and will send it to the front-end to populate the two options for every question. A progress bar will track and display the users' progress throughout the questionnaire and will allow them to go back to change previous answers as required. On the final question, the next button will turn into a finish button. The answers will be recorded and sent to the back-end 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



The user can access the 3 different functions of the robo-advisor or they can navigate to view their user profile, all from the navigation bar or home page screen. The home page will be extremely simplistic with only the navigation bar and 3 square elements on the page that will lead them to the different functions of the robo-advisor. Currently, the 3 functions of the advisor (as laid out by the Capstone Project requirements) have been respectively named Method 1, Method 2, and Method 3. Since this is not intuitive and gives no information on their function, they will be renamed by the Business Logic team members and appropriately implemented in the prototype and final product.

Navigation Bar

The navigation bar will be a sticky header element that will be 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.


User Profile

User Profile

• ON Change of Profile info [with Confirmation] update DB through back end.

Alpha Factory

Advisor About Contact Us John Doe




John Doe

J. Doe@gmail.com

Risk Tolerance: High

Date of Birth: 12/24/1990

Change Password



History

Action	Date	Result
Portfolio Produced	Jan. 21, 2018	109% return with 2% Var
Portfolio Checked	Feb 20, 2017	-8% return, 30% risk

Are You Sure?

Manually Changing RT can affect your returns.

Cancel Confirm

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. The history is currently an experimental idea and will be implemented

only if the development of it doesn't require excessive time within the context of the project deadline. The user must be able to edit the information (aside from name) upon clicking an edit button or pressing change password. The risk appetite can also be manually changed here but with a warning. The resulting changes will be recorded and sent to the back-end 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

Method 1 [To be renamed] (Business Logic)

Alpha Factory

Advisor About Contact US John Doe

Portfolio Risk and Returns [Method 1]

