A FRAMEWORK FOR EVALUATING AI-GENERATED CONTENT USING HUMAN FEEDBACK

Chris Vijn[1], Reinier Bos[2], Maciek Wrona[3], and Diego Jimmy Sourbag[4]

[1]**s3691926**
[2]**s3703177**
[3]**s3679861**
[4]**s3699420**

Executive Summary

The adoption of LLMs in business and in the general public paved the way to create a lot of new products and tools utilizing this new technology. For Pegasystems, this meant the creation of their AI-powered workflow builder called Pega GenAI Blueprint™. A common problem that comes with the implementation of these LLMs and tools using these LLMs is the evaluation of their output. Without trustworthy evaluations, it is hard to optimize your models for your users thus posing a threat to the product's performance.

Pegasystems experiences the same problem regarding their product. This research aims to provide a tool that is able to use mass human evaluation to effectively evaluate the output of their tool using different AI models and other parameters to derive what works and what does not. This could be used by Pegasystems to evaluate their Pega Blueprints. During this research, a tool was thus made that users can use to give their opinion on two different Blueprints created using the same prompts but different parameters and/or models. The data gathered by this mass user evaluation can then be used to derive impactful insights in order to further streamline Pega GenAI Blueprint™ for its users.

The tool is designed in the form of a web app that uses two separate windows to show the user the two Blueprints. A user can then choose ”first better”, ”second better”, ”both good” and ”both bad”. The answers are used to rank the different model configurations using an Elo model that is often utilized in competitive sports like chess. There was not enough time to complete the research by gathering a good amount of data, however, the way in which this data could be used is explained in detail in the discussion.

Keywords: LLM evaluation, Blueprint ranking, Pegasystems, Elo model, generative AI, user feedback

1 INTRODUCTION

In recent years, the world has experienced a steep increase in production and usage of Artificial Intelligence (AI), mainly Large Language Models (LLMs); powerful tools to improve the quality of life and work. This has led to a vast number of business opportunities. Pegasystems has taken advantage of this revolutionary technology and applied it in the business context, which resulted in the creation of Pega Blueprint; an application of the AI tools for generating workflows.

Pega Blueprints are designed to automate your workflows by helping business & IT leaders to create a vision for an enterprise-changing software solution. Usually, this process could take weeks of collaboration between the two parties. With Pega Blueprints you can fill in a prompt, and the tool will provide you with a detailed workflow of how the software landscape could look when implementing the software that you want. The tool helps these teams in creating a vision and start big projects like these, as explained in Pegasystems' own article, ”The hardest part is where to begin” (Akgonul (2024)). Pegasystems alleviates this pain by offering an AI-powered tool to do this for you.

Pega Blueprints leverages the power of generative AI and combines it with curated templates provided by industry experts of Pegasystems. They are still actively developing their tool, as generative AI is evolving rapidly. Pegasystems does this by training a variety of different models (mainly OpenAI) on these templates. They do, however, not have a tool to decide which of the models performs the ”best” in the eyes of users. At this moment, the effectiveness of their trained models is decided solely by the developers and by evaluations done with users on a smaller scale.

Many LLM developers have already come upon a similar, if not the same, problem. To keep track of the quality of LLMs, they must be evaluated in some way. Previously, statistical benchmarking was used, but soon enough limitations were observed, such as lack of flexibility and limited real-world adaptability (Zheng et al. (2023)) as well as contamination of test sets (Yang et al. (2023)). Several academic researchers addressed those limitations by creating the Chatbot Arena (Chiang et al. (2024)). This is a website where the evaluator enters a prompt and afterwards 2 LLMs produce an output. Thereafter the evaluator chooses the preferred output: the ”winner”. Using an Elo system, LLMs are ranked on the leaderboard. Anyone can be an evaluator on Chatbot Arena: https://lmarena.ai/.

In this research, the focus is on the evaluation of Pega Blueprints. As mentioned previously, the quality of the Blueprints is hard to determine; it is subjective whether a Blueprint is good or not. This has currently only been investigated by Pegasystems' employees, which resulted in the product they now have.

The goal of this research is to quantify user feedback on the quality of Blueprints. By doing this, different configurations of the Blueprint generator can be evaluated to see the effect it has. This allows Pegasystems to evaluate different configurations and get the best Blueprints.

One solution would be to integrate the Blueprint generator with Chatbot Arena. However, integrating the Blueprint with a third-party tool is not a feasible solution due to safety concerns. Therefore, the solution proposed was to develop an application to quantify user evaluations. This application would be similar to Chatbot Arena, but tailored for Blueprint evaluation.

1.1 Problem statement and research question

In summary, the following problem statement follows from the previous:

- **Problem statement:** It is hard to determine the quality of LLM output, as it operates as a black-box process. However, this is valuable information for improving results. Pegasystems currently lacks a standardized, objective, and in-house mechanism to evaluate the effectiveness and performance of their Blueprints. This lack of an evaluation framework makes it challenging to ensure the quality and reliability of AI-generated Blueprints.

This leads to the following research question:

- **Research question:** ”How can a standardized evaluation framework be developed to objectively assess the quality and performance of AI-generated content?”

1.2 Approach

The approach used during this research is structured and iterative such that clarity is ensured throughout the entire process. The starting point is the gathering of detailed requirements to understand the objective and constraints. From there, we define use cases that outline specific scenarios. During the design phase, Pegasystems’ feedback was actively incorporated to ensure that the solution aligns with best practices and stakeholder expectations. Once the design was refined, the focus shifted to practical implementation. This is the development of the evaluation tool, which includes the implementation of the ranking system and the identification of the specific questions that need to be addressed prior to the user’s utilization of the tool.

1.3 Reader’s guide

The remainder of this report has a familiar structure. First, the background is explored, providing academic context, and introducing the Chatbot Arena. This section delves into its purpose, functionality, and origins, laying the groundwork for subsequent discussions. Next, the focus shifts to the data, briefly outlining the datasets utilized during the research, and explaining the role of the API in generating and retrieving Blueprints. Following this, the framework is presented that encompasses the enumeration of the tool’s requirements, its use cases, and the design. This section also addresses the Elo ranking system and the implementation of the evaluation tool. Afterwards, the results section will showcase the results of the research, highlighting the operation of the evaluation tool and the deliverables of this research. The discussion then considers the broader implications of the research, its limitations, and potential areas for improvement. This section also explores opportunities for future research and the value it could bring to Pegasystems. Finally, the conclusion summarizes the key findings and achievements of the study.

2 BACKGROUND

As generative AI is still a fairly new technology that is still very actively being developed, it will be key for Pegasystems to keep fine-tuning their models. Pegasystems uses the OpenAI API to build its Blueprints by training OpenAI models on their curated data. However, it is hard to benchmark and gain insight into how well the different curated models perform. This is in its essence not a new problem; machine learning as a field knows numerous models that have the same problem. The models can be seen as black boxes, they get input and you get an output, but what happens in between and how the model comes to its conclusions is hard to derive. This is also the case for LLMs as well, especially concerning the BP making tool that Pegasystems offers to its clients. Because of this aspect of LLMs, it is key to evaluate which types of training and fine-tuning and/or which models produce the best outcomes, as deriving how models come to their outputs is hard. There have already been several attempts at making tools that do this like Chatbot Arena which we mentioned already.

