

# Performance Engine: Analysis and Evaluation

ETHAN YANEZ and DEVON STETTLER, University of Florida, USA

Our project aims to improve the way financial advisors generate performance reports relative to both customization and efficiency. We will do this by providing a simple interface and basic functionalities for selecting and placing report components, as well as integrating the OpenAI API to generate entire reports from layman's terms requests. We will be taking a minimalist approach to the UI design, ensuring that even those with very little experience will be able to quickly adopt the platform. This new way of generating reports will allow advisors to more quickly display personalized metrics to their clients without the need for extensive training or outsourced labor.

## 1 INTRODUCTION

Financial advisors are responsible for tracking markets and optimizing their clients' portfolios. With this comes the need to display the performance of their managed assets to clients. While large firms can outsource or hire internal departments for IT, smaller firms are left to rely on their portfolio management software. The current and most popular options in the market are very limited in their customization options, mostly providing static report templates. Changes to these templates or entirely custom reports can be requested from their service provider, but these often have a lengthy turnover and come with a fee. The need for customization derives from the varying investing strategies employed by advisors, as well as the diverse needs of their clients. There are several ways to approach wealth management, such as value strategy, in which advisors look for assets at a "discount", which entails that their price is lower than their intrinsic value. A relative client for such a strategy would likely want to see the price-to-earning ratios, which are the key trait of an undervalued stock. There are also growth strategies, in which advisors select stocks that are seeing (or likely to see) growth in earnings year over year, and clients would want to see long-term returns. If the growth strategy taken is also aggressive, a client would likely want to have some idea of the aggregate risk of their investments. There are also more passive investing strategies, such as those that focus on high dividend yields for stocks or coupon rates for bonds. A client whose advisor is employing such a strategy would likely focus more on income, as they are not focused on the appreciation of their assets relative to price. These are only a few of the investing strategies used today, and in practice, most advisors will likely be employing some mix of them. This trend implies the need for advisors to customize visualizations for the varying needs of their clients.

The target user of this project is financial advisors, and specifically those who are part of smaller firms. It is worth noting that the Investment Adviser Association [1] reported that in 2023, 92.7% of SEC registered advisors employed 100 or fewer people, with a median of 8. Additionally, smaller advisors accounted for a higher proportion of employment relative to assets managed, and 93.6% of new SEC registrants had less than \$1 billion in assets under management. With such a large presence of smaller firms in this industry comes a need for flexible software that does not require a high extent of technological savvy. Advisors should have a simple way of generating custom reports instantly, without having to invest weeks into learning new platforms. In this paper, we will present our progress so far in building such a service and conducting evaluation, as well as our future plans.

## 2 RELATED WORK

The bulk of related literature incorporated into our project design process involved topics surrounding learner-centered design, as well as modern design principles and considerations for software applications that make use of AI technology. Lallé et al. [3] presented an accurate method for measuring users' learning curves when first using a visualization

---

Authors' address: Ethan Yanez, ethan.yanez@ufl.edu; Devon Stettler, devon.stettler@ufl.edu, University of Florida, Gainesville, Florida, USA.

tool. With this data, they were able to make more precise decisions on when and where to add adaptive support for an interface. This study involved observing and measuring physiological activity such as gaze and pupil dilation. Focusing more on the implementation of adaptive interface, Ringbauer et al. [5] conducted a study observing the improved experience of users with disabilities or impairments when taking a "design for one" approach to UI/UX development. While we will not be able to conduct processes like those from these studies, we can use metrics like time between actions to figure out where users tend to get stuck. With this data, we can implement adaptive support on a small scale, such as pop-ups with additional instructions at sections of the report building process where users tend to have trouble figuring out what to do next. Deciding when to display a text box with extra help can be based on inactivity for a certain amount of time.

Adjacent to the idea of improving learning curves, we would like to minimize the amount of visual overload in the interface, even for experienced users. A study by dos Santos et al. [2] looked at how elderly people with 'computer anxiety' can be further inhibited by information overload. The study measured gaze and pupil dilation to determine when users were overstimulated, and with resulting data, the researchers offered design practices to avoid such negative user experiences. Not only will our target user skew toward elderly, they are also less likely to be technologically inclined. We have adopted some of the recommended practices by the researchers in our own design process, such as excluding any visual components that are not necessary for the current task at hand.