Enter the assets and weights of the Portfolio you wish to analyze

Assets

Stocks

Bonds

ETF

+

weight

Stocks 20 Bonds 40 ETF 20

Percentages ☒

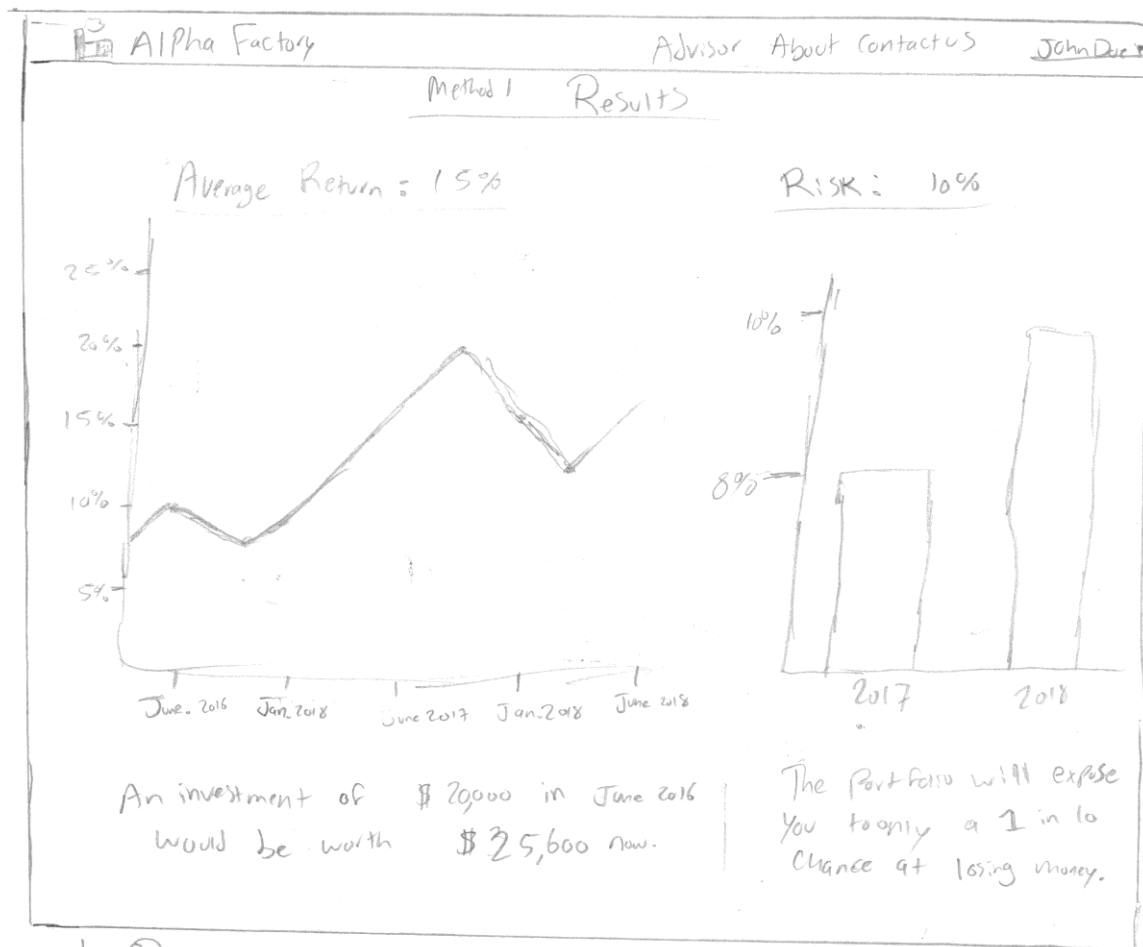
Analyze

⊕ Adds more Assets if BusinessLogic want's to add this functionality

↳ Clicking Analyze sends info to back-end to be run with appropriate code and update DB.

Upon selecting Method 1, the user will be led to a page where they will have to fill out the information required to run Method 1 through the back-end [to be decided exactly by the Business Logic during later design iterations]. 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 backtested and stored in the database for future reference.

Method 1 results



- ↳ Data and Graphs to be decided by business logic.
- ↳ Might change to display a more informative + understandable measure of risk.

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

Method 2 [to be renamed]