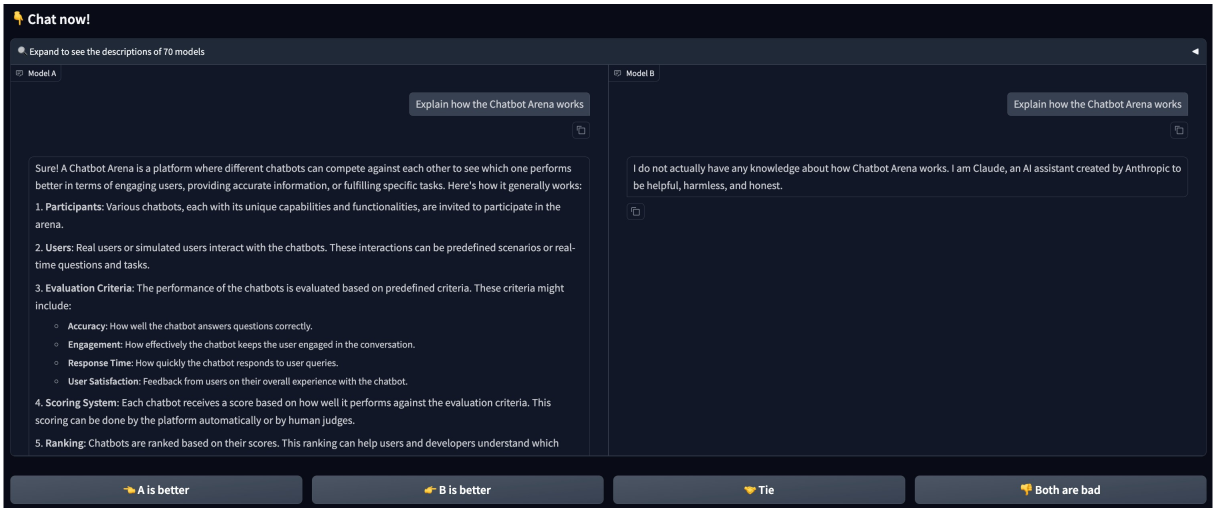
2.1 Evaluation methods

Understanding how to evaluate LLMs can be broken down into three questions: ’What to evaluate?’, ’Where to evaluate?’ and ’How to evaluate?’ (Chang et al. (2024)). This research focuses on the question of how to evaluate. Evaluations can be conducted using automatic or human methods. Automatic evaluation involves using established metrics such as BLEU, ROUGE, and BERTScore to measure the quality of LLM-generated content. This method is widely adopted for its efficiency and effectiveness in deterministic tasks, such as mathematical problem-solving and natural language understanding. In addition, automatic evaluations save significant time and resources compared to mass human evaluations. Recent advances have explored training LLMs to act as judges for evaluating other models’ outputs, although these methods are still in development.

However, the capabilities of LLMs often surpass traditional natural language processing metrics, making human evaluation essential for tasks requiring subjective judgment or real-world application. Although automatic methods such as BERTScore provide consistent results, they have been found less reliable than human evaluations, which offer a more comprehensive understanding of model performance in diverse scenarios (Chang et al. (2024)). Human evaluations are particularly valuable for capturing the nuances of user expectations and applications, although they are not without challenges. Bias and variance in human judgments, influenced by factors such as cultural and professional background, can introduce instability in results. To address this, expert oversight is critical to validate evaluations and reduce bias, ensuring that conclusions are reliable. Strategies such as aggregating feedback from diverse evaluators and implementing detailed evaluation guidelines can help mitigate variability.

2.2 Applied Evaluation methods

Chatbot Arena is a place where different Large Language Models can be evaluated. It works by entering a prompt. This prompt is input for two random LLMs. The LLM answers are displayed next to each other and the user can review the output by choosing the better one, a tie, or both bad. An example is visible in Figure 1.



Explain how the Chatbot Arena works

Sure! A Chatbot Arena is a platform where different chatbots can compete against each other to see which one performs better in terms of engaging users, providing accurate information, or fulfilling specific tasks. Here's how it generally works:

1. **Participants**: Various chatbots, each with its unique capabilities and functionalities, are invited to participate in the arena.

2. **Users**: Real users or simulated users interact with the chatbots. These interactions can be predefined scenarios or real-time questions and tasks.

3. **Evaluation Criteria**: The performance of the chatbots is evaluated based on predefined criteria. These criteria might include:
   - **Accuracy**: How well the chatbot answers questions correctly.
   - **Engagement**: How effectively the chatbot keeps the user engaged in the conversation.
   - **Response Time**: How quickly the chatbot responds to user queries.
   - **User Satisfaction**: Feedback from users on their overall experience with the chatbot.

4. **Scoring System**: Each chatbot receives a score based on how well it performs against the evaluation criteria. This scoring can be done by the platform automatically or by human judges.

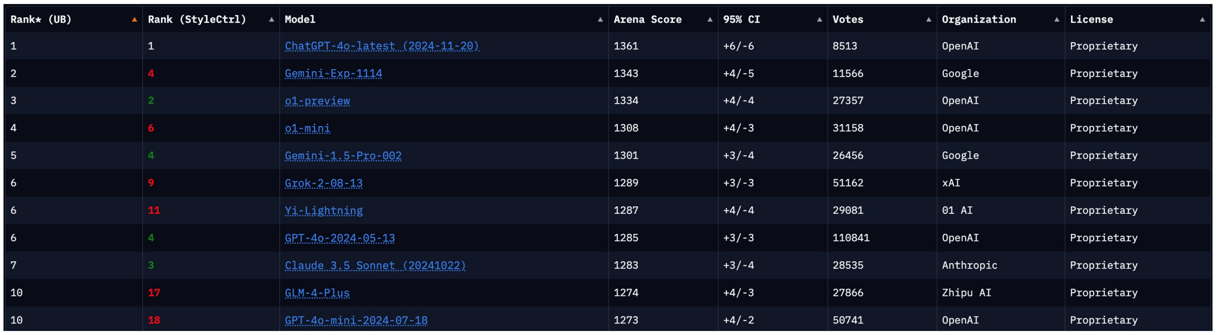
5. **Ranking**: Chatbots are ranked based on their scores. This ranking can help users and developers understand which

Explain how the Chatbot Arena works

I do not actually have any knowledge about how Chatbot Arena works. I am Claude, an AI assistant created by Anthropic to be helpful, harmless, and honest.

Figure 1. Chatbot Arena

The models are compared with each other using an Elo model. This is a model where each model receives an initial rating. In each evaluation model, model A is displayed next to model B. If model A wins, the score of model A will increase, while the score of model B will decrease. This works in both ways.



