Leveraging Large Language Models to Enhance User Experience Correlated to Experiment Metrics for E-commerce Notifications

EMILY HA AND HANNIBAL LIND
BACHELOR´S THESIS
DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

Bachelor's Thesis

Leveraging Large Language Models to Enhance User Experience Correlated to Experiment Metrics for E-commerce Notifications

By

Emily Ha and Hannibal Lind

Supervisors:

Ikea: Tobias Friberg, LTH: Christin Swahn

(Template by Peter Nilsson, modified by Christian Nyberg)

Department of Electrical and Information Technology
Faculty of Engineering, LTH, Lund University
SE-221 00 Lund, Sweden

Abstract

Large Language Models are commonly used in the present digital world and the usage of them continue to grow at a fast pace. This thesis investigated the possibility of leveraging Large Language Models to summarize nudge experiment result metrics within IKEA's internal nudging framework portal. The research focused on evaluating the prompt engineering techniques: Zero-Shot, Few-Shot, and Role Prompting, to determine which approach that enhances the user experience the most. A prototype with the LLM summarizing feature was implemented and integrated into the IKEA self-service portal. The prototype used the Gemini 2.0 Flash-Lite model and was accessed through Google's Vertex AI API. Moreover, the prompting techniques were evaluated on comprehensiveness, helpfulness and decision helping, through a usability test. The results showed that Few-Show prompting was interpreted as the most effective technique across all categories. In addition, Role Prompting showed some strength in its ability for highlighting key points. Zero-Shot prompting showed to be the technique that generated the least effective response. The findings suggested that the most effective prompting technique include structured and contextual examples.

Keywords

Large Language Model, Gemini, Vertex AI, Prompting techniques, Nudge and Metrics.

Sammanfattning

Språkmodeller används flitigt i den nuvarande digitala världen. Detta examensarbete fokuserade på att undersöka hur språkmodeller kan användas för att sammanfatta nudge experiment-resultat data för IKEAs interna portal. Arbetet fokuserade på att evaluera de olika prompt-teknikerna: Zero-Shot, Few-Shot och Role Prompting, för att avgöra vilken som bäst förbättrar användarupplevelsen på portalen. En prototyp med sammanfattnings funktionaliteten av en språkmodell implementerades till IKEAs self-service portal. Prototypen använde språkmodellen Gemini 2.0 Flash-Lite via Vertex AI API:et. De olika prompt-teknikerna evaluerades genom ett användbarhetstest, där deras genererade svar bedömdes utifrån hur omfattande, hjälpsamma och beslutsstödjande de var. Slutligen visade resultaten att Few-Shot prompt-tekniken var den mest effektiva i samtliga kategorier. Role Prompting-tekniken visade styrkor i dess förmåga att lyfta fram nyckelpunkter, medan Zero-Shot prompt-tekniken genererade de minst effektiva svaren. Studiens resultat tyder på att de prompt-tekniker som inkluderar strukturerade kontextuella exempel är mest effektiva i att generera svar som förbättrar användarupplevelsen.

Nyckelord

Språkmodeller, Gemini, Vertex AI, prompt-tekniker, Nudge.

Acknowledgments

Foremost, we would like to extend our sincere gratitude to our supervisor at IKEA, Tobias Friberg, for his continuous support and encouragement throughout this work. We also wish to thank the Nudging Framework team at IKEA for their interest and willingness to support our research. In addition, we would like to give out gratitude to the Customer Order 360 team for being supportive in the beginning of our journey. Lastly, we would like to express our appreciation to our LTH supervisor, Christin Swahn, for her valuable guidance and support.

Contents

1. Introduction

This chapter introduces the context and motivation for the thesis. It presents background of the problem, purpose of the thesis, research questions and background on the IKEA team that the thesis was conducted under.

1.1. Background

This thesis researches the possibility to leverage a Large Language Model (LLM) to enhance the user experience by generating summarizations of experiment result data from a nudge on one of IKEAs internal framework websites.

A nudge in this context refers to a user interface component, such as a notification or prompt that appears on the IKEA website. A nudge is created through a self-service portal and used mostly by e-commerce and marketing teams. The self-service portal is a website developed by the Nudging Framework team, where they host their nudge framework tool for other internal IKEA users to use. The self-service portal enables creation of nudges, selection of metrics to view during the experimentation and finally lets the user deploy if it is deemed a success.

Before deploying a nudge to the IKEA website, an experiment is conducted to collect performance metrics on how the nudge performs. Based on the experiment's performance metrics the nudge creator decides whether to deploy the nudge to the IKEA website or not. Analyzing the raw metrics can be complex and time-consuming for the nudge creators.

This thesis research different prompting techniques to generate concise summaries on nudges experiment result metrics using an LLM. Each prompting techniques output will be evaluated through user testing. The goal of enabling summarization of the metrics is to improve interpretation and enhance user experience based on the categories of comprehensiveness, helpfulness and decision support.

1.2. IKEA Nudging Framework Team

The Nudging Framework team at IKEA develops and maintains the self-service portal that enables users to create and deploy nudges to the IKEA website. The team make sure that the platform allows users to customize nudge design, define the behavior of the nudge, deployment of nudge and overall management of the nudges.

Beyond managing the portal, the Nudging Framework team is responsible for storing and processing experiment data and data tied to the nudges performance.

1.3. Purpose

The purpose of this thesis is to explore the possibility of using a Large Language Model (LLM) to generate a description of the metrics that a nudge generates in its experiment phase on IKEA's websites. To then see how the generated description can be integrated into the Nudging Framework team's portal, to make the performance of the nudge during its experiment phase easier to get a grasp on for the nudge creator. The expected result of this thesis is to have an LLM generate a description of the metrics on the Nudging Framework portal, with the aim of providing the nudge creator with a clearer and easier understanding of the metrics of the nudge.

1.4. Goal and objectives

The goal of the thesis is to explore the possibility of using an LLM to generate a description of the metrics of a nudge, using prompt engineering. The aim is to create accurate and concise summaries that improve the readability and interpretation of the performance metrics for nudge creators. To achieve this, different prompt techniques will be evaluated to identify the most effective approach for generating summaries. The generated summaries will be assessed through user testing.

1.5. Problems to Research

The hypothesis in this thesis is that a LLM summarization of nudge experiment result helps the user in decision making.

To research this hypothesis, the following research problems are defined:

1. How can a Large Language Model be utilized to produce summarization descriptions of a nudge's experiment result?
2. What is needed to develop a prompt template?
3. Which prompting technique enhances the user experience on the portal the most based on comprehension accuracy, decision support and summary helpfulness.

1.6. Motivation

The motivation and reasoning for this thesis topic stems from the opportunity to gain hand-on experience with the expanding topic of ai and machine learning, specifically large language models and prompt techniques, while simultaneously giving an opportunity to expand our knowledge in full-stack development. The thesis offers a chance to bridge our academic learning with a practical implementation in a real-world scenario. Being able to conduct the thesis in collaboration with IKEA is particularly suitable for several reasons. One major reason being that IKEA is a large-scale operating company and provide an ideal setting to explore how large language models can effectively summarize and present large metrics in a user-friendly format. The other major reason being that previous exposure to IKEA's operational environment provides a head start in understanding some of the challenges and requirements for the thesis. The rewarding aspect of conducting it at IKEA is that it will be on a large operation scale and have the potential to make meaningful impact for the creators of the nudges on the Nudging Framework portal.

For IKEA the motivation for this thesis is to get a more concise description of the metrics, this to make it easier for the nudge creator to understand the gathered metrics easier. Not only that but also to gain further knowledge within IKEA around the topic of LLM and prompt engineering. To be able to see more ways that LLMs and prompt engineering can be used within the company, particularly regarding making technical parts more easily readable by using LLMs. IKEA works towards having as many of its techniques

developed in-house, which presents us with the opportunity to explore the usage of a LLM within their nudge metric systems.