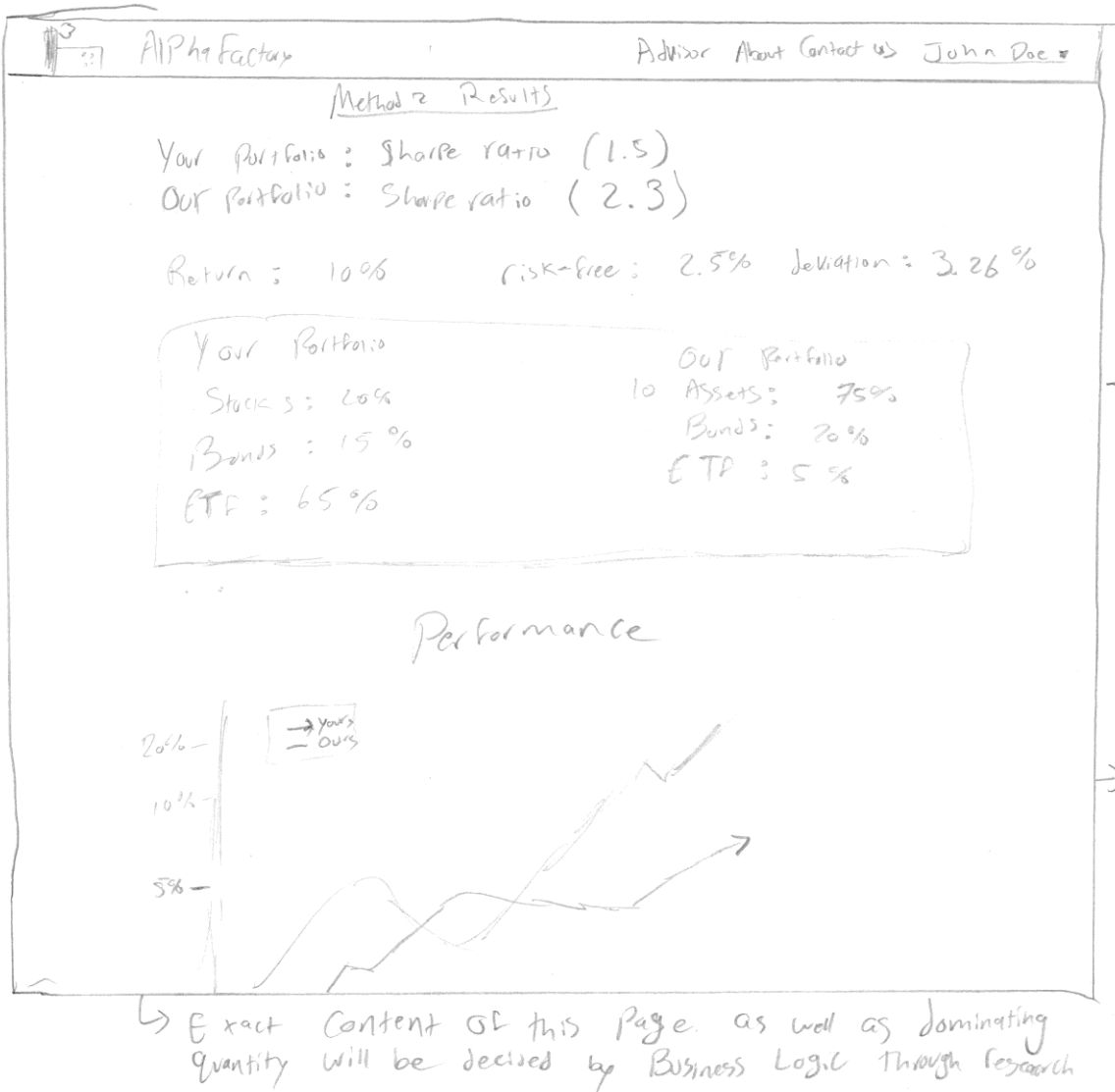
The wireframe shows a web page with a header bar containing 'Alpha Factory' on the left and 'Advisor About Contact Us John Doe' on the right. Below the header, the main content area has the title 'Dominating Portfolio' and the slogan 'We always want better for you!'. A central box titled 'Select Portfolio' contains a 'history:' section with two portfolio cards: 'Portfolio 1 Jun. 2018' and 'Portfolio 2 July 2018', each with a close 'X' button. To the right of these cards is a plus sign button. Below the history section is a label 'Set number of assets:' followed by a text input field containing the number '10'. At the bottom of the central box is a large 'Search' button.

(+) Logic to be decided by Business Logic

↳ Search will send the information to backend to run appropriate code and update DB.

Upon selecting Method 2, the user will be led to a page where they will have to fill out the information required to run Method 2 through the back-end. 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 backtested and stored in the database for future reference.

Method 2 Result



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.

Method 3 [Portfolio Generation]

Alpha Factory Advisor About Contact us John Doe

Portfolio Generation

Tell us what your goals are and we will find a Portfolio to help you achieve it!

Initial investment:

Financial Goal:

When would you like to achieve this by.