Rank* (UB)	Rank (StyleCtrl)	Model	Arena Score	95% CI	Votes	Organization	License
1	1	ChatGPT-4o-latest (2024-11-20)	1361	+6/-6	8513	OpenAI	Proprietary
2	4	Gemini-Exp-1114	1343	+4/-5	11566	Google	Proprietary
3	2	o1-preview	1334	+4/-4	27357	OpenAI	Proprietary
4	6	o1-mini	1308	+4/-3	31158	OpenAI	Proprietary
5	4	Gemini-1.5-Pro-002	1301	+3/-4	26456	Google	Proprietary
6	9	Grok-2-08-13	1289	+3/-3	51162	xAI	Proprietary
6	11	Yi-Lightning	1287	+4/-4	29081	01 AI	Proprietary
6	4	GPT-4o-2024-05-13	1285	+3/-3	110841	OpenAI	Proprietary
7	3	Claude 3.5 Sonnet (20241022)	1283	+3/-4	28535	Anthropic	Proprietary
10	17	GLM-4-Plus	1274	+4/-3	27866	Zhipu AI	Proprietary
10	18	GPT-4o-mini-2024-07-18	1273	+4/-2	50741	OpenAI	Proprietary

Figure 2. Chatbot Arena Ranking

Figure 2 shows the result of the evaluation in the Chatbot Arena. Different models are ranked on the basis of the results of the user evaluation. This shows how the subjective output of LLM can be quantified and ranked. This Chatbot Arena is an inspiration for the Blueprints evaluation

framework.
As mentioned in the introduction, we took inspiration from a website called Chatbot Arena, this is a project that is working on by three universities:

1. UC Berkeley

2. Stanford University

3. UCSD

Outside of the contributions of these universities, there were also many companies and other third parties involved in the eventual success and launch of Chatbot Arena such as companies like Hugging Face and other big AI companies and communities who helped and/or sponsored this project. In the paper written about Chatbot Arena, we will refer to (Chiang et al. (2024)) there are a multitude of problems described when it comes to navigating their and our problem, making an evaluation tool.

Chatbot Arena used human preference evaluation, which is exactly what it sounds like, namely, the gathering of a massive amount of data using a normal ’free’ human evaluation. The user (in Chatbot Arena) is presented with two windows, each showing the output of one of the models, while keeping the prompt completely open to be edited by the user, guaranteeing a creative incentive for the user to use the tool and ensuring that the researchers receive a set of diverse data to garner new insights from. Our evaluation tool will be building on this principle; however, Pegasystems also specified that our evaluation tool needs to be modular in such a way that it could be used to evaluate on other criteria outside of just evaluating which model creates the best Blueprints. This makes our case a bit more complex as we need to take different questions and possibilities into account.

PandaLM is another form of evaluating the output of LLMs. It is an LLM with the sole purpose of evaluating other LLMs on how good their output is (Wang et al. (2023)). PandaLM does the same thing as Chatbot Arena but without human touch. You feed the model a set with data consisting of instructions and inputs combined with the output of two different models. The data also has human-generated features of the said input, the output then being a type of text classification by the models that are being tested by PandaLM. The same problem arises in this case, the judge LLM: Panda LLM is still a black box, meaning that the way of generating its answers can not be derived. In order to fine-tune PandaLM human evaluations were once again used.

Several frameworks are also already in place that aim to assess the effectiveness of LLMs when it comes to answering the prompts. Another framework makes use of the completion of several different tasks to benchmark their trained model (Radford (2018)). They chose these tasks in such a way that you can derive useful information from the success of the task. The level of output can then be assessed and used for further tuning of hyperparameters or for training with other types of data to further train the model to be able to handle these tasks. The tasks from which they chose to derive their insights were as follows:

1. Natural Language Inference
2. Question answering and commonsense reasoning
3. Semantic Similarity
4. Classification

They would use the information derived from letting the model do these tasks to test and then to pre-train and fine-tune the hyperparameters for their model to enhance its ability to do these tasks. Pegasystems could use a technique like this as well to determine certain weak points in their model when doing tasks in making BPs for different types of company, for example (i.e. banks, insurance companies, etc.). With this understanding of the existing tools, we will form the methodological approach for our evaluation framework.

3 FRAMEWORK

Building on the theoretical foundation, the practical implementation focuses on adapting these principles for Pega Blueprints.

3.1 Requirements

The practical implementation of theory consists of designing a evaluation tool, where users can evaluate the Blueprints. The functional requirements for the evaluation tool are:

1. **Compare Blueprints**: A user should be able to compare two different Blueprints, which are generated based on his own requirements.
   Priority: High
   Acceptance criteria: A user can generate two Blueprints with his own input and view them side-by-side.

2. **General questions**: An user should fill in a few general questions, which can be taken into account when analyzing the results
   Priority: Medium
   Acceptance criteria: A user answers a few question before generating Blueprints, and the responses are used in the evaluation process.

3. **Everything variable**: Everything should be designed as variables, which can be defined by Pegasystems. Therefore, they can change the functionality of the application in the future or use it for evaluating something else.
   Priority: High
   Acceptance criteria: All functionality is configurable via variables, with no hard-coded parts in the back end, allowing Pegasystems to update logic or criteria without code changes.

The non-functional requirements are:

1. **Pegasystems' design style**: The design should be in line with other Pega products, as they want to present a professional product to users.
   Priority: Medium
   Acceptance criteria: The design follows the Pegasystems UI / UX and branding guidelines, and Pegasystems agrees on the delivered design.

2. **Minimize bias**: Everything in the evaluation tool should be designed to minimize bias, as that would influence the results.
   Priority: High
   Acceptance criteria: The tool must ensure fairness by providing consistent and unbiased results across diverse inputs.

3.2 Use cases

The requirements for the system are defined, and two use cases were designed, to get a good view of the usage of the system. The first use case is for a user who evaluates a Blueprint. The second use case is for a Pegasystems employee who analyzes the results.

3.2.1 Use case 1

- **Actor:** User

- **System:** Pega Blueprints, Blueprint evaluation software

- **Goal:** Comparing two different versions of the Blueprint. The evaluation software calls the Blueprint API with two different model variables. The Blueprint API returns two Pega Blueprints that the user can evaluate.

- **Preconditions:** User knows how a Blueprint works and is willing to evaluate them. Additionally, the model variables, which are input for the Blueprint API, are mapped to different Blueprint configurations. This is done by the Blueprint development team.

- **Scenario:**

  1. The user is asked to evaluate the Blueprints and is willing to do so.

  2. The user opens the blueprint evaluation software.

  3. User answers a few general questions about himself and his position

  4. The user fills in the requirements for his Blueprint

5. Blueprint API is called twice by the evaluation software with the same user information, but two different variants.

6. Two Pega Blueprints are returned to the evaluation software and are shown to the user.

7. User evaluates both Pega Blueprints by giving his preference; first better, second better, both good and both bad

8. User input is saved based on the variant.

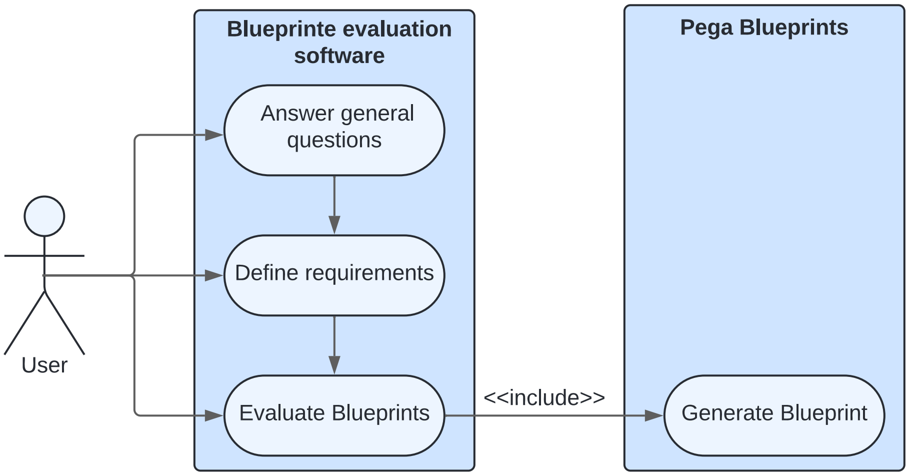


Figure 3. Use case 1

3.2.2 Use case 2

- **Actor:** Pegasystems Employee

- **System:** Blueprint evaluation software

- **Goal:** Evaluating the different Blueprint versions by a Pegasystems employee. This is done with the output of the Blueprint evaluation software. The output shows the different models with their ratings, which can be interpreted to conclude the best configuration for generating Blueprints.