With the inclusion of AI technologies into modern-day software applications comes a necessary consideration of user perception of such integrations, relative to trust and accuracy. A paper by Long et al. [4] discusses what level of ability is required for effective interaction with and evaluation of AI, as well as how to design "learner-centered" AI technologies that are conducive to user comprehension. Particularly, what interested us is the idea of how to maximize trust and positive perception of AI. A study conducted by Shin [6] looks more specifically at the idea of what factors improve perceptions of AI, such as explainability, causability, transparency, and fairness. With targeting workers in the financial industry comes a need for transparent practices, as compliance is of utmost importance. Users must trust that their data is accurate and secure, as the lack of either trait can lead to legal troubles. Achieving data privacy is less relevant to interface design, as it is more closely related to cybersecurity. For the purposes of this paper, we will forego any explanations of how data will be secure in a production environment. More relevant topics involve those of how users will perceive the security and accuracy of our service, regardless of how the backend implementations are actually done. With this in mind, we intend on including info icons in the prompt input/text box portion of the service, which will inform users that the underlying AI technology is not being fed any of the financial data, just information about chart types, parameters, and styles.

### 3 USER REQUIREMENTS

Our service intends to allow financial advisors to quickly and easily generate performance reports for their clients. The basic requirements include functionalities for selecting data types to calculate, chart types to display, entities (financial accounts or individual positions/investments) to include, select a date range to aggregate data from, organize charts within a report, and print a final report. Our service currently pulls real price data and simulated trades from the database, but in a production environment, there would be some internal or third-party team that performs a reconciliation process for importing daily data files.

A sample run of using our service would have the user first enter the landing page, where they can select to start from a blank template or use an AI-generated report. If they choose to start from scratch, they would select the entities that will be available to them when creating the report. After finalizing their selections, they would be routed to the

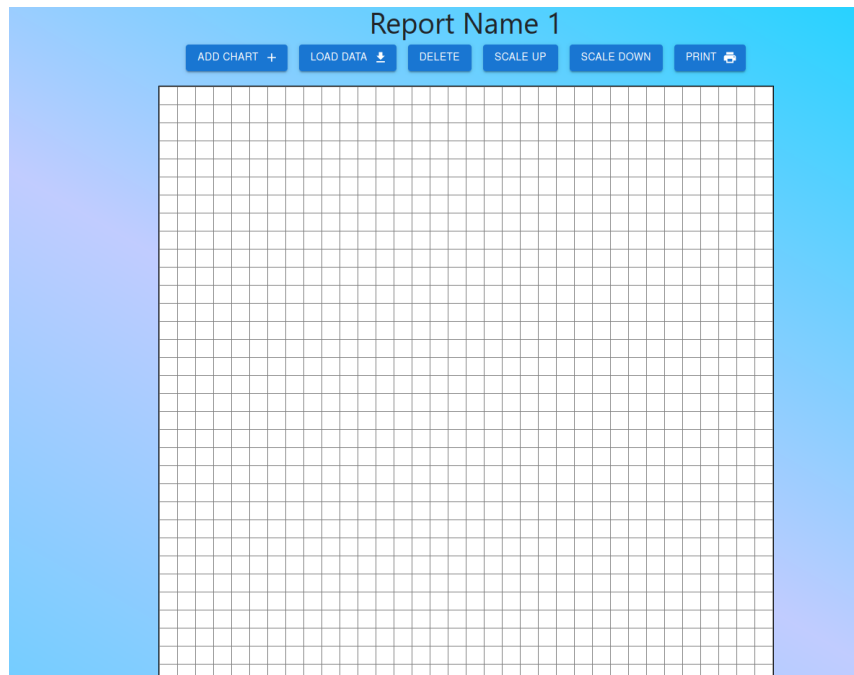


Fig. 1. Grid UI and header.

editing page, where they can select to add a new chart. Each new chart requires the selection of entities, data type, chart type, and date range. Once the advisor has selected all the charts they would like and organized them, they can populate the report with data and save it as a PDF. Going back to the start, if the user decides to use the AI functionality, they must enter a prompt, then select the entities and date range. After selecting to create the new report, they are routed to the editing page with charts pre-populated based on the integrated LLM's response to the prompt.