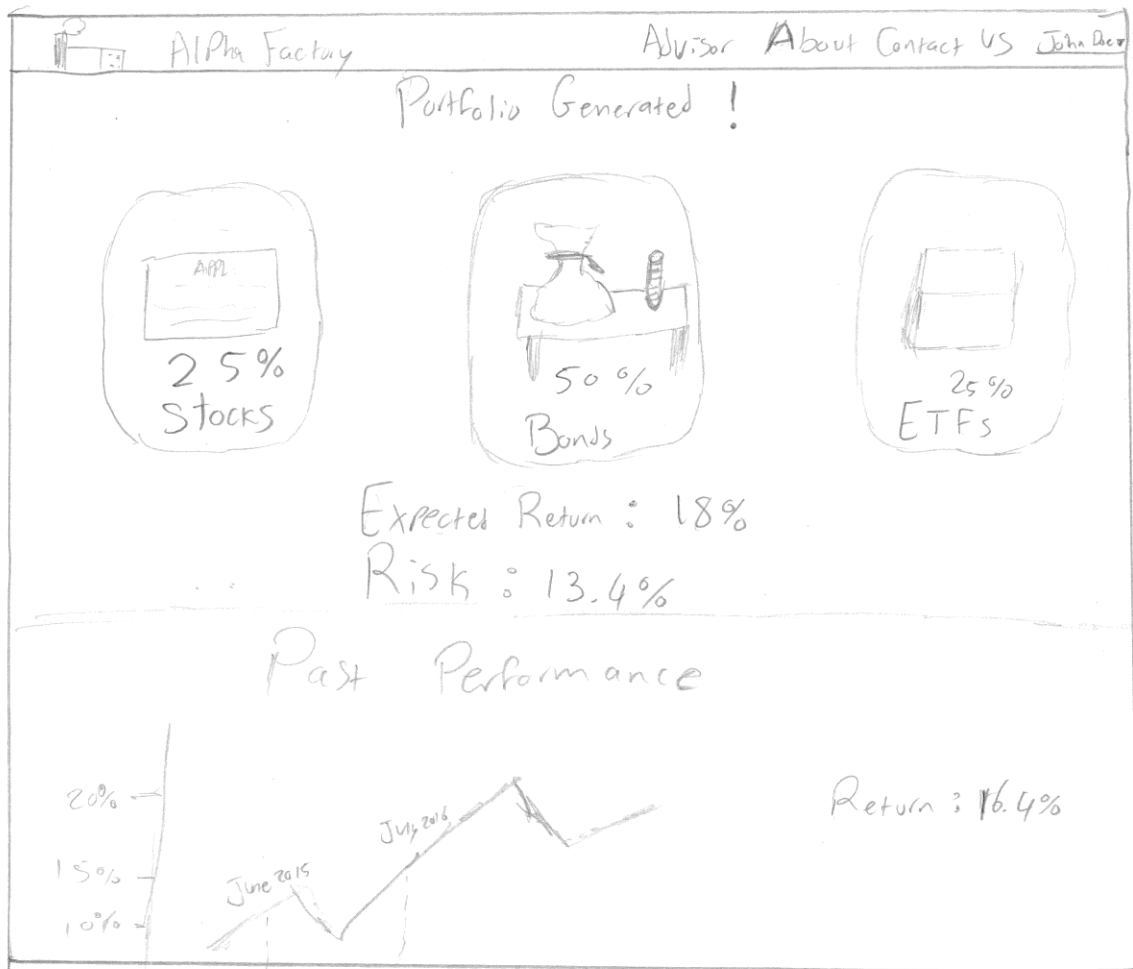
5 Years

→ Questions might be changed according to Business Logic.

- ↳ Clicking Generate calls back-end Python Code with user info and form info.
- ↳ returns results and updates DB.

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 back-end. 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 backtested and stored in the database for future reference.

Method 3 Results



Option to
run
Method 1/2
from this
Page
to be
considered.

↳ Exact results to be determined by Business logic and back end.

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.

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 back-end to set pre-determined methods for the different time horizons so that it can better support them instead of having a back-end 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 , to be 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 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 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 drawbacks. Table 6 below displays some of the perks and drawbacks of each of the three options.