- **Preconditions:** Enough Pega Blueprints have been evaluated to provide significant output.

- **Scenario:**

1. Pegasystems employee opens Blueprint evaluation software.
2. Pegasystems employee navigates to the results section.
3. Blueprint evaluation software displays a table of results for all evaluated variants.
4. Pegasystems employee filters the results
5. Pegasystems employee analyzes the difference between different models.

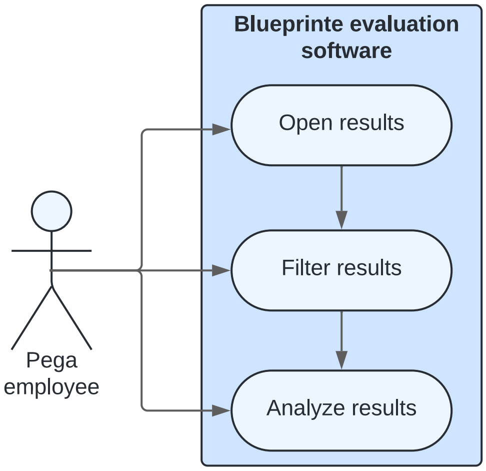


Figure 4. Use case 2

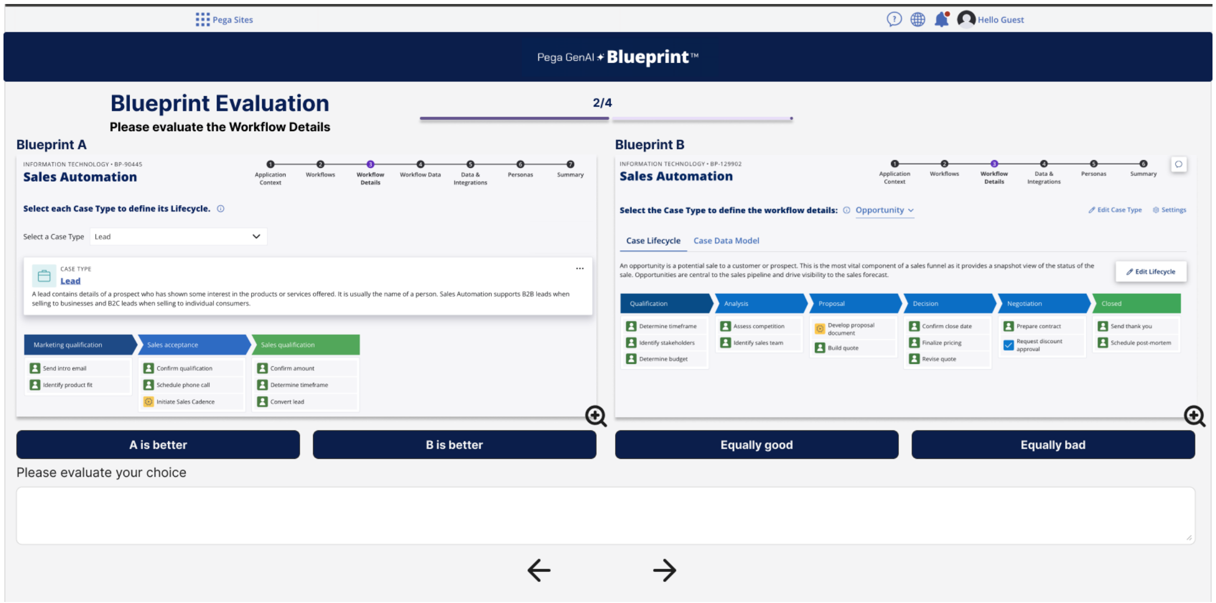
3.3 Design

Design is an important aspect of the evaluation tool, as it can impact the user's opinion on the Blueprints. Additionally, it is the product that Pegasystems customers will see, and they want to present them a professional product. Therefore, the evaluation tool will have a design that sticks to the design principles of Pegasystems as much as possible. We made a few choices in the design to improve usability and keep bias low.

First of all, we chose to place the Blueprints next to each other, instead of below each other. That is because a side-by-side comparison minimizes bias compared to a stacked comparison (Humphries et al. (2021)). However, Blueprints cover a whole page, so it is harder to see detail in a smaller frame. Therefore, we added a zoom functionality on the bottom right of both Blueprints. With this feature, users can expand the Blueprint and see details if needed.

To evaluate and compare different configurations of the Blueprints, such as variations in the underlying LLM (e.g., GPT-4o, 4o-mini, or 3.5 Turbo) or adjustments to hyperparameters (e.g., temperature), the tool allows users to provide feedback in two ways. First of all, they have the possibility to choose one of the options displayed below the Blueprints. Users can express if they like one Blueprint over the other, or rate them as ’equally good’ or ’equally bad’. We chose to name the Blueprints as ’A’ and ’B’ instead of ’1’ and ’2’ to prevent bias, as ’1’ and ’2’ might indicate that the first is higher ranked. Additionally, users can leave an explanation below that gives a better understanding of their choice. This could also be replaced by a question to specifically focus on one aspect of the Blueprint. However, these questions should not be too guiding in the decision. If these questions are added, they will be provided by Pegasystems.

A Blueprint consists of different stages. It is very hard for a user to compare results that consists of multiple parts, and it would be hard to analyze, as you do not know what the user based their decision on. Therefore, we chose to compare a few specific pages of the Blueprint. The user is asked to compare each page. The amount and specific pages will be provided by Pegasystems. A progress bar will be displayed at the top, to make users aware of the length, and thereby increase the chance that they will complete the whole evaluation, as it will only be a few steps. The complete design is visible in Figure 5.



Blueprint Evaluation

Please evaluate the Workflow Details

2/4

Blueprint A

INFORMATION TECHNOLOGY • BP-90445

Sales Automation

Select each Case Type to define its Lifecycle.

Select a Case Type: Lead

CASE TYPE

Lead

A lead contains details of a prospect who has shown some interest in the products or services offered. It is usually the name of a person. Sales Automation supports B2B leads when selling to businesses and B2C leads when selling to individual consumers.

Blueprint B

INFORMATION TECHNOLOGY • BP-129902

Sales Automation

Select the Case Type to define the workflow details: Opportunity

An opportunity is a potential sale to a customer or prospect. This is the most vital component of a sales funnel as it provides a snapshot view of the status of the sale. Opportunities are central to the sales pipeline and drive visibility to the sales forecast.

Please evaluate your choice

Figure 5. Evaluation tool design

3.4 User information

The Chatbot Arena is an example of how to rate subjective results of an LLM. This evaluation principle will be the basis for evaluating Blueprints. However, a Blueprint has more detail than an LLM response. Therefore, we need to enhance this evaluation principle to make it suitable for evaluating Blueprints. First, several factors impact the quality of an evaluation. To get a sense of this, we will ask respondents to answer these questions in advance. These questions are provided by Pegasystems:

- What line of business do you work for? Are you more on the business or IT side?

