

Mutual Fund Calculator

A web application designed to help users estimate potential returns on their mutual fund investments.

- Users will be able to select a mutual fund, input an initial investment amount, and input the investment duration to calculate the future value of their investment.

Introduction:

This project will allow the student to learn about mutual fund (MF) trading, model investments, and predict returns over time. This project is intended for college students at a sophomore level of their computer science (or equivalent) degree. On completion, the student should have a strong foundation in full stack engineering, as well as enhanced financial literacy in the domain of mutual fund trading.

Terminology:

- **Mutual Fund:** an investment vehicle that pools money from multiple investors to purchase a diversified portfolio of stocks, bonds, or other securities (according to the fund's stated strategy)
 - Example: ClearBridge Large Cap Growth Fund
- **Beta:** denotes volatility or systematic risk of a security or portfolio compared to the market.
 - Assuming the S&P 500 is the "market" and has a beta of 1, stocks higher than 1 are interpreted as more volatile than the "market".
- **Return Rate:** net gain or loss of an investment over a specified period, expressed as a percentage of the investment's initial cost.

Definitions from <https://www.investopedia.com/>

Mutual Fund Predicted Future Performance Formula:

Use the S&P historical return over the past 5 years, the selected MF's historical beta, and the Capital Asset Pricing Model to predict the future value of an investment.

$$FV = P * e^{rt}$$

- P = principal (initial investment)
- r = rate = risk free rate + beta * (expected return rate – risk free rate)
- t = time

Risk Free Rate → US Treasury Interest Rate

<https://fred.stlouisfed.org/series/DGS10>

Expected Return Rate → Use historical avg. returns from chosen mutual fund for previous year

$$\frac{\text{last day of year value} - \text{first day of year value}}{\text{first day of year value}}$$

- Get average rate for previous year:

Beta → get calculated beta from an open-source API (ex. Newton analytics)

- Example: <https://api.newtonanalytics.com/stock-beta/?ticker=VFIAX&index=^GSPC&interval=1mo%E2%80%8B&observations=12>
 - Ticker is mutual fund symbol
 - Index is S&P 500
 - Keep index, interval, and observations the same as example

Structure:

- Build the services layer of the backend that expose RESTful APIs for the frontend.
 - GET list of mutual funds

- Hardcode mutual funds – can google any mutual funds you want to add (make sure they are exposed in the newton API as you will need the beta for the mutual fund to calculate the future investment)
- Some examples: <https://www.marketwatch.com/tools/top-25-mutual-funds>
- GET future value of investment amount
 - Take in the mutual fund, initial investment amount, and time as parameter
 - Hardcode in risk free rate
 - Connect to APIs that
 - Expose beta for a given mutual fund: <https://www.newtonanalytics.com/docs/api/stockbeta.php>
 - Calculate expected return rate from last year of mutual fund data
- Build the frontend & connect to backend endpoints to retrieve data & populate the UI
 - Dropdown component to select an MF to invest in
 - Input component to write the initial investment amount
 - Input component to choose the time horizon
- Create a presentation showcasing the finished product and learning milestones

Timeline:

- Week 1: Project set up
- Week 2: Backend API Development
- Week 3: UI
- Week 4: Wrap up, add bonus features, prepare presentation.

Prerequisites

- Install:
 - IntelliJ IDEA: <https://www.jetbrains.com/idea/download/?section=windows>
 - Git: <https://www.jetbrains.com/help/idea/set-up-a-git-repository.html>
 - Angular CLI & Node.js: <https://angular.dev/tutorials/first-app>
 - Java8: <https://www.java.com/en/download/>

Possible Bonus Features:

- Junit Test on backend
- Jasmine test on frontend
- Allow selection of multiple mutual funds to compare future predictions.
- Broaden scope to include ETFs, etc.
- UI enhancements such as historical graphs, comparisons across funds, animations, etc.
- (Advanced Option) Using the service provider of their choosing (ex. Google Cloud), create a SQL server instance, database, and table to write investments into. Add additional endpoints to read from and write to the db. On the UI, students choosing this extra feature can add additional display features such as an AG grid of past investments or a graph tracking investment value(s).
- Up to you, get creative!

AI Challenge:

Use ChatGPT (or model of your choice) to broaden the scope of this application's capabilities.

- Example Prompt: Given a list of tickers [y], a risk tolerance parameter p, generate for me a complete portfolio optimizing for the best returns over t years.

**y, p and t can be input parameters from the specifications above*

Feel free to be creative and add or take away parameters, or just come up with your own idea :)

Set Up Instructions:

- 1) Follow these steps to connect to OpenAPI: <https://platform.openai.com/docs/libraries>
- 2) Create a free ChatGPT account, and then follow the steps in this link to generate an API key:
<https://platform.openai.com/api-keys>