Library	Pros	Cons
Bokeh	<ul style="list-style-type: none"> • Wider range of interactions than Dash • Large number of examples on library usage and integration with Flask 	<ul style="list-style-type: none"> • More difficult to implement than Dash • Learning curve can be steep with little documentation and implementation
Plotly.js/Dash	<ul style="list-style-type: none"> • 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 	<ul style="list-style-type: none"> • 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	<ul style="list-style-type: none"> • 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 	<ul style="list-style-type: none"> • Limited graphs available to display • Graph customization is minimal

Table 6: 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. [43] 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	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• Very simple and minimalistic• Beautiful cascade of colours and other options for the elements	<ul style="list-style-type: none">• Less comprehensive than Bootstrap or Foundation• No Icons available• Licensed by Yahoo
Foundation	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• Learning curve is steeper than Bootstrap without prior experience in CSS

The decision on which front-end 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 Front-End development Technologies

Library	Pros	Cons
Angularjs	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• Usually used for SPAs (Single Page Applications)• Client-side rendering• Steep learning curve with no prior experience
Jinja2	<ul style="list-style-type: none">• Default templating framework in use with Flask• Simple, easy to use• Allows for reusable elements (navbar, pages)• Allows for server-side rendering	<ul style="list-style-type: none">• Limited integration with other front-end technologies like Angular• Harder to create a seamless user experience using only templates
React.js	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• Integration with flask can be complex at times• Newer framework with not as much support/documentation for bugs as Angular

The hardest choice to make came from deciding what type of front-end 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.” [44] 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 front-end 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 front-end (client side) and Flask would have to be used as a RESTful API which would also need the appropriate

coding, or “wiring”, for the back-end (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 “Test Folder”.

My string: Wheeeee!

Value from the list: 3

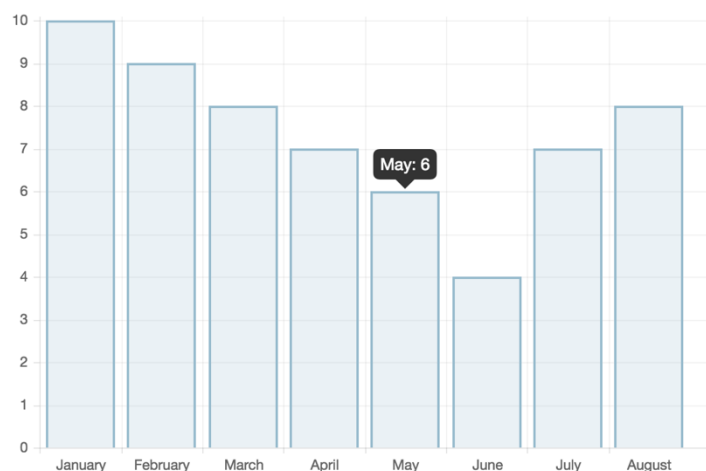
Loop through the list:

- 0
- 1
- 2
- 3
- 4
- 5

Sample Button

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.

Flask Chart.js



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.

Back-End

Choice of Database / Data Source for Raw Data

To store its various forms of data, Alpha Factory uses 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 are 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: In the absence of SQL, 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 RDBMS will be discussed more in 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 DBs. [45] The specific choice of MongoDB over other NoSQL DBs was done in an extremely similar manner to the method mentioned above and a comparable table between other NoSQL DBs was thoroughly examined to conclude using MongoDB. [46]

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 dataset is expected to be similar, and in addition to similar formats, with each asset being its own 'object', any interactions between the assets can be done seamlessly, like finding different ways to track the relation of different assets over various different horizons for the purpose of better predicting the movement of our asset universe as a whole.

In addition to the raw data, the computed data related to each asset needs to be stored on our databases as well, such as parameters for returns, volatility and correlation, as well as the data pertaining to these assets in the various portfolios that get created through Alpha Factory's business logic and processes. Once again, the choice to use an object-oriented database, and MongoDB in particular, allows a very straightforward organization for this data. The plan for how these objects are 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. So, within the overlying object for each asset, the raw data will be contained in another object type that is then embedded in its asset's structure. Lastly, as the calculated values are also a part of each asset's object, they are also included, but in such a way that the data and computed values are two separate objects that have a chance to interact with one another. Figure 4 shows how this might look within MongoDB's structure, and this practice of nested objects and collections pertains to most of the data that will be found in the database:

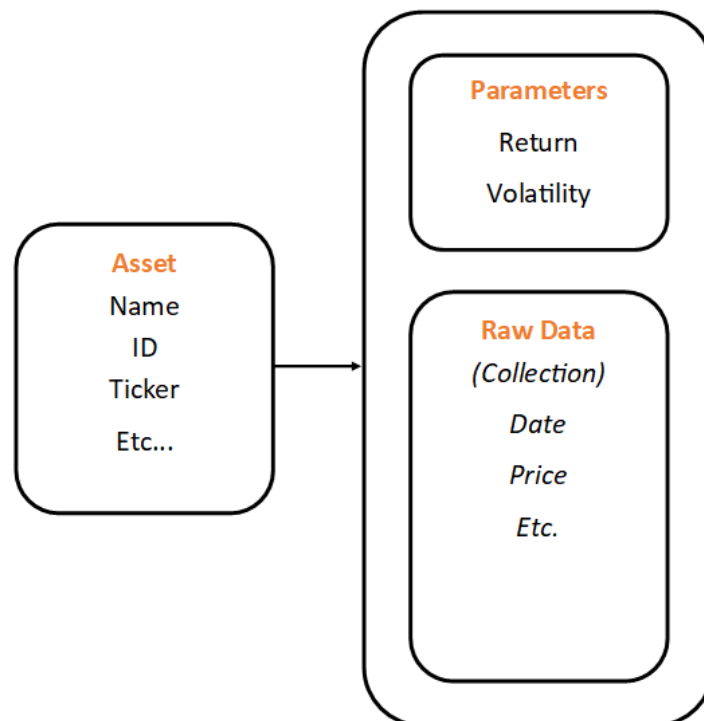


Figure 4: Layout of an Asset Object with a MongoDB database

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:** If there is missing data for a certain asset, there are two ways that it can be addressed. If the time horizon for which the data is missing is small enough, then the whole data point can be given a constant value that can be read as a valid data value, but which tells the user that the point is missing so that it will be exempt from calculations. For example, if the price for a

certain asset couldn't be found or brought into the database for the span of a week, then the values for that week would be given a valid date, but the price would be given a value of NULL or N/A, and checks within the rest of the code would be made to find and omit any of these values. Missing data over a select range of data can also be proxied by taking the average values between the two dates on either side. Furthermore, missing data for our ETFs can also be proxied using the underlying index, especially if the selected ETF closely tracks it. Dealing with missing data will ultimately be selected on a case-by-case basis from one of the following options.

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 current solution in place would be a manual investigation and fix 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. In the case of Alpha Factory, we chose to support the application with Atlas, using the Google Cloud Platform as the actual host for the database. With the first 512 MB free through Atlas, we decided that using Atlas' services, which includes all the benefits listed in Table 7 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, 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 7: Benefits of hosting a MongoDB database on Atlas

Flow Chart and Decisions on Technologies Used

The following are flowcharts that go into the processes that are entered into when making interaction with the database 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.