- Do you have any previous experience with Pegasystems?
- Are you looking to modernize a legacy piece of technology?

Additionally, we will ask the standard questions for generating a Blueprint, which are:

- Industry
- Sub-industry
- Department
- Application purpose
- Functional description
- Organization name
- Language

This information is necessary to generate a Blueprint through the Blueprint API. Once this information is acquired, the actual Blueprints can be generated.

3.5 Ranking of Blueprints

Once the Blueprints is evaluated, it has to be translated into a score, to quantify subjective evaluation. The Blueprints will be ranked, based on an Elo ranking system for chess players (Elo and Sloan (1978)). This is similar to the system used in the Chatbot Arena ranking system. For each comparison of Blueprints, the user has the following choices:

- A is better
- B is better
- Equally good
- Equally bad

For each step of evaluation, an Elo rating will be initialized. It will be set to 1200, representing an equal starting point. Each comparison will have the following outcome:

- A is better: Result A = 1, Result B = 0
- B is better: Result A = 0, Result B = 1
- Equally good/bad: Result A = 0.5, Result B = 0.5

Then the probability of winning has to be calculated. This is done with the formula:

$$E_A = \frac{1}{1+10^{(R_B - R_A)/400}}, \quad E_B = \frac{1}{1+10^{(R_A - R_B)/400}}$$

where R_A and R_B are the Elo ratings of the first and second configurations, respectively. Ratings will be updated using the formula:

$$R'_A = R_A + K \cdot (Result - E_A), \quad R'_B = R_B + K \cdot (Result - E_B)$$

Where K = 32, which is suitable for moderate adjustment (Elo and Sloan (1978)).

The Blueprints generated by the Blueprint API are assigned an *experimentID* and *variantID*, which will be used to store the Elo score. However, the API does not provide the actual meaning or description of these IDs. During the analysis phase, Pegasystems will provide a list detailing the meaning of each *experimentID* and *variantID*. This information will be added after the user ranking process.

Every time a Blueprint is evaluated, the ranking of the Blueprint experiment and variant will be updated. If a significant number of users have evaluated the Blueprints, it will give a good view of the performance of the variants. The overview will be displayed in a similar style as the Chatbot Arena ranking page (see: Figure 2). The ranking page will display an overview for the total ranking of the Blueprints, which is the sum of the separate rankings for each page. It will also be possible to show the ranking for a specific Blueprint phase.

3.6 Implementation

The developed software integrates both the back-end and the front-end components to provide a seamless experience in the evaluation process.

3.6.1 Back-end Architecture

The back-end is implemented using the python library Flask. This is a lightweight web framework. It is designed to handle API request from the front-end. Secure communication between the front-end and back-end is highly important. To enable this secure communication, Cross-Origin Resource Sharing (CORS) is used. The primary tasks of the backend include generating the Blueprints and updating the score through an Elo system. These generated Blueprints are based on user-defined criteria such as industry. The Elo score is updated on the basis of user input.

The tool integrates directly with the Pega Blueprint API. The back-end queries twice per evaluation session to generate the two different Blueprints using different model configurations. This allows for direct generation of the Blueprints. The back-end generates two Blueprints by calling this API with slightly varied parameters after receiving the user input. The Blueprints are then randomly displayed to minimize the potential bias in the evaluation.

3.6.2 Front-end Architecture

The front-end is implemented in a React.js application. This provides a dynamic, user-friendly interface for interactions of the evaluation tool. React.js is suitable for fulfilling the following requirements:

- **User Input:** The selection of specific requirements prior to generating the Blueprints, such as industry.
- **Dynamic Visualization:** A real-time scoreboard displays the rankings of the Blueprint models, inspired by the Chatbot area.
- **Responsiveness:** The tool is designed to be responsive. This ensures a smooth experience through different devices.

The interface allows users to input their specific parameters for Blueprint generation. The generated Blueprints are displayed in random order side by side. Side-by-side comparison is used to minimize biases, improve clarity, and allow users to easily make comparisons between the different Blueprints. The front-end includes a feedback mechanism for users to indicate their Blueprint evaluation. The scoreboard component dynamically updates the rankings based on the back-end Elo score calculations. This provides users with immediate information on how their evaluations impact overall rankings.

4 RESULTS

This research resulted in the creation of an evaluation tool for Pega Blueprints. The software ranks the models used for the blueprints, using an Elo rating. This method is inspired by methods used in chess rankings. This method allows for the creation of a numerical score from subjective human judgment. The reliability and stability of the scores will increase with time as more evaluations are made.

4.1 Deliverables

The following key outcomes were delivered:

- **A modular evaluation tool:** The tool allows the user to qualify different blueprints created by randomly assigned blueprint models.
- **Dynamic Visualization:** The evaluation results are presented dynamically, similar to Chatbot Arena. This enables stakeholders to identify trends
- **User-Centric Design:** The tool allows for user feedback, to collect more details about the decision made.
- **Quantifiable Feedback**: Subject preferences are converted to numerical Elo ratings to provide actionable insight to optimize the Pega Blueprint configuration.

4.1.1 Usage Instructions

To utilize the tool:

1. **Generate the blueprints**:
    - Choose between the different options for the required field such as industry, sub-industry, application purpose, and other parameters through the interface of the web application.
    - Two blueprint variants are generated based on the user input of two different blueprint models. These blueprint models are randomly assigned to prevent bias; see 6 in the Appendix.
2. **Compare and evaluate:**
    - Evaluate the two differently generated blueprints by providing feedback for the different stages. For each stage of the blueprint, the user should provide their feedback by choosing the best-fitting statement; see 8 in the Appendix. The options are as follows:
        - First Better
        - Second Better
        - Both Good
        - Both Bad
    - The user has the option to elaborate on its choice by filling in the open text form.
3. **Review Scores:**
    - The system updates the Elo ratings dynamically based on the user input. This input is reflected in the updated ranking in the scoreboard; see 7 in the Appendix.

This tool provides Pegasystems with a scalable framework to continually test and improve the quality of its blueprint models.

5 DISCUSSION

This study explores the development of an evaluation framework for Pega Blueprints, drawing heavily from the methodologies implemented by UC Berkeley, Stanford University, and UCSD in Chatbot Arena (Chiang et al. (2024)). Despite significant progress in defining the conceptual foundation and creating a design for the evaluation tool, as well as its development, several aspects require reflection and consideration.

5.1 Improvements

One of the project requirements was the creation of a modular framework to make subjective evaluations possible based on various criteria. Although the theoretical approach outlines this flexibility, the practical implementation remains to be tested. An adjustable and flexible design will be crucial to supporting future use cases or additional evaluation criteria. The effectiveness of this modularity will depend on the thorough testing of the user and continuous improvement.

Furthermore, the tool's success will also depend on its usability and the willingness of Pegasystems' customers, which are eventually/ultimately the target audience, to engage with it. The initial design is based on Pegasystems' already existing Blueprint interface, however, usability testing is essential to identify potential bottlenecks. Engaging users to provide meaningful evaluations, especially for Pegasystems' long multiphase workflows, may require some creative incentives or a careful selection of the users.

There is also potential in improving the Elo system itself. Currently, the framework treats the evaluation choices ”Equally good” and ”Equally bad” as equivalents in the ranking system. Although this approach simplifies the scoring process, it overlooks the contextual differences between these two scenarios.