From pilot runs and preliminary, informal evaluations, we decided to make some minor changes to the functionality of the service. Initially, we allowed both account and position data to be displayed on a single chart. A professional reference (developer for a portfolio management software produce) recommended that we restrict charts to a single entity type, as the average advisor would seldom be interested in comparing account performance against that of a position due to the inherent difference in balance (money in the entities). We also discovered through internal evaluations that having a landing page made the flow of the user experience clearer. Originally, we had all functionality occur from the editing page, but this would lead inexperienced users to potentially use the AI functionality after having created some charts. This is not an intended workflow, AI-generated chart sets are meant to be either the start of the report editing process, or the final product.

#### 4 UI AND SYSTEM DESIGN

A key aspect of our design approach is to achieve a high level of skeuomorphism. In this case, we are attempting to match the appearance of the report once printed (or in digital PDF form) by defaulting the grid to the 8.5 by 11 US

Letter aspect ratio, as shown in Figure 1. By doing so, and allowing users to place charts directly on the grid, they can see in real-time what their clients will see.

Our interface is intended to be as simple as possible, as financial advisors tend to be less familiar with modern computer software. “Simple”, in this instance, is referring to a minimalist approach to design, avoiding any unnecessary clutter in the interface. Objects on the screen will only be presented upon request, with the initial layout showing only the grid and buttons to add a new chart, fill data, edit sizing, delete a chart, and print to PDF (as shown in Figure 1). This is distinctly different from a program like Excel, which has hundreds of buttons and features that are unnecessary for our target demographic. We are attempting to develop a system that appeals to a wide demographic of users, with a learning curve that does not require much interaction for those who do not care to learn the entire system and its different customization options. Instead, they can choose to use the AI-generated reports. On the other end of the spectrum, those who want to fully customize each individual chart can do so by creating new charts and modifying styles at their own discretion. As mentioned, we are striving to be in the middle ground of standard financial reporting tools, which often include only presets, and tools like Excel, where users are first presented with an interface with many different buttons and options.



Fig. 2. A report with several charts with data loaded. Selected charts are indicated by the dots around their edges, and can be deleted and scaled.

#### 4.1 Report Editing

When editing the report, it is important that the user can intuitively determine what they need to do to create complex layouts. Since this tool is ultimately built around creating layouts that fill data from external sources, the UI is built such that edits feel natural and consistent with the expectations a user already has, based on prior experience with similar software. When building layouts, the user is mostly adding and arranging visualizations, so the tools for these actions

**Add Chart**

Entity Level  
☒ Account ☐ Position

Select the target account(s)/position(s)

- ☐ John Adams IRA x5353
- ☒ John Adams Individual x6559
- ☒ Mary Adams 401k x6237
- ☐ Mary Adams Individual x1482

Enter the required information and select chart type.

Type: Asset Allocation

03/18/2025

Number of points: 3

Chart Type: pie

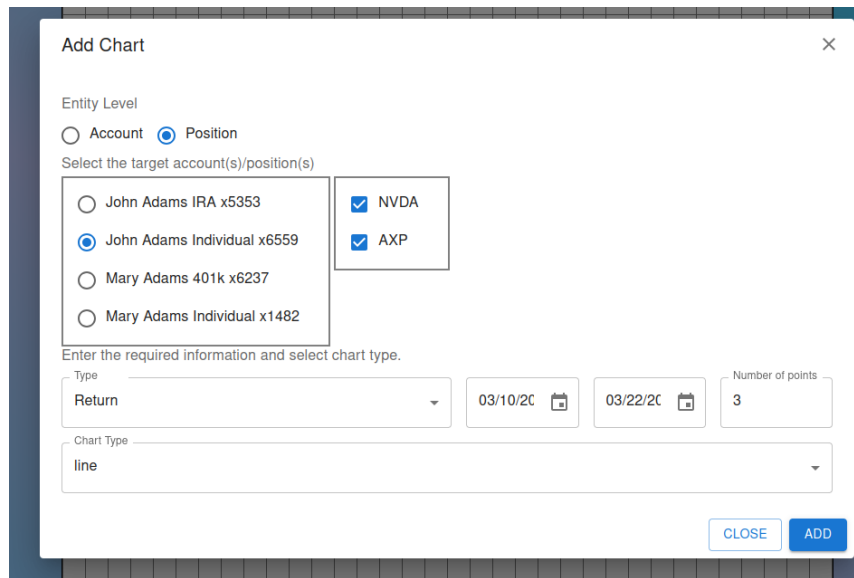
CLOSE ADD

Fig. 3. Add dialog with multiple accounts selected, and a data type that has only one date.

are immediately obvious to the user. Arranging the visualizations is performed by dragging them around along the grid in the editor, and resizing actions are executed with a simple set of "scale up" and "scale down" buttons. First, they will be asked which account ID they are targeting. Since they already selected the IDs that will appear here if they chose to start from blank, the dialog, shown in Figure 4 and 3, will only provide these IDs to reduce confusion, since advisors often have to manage many accounts. The data types available will be preconfigured for each entity type in the system, so advisors can simply choose from the options that have already been assigned, as shown in Figure ???. To open the add menu, a button will be located in the header. In case the user accidentally clicks out of the dialog, whatever they had entered will remain in place until they choose to either remove it, or click the submit button to add the visualization.