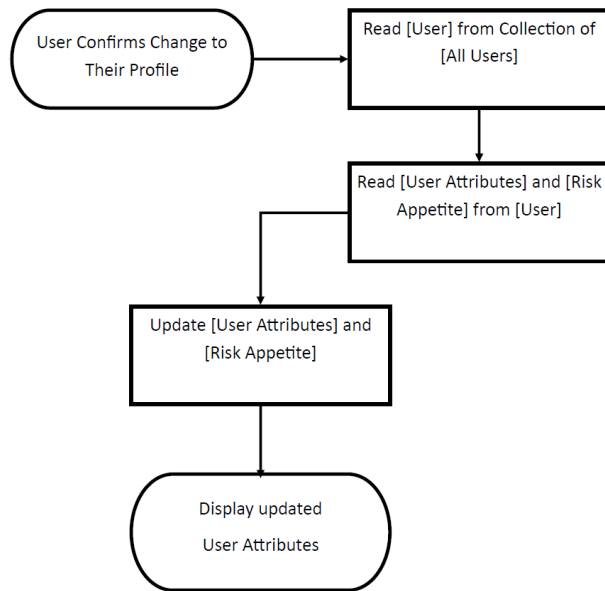


Figure 4: Flowchart for Changing User Info

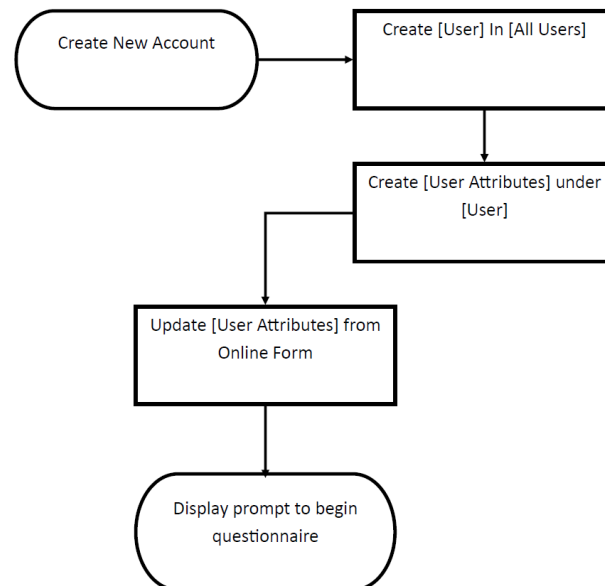


Figure 6: Flowchart for Creating a New User Account

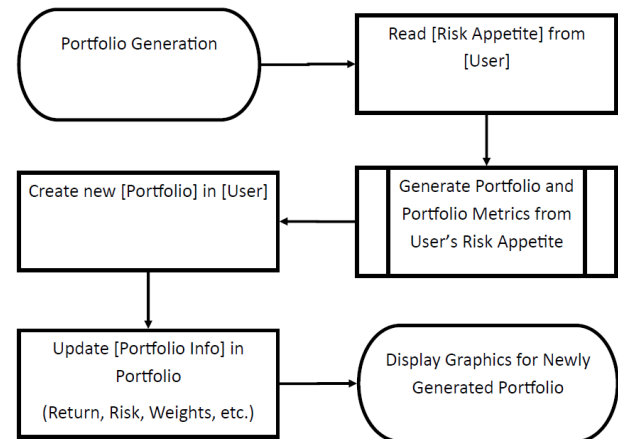


Figure 5: Flowchart for Portfolio Generation

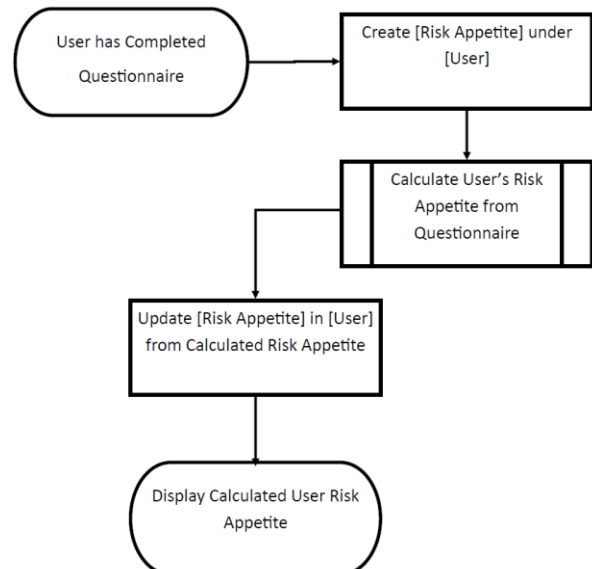


Figure 7: Flowchart for Completed Questionnaire

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 8 below:

Advantage	Explanation
Wider variety of allowed data types	<ul style="list-style-type: none">• 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
More coherent modelling capability	<ul style="list-style-type: none">• 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	<ul style="list-style-type: none">• 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 8: Advantages of an Object-Oriented Database Management Systems

Analysis of Service Environment

The service environment must be a holistic environment that can support all the technologies/packages selected for the front and back-end. Since we will be using Anaconda as the package manager, we will use the installation procedures and process related to the conda package manager. First things first, the top priority is to ensure that the Python version is consistent across all developer environments. The Python version that has been selected is the Python version that comes with the installation of Anaconda version 4.5.11 which is Python 3.6.5. The service environment will be the default root/base environment and the required packages can be broken up by the side they are servicing.