When users indicate that two outputs are ”Equally good”, it means that both outputs meet or exceed expectations, potentially suggesting good performance from the models. In contrast, choosing the ”Equally bad” option implies that both have failed to meet the evaluators' expectations, indicating that there is room for improvement in certain areas. Therefore, by giving the same score to those outcomes, the tool loses the ability to differentiate between cases where the tested models or tuned parameters perform equally well compared to equally poor. The absence of this differentiation can weaken the precision of the gathered results and obstruct the more in-depth analysis that follows.

For instance, understanding how frequently outputs are considered to be ”Equally bad” could aid in identifying systemic problems within the Blueprint generation process, such as limitations in certain model configurations or misalignment with user expectations. However, tracking instances of ”Equally good” can highlight areas of consistent success and provide information about the best practices for future iterations.

To address this issue, a revision of the current ranking algorithm would be needed to handle these two evaluation choices separately. For example, assigning distinct weights to each category would allow the system to represent their varying significance. This adjustment would allow Pegasystems to gain deeper insights from the evaluation process, which would put more focused model optimization and ultimately improve the quality of the produced Blueprints.

5.2 Limitations

5.2.1 Insufficient details

One of the significant limitations faced during this research was the limited and incomplete disclosure of the data and information provided. Although it was preferred to evaluate and compare different configurations of the Blueprints to fully test the evaluation tool, the scope of the changes that could be applied was not fully disclosed. We were informed solely on the two main variables for evaluation: the changes in de underlying LLM used in the generation process of a Blueprint and the adjustments to hyperparameters. An example of the former would be the use of GPT-4o, 4o-mini and 3.5 Turbo, as well as the switch to Haiku. When it comes to the latter, only the temperature parameter was mentioned. Beyond these, no additional details about other potential variables that influence the output of the Blueprint were provided, such as specific training data sets or fine-tuning strategies. This will be done by Pegasystems' professional team.

5.2.2 Reliance on sufficient evaluations

A limitation of the evaluation tool is its reliance on a sufficient number of evaluations to produce meaningful and accurate results. The precision of the Elo-based classification system depends on consistent and numerous comparisons between variants, as a normal distribution is present (Elo and Sloan (1978)). When some outputs or configurations receive fewer evaluations, their ratings may not be able to accurately depict their true performance. This may skew the overall ranking and insights. This limitation can lead to biased or incomplete conclusions, especially if some outputs are overlooked or not evaluated with the same frequency as others.

5.2.3 Generalization

A major limitation is the generalization involved, as users' prior knowledge, experience, or personal preferences have not been taken into account. These factors can influence how users perceive and interact with the Blueprint. For example, a user might prefer a specific workflow configuration or approach based on familiarity, even if an alternative option has objectively better efficiency or accuracy. By not taking these elements into account, the evaluation tool risks overlooking user satisfaction or the practical usability of the tool in real-world scenarios; therefore, the Blueprint's objectivity might suffer. To mitigate this, Pegasystems could include a diverse range of users with diverse backgrounds, roles, and expertise to ensure a representative data set. Adjusting for biases using statistical methods, such as normalizing scores, could also improve the fairness of evaluations. Grouping users based on shared preferences could allow for more personalized configurations, and providing feedback to users on how their evaluations compare with others might encourage more consistent and objective scoring.

5.3 Future Research

5.3.1 Comparing performance of different LLMs

Using the created evaluation tool, Pegasystems can compare the outputs of various LLMs to determine which generates the most effective Blueprint or Blueprint element. This analysis was not feasible due to the limited scope of the disclosed variables, as detailed in ***5.2.1 Insufficient details***. By presenting users with outputs from different models, such as OpenAI's GPT-4o, 4o-mini, and 3.5 Turbo as well as various LLMs from other companies, and using pairwise comparisons, the tool's scoring system can rank the models in terms of preference and quality. This analysis allows Pegasystems to understand how each model performs in generating their Blueprints. Over time, this analysis could help Pegasystems in selecting the optimal LLM for their Blueprint or Blueprint element generation and in identifying areas where additional training is needed.

5.3.2 Hyperparameter tuning

Pegasystems will be able to use the evaluation tool to explore the impact of hyperparameter adjustments on the quality of the generated Blueprints. For example, the temperature parameter influences the creativity and coherence of the output (Wang et al. (2024)), while token limits affect the length and detail (Levy et al. (2024)). By systemically testing the generated outputs with varying parameter settings and having humans rank these variations, the tool can bring optimal configurations for specific use cases to light.

5.3.3 Evaluation across industries and use cases

The needs of the Blueprint users vary between departments and between industries. Using the developed evaluation tool, Pegasystems can analyze how well its Blueprint delivers to those specific domains. By dividing evaluation results based on the segmented users' industry or use case, Pegasystems can directly identify patterns in preferences, common pain points, or even requirements that are specific for a certain industry. Insights from this analysis would enable Pegasystems to fine-tune the generation of Blueprints for domain-specific accuracy and utility.

5.3.4 Custom configurations for clients

A step further would be to make custom configurations per client. Different clients have different needs and preferences even within the same industry (Makki et al. (2018)). Pegasystems can use this evaluation tool to help them tailor the outputs in order to meet these unique and specific requirements. By anonymously collecting feedback from individual clients, Pegasystems could possibly identify patterns in their preferences and then recommend customized configurations of the Blueprint generation process.

5.3.5 User preference trends analysis

Understanding user preferences is important for improving the Blueprint generation process. After collecting and analyzing the feedback through the evaluation tool, Pegasystems can also identify broader trends in what users value most in their Blueprints. For example, users may be more in favor of concise and structured outputs than those that are lengthy and detailed. This way, the generalization limitation could be addressed to some extent.

6 CONCLUSION

This research focused on evaluating the quality and performance of AI-generated content, specifically focusing on Pega Blueprints. Pegasystems, using generative AI technologies, faced difficulties in standardizing the evaluation of their Blueprint models, with subjective judgments playing a significant role in determining the effectiveness of different configurations.
This section presents the achievements of the research, the results derived from the evaluation process, and potential avenues for future work.

6.1 Achievements and results of the research

The primary objective of this research was to design and implement an evaluation tool that could provide a standardized, objective, and scalable method to evaluate Pega Blueprints. However, achieving this goal required addressing several key challenges, such as minimizing bias in human judgments and ensuring that the tool is scalable for continuous future improvement.

The main challenge was to identify a way in which the system could collect human feedback in a clear and structured way without compromising on flexibility. This was achieved by implementing a side-by-side comparison approach for the development of the evaluation tool. By placing two different Blueprints next to each other, the tool minimized bias that is associated with ordering or stacking comparisons. As such, users can provide feedback on which blueprint they prefer, or rate them as ’equally good’ or ’equally bad.’ This helps Pegasystems identify which configurations are more successful in meeting user needs.

Furthermore, the tool provides the flexibility to collect additional qualitative data through open text boxes, allowing users to explain the rationale behind their choices. This addition was needed for capturing nuanced insights into the users’ preferences that could inform Pegasystems on the spots for improvement of the Blueprint generation process.

In addition to the side-by-side placement mentioned above, other measures were incorporated to minimize bias. For example, the random assignment of Blueprints to each evaluation session also minimized selection bias, ensuring that each model had an equal chance of being evaluated in any given session. Moreover, to ensure that user feedback was as unbiased as possible, the tool did not include pre-defined rankings or instructions that could lead to unfair selection of the user. Additionally, no numerical values were used for depicting the different Blueprints to mitigate the options ’1 is better’ or ’2 is better’ that might suggest any intrinsic superiority. Instead, users are presented with neutral feedback options like ’A is better’ or ’B is better’ or the previously mentioned options for equal standings. This design choice provides a more objective evaluation environment.