## 4.2 AI Generated Reports

In addition to the preset options, users can also forego the process of building a report by making use of our AI integration. The user will select this option from the landing page, where they will be prompted to insert a prompt for what report they would like to see. The underlying technology for this feature will involve calling to OpenAI's API, making use of their text generation feature to interpret requests. On the backend, each user will create an instance of a chat log with the LLM model (gpt-4.1-mini in this instance), but with only a single message, that being the prompt. The user's prompt will be parsed into a larger prompt that we store internally, which also includes detailed instructions as to what the output can be. The only information the API will need to provide is a layout of charts (pairs of charts and their relative data types). Our API will receive this information in a JSON format, add the necessary sizing and positioning parameters, send and then send it to the UI to render the report. The level of detail that is used in the prompt can vary, as the LLM will be instructed to produce an output regardless.



**Add Chart** [X]

Entity Level  
☐ Account ☒ Position

Select the target account(s)/position(s)

☐ John Adams IRA x5353
 ☒ NVDA

☒ John Adams Individual x6559
 ☒ AXP

☐ Mary Adams 401k x6237

☐ Mary Adams Individual x1482

Enter the required information and select chart type.

Type: Return 03/10/20 03/22/20 Number of points: 3

Chart Type: line

CLOSE ADD

Fig. 4. Add dialog with a single position and account selected, and a type with a date range.

## Create a Report

☐ Blank ☒ AI

Enter a prompt for the AI to generate a report:

AI Prompt

Please make me a returns report for these accounts.

Entity level to use with this template:

☒ Account ☐ Position

Select the target account(s)/position(s) to use with this template.

- ☒ John Adams IRA x5353
- ☒ John Adams Individual x6559
- ☒ Mary Adams 401k x6237
- ☐ Mary Adams Individual x1482

Select the date range to use with this template:

03/20/2025

03/22/2025

CREATE

Fig. 5. Creating a report with AI.

### 4.3 Finalizing Reports

Once the user has configured their preferred layout, they will be able to load data and export their document to a PDF file. Since the application runs as a webapp, the PDF export can be directly performed by the browser's print to PDF functionality. Since it is built into the browser, this can be performed in a way the user would intuitively expect, in addition to another header button that will open the browser's print menu. Additionally, once they've exported the report and seen what it would look like to their client, they can go back and conduct further edits.

### 4.4 Backend Operations

The focus of this paper is to detail the UI design of our service and how we intend for user interaction to occur. However, it is worth noting, briefly, how the API and backend (database) of our service will be structured, as well as the underlying technologies we are using. Our service was ported to the internet from our local machines, accessible via an HTTP link. The UI is built in React with TypeScript, and it is responsible primarily for allowing the user creation and movement of charts. The financial data that populates the reports will be received from the API, as well as formatting and layouts AI-generated reports. The API, which is built in Python, is responsible, mostly, for being a middle layer between the UI and database. By this, we mean that it will store class structures for the data that is transferred, but perform minimal calculations with the raw data. The database software we are using is SQL server, which is storing real investment and price data, as well as simulated trading activity generated by us. The actual calculation of outputs, like return and market value, are done via functions in SQL.

## 5 EVALUATION METHOD

The goals for our formal user study were specifically to identify what our service lacked relative to the baseline, as well as how it compared in time and cognitive workload requirements. For this usability testing procedure, we used Excel as the baseline. Originally, we planned on using another performance reporting tool. However, due to lack of resources, we were unable to gain access to such a software product and viable participants for it. As mentioned prior, our service intends to be an improvement on other low-customization reporting tools, as well as platforms like Excel and PowerBI for building reports from scratch. We altered our original plan for evaluation and desired insights to more properly fit this new baseline. For this new method, our primary hypothesis is that our new system will lead to faster overall times to generate a performance report, as well as simpler user experience.