Client Side (Front-End)

For the front-end, we will require the packages for Jinja2, Bootstrap, Flask and Python. Since we already have python installed, we simply had to select to install the remaining packages and their versions which can be seen in the table below.

Package	Version	Installation
Jinja2	2.10	Comes with Anaconda installation
Bootstrap	4.1.3	conda install -c conda-forge bootstrap
Flask	1.02	Comes with Anaconda installation

Note: To install Bootstrap, use of conda-forge is required.

Server Side (Back-End)

For the back-end, we will be using MongoDB and we will therefore require the associated DB package as well as PyMongo to best serve our Mongo querying/inserting needs through Python.

Package	Version	Installation
Mongodb	4.0.2	conda install -c anaconda mongodb
PyMongo	3.7.1	Comes with Anaconda installation

The MongoDB daemon can then be started by typing “mongod” (if through brew) into the terminal and the db will be up and running for usage with Python.

Project Plan and Timeline

The official launch and presentation of Alpha Factory will occur on December 3rd, 2018 from 3-6pm in room 440 of the Myhal Centre for Engineering Innovation & Entrepreneurship at the University of

Toronto. The Final Report and documentation of Alpha Factory will be released shortly after, but no later than 5pm on December 5th, 2018.

To ensure timely delivery, Alpha Factory has imposed an internal deadline of November 29th, 2018. Currently, they aim to have a draft of the Final Report and complete robo-advisor platform by that time. To meet these deadlines, the team has laid out the following timeline with the corresponding milestones.



Figure 8: Project Timeline

Additional notes regarding select milestones from Figure 8 above are provided in Table 9 below.

Item	Detailed Description
Oct 15: Business Logic complete	<p>A completed Business Logic constitutes:</p> <ul style="list-style-type: none"> • Completion of asset selection and financial data location • Completion of portfolio generation algorithm • Completion of parameter estimation • Complete justification for all design decisions made including asset selection, portfolio generation and parameter estimation • Thorough understanding of all models and methodology used completed as part of Literature Review

Item	Detailed Description
Oct 27: Front and Back-End complete	<p>A completed Front-End constitutes:</p> <ul style="list-style-type: none"> Ability to gather user inputs and feedback Ability to hook up to Business Logic and display computed data (e.g. graphs, tables, numbers, etc.) Complete user-friendly design and user interface <p>A completed Back-End constitutes</p> <ul style="list-style-type: none"> • Ability to clean data and/or deal with missing data • Database selection and initialization including the ability to store raw and computed data from the financial data selected and generated by Business Logic
Nov 17: Integration between Business Logic and Front-End / Back-End	<p>With the subsystems integrated, completion of this stage implies the ability to:</p> <ul style="list-style-type: none"> • Compute and present the final outcome (e.g. optimal portfolio given the user inputs) • Perform all the required analytics (e.g. backtesting, portfolio performance metric calculations, etc.) • Validate portfolios generated by the system

Table 9: Detailed descriptions for select milestones

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_ZjRbvT9Q5mh1wflPbcaAs6WEALw_wcB&gclsrc=aw.ds&dclid=ClzX0d7D790CFcLZwAodXdwGHw.
- [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 Weather 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%2BBIQ&kw=questrade%2Bportfolio%2Biq&gclid=Cj0KCQjwgOzdBRDIARIsAJ6_HNm_d72jElqqe10vUVIj7lQ-Y5B_HBkvok7TEJ05Fzlz5csy9iHjOd0aAoQAEALw_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>