To ensure that the evaluations were quantifiable, the research aimed to translate subjective human preferences of the users into numerical scores. This has been accomplished by integrating the Elo rating system. The Elo system provided a mathematical solution to calculate the relative performance of two competing Blueprints and, therefore, the performance of the underlying AI models.

The implementation of this Elo ranking system allowed for a relatively reliable yet dynamic ranking of the Blueprints, which should improve as more users provide their feedback. This will ensure that the scoring is not static and will evolve over time, reflecting the accuracy of users' preferences with more reliability and stability as the number of evaluations increases.

As such, key deliverables include a modular evaluation tool and a user-centered design to minimize bias in the evaluation process. The flexibility of the tool provides adaptability to future updates in Pegasystems' models or even additional evaluation criteria. Furthermore, Pegasystems can improve its Blueprint generation process and choose the most suitable AI models over time through the collection and analysis of user feedback.

6.2 Future work

Future work could focus on expanding the comparison of AI models to incorporate a wider range of LLMs from various sources. By broadening the model comparison scope, Pegasystems can gain a better understanding of the strengths and weaknesses of each model and fine-tune their Blueprint generation accordingly. This will also enable Pegasystems to evaluate the performance of LLMs across various domains, ensuring the selection of the most optimal models for specific use cases and tasks. Furthermore, the tool's ability to evaluate outputs across different LLMs offers the opportunity to explore the tuning of hyperparameters in more detail. Future versions could focus on testing additional hyperparameters to optimize performance. By adjusting these parameters and analyzing the results, Pegasystems can refine its Pega Blueprint to generate more effective outputs for specific use cases as well. Other future work could include custom configuration for clients, as well as usability testing and user experience enhancement.

REFERENCES

Akgonul, K. (2024). Pega genai blueprint is changing how enterprises approach transformation. Accessed: 2024-12-20.

Chang, Y., Wang, X., Wang, J., Wu, Y., Yang, L., Zhu, K., Chen, H., Yi, X., Wang, C., Wang, Y., et al. (2024). A survey on evaluation of large language models. *ACM Transactions on Intelligent Systems and Technology*, 15(3):1–45.

Chiang, W.-L., Zheng, L., Sheng, Y., Angelopoulos, A. N., Li, T., Li, D., Zhang, H., Zhu, B., Jordan, M., Gonzalez, J. E., et al. (2024). Chatbot arena: An open platform for evaluating llms by human preference. *arXiv preprint arXiv:2403.04132*.

Elo, A. E. and Sloan, S. (1978). The rating of chessplayers: Past and present. *(No Title)*.

Humphries, A., Chen, Z., and Cave, K. R. (2021). Both feature comparisons and location comparisons are subject to bias. *Attention, Perception, Psychophysics*, 83(4):1581–1599.

Levy, M., Jacoby, A., and Goldberg, Y. (2024). Same task, more tokens: the impact of input length on the reasoning performance of large language models.

Makki, M., Landuyt, D., Lagaisse, B., and Joosen, W. (2018). A comparative study of workflow customization strategies: Quality implications for multi-tenant saas. *Journal of Systems and Software*, 144.

Radford, A. (2018). Improving language understanding by generative pre-training.

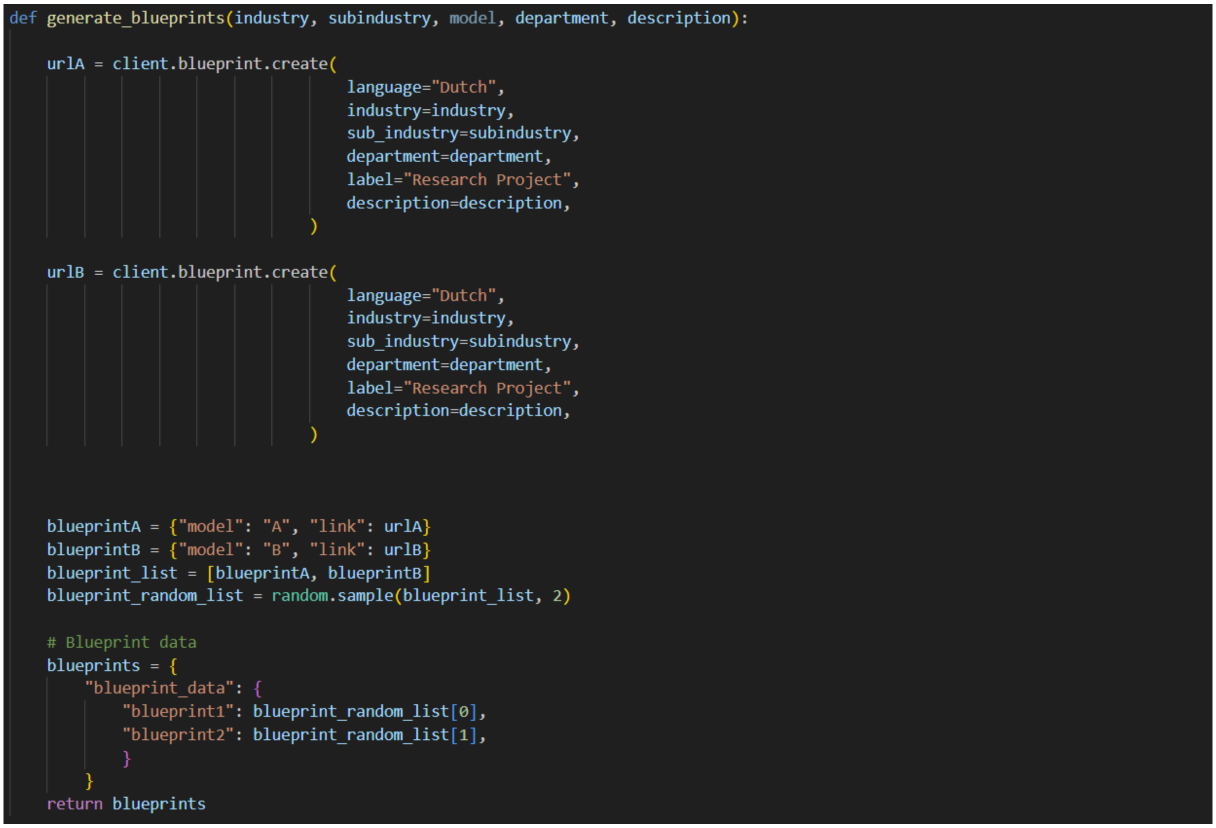
Wang, Y., Yu, Z., Zeng, Z., Yang, L., Wang, C., Chen, H., Jiang, C., Xie, R., Wang, J., Xie, X., et al. (2023). Pandalm: An automatic evaluation benchmark for llm instruction tuning optimization. *arXiv preprint arXiv:2306.05087*.

Wang, Y., Zhang, Z., Chen, H., and Shen, H. (2024). Reasoning with large language models on graph tasks: The influence of temperature. In *2024 5th International Conference on Computer Engineering and Application (ICCEA)*, pages 630–634.

Yang, S., Chiang, W.-L., Zheng, L., Gonzalez, J. E., and Stoica, I. (2023). Rethinking benchmark and contamination for language models with rephrased samples.