### 5.1 Population

Our study was conducted with 6 participants. None are official financial advisors, but they all have a background in finance. While these individuals do not have hands-on experience servicing clients with portfolio management, they were all able to display a sufficient understanding as to what the average investor would like to see about their portfolio. All participants also have experience with Excel - not particularly for building performance reports, but every individual was familiar with the basic functions for simple arithmetic and aggregating data.

### 5.2 Study Conditions

All participants were part of the same group. Each user was allotted a maximum of 15 minutes of use per-session and were informed that they could take as much time as they like within those boundaries. Each participant was given the same briefing before their session began, and a think-aloud protocol was established. Participants were encouraged

## Create a Report

☒ Blank ☐ AI

New Report Name

Report Name 1

Select the entities to use in your report.

- ☒ John Adams IRA x5353
- ☐ AAPL
- ☒ RCIT
- ☐ UBER
- ☒ T
- ☐ CI
- ☒ John Adams Individual x6559
- ☒ NVDA
- ☐ XOM
- ☒ AXP
- ☐ BABA

Fig. 6. Example showing the creation of a blank report.

to convey their thoughts as they were running through the process, but the primary investigators (ourselves) only responded to actual questions. It was required that questions be answered for both sessions, using our service and Excel, so we familiarized ourselves with all of the functions and processes needed in the latter platform to produce the same charts that our service is capable of producing. Additionally, all sessions involved the same order of usage between the two platforms.

### 5.3 Participant Procedure

Each user was first sent a link to a Zoom meeting. Once they joined and clarified that they were ready to begin, they were given a briefing. This briefing had us inform the participants that they would first use our service, then Excel. The goal when using each platform was to create a report that they deemed sufficient. The sufficiency of their final product was up to their idea as to what an investor would want to see about their portfolio's performance. Outside of this, there were no restrictions on what the final product must be.

Once the briefing was completed, the users were sent a link to our service, as well as the Excel file to work from, in the Zoom meeting's chat box. The Excel file included all the necessary data to calculate performance metrics (all table data from our SQL database). Once the user was ready to begin, they were instructed to click the link and share their screen. We then instructed the user to start a timer in another window and began recording the Zoom meeting.



Once the user had gone through both platforms, we asked them about their experience, then thanked them for their participation and closed the meeting. The recordings were then used by us to collect the data for our metrics.

#### 5.4 Metrics

Our primary metrics were mostly quantitative, the first being time taken to complete individual tasks/steps. We considered these steps for both platforms to be the time it takes for users to pick initial parameters on the landing page, add new charts and organize them, populate charts with data, and then print to PDF. We also measured the number of "pauses" each user had during the use of each platform. In this case, a pause is specifically referencing the user not clicking any components on the screen or typing anything out for more than 15 seconds. Our other primary quantitative metrics included overall time taken to reach a satisfactory report, number of questions asked, and error rate. To clarify, we considered the error rate to be the number of errors when attempting to complete an action versus successes. For the purposes of this study, an "action" is considered any process of interacting with the UI that leads to an output, which included selecting parameters, moving charts, resizing charts, calculating metrics, and printing reports. Our primary qualitative metrics included the specific questions asked by each user during their sessions, as well as their feedback after using both platforms.

Our only secondary quantitative metric was the number of times that a user went back to make more edits after reaching a final product. We considered an instance of this event to specifically be the user going to export the report to PDF, then going back to the platform to continue editing. We considered this metric to be secondary, as the users were instructed to create a report that they personally deem sufficient, which can greatly vary between users. We also considered one secondary qualitative metric to be the facial expressions and body language of participants as they were using each platform. This, again, can be quite subjective, especially given that we did not reference any expertise from fields like psychology or sociology.

#### 5.5 Informal Evaluation (Pilot and Revisions)

After our official switch of baseline, we conducted a pilot run for our planned evaluation. This involved us playing the role of the participant, and the other that of the PI, then switching. We then conducted an informal version of the pilot with a colleague outside of this study. Our original plan had us responding to any spoken comment from the participant, however, we found that this created a large bias between platforms. As we mentioned, all participants are familiar with the use of Excel and its functions, so there were likely to be less confusions during its use. If we had responded to every comment, that would have a large impact in our metric of number of questions asked, as any indicator of confusion has the potential to become an official question. So, we decided to only speak during the session when participants asked a direct question.