From a societal perspective the thesis is motivated by making metrics of a nudge or a similar product easier to read and understand, by being able to utilize an LLM and prompt engineering. By having a summarizing description of the metrics, which would make understanding how a nudge performed easier. This could be further developed and implemented for other types of systems where you evaluate metrics.

1.7. Delimitations

The scope of this thesis is restricted to the IKEA Nudging Framework team's nudge portal and its associated metrics and backend system. The focus is solely on the development and integration of using prompt engineering on a LLM to generate a summarizing description of the metrics of a nudge.

1.8. Division of labor

Through the thesis both authors have been working equally and mostly together throughout each phase, as indicated in Table 1. Having worked mostly onsite together at the IKEA office. While working from home discussions about implementations and help was done through messages and video call session.

TABLE I. DIVISION OF LABOR.

Author	Emily Ha	Hannibal Lind
Local Development	50	50
Portal Integration	50	50
Usability Test	50	50
Report	50	50

2. Theoretical Background

This chapter includes the key concepts and technologies relevant to the thesis. Explaining Large Language Models and prompt engineering techniques. Additionally, introduces Google Cloud Platform and Vertex AI. The chapter also explains what a Nudge is and briefly about metrics used in the thesis, it also introduces Optimizely, Local Development and what a Controller-Service based backend system is. Additionally, it describes some fundamentals about evaluation of AI generated texts. In addition, it also describes briefly some frontend components and general technologies used in this thesis.

2.1. Artificial Intelligence Models

Artificial Intelligence (AI) models are programs that are trained to perform tasks that simulate the human brain such as that predict, recognize, and make decisions without human intervention. AI models are capable of analysis of large datasets, learn from data, predictions based on learned information [1].

2.2. Large Language Models

Large Language Models (LLMs) are a specific type of Artificial Intelligence program trained with a large set of data to be able to perform tasks related to human language. By analyzing large amounts of data, LLMs learn to recognize patterns, predict word sequences, generation text for the given input context, content summarization, code generation, sentiment analysis, language translation and more [2]. Large Language Models are used in this thesis to enable summarization text on IKEAs internal portal.

2.2.1. Large Language Model Inference

Large Language Inference is about using Large Language Models to generate human like responses. Summarization is a key task in Natural Language Processing. LLMs are vastly used for their ability to summarize a large amount of textual information into a concise

shorter version. It helps with handling an overload of information and can speed up and enhance decision-making. LLMs are ideal for summarization as it is capable of understanding the context of large inputs, generates output summaries that with a human tone and can handle different types of input such as text and data.

The text provided as input to the LLM is called a *prompt*. The LLM based on the prompt will then generate a response known as the *completion* [3]. A pre-defined structure for a *prompt* is called a prompt template, which allows for dynamic substitution of variables with specific values within the prompt. The LLM processes input prompts, a process named tokenization. Tokenization is the process where input text is divided into smaller tokens which is smaller units that can consist of a word or subworld. Tokenization of the input helps the LLM to handle and process input more effectively. Once tokenized, the tokens are encoded into a numerical value in the form of vector that the LLM understands. The LLM will analyses the relationship between tokens through their transformer layers to generate a relevant output response [2].

2.2.2. Implementing Large Language Model Applications

In today's digital world it is common to utilize LLMs through Application Programming Interfaces (API) [3]. In this thesis the utilization of LLMs is researched to see how it affects the user experience on the IKEA system when summarizing a nudge's experiment metrics.

2.3. Google Cloud Platform

Google Cloud Platform (GCP) is a cloud computing service provided by the technology company Google. GCP provides tools for computing, storage, networking, machine learning and artificial intelligence. GCP enables developers to run, develop and manage applications that are scalable.

For artificial intelligence (AI) and machine learning (ML) applications, GCP provides a range of services. The one used in this research is Vertex AI

2.3.1. Vertex AI

Vertex AI is a machine learning platform on GPC [4], used to build, train, deploy and manage ML models. The platform provides access to pre-trained Large Language Models. These LLMs are optimized for NLP tasks.

This thesis utilizes Vertex AI and its LLM interference to generate summaries for the nudge performance metrics. Instead of training a model from scratch, pre-trained LLMs on the platforms are evaluated and selected through Vertex AI's API.

Vertex AI API

Vertex AI API is a cloud service that allows developers to utilize Google's pre trained models by sending API requests and getting a response form the LLM service. Vertex AI API can be accessed through different libraries such as C#, Go, JavaScript and Python. To maximize the Vertex AI API, Google authentication, the LLM model name, an API key, and a request body containing the prompt are required in the API request. If the API request is successful, the Google server with the LLM will send back an API response containing the generated summarized response.

In this thesis the Vertex AI API is implemented in the local test environment using the Python Library and in the IKEA portal's code base using the JavaScript library. To enable sending in prompts containing the instruction to summarize a nudge's experiment result data and to get the LLM's summarized text response that will be integrated into the portal's user interface [5].

Vertex AI API Call

Interaction with the Vertex AI API was achieved through the function *model.genereateContent(prompt)* [6]. This function call comprises two parts: the *model* and the function *genereateContent(prompt)*. The model is a variable holding the object instance of the selected language model. g*enerateContent(prompt)* is a method that operates on the *model* and requires a *prompt* argument, which contains the instructions and data for the LLM.

2.3.2. Gemini

Gemini is Google's main line of LLMs offered through their Vertex AI platform. As of the time of writing this thesis, Google provides three different Large Language Models: Gemini 2.0 Pro, Gemini 2.0 Flash and Gemini 2.0 Flash-Lite. This thesis leverages the Gemini 2.0 Flash-Lite model, due to its cost-efficiency and applicability for summarization tasks [7].

Gemini 2.0 Flash-Lite

The LLM Gemini 2.0 Flash-Lite is designed to be cost-efficient. The model can process sequences of input up to one million tokens. Prompt inputs with larger input sizes require a model that has a higher capacity for input size limit. Additionally, the model is limited to generating text output. The text output's length limit is eight thousand tokens. The Gemini 2.0 Flash-Lite model has a knowledge cutoff as of June 2024, meaning the model's training data does not include any events or information past the point of June 2024 [8].

2.4. Prompt Engineering

This section will explain briefly what prompt engineering is and why it is used when prompting an LLM. It also includes brief descriptions of some prompting techniques that were used in this thesis.

2.4.1. Definition of Prompt Engineering

Prompt engineering is the technique of using optimized prompts to guide LLMs, to generated desired responses [9], [10]. It involves using specific strategies to help the LLM to understand the input to be able to generate the desired output. Furthermore, prompt engineering allows the focus to be spent on the desired output than finetuning a model for a desired task. A valuable concept in prompt engineering is implementing the technique into a prompt template. This is done by structuring the prompt using the technique by including certain instructions, context and examples that guide the LLM to prevent misinterpretation and hallucination. [9], [10], [11].

2.4.2. Prompting Techniques

There is a various amount of prompting techniques that can be used to effectively prompt an LLM or AI model. These techniques differ from each other and enhance the prompt differently, but all have the common goal of making an efficient prompt that guides the LLM or AI model towards a specific outcome.

Zero-Shot

Zero-Shot prompting is a prompting technique that involves providing an LLM with a question or with instruction without providing any examples of the output. The technique relies on the LLM and its pre-trained knowledge to understand the instructions given and generate a response from those. In general, the technique functions based on the LLM already having enough general understanding to comprehend the data and the instructions given [12], [13].

Few-Shot

Few-Shot prompting is a prompting technique that at its core, resembles a Zero-Shot prompt, but has enhanced the prompt by providing examples of the output or the data that the LLM is to comprehend. The examples given to the LLM demonstrate how the output should look like and how the input should be formatted in terms of style, pattern and format. The examples are used to teach the LLM the specific requirements of a task, to perform better without having to update the underlying parameters of the model [14].

Role Prompting

Role prompting is a prompting technique that makes the LLM assume a specific role, or viewpoint before it should answer the question or perform the task given [15]. By giving the LLM a specific role, the LLM response will change in its style, tone, focus of what it deems important in the information given, depending on what the LLM determines the roles prioritize being. The LLM leverages its training to determine how it should interpret the role given.

2.5. Nudge

A nudge in this thesis, refers to a user interface (UI) component that appears on the IKEA website. The nudges are used to guide users towards certain business and sales-related actions, such as purchasing a product or completing a checkout process. Within IKEA's internal systems, there is an internal self-service portal where users such as e-commerce and marketing workers can create nudges.

The creation of a nudge on the self-service portal includes tailoring the design of the nudges - such as color, header text, fonts, images and buttons. In addition, the user also defines the logic of the nudge, such as where the nudge should appear, in what domain on the IKEA website it should be present on, based on what events on the website that triggers the nudges appearance and for how long the nudge should be visible on the website for. During the creation a hypothesis is also made that states for what the creator think the nudge will improve or make the customers aware of.

Before a nudge is deployed to the live production website, it goes through an experiment phase to evaluate its effectiveness. During this phase, experiment performance metrics are collected to measure the nudge's impact on user behaviors. It is during this phase where seek to understand if the nudge proves the hypothesis. After the experiment phase, the user can analyze the gathered performance metrics results and draw conclusions whether the nudge should be deployed or not, based on whether the nudge has satisfied its purpose or driving business and sales-related actions.

However, analysis of the experiment metrics result can be complex, as it demands deep understanding and ability to interpret the data points. The complexity can lead to indecisive decision making, especially for users without strong analytical backgrounds. Therefore, this thesis explores how LLM can improve the user's decision making by generating concise summaries of the performance data of the nudge experiment.

2.5.1. Nudge Experiment Metrics

When the nudge is in its experiment phase different metrics are gathered and collected, to give the evaluation of the nudge performance a foundation to stand on. The different sites that a nudge

can be deployed on have different obligatory performance metrics that it keeps track off, these in addition to the metrics that the nudge creator want to track and see how the nudge affect them. The metrics can range from tracking the log in, click through, checkout totals and checkout completion.

The metrics tracks the performance of the page with and without the nudge or different designs of the nudge and tells the improvement or deterioration that the nudge did to the selected metrics. Both to those chosen by the nudge creator and also the specific ones for a site.

2.5.2. Default metrics

Within the nudging framework, certain metrics hold greater significance than others and are deemed essential for consideration when drawing conclusions about the results of an experiment. These metrics are termed “default metrics” and serve as key indicators of a nudge performance, exemplified by metrics such as exit rate and bounce rate.

2.6. Optimizely API

Optimizely is a platform used for the experimentation of websites. It allows developers to experiment different variations of a website and is used to manage the experiments and gives metrics based on the experiment. Developers can manage and extract experiment data through Optimizely's API and additionally the experiments can be managed through an internal dashboard that Optimizely provides [16].

2.6.1. Extraction Function

An extraction function is a method that fetches information based on a given set of parameters, which specifies what information to fetch. The parameters are used to locate the specific information by the API and then return the located data.

The extraction function used during this thesis is the endpoint “/experiments/experiement ID/results” within the Optimizely API. The endpoint contains three parameters: *experiments, experiment ID* and *results.* The *experiments* parameter is used to specify the scope it will extract the information from, in this case the experiments. The

experiment ID parameter is used to specify which exact nudge experiment that it will fetch. The *results* parameter is used to determine what is requested from the specific nudge. These parameters combined leads to the extraction function returning the results of from the nudge experiment.

2.7. Local Development

This methodology involves implementing and testing functionality within a controlled and constraint environment that simulates the final production setting. The controlled and constraint environment can be a computer running the services to emulate the final traffic. The purpose of this methodology is about enabling earlier verification of functionalities by being able to showcase the functionality early in the development. Which enables the developer to earlier identify challenges and constraints during the development [17].

For this thesis the local test development environment consisted of a computer running a website running on a localhost webserver to simulate the portal. That combined with a local backend service to emulate the call to Vertex AI with the Python Vertex AI library.

2.8. Controller-Service backend architecture

The controller-service architecture is pattern that enables the separation of concerns; by splitting up the logic handling in the application. The controller layer handles the incoming HTTP requests, processes inputs and return of the response. The service layer is responsible got handling the communication with external sources such as databases, external backend services or APIs. The separation between the controller and service layer enables a structure for development and ensures scalability and modularity when it comes to backend architecture [18].

2.9. Evaluation of AI-Generated Outputs

In this thesis some evaluation of the output from the LLM has been done. In broad schemes the main method for evaluation was performed by humans, which means that humans would compare and

evaluate the output from the LLM. This is stated to be the standard for evaluating natural language generation tasks [19].

2.9.1. Human Evaluation

Human evaluation is the standard for evaluating the output of an LLM, especially when given a task containing natural language generation. This thesis revolves around these natural language generation tasks, which meant that evaluation performed by humans would be the optimal way to evaluate the output from the LLM. Within the output certain parts are evaluated such as the sentence's readability, the LLM's interpretation of the data and its ability to follow the instructions given.

To evaluate the accuracy of the output, the most efficient way is to compare the data inserted and cross reference it with the generated output, to check for differences or misreadings. In this thesis the evaluation was performed by the authors, which compared the values and data that was included in the prompt to the generated response from the LLM.

2.10. User Experience

This section explains what user experience entails and how it is can be conducted.

2.10.1. Defining User Experience

The term user experience involves how a user interacts with a system or service. It incorporates the user's perception of using the system on different categories depending on the system. For example, user experience of a system could be measuring efficiency, accessibility, usefulness and etc. [20].

2.10.2. User Experience Metrics

Metrics are a way of measuring and evaluating things. User experience metrics refer to metrics that are tied to a user's perception and usage of a product or system. User experience metrics often obtain aspects such as efficient, errors and satisfaction of system usage from a user. Using user experience metrics help pinpoint

useability problems, design alternatives and help guide data-driven decision making throughout product development [20].

2.10.3. Usability Testing

Usability testing is a way of testing the functionality of a project or product. The test observes a real user's interaction and perception of using the product. Usability testing is commonly conducted to gather data for user experience metrics. In this thesis, usability testing will be conducted to get data on which prompting technique that generates the preferred output, based on the different user experience metric categories [21].

2.10.4. Survey for Usability Testing

To gather data on the user's experience a survey can be done to gather data on pre-determined user experience metrics. A survey can be formatted in many different ways. In this thesis the survey will be in the form of a Google form [21, pp. 327-376].

2.11. Frontend components

Modal Component

A modal component is a window that appears on top of the current page. It is used to display information without navigating away from the current site [22].

Segment Controller

A segment controller is an interactive UI component that displays two or more segments. These segments can contain text or images and can be used to as buttons to control web page elements [23].

2.12. Tools used

cURL commands

Command line tools that enable HTTP requests to URLs. Commonly used to test APIs [24].

React

JavaScript library used for user interface development [25].

TypeScript

Programming language built on JavaScript as it adds static types [26].

Types in TypeScript

Within TypeScript it is possible to create custom types. That allow the creator to represent data is more complex structures more suited for the specific task [27].

JavaScript

Programming language commonly used for web page development [28].

JSON

A file format that uses the JavaScript object notation to structure data more clearly. That is commonly used within APIs and within files to store data [29].

Npm

Npm standing for Node package manager is a tool used to install and manage packages in JavaScript applications [30].

Node.js

Node.js is a JavaScript runtime that allows JavaScript to run outside of the browser. Node.js runs on Google's V8 engine and enables execution of JavaScript code outside of the browser, allowing for server-side scripting. Enablement of server-side scripting unifies the web application development to be able to be developed in only JavaScript.

Node.js handles server-side operations with a single thread. The single thread enables asynchronous operation by using the event-loop. Starting Node.js also initializes the event loop. The event loop enables the application to handle multiple incoming requests and I/O tasks asynchronously making it a non-blocking mechanism. It enables this by delegating incoming tasks and requests and then the event loop continuously check for tasks execution callbacks [31].