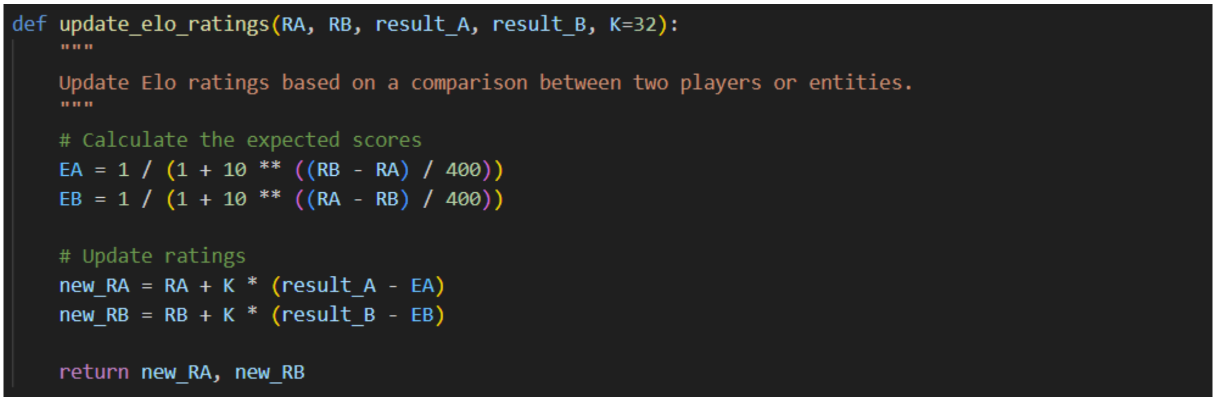
Zheng, L., Chiang, W.-L., Sheng, Y., Zhuang, S., Wu, Z., Zhuang, Y., Lin, Z., Li, Z., Li, D., Xing, E., Zhang, H., Gonzalez, J. E., and Stoica, I. (2023). Judging LLM-as-a-judge with MT-bench and chatbot arena. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*.

7 APPENDIX



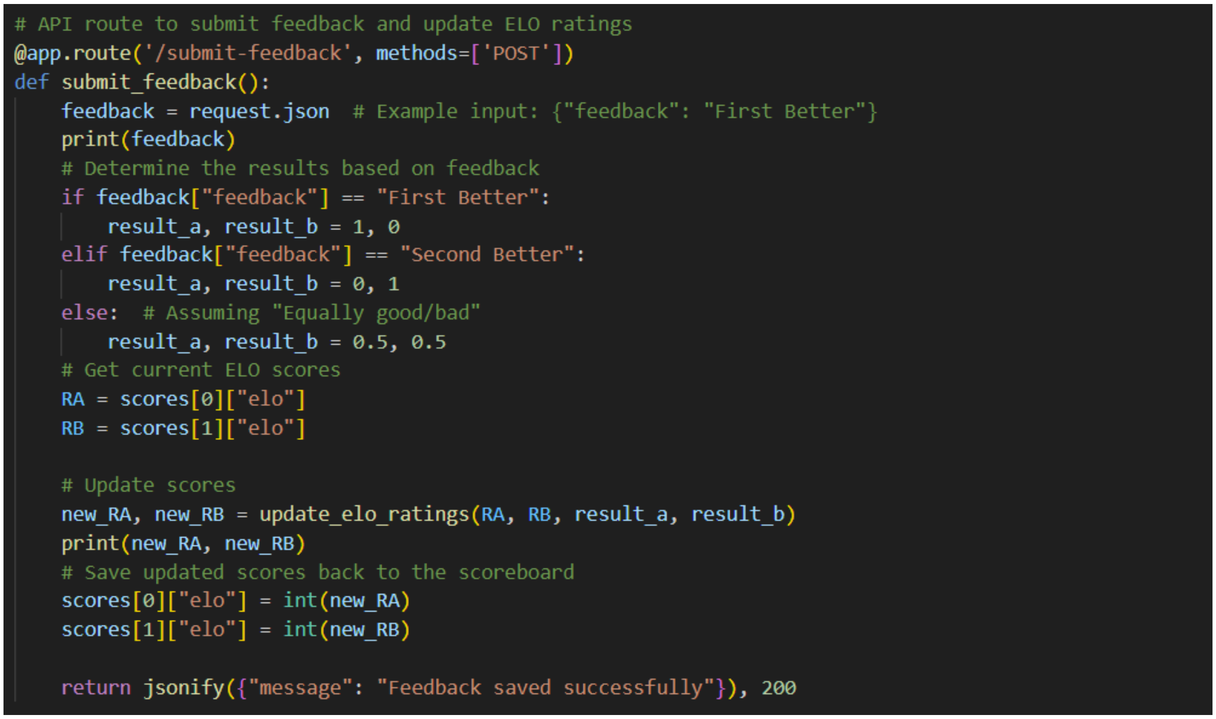
```python
def generate_blueprints(industry, subindustry, model, department, description):

    urlA = client.blueprint.create(
                                    language="Dutch",
                                    industry=industry,
                                    sub_industry=subindustry,
                                    department=department,
                                    label="Research Project",
                                    description=description,
                                )

    urlB = client.blueprint.create(
                                    language="Dutch",
                                    industry=industry,
                                    sub_industry=subindustry,
                                    department=department,
                                    label="Research Project",
                                    description=description,
                                )



    blueprintA = {"model": "A", "link": urlA}
    blueprintB = {"model": "B", "link": urlB}
    blueprint_list = [blueprintA, blueprintB]
    blueprint_random_list = random.sample(blueprint_list, 2)

    # Blueprint data
    blueprints = {
        "blueprint_data": {
            "blueprint1": blueprint_random_list[0],
            "blueprint2": blueprint_random_list[1],
            }
        }
    return blueprints
```

Figure 6. python function generate_blueprints



```python
def update_elo_ratings(RA, RB, result_A, result_B, K=32):
    """
    Update Elo ratings based on a comparison between two players or entities.
    """
    # Calculate the expected scores
    EA = 1 / (1 + 10 ** ((RB - RA) / 400))
    EB = 1 / (1 + 10 ** ((RA - RB) / 400))

    # Update ratings
    new_RA = RA + K * (result_A - EA)
    new_RB = RB + K * (result_B - EB)

    return new_RA, new_RB
```

Figure 7. python function update_elo_ratings



```python
# API route to submit feedback and update ELO ratings
@app.route('/submit-feedback', methods=['POST'])
def submit_feedback():
    feedback = request.json  # Example input: {"feedback": "First Better"}
    print(feedback)
    # Determine the results based on feedback
    if feedback["feedback"] == "First Better":
        result_a, result_b = 1, 0
    elif feedback["feedback"] == "Second Better":
        result_a, result_b = 0, 1
    else:  # Assuming "Equally good/bad"
        result_a, result_b = 0.5, 0.5
    # Get current ELO scores
    RA = scores[0]["elo"]
    RB = scores[1]["elo"]

    # Update scores
    new_RA, new_RB = update_elo_ratings(RA, RB, result_a, result_b)
    print(new_RA, new_RB)
    # Save updated scores back to the scoreboard
    scores[0]["elo"] = int(new_RA)
    scores[1]["elo"] = int(new_RB)

    return jsonify({"message": "Feedback saved successfully"}), 200
```

Figure 8. Python function submitfeedback