Additionally, our original briefing asked participants to create a report that they deem sufficient to show to an investor. Allowing the level of sufficiency to be decided by the user was still desired, as there is a large variance in what advisors tend to show and investors like to see. We did find, though, that this method presented two issues, the first being a potential tendency for participants to prolong their use of our platform out of interest in its capabilities. The other concern was how this would impact participant fatigue, as every user first started with our service, then moved to Excel. To attempt to address these concerns, we added an additional instruction in the briefing asking that the participant take any amount of time within the 15-minute time constraint, but also be encouraged to end their individual sessions as soon as they reached an output that they deemed sufficient (as opposed to making minor tweaks on top of a sufficient final product).

In general, cognitive walkthroughs allowed us to address hypothetical, one-off situations with minor tweaks. Specifically, we wanted to ensure that a user's action led to an expected outcome. Due to our minimalist approach to the UI, options for different actions are clear and feedback is usually immediate, but this presents an issue when it comes to making a component's purpose very clear. The modules for selecting parameters were designed with this idea in mind, ensuring that the entity-level a user was running a report or creating a chart at was clear, and that each parameter for an individual chart (chart type, data type, and date range) were selected in a specific order such that only those parameters that are compatible with those selected prior are presented as options.

## 6 RESULTS AND ANALYSIS

Table 1. Mean Quantitative Metrics Across Groups

Metric	Baseline	New System	Improvement
Overall Time (s)	672.5 $\pm$ 105.58	500.3 $\pm$ 142.6	34.4%
Error Rate (%)	8.2% $\pm$ 3.6%	9.5% $\pm$ 3.2%	-13.8%
Pauses	2.8 $\pm$ 1.77	4.3 $\pm$ 2.6	-34.6%
Restarts	2.2 $\pm$ 0.69	1.5 $\pm$ 1.7	44.4%
Questions	8.5 $\pm$ 4.11	7.2 $\pm$ 3.8	18.6%

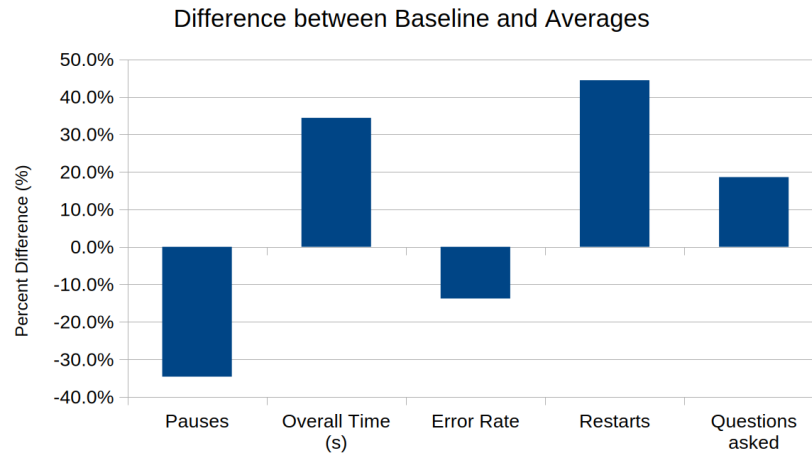


Fig. 7. Percent difference between each of the baselines and the new averages.

When looking at the mean values of the metrics between the baseline and our new system (as seen in Table 1), we note that our system did not outperform Excel on all fronts. We note, though, that the only lack of improvement is seen in number of pauses and error rate, both of which indicate some impact of the participant's greater familiarity with Excel. These findings support our primary hypothesis on one end, as average overall completion times saw a decrease. On the other hand, we can not definitively conclude that the use of our platform was internally felt as a simpler process for users.

## 6.1 Efficiency of Report Generation

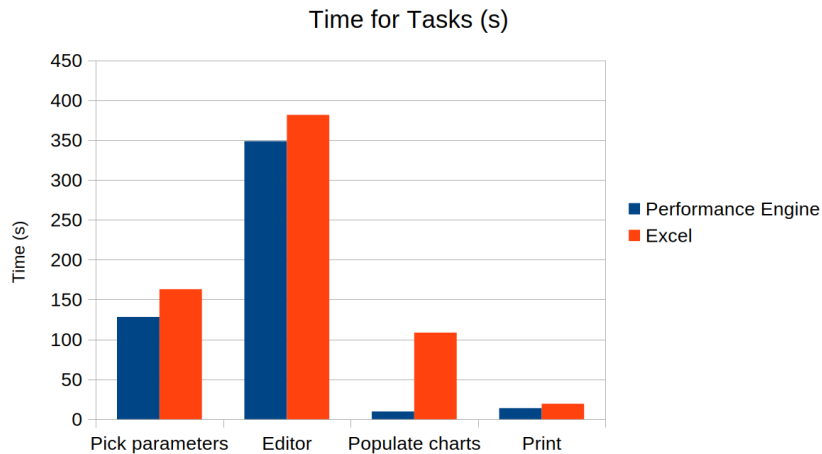


Fig. 8. Average times for each task.