Vertex AI SDK npm package

The vertex AI SDK is made for projects that run on Node.js and provides boilerplate to help integrate Google Cloud's Vertex AI Services. With Vertex AI SDK, projects can build AI features such as

text generation by communicating with the Google Large Language Models through an API [32].

3. Method

This chapter describes the methodology used to conduct research, software development and user testing throughout the thesis. It outlines the different stages and development phases including local development and integration development. Additionally, explain the process and prompt technique research, implementation of the corresponding prompt templates and evaluation of their effectiveness through usability testing.

3.1. Defining methodology and workflow

The primary step in the thesis is defining the methodology and workflow approach suitable and plausible in terms of scope and time, to complete research, pre-studies, development, integration and usability testing.

3.1.1. Workflow cycle

The defined workflow during the thesis will be conducted in the following defined structure.

1. Pre-studies
2. Local Development
3. Portal Integration
4. Utility Feature Implementation
5. Prompt Template Development
6. Usability Testing
7. Evaluation

3.1.2. Communication during Thesis

Over the course of the thesis the communication between the authors has occurred mostly in-person while working together at the IKEA office. Beside spoken communication, messages online has been used to communicate and development of the thesis implementation and to discuss lighter topics quickly.

The communication between IKEA and the authors, has been going on mostly through Slack and in-person workdays and meetings. Meetings were set up with the Nudging Framework team and the IKEA supervisor to determine the direction, give iterative feedback on the thesis and to update on the progress.

3.2. Pre-studies

This section will go over the work needed to be researched and understood before the thesis implementation could begin. That meant going over and reading the documentation for the APIs, both Optimizely and Vertex AI. Additionally, understanding the core nature of the nudges experiments and the self-service portal.

3.2.1. Understanding Nudges

To conduct the thesis the first step was to understand what a nudge is and what an experiment for it is. Therefore, sessions with questions to the Nudging Framework team were held, about nudges and their purpose and how their experiments are conducted. These conversations helped with understanding how the experiment data is generated and how the nudge creator uses the data.

3.2.2. Understanding the Self-Service Portal

To better understand the self-service portal and how nudges work on the portal, meetings with the Nudging Framework Team were held, and documentation of the nudge system was reviewed. Majority of this phase focused on how a nudge experiment is carried out. Therefore, multiple mock nudges were made on the self-service portal, by following the instructions on the portal. This gave nudges that could later be connected to metric data by the Nudging Framework team, enabling fetching from Optimizely.

3.2.3. Researching Prompting Techniques

Preliminary research of different prompting techniques was conducted to determine which techniques best fit the task of this thesis. Based on applicability to the chosen Large Language Model, Gemini 2.0 Flash-Lite, and the summarization task, the selected

prompting techniques were Zero-Shot, Few-Shot, and Role Prompting (see Section 2.4.2, Prompting Techniques).

3.2.4. Researching Tools

This section goes over how research was performed on some of the tools that would be needed during the development. That includes research of the Optimilzey API and Vertex AI API.

Researching Optimizely API

Optimizely serves as the Nudges platform to store and compute results for the metrics. Initial research was necessary to understand the structure of the metric results and their retrievability from Optimizely's API (see Section 2.6, Optimizely API). The initial phase of research involved a review of the documentation to identify potential ways to extract the data. The function used is an extraction function (see Section 2.6.1, Extraction Function), that fetches the experiment results for a specific nudge. As an extraction function was chosen, further analysis of its response was constructed. This to understand how to get the interesting values out of the response.

Researching Vertex AI API

Vertex AI API was researched to get a fundamental understanding how to utilize the tool, to be able to perform the request to the LLM (see Section 2.3.1, Vertex AI). The initial research was done through reading the documentation for the API and understanding how to send a request to it, and how to read the response.

3.3. Local Development

This phase will include the steps taken during the local implementation phase of the thesis, this includes the local test implementation, the integration to the portal, the refinement of the prompts and how initial evaluation was performed of the LLM output.

3.3.1. Fetching Nudge Metric Results

The first step taken was to fetch the Nudge experiment result data from Optimizely, this was done by the determined function within the Optimizely API, (see Section 2.6.1, Extraction Function). The data

was then fetched utilizing a Python script which called to the function, as the function requires an *experiment ID* which cannot be access dynamically during this phase. It was decided to use a static *experiment ID,* to fetch a single Nudge experiment result.

3.3.2. Parsing Metric Results

To refine the response from the extraction function from Optimizely, the response was parsed using a Python script. The refining composed of reducing the amount of information from the Optimizely response. This was done by creating a new object within the code that only contained the essential values that was needed for summarization, for the LLM. The Python script generates a new JSON object composed of only those essential key values.

3.3.3. Formatting the Parsed Metric Results

To make the data more easily digestible for the LLM, a reformatting of the JSON was done to visualize it as a table, see Fig 1. This reformatting was performed by a Python script.

Metric	Scope	Winning Direction	Status	Samples	Improvement	Lift Status
checkout_complete	visitor	increasing	Nudge visible	-------	----------	N/A
			Nudge hidden	-------	----------	N/A
bounce_rate	visitor	decreasing	Nudge visible	-------	----------	N/A
			Nudge hidden	-------	----------	N/A
exit_rate	visitor	increasing	Nudge visible	-------	----------	N/A
			Nudge hidden	-------	----------	N/A

Fig. 1, Table used during the local testing phase to structure metric data.

3.3.4. Google Authentication for Vertex AI

To enable request to be sent to Vertex AI, initial setup of the Google Cloud SDK and Vertex AI API was configured for the local testing environment. Configuration of the Google Cloud SDK and Vertex AI API during the local testing phase helped to minimize potential errors during the following portal integration phase. To set up the authentication for Vertex AI, credentials for the environment variables and authentication keys were generated on the Google Cloud Platform.

3.3.5. Prompting Vertex AI

When the authentication for Vertex AI was done, the test application could now make request to the Vertex AI API, through the Python Vertex AI library, (see Section 2.3.1, Vertex AI). The initial prompt made through the test application was following the Zero-Shot prompting technique and consisted of the instruction "Summarize the data in short" and the table containing the Nudge metric results, as shown in Fig. 1. This prompt was then sent to the LLM model Gemini 2.0 Flash-Lite, (see Section 2.3.2, Gemini).

3.3.6. Showing LLM Response

To showcase the LLM response and also show how it could look on a portal, a local Python web server was started and website developed to display the response coming from Vertex AI. To demonstrate how an LLM response could be added to a website and showcase the functionality to be implemented later in the self-service portal. This part was conducted to have a mockup and enable initial feedback of the functionality from the Nudging Framework team.

3.4. Portal Integration

This phase includes the steps performed to integrate the LLM service, prompt processing, and Vertex AI authentication from the local development into the self-service portal. It involves setting up the self-service portal on local machines, developing the backend controller and service, and developing the frontend for showcasing the LLM response.

3.4.1. Set Up NFW Self-Service Portal

The steps taken to set up the nudging framework self-service portal, (see Section 2.5, Nudge), was to clone the development repository from IKEA Ingka's GitHub. Then the documentation was followed on how to run it locally.

3.4.2. Developing Backend

This phase includes the steps taken to integrate the LLM into the backend. The taken steps where to develop a controller and a service for the LLM and for the Vertex AI, and to extending the existing

Optimizely API service. The backend was developed in TypeScript and runs on Node.js.

Vertex AI

The Google Cloud Vertex AI SDK npm package was used in the development to enable the Vertex AI API from the self-service portal's codebase. The following command was used to install the npm package:

```
npm install @google-cloud/vertexai
```

The backend was able to send prompts and receive the generated output from Vertex AI, with the Vertex AI SDK in place. Thus, completing the connection between the codebase and Vertex AI API.

3.4.3. LLM Service backend

To enable summarization of the nudge metric results a backend service was developed, consisting of both a controller and a service. This was done to follow the structure of the self-service codebase. The LLM controller in the backend service handles the incoming requests from the frontend. It then interacts with the LLM service and Vertex AI service to send the prompt and get the output response from the Vertex AI model. The backend service handles the prompt requests and enables the response to being outputted to the frontend interface, via the controller. These functions were then tested by using curl commands to confirm that the LLM controller and service worked with the Vertex AI credential set up from the local implementation.

3.4.4. Extending Optimizely API

To enable getting the nudge experiment results from Optimizely and give it to the LLM, it is needed to extend the existing Optimizely API service in the self-service portal. The metrics needs to be included in the call to Vertex AI through the LLM backend service. The added functionality in the Optimizely service is the extraction function (see Section 2.6.1, Extraction Function). That included the call to the Optimizely API to request the nudge metric results. The response given here was a JSON that would be sent back to the frontend interface.

3.4.5. Usage of IKEA frontend components

Development of user interface component was done to enable the visualization of the LLM's generated response on the internal self-service portal. In specific, a modal component (see Section 2.11, Frontend Components), that contained the response was developed and integrated to the self-service main parental component.

Additional components such as the segment controller, (see Section 2.11, Frontend Components), that displays and enables different prompting technique options as shown in Fig 2.

Fig. 2, The segment controller that select the prompting techniques.

Fig. 3, IKEA Skapa loading ball component.

The loading ball component, shown in Fig. 3 is used to display the loading response from Vertex AI, it is also a component from IKEA SKAPA library.

3.4.6. Developing Frontend Components for the LLM Summary Feature

This section goes components or functions developed to enable the LLM summarization feature. By connecting the backend to the frontend and functions that worked on the frontend to refine the data structure fed to the LLM.

Connecting Frontend and Backend

To connect the developed LLM backend service with the self-service portal user interface, HTTP endpoints were implemented in the backend of the application. The endpoints allow the frontend to send data as the *experiment ID* and the *hypothesis* through HTTP requests. The frontend calls the LLM controller which sends back the LLM output response, which is then rendered in the user interface making it visible on the self-service portal. Utility features

Utility functions

This section will go over the utility functions created in the frontend to assist with some data mapping and string building, in the form of a table generator and a prompt builder.

Mapping Metric Results

Using the JSON response from Optimizely showcased a limitation with the LLM's input capacity, as included data became too large. It was first tried to minimize it and map out the key values wanted. To make the JSON smaller and fit within the LLM's input capacity. This mapping followed the same principles as the one performed during the local testing, but added a few more keys, to be able to give the LLM more context and data to summarize on.

The first step in mapping the data in TypeScript was to create the types for each layer within the JSON. Once the types had been created functions to map the JSON to each type was developed and integrated into the portal as a utility feature.

Formatting Table for Metric Results

Mapping the results did not suffice in making the data fit within the LLM's input capacity. Going back to the local implementation again to take inspiration for the table created there, a utility function was added in the frontend to format the mapped JSON into a table. This method now ensured that the data fitted within the context window.

The format of the table in the local test application as shown in Fig. 1, did not fit with the new mapped structure of the JSON. Additionally, it was thought of better to include more columns describing each row individually, instead of having a row for each

design for the nudge as done during the local test development. Easiest seen in the new table, with the row now including both designs of the nudge and the values. This made the overall table include less rows but more columns, as shown in Fig 4. This new table and table structure made the data clearer and more structured for the LLM to parse.

Metric	Primary	Aggregator	Scope	Winning Direction	Significant
checkout_complete	true	unique	visitor	increasing	false
bounce_rate	false	sum	visitor	decreasing	false
exit_rate	false	count	visitor	increasing	false

Design 1	Design 2	Samples D1	Samples D2	Improvement	Lift Status
Nudge hidden	Nudge visible	---	---	N/A	better
Nudge hidden	Nudge visible	---	---	N/A	better
Nudge hidden	Nudge visible	---	---	N/A	better

Samples Remaining	Confidence Lower Bound	Current Confidence	Confidence Upper Bound
---	undefined	0	undefined
---	undefined	0	undefined
---	undefined	0	undefined

Fig 4, Table used to structure the metric data for the LLM, stacked vertically.

Prompt Builder

Three distinct prompt builder functions were implemented to be utilized during the usability testing phase. Each prompt building function addressed one of the chosen prompting techniques: Zero-Shot, Few-Shot and Role Prompting, (see Section 2.4.2, Prompting Techniques). The separation into distinct functions was necessary for the testing to enable the selection mechanism on the frontend, making the interaction easier for the test participants.

3.4.7. Developing Prompt Templates

This section will go over the development steps taken when implementing the different prompting functions and their respective techniques.

Zero-Shot Prompt

The first prompt template function implemented was to utilize the prompting technique Zero-Shot, (see Section 2.4.2, Prompting Techniques), and act as a foundation to build upon for the other

prompt building functions. Using Zero-Shot technique meant giving the LLM limited instructions and let it comprehend most of the context by itself. The initial implementation of the function used the prompt: “Summarize this in short”, combined with the nudge metric result table. To follow the principle of Zero-Shot as much as possible by giving the LLM only itself to interpret the data and to generate a response.

Initial evaluation of the response from the LLM showed that it was too inconsistent. Additional instructions were added to give LLM loose guidelines to make it follow the Zero-Shot technique, to enable a more consistency. The added instructions followed the guidelines given by the Nudging Framework team about the data, and included which metrics that always should be included and that the generated output should be in markdown, to suit the frontend better.

Few-Shot Prompt

The second prompt template function was developed around the prompting technique Few-Shot, (see Section 2.4.2, Prompting Techniques). The technique uses examples of both the input and the output to guide the LLM in how to understand the data. This meant giving an example of the table that would be added to the data in the prompt. The example had added instructions of what each column in the table described and how to interpret it, for example the data being written as 0.9 should be interpreted as 90%. The other given example to the LLM was how it should structure the output, this example outlined where the LLM should output the *hypothesis*, the default metrics, the highlighted metrics and a summary and conclusion.

Role Prompting Prompt

The third prompt template function was developed to implement the prompting technique Role Prompting, (see Section 2.4.2, Prompting Techniques). The technique relies on assigning the LLM a specific persona, to guide the output in tone and style. Initially the function was conceived as an extension to the Few-Shot prompt function combined with assigning the role “data scientist” and to output in the tone of “IKEA”. The first iteration of the function produced results that were too indistinguishable from the Few-Shot function, which could limit the ability to draw comparisons during the usability testing phase. The prompt function was then redesigned to

enhance its distinctiveness by removing the output example and giving more control to the LLM. This resulted in the output being very inconsistent which also would impact the usability testing negatively. The prompt was refined once more to include a small set of instructions outlining the essential elements required in the output.

Evaluation of LLM Response

This section will go over how evaluation of the LLM response was performed, with most focus on evaluating its accuracy to the data. Following the golden standard for evaluation of text generated is to utilize comparisons with humans as evaluators.

Cross-referencing

During the thesis the LLM responses have been evaluated using the evaluation technique named cross-referencing, which states that the actual data and the output from the LLM is put side by side. To then go over each data point that the LLM included in the response and compare it to the values in the table and value from the Optimizely response. This was done to ensure that the LLM interpreted the values correctly in a significant majority of instances.

3.5. Usability Testing

This part describes the steps taken to design and conduct the usability test. It covers the making of the survey, selection of participants and how the usability testing was conducted.

3.5.1. Creating Survey

Google Forms was used to collect results from the usability test, chosen for its simplicity and the author's prior knowledge with it. The survey was based on the research questions outlined in the thesis (see Section 1.5, Problems to Research), mainly focusing on pinpointing which prompting technique enhances the user experience. The survey includes the following questions:

- Select which prompting technique that generated the most comprehensive summary.
- Select which prompting technique generated the most helpful summary.

- Select which prompting technique that generated the most supportive summary in terms of decision making
- Select one or more reasons for their choice of prompting technique
- Optionally give additional comments and feedback.

The usability test included the developed prototype that displayed the LLM-generated summaries for two nudges, referred to as Nudge A and Nudge B. These were provided by the Nudging Framework team within their development suite and included the hypothesis and metric data for the nudge. To collect responses for both nudges, the survey was designed to repeat the same set of questions for each nudge separately. This allowed participants to evaluate and compare the prompting techniques for two different nudges independently.

3.5.2. Participants Of Usability Testing

Six participants were selected for the usability test by the thesis authors. All participants were selected based of schedule availability and varying knowledge of the self-service portal and the nudge framework. All six participants were software engineers at IKEA, some with deep familiarity with the nudging framework, and others with no prior knowledge it. Getting the users of the nudging framework (such as business and marketing workers at IKEA) was not possible due to availability and time constraints.

The variation in familiarity with the nudging framework and self-service portal allowed for broader perspectives and inputs on how the LLM summarization prototype was perceived. Furthermore, having participants with varying familiarity with the nudging framework helped to identify usability challenges, both for experienced and inexperienced users. This allowed for different perspectives when evaluating the LLM-generated summaries.

3.5.3. Performing Usability Tests

The usability test was conducted in-person together with the users face to face, either in pairs or individually. Initially, an introduction by the thesis writers was given to the users on how the survey works and an overview of the self-service portal.

The prototype with the fully developed LLM service was shown on one computer. The users were free to click around and use the prototype freely during the whole test. Each user brought their own computer to answer the survey, whilst having the opportunity to experience and test the prototype. Each session for usability testing was booked for 30 minutes. The users were free to ask any questions to the thesis writers regarding the prototype or the test, throughout the whole testing session.

3.6. Criticism of References

The information used for this thesis was obtained from various either website, publications or books. These source references can be found in Section 7. References.

Sources [1], [2], [13], [14] and [15] were from the official site of IBM (Internation Business Machine Corportation), which is a major provider of AI, cloud services. They also have a long history of innovation and is a major researcher within the AI field. This makes them a credible source, in particular for AI and LLM, and prompt engineering. However, it is important to note that their publications and information may reflect on their favorable corporate interests.

Source [3] is a published book which can be considered as a credible source due to it having to be reviewed and published.

Sources [4]-[9] were from the official site of Google or their products official sites, such as Vertex AI and Deepmind. As the technology and services used in the thesis such as the Gemini models and Vertex AI API are developed by Google, their official documentation serves as the primary source. Hence, making it a credible source.

Sources [10] and [13] is from the official Microsoft documentation page or from their official GitHub course about AI. Microsoft is a major provider in cloud services, software applications and has a long history of innovation. They are a major investor in AI through their investment in OpenAI. This combined with them being a major contributor to the technical sector makes them a credible source.

Source [16] is the official documentation for Optimizely experimentation REST API's, this is a credible source as it is used as

a backbone for the API that is implemented for IKEA based on the Optimizely experimentation REST API. As this is the official documentation of the foundation, provided by the company providing the service, it makes it a credible source.

Source [17] is an article written about local development and its benefits, written by the company Cloudomation. The company provides educational content aimed for developers about workflows and automations for development environments. While the company also offers tools for development workflows, these tools are not pushed in the article. The source is deemed credible enough to support a explanation of local development.

Source [18] is a paper published on reserachGate.org, which may lack in academic credibility due to the paper's limit in peer review. While the paper's main focus is not on the controller service architecture, which is the reason it is referenced, it still gives insight into the specific architecture and its structure.

Source [19] is a paper published on arXiv.org which is an open-source repository for researcher to publish their papers before they are peer-reviewed and published. The paper has received 3588 citations on Google Scholar as of 2025-05-12, and the authors are affiliated with reputable universities it can be deemed as a sufficiently credible source.

Source [20] is a published book, published by Elsevier which is a reputable publisher. The book being published means that it has been reviewed by other parties and that makes it a credible source. The book aims to explain and inform about user experience and how it can be measured. The book is not a corporate source and is less prone to carry business related biases making it a neutral and reliable source.

Source [21] is a published book, published by Morgan Kaufman which is an imprint of Elsevier science. which is a reputable publisher. The book being published means that it has been reviewed by other parties and that makes it a credible source. The source is aimed to inform about Usability testing and how it is conducted. As the source is not a corporate source it is less likely to carry corporate biases, which makes it an academically reliable source.

The sources [22]-[32] that are used in the technical background are used to present tools that have been used within the scope of the thesis. The sources, lead either to the official documentation of the

tool, or a commonly used source that explains the tool. The sources [22] and [23] comes from the Material UI documentation. Source [24] from the official curl documentation. Source [25] from the official React documentation. Sources [26] and [27] from the official Typescript documentation. Source [28] from the official JavaScript documentation from Mozilla. Source [29] come from the official JSON organization. Sources [30] and [32] come from the official npm documentation and source [31] come from the official Node.js documentation. As these all come from the official documentation of the tool, they can be deemed as credible sources.

3.7. Usage of AI

The code and the report have been written by the authors of the thesis. During the implementation of both the local development and portal integration, AI has been used as a tool to assist in figuring out errors in the code. During the writing of the thesis report, AI has been used to aid with grammar and structure.

4. Results

This chapter contains the results of the implementation work of the LLM service conducted in this thesis, and the results of the usability tests.

4.1. Implementation Result

This section covers the implementation results of the developed LLM service. It presents the prompt templates and the integration of the self-service portal.

4.1.1. Prompt Templates

Three prompt templates were developed based on the selected techniques: Zero-Shot, Few-Shot, and Role Prompting, (see Section 2.4.2, Prompting Techniques). In all the figures that showcases the prompt templates, there is two parts that are shown as in quotation marks. The “hypothesis” represents the variable *hypothesis* that is added to the prompt, “experiment data” represents the variable that represents the *metric result table* for the nudge.

Zero-Shot Prompt

```
Summarize the experiement data in short.
Always include the first metric in the data, this is the primary metric.
Must include checkout_complete, bounce_rate and exit_rate.
Do not include metrics starting with cweb or the experienemt id.
Include metrics where that are significant market in the data.
Write the response in **markdown** and write the metric names in bold.
Write percentages in plain text, 0.1 increase indicates 10%. Connected an analysis to the hypothesis.
Maximum of 15 sentences. State which experiment id the metrics belong to.

Hypothesis: "hypothesis"
Experiment data: "experiment data"
```

Fig. 5, The Zero-Shot prompt.

Shown in Fig. 5 is the final prompt template used for the Zero-Shot technique.

Few-Shot Prompt

```
Example:
  Experiment data:
  Aggregator – What the metric is measuring towards
  Scope – For whom the metric targeted
  Winning direction – Tell if the metric increased or decreased
  Improvement – The percentage increase or decrease, written as 0.1 for 10%.
  Lift status – Tells if design performed better or worse than the default design or nudge off.
  Significant – Tells if the metric data should be heavily consider while making the conclusion.
  Visitor Remaining – The amount of samples still needed to determine if the metric is significant
  Confidence Lower Bound / Upper Bound – The upper and lower bounds for the improvement. Written as 0.1 for 10%.
  Current Confidence – The current significant value which will be within the Lower and Upper bounds. Written as 0.1 for 10%.

  Metric | Aggregator | Scope | Winning Direction | Significant
  | Design 1 | Design 2 | Samples Design 1 | Samples Design 2 | Improvement | Lift Status
  | Visitors Remaining | Confidence Lower Bound | Current Confidence | Confidence Upper Bound
  Metric A | unique | visitor | increasing | false
  | Off | On | 41442 | 76775 | 0.04 | better
  | 89278 | −0.025 | 0.56 | −0.025
  Metric B | count | visitor | increasing | true
  | Off | Default design variant | 41442 | 76775 | 8.023| better
  | 0 | 7.47 | 1 | 7.47
```

Fig. 6, Part 1 of the Few-Shot prompt. Including the description of the table and example data.

```
  For the metrics showed, include the significance, the percentage increase as 10% for example.
  Include the winning direction for the metric and if it is better or worse.
  Always include for default metrics.

  Desired markdown output format:
  # Experiment ID
  ## Hypothesis
  State the hypothesis of the nudge
  ## Primary and Significant metrics
  Metric A
  Metric C
  ## Default metrics
  Exit rate
  ## Analysis and conclusion
  Your analysis and conclusion based on the hypothesis and metrics provided.
```

Fig. 7, Part 2 of the Few-Shot prompt. Including the output example and instructions for output for the LLM.

```
Summarize the experiement data in short.
  Always include the first metric in the data, this is the primary metric.
  Must include checkout_complete, bounce_rate and exit_rate as default metrics.
  Include metrics where they have significant as true in the data.
  If primary metric is in default metrics, show it **only** under the Primary metric.
  Write the response in **markdown** and write the metric names in bold.
  Write percentages in plain text, 0.1 increase indicates 10%.
  Include which aggregator the metric is for.
  Do not include metrics starting with cweb or the experienemt id, even if they are significant.
  Maximum of 15 sentences.
  Connected an analysis to the hypothesis.
  If most metrics and the primary one isn't significant, then state that the test should run more.

Hypothesis: "hypothesis"
Experiment data: "experiement data"
```

Fig. 8, Part 3 of the Few-Shot prompt. Including the instructions for the LLM.

Figures 6, 7, and 8 together illustrates the final Few-Shot prompt template. Fig. 6 includes the description of the metric result table and example metrics. Fig. 7 includes the output example and instructions for the output. Fig. 8 includes the instructions for the summarization task to the LLM and contains the placeholder variables for the *hypothesis* and the *metric result table*.

Role Prompt

```
Please follow these instructions clearly.

| Field Name               | Description                                                                  |
| :----------------------- | :--------------------------------------------------------------------------- |
| Metric                   | The name or identifier of the metric being measured.                         |
| Primary                  | Indicates if this is the primary metric for the experiment.                  |
| Aggregator               | How the metric is calculated (e.g., 'unique', 'sum', 'count').               |
| Scope                    | The target of the metric (e.g., 'visitor').                                  |
| Winning Direction        | Whether the metric should increase or decrease ('increasing', 'decreasing'). |
| Significant              |  Indicates if the improvement is statistically significant ('true', 'false'). |
| Design 1, Design 2       | Names of the two design variants being compared.                             |
| Samples Design 1, Design 2 | The number of samples used for each design.                                |
| Improvement              | The percentage change in the metric (e.g., 0.1 for 10%).                     |
| Lift Status              | Whether the design performed better or worse than the baseline.              |
| Visitors Remaining       |  The estimated number of samples needed to reach statistical significance.   |
| Confidence Lower Bound   | The lower bound of the confidence interval for the improvement (e.g., 0.1 for 10%). |
| Current Confidence       |  The best confidence in the improvement (e.g., 0.1 for 10%).                 |
| Confidence Upper Bound   | The upper bound of the confidence interval for the improvement (e.g., 0.1 for 10%). |

Metric Output Example:
Metric A:
Explain target and aggregator, state how many samples and how many is left until significant, state the improvement and direction.
If a metric is significant, state the confidence levels.
Do not format as a list, **ONLY** sentences explaining.
```

Fig. 9, Part 1 of the Role Prompting prompt. Including the detailed table descriptions and the output instructions.

```
[Table from earlier section]

Experiement Data : "experiement data"
Hypothesis : "hypothesis"

Instructions for Analysis and Task instructions:
- Always include the 'Primary' metric.
- Guardrail metrics are the following, checkout_complete, bounce_rate, exit_rate.
- Always include metric that are 'significant' in the data.
- Do not include metrics starting with cweb, or the 'experiementid' in the response, even if significant.
- Keep the metrics shown in output to primary, guardrail and signifiant metrics.
- State the Hypothesis in the output.
- Connect analysis to the hypothesis. If the data isn't sufficient, then say so.
- Make sure this is between the metrics and the analysis '---', a dividing line.
- Metric names should always be in bold.

Please follow these instructions carefully:

1. You are an LLM trained to perform Data Analysis.
2. Analyze the provided 'data' according to the 'analysis_instructions' in short.
3. Connected the analysis to the 'hyopthesis' and give a short conclusion of max 5 sentences.
4. Maintain a happy, helpful, and enthusiastic tone of IKEA throughout your response.
```

Fig. 10, Part 2 of the Role Prompting prompt. Including the instructions.

Figures 9 and 10 together illustrated the final Role Prompting prompt. Fig. 9 includes the detailed description of the table and the

instructions for the output. Fig. 10 includes the instructions for the summarization task to the LLM and contains the placeholder variables for the *hypothesis* and the *metric result table*.

4.1.2. Portal Integration

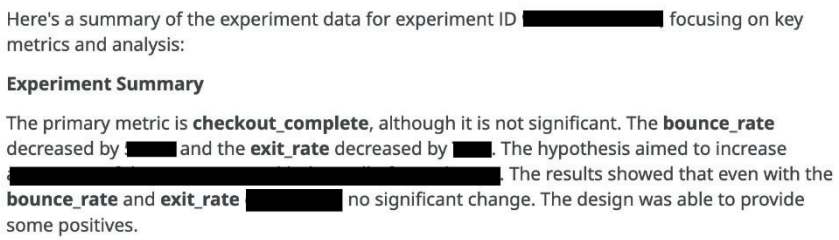
The backend integration included developing a new controller and service for the LLM summarization service. The LLM controller and service were both developed in a separate git branch of the self-service repository and built using Node.js and TypeScript. The developed LLM backend service handled the prompt construction, authentication to Vertex AI, sent requests to Vertex AI, and returned the generated response from the LLM.

The frontend integration involved developing new UI components to display the response from Vertex AI. A modal component was used as a parental component for displaying the response. Additionally, a segment controller shown in Fig. 2 and loading ball component shown in Fig. 3 from the IKEA UI library were used to enable switching between the prompting techniques and visualize the loading state. The LLM frontend components were linked to the backend service through HTTP requests, and the LLM-generated summarized were shown in markdown format on the modal component.

4.1.3. LLM Responses

The following figures are examples of how a response from each prompting technique, in each figure confidential information is blacked out.

Zero-Shot


Here's a summary of the experiment data for experiment ID [redacted] focusing on key metrics and analysis:

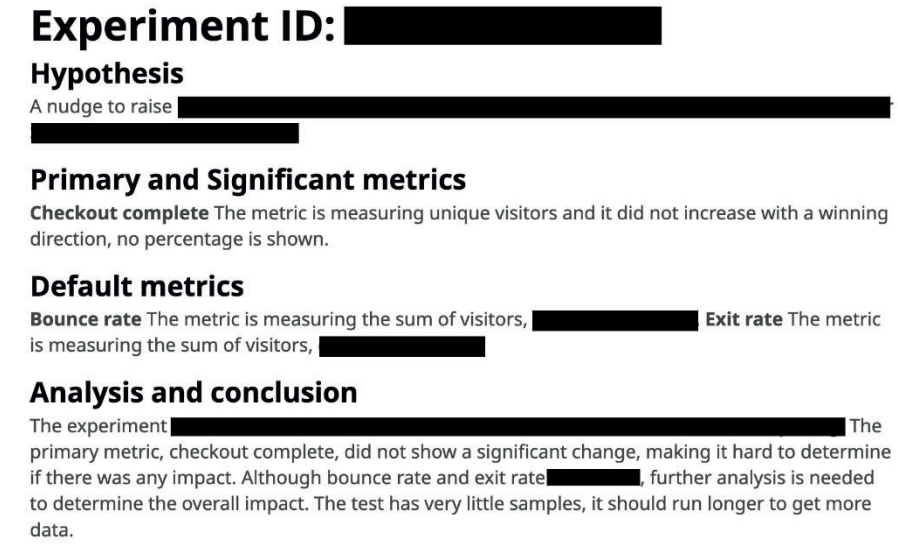
Experiment Summary

The primary metric is **checkout_complete**, although it is not significant. The **bounce_rate** decreased by [redacted] and the **exit_rate** decreased by [redacted]. The hypothesis aimed to increase [redacted]. The results showed that even with the **bounce_rate** and **exit_rate** [redacted] no significant change. The design was able to provide some positives.

Fig. 11, The response from the LLM, using the Zero-Shot prompt.

Generated response by the LLM based on the Zero-Shot prompt is shown in Fig. 11.

Few-Shot



Experiment ID: [redacted]

Hypothesis

A nudge to raise [redacted]

Primary and Significant metrics

Checkout complete The metric is measuring unique visitors and it did not increase with a winning direction, no percentage is shown.

Default metrics

Bounce rate The metric is measuring the sum of visitors, [redacted] **Exit rate** The metric is measuring the sum of visitors, [redacted]

Analysis and conclusion

The experiment [redacted] The primary metric, checkout complete, did not show a significant change, making it hard to determine if there was any impact. Although bounce rate and exit rate [redacted], further analysis is needed to determine the overall impact. The test has very little samples, it should run longer to get more data.

Fig. 12, The response from the LLM, using the Few-Shot prompt.

Generated response by the LLM based on the Few-Shot prompt is shown in Fig. 12.

Role Prompting

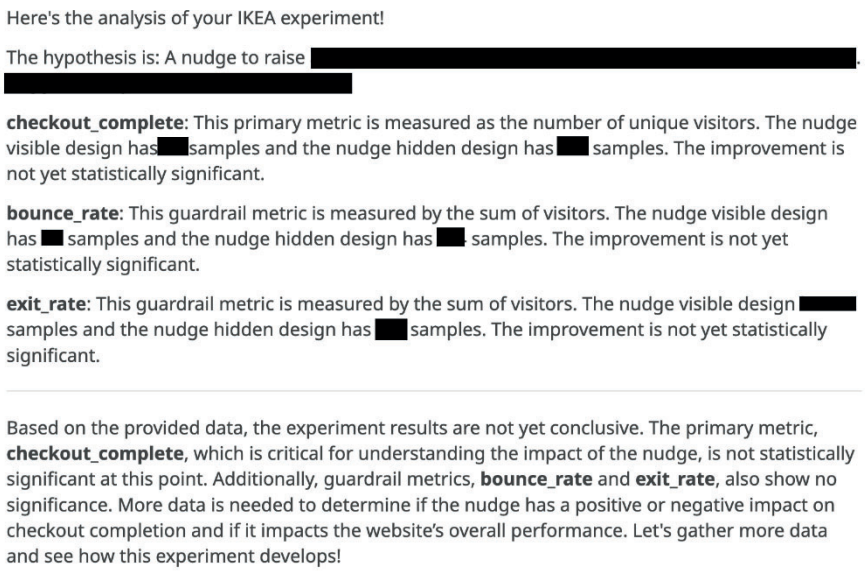


Fig. 13, The response from the LLM, using the Few-Shot prompt.

Generated response by the LLM based on the Role Prompting prompt is shown in Fig. 13.

4.2. Usability Test Results

Each sub header represents a question in the survey and includes the corresponding result for each nudge.

4.2.1. Most Comprehensive Prompting Technique

Nudge A

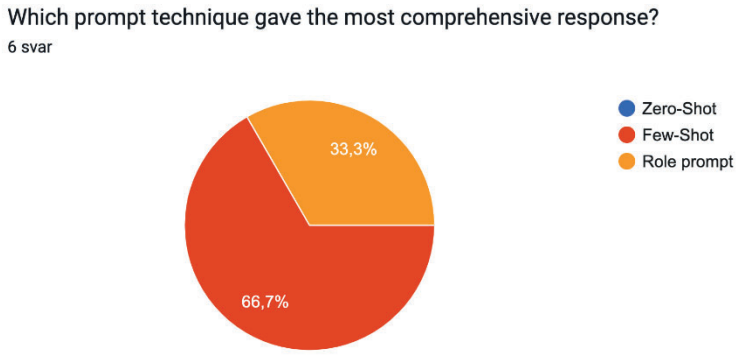


Fig. 14, Pie chart of responses for the most comprehensive prompting technique for Nudge A.

Shown in Fig. 14, the six responses from the usability test, 66,7% chose Few-Shot to have given the most comprehensive response. 33,3% chose Role Prompting. No respondents chose Zero-Shot.

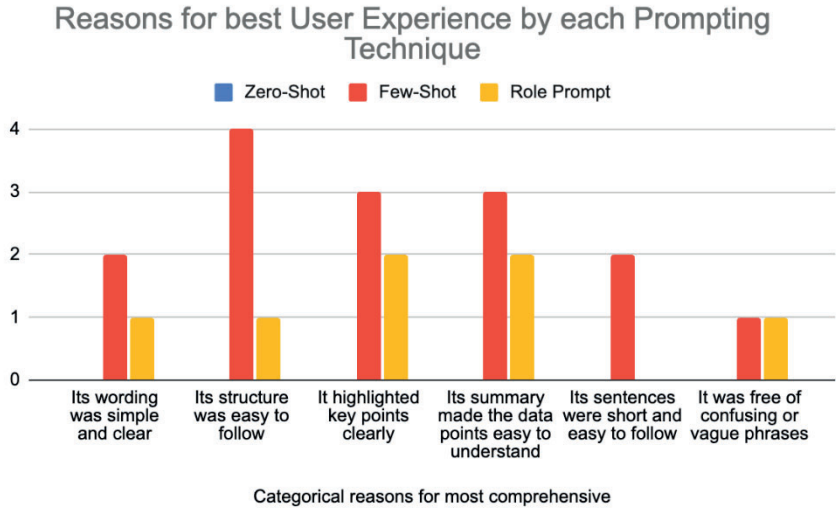


Fig. 15, Bar chart of selected reasons for which prompting technique that gave the most comprehensive response for Nudge A.

Shown in Fig. 15 is the reasons selected to why Few-Shot gave the most comprehensive response was: “Its wording was simple and clear” (2 votes), “Its structure was easy to follow” (4 votes), “It highlighted key points clearly” (3 votes), “Its summary made the data points easy to understand” (3 votes), “Its sentences were short and easy to follow” (2 votes), and “It was free of confusing or vague phrases” (1 vote).

The reasons selected to why Role Prompting’s response was the most comprehensive: “Its wording was simple and clear” (1 votes), “Its structure was easy to follow” (1 votes), “It highlighted key points clearly” (2 votes), “Its summary made the data points easy to understand” (2 votes), and “It was free of confusing or vague phrases” (1 vote). There were no reasons selected for Zero-Shot giving the most comprehensive response.

Nudge B

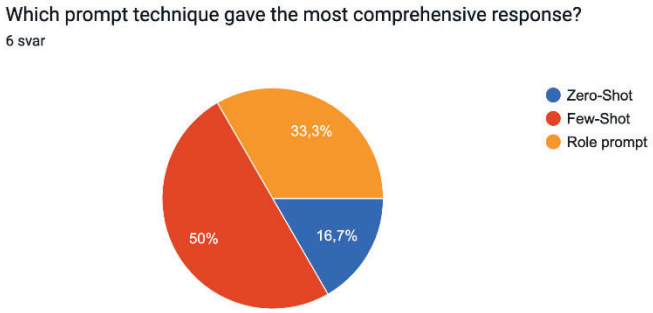


Fig. 16, Pie chart of responses for which prompting technique that gave the most comprehensive response for Nudge B.

Shown in Fig. 16 is the six participants choices from the usability test for Nudge B. The Few-Shot prompting response was perceived as the most comprehensive, chosen by 50% (3 votes) of the participants. Compared to the 33.3% (2 votes) that chose the Role Prompting response and the 16.7% (1 vote) who chose the Zero-Shot response.