As mentioned, we see from the data that the mean time to generate a performance report showed a decrease with the new system. When looking at the individual times taken for each task (as seen in Figure ??) we notice that most of the difference is accounted for by the time it took users to populate charts with data for each platform. This was expected, as for our service, there is one button that populates all charts on the grid with data, whereas in Excel, the user must enter the different formulas for calculating performance metrics at least once (use of the AutoComplete feature allowed them to avoid having to type out many of the same formula). We do, however, see marginal improvements on every task when users were on our system. We attribute this mostly to the very minimal number of options and modules in our UI, as well as the avoidance of needing to type out functions to select groups of data and perform calculations on them. This, again, was expected, as we are simply providing features that Excel does not which automate certain processes.

Relative to our metrics that showed a decrease in efficiency, we consider these mostly to be due to user's lack of experience using our system. This is to be expected, as we gave no preliminary training session or documentation to participants. It is, though, worth investigating what could have minimized this difference. Assuming that our system had the potential to do better in these metrics, we looked into what features could have been changed. The most common questions during each session with our system were regarding the selection of entities, and it is also what accounted for most of the pauses and errors. As mentioned, charts can only be run on the account or position-level, not both, so when users select entities, they can only choose from a single entity type. We thought this to be somewhat clear in the individual add-chart module, as they are locked into only selecting from one entity type. However, from the landing page, where users are prompted to select all entities that will be available to them in the editor, they can select from either entity type. We may see a decrease in confusion over this aspect of the system if the landing page gave a clearer indication of what would actually be happening with the entities they select, specifying that they are choosing the data available to them in the editor through a text component, or changing the way entities are selected.

## 6.2 User Experience

Outside how quickly a user could generate a report, we also investigated the second part of our primary hypothesis, that being the overall user experience. We did see a lower number of questions asked by users on our system, however, we attribute much of this to the simpler UI, as Excel inherently has many more subtasks that need completing. As mentioned, most questions were regarding the selection of entities, however, there was at least one question about every feature in our system. One expected question was regarding the number of chart and data types available, of which there are less than 5 for each. We note this not as something to be fixed, but rather expected to come if the project saw further development. There were also questions about customization of charts, which so far only included positioning and resizing, however, future development would add customizations of individual graph components. The last common question, which did call for a revision of previous ideas, was regarding the capabilities of the AI generated reports. As mentioned, one can currently only start with an AI report, not use it from the editing page. Allowing users to use the AI feature from the editing page was not an intended feature in the future, however, several participants claimed that this would be useful, as it would allow them to flow freely between custom additions of charts and ones generated just from a prompt based on how much they would like to customize each individual chart.

The general feedback after each session reflected both pros and cons of our platform versus Excel. The positive feedback indicated that they did prefer the UI being less cluttered, as they could narrow in on their specific task of generating a report. Additionally, they claimed that it was much more convenient to have the calculation of performance metrics built in, rather than having to know the Excel functions. One con that was mentioned frequently was the lack of customization, which we mentioned is planned to be added with more features in the future. One useful concern that some participants brought up regarded the selection of entities for each individual chart. Our original idea was to have each chart be individualized, requiring all parameters when being created, as other similar service in the industry require that every chart in a single report have the same entities. From the feedback, though, we gathered that this is a functionality that users would like to have access to, as sometimes a single report would contain every entity.

## 6.3 Summary of Findings

Overall, we found that users were able to more quickly generate a sufficient performance report with our platform compared to Excel, mostly attributed to the lower number of components in the UI and automated subtasks. We note, though, that while participants in the study tended to ask fewer questions when using our platform, there was still a higher error rate and number of pauses on average. We theorize this to be due to some lack of expected functionalities, mostly regarding the selection of entities. Because the UI of our platform is less cluttered, it appears to be very simple, which leads to fewer questions from users. However, this also leads users to make confident assumptions of the functionality of several components. Our future work intends to address this concern.

## 7 CONCLUSION AND FUTURE WORK

Our goal is to make the generation of financial performance reports a task that requires as little technical knowledge as possible, while still allowing the final product to contain exactly what advisors are looking for. We aimed to build out a service that accommodates varying levels of desired effort, making the interface feel like a digital sketch pad, and in some ways it seems that we did that. Financial advisors already wear many hats as it pertains to wealth management, client relationship management, and making executive decisions for their firm, so displaying performance to their clients should be as simple and customizable a task as possible.

For future development, there is room for more features to be added, and bugs ironed out as always. The use of user-saved templates had to be dramatically scaled back in favor of the AI generation, due to our time constraints, but would be especially useful for power users of this software, since frequent rebuilding of reports could be avoided. Additionally, live edits of things like the document title, a scaling system in line with modern photo editing and design software (like one with handles), could be an improvement over the scaling buttons, since some users instinctively went to scale charts in this way. Finally, regarding future development that was not previously intended, we do plan to look into how to alter certain components in the UI which present a disconnect between intended and actual functionality for users. This will likely entail tweaks to how users select parameters, with the end goal of providing expected outcomes. After the addition of more customization options and a change to parameter selections, we would have the next version of our service. With this version, a great follow-up study would have us compare against other performance reporting tools in the industry of portfolio management software providers. This study allowed us to gain insight into how users prefer our service over one that requires much more work and individual task completion, but we are still interested in seeing how our service compares in efficiency and user satisfaction to what many real advisors would be used to.

## **8 APPENDIX A: INFORMED CONSENT AGREEMENT**

Attached to Canvas submission.

## **9 APPENDIX B: ROLES AND CONTRIBUTIONS**

For development, we have divided work between frontend and backend, and worked together on evaluating our user-facing design. Frontend development was mostly done by Devon, writing React code in Typescript. He also made the original mockup sketch in Figma based on the shared design ideas created in the original brainstorming step. Backend development was done by Ethan, writing Python code to interface with a SQL database, and setting up the Flask API endpoint. He also wrote a significant amount of the original Pitch. Internal evaluation was done at all steps of the UI design process, and ideas were floated back and forth by both of us. We both contributed to the evaluation process itself, being present in the Zoom calls where we monitored the users' actions in the interface, taking our notes and recording the metric information.

## **10 APPENDIX C: COMPLETED CONSENT AGREEMENTS**

Attached to Canvas submission.

## **11 [EXCLUDED] APPENDIX D: PHOTOS FROM EVALUATION**

## **12 APPENDIX E: LINK TO CODE REPOSITORY**

Our frontend repo (<https://github.com/ethany80/PerformanceEngineUI>)

Our backend repo (<https://github.com/ethany80/PerformanceEngineAPI>)

## **REFERENCES**

[1] Barr, K. L., Bernstein, G. C., Grossman, L. L., Wise, M., Rickwalder, J., Honea, G., Wijegunawardana, H. (2024). Investment Adviser Industry Snapshot 2024. <https://www.investmentadviser.org/wp-content/uploads/2024/06/Snapshot2024FINAL.pdf>

- [2] dos Santos, T. D., de Santana, V. F. (2021). Gaze interaction and people with computer anxiety. Proceedings of the XX Brazilian Symposium on Human Factors in Computing Systems, 1–12. <https://doi.org/10.1145/3472301.3484319>
- [3] Lallé, S., Toker, D., Conati, C., Carenini, G. (2015). Prediction of users’ learning curves for adaptation while using an information visualization. Proceedings of the 20th International Conference on Intelligent User Interfaces, 357–368. <https://doi.org/10.1145/2678025.2701376>
- [4] Long, D., Magerko, B. (2020). What is ai literacy? competencies and design considerations. Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems, 1–16. <https://doi.org/10.1145/3313831.3376727>
- [5] Ringbauer, B., Peissner, M., Gemou, M. (2007). From “Design for all” towards “design for one” – a modular user interface approach. Lecture Notes in Computer Science, 517–526. [https://doi.org/10.1007/978-3-540-73279-2\\_58](https://doi.org/10.1007/978-3-540-73279-2_58)
- [6] Shin, D. (2021). The effects of explainability and causability on perception, trust, and acceptance: Implications for explainable AI. International Journal of Human-Computer Studies, 146, 102551. <https://doi.org/10.1016/j.ijhcs.2020.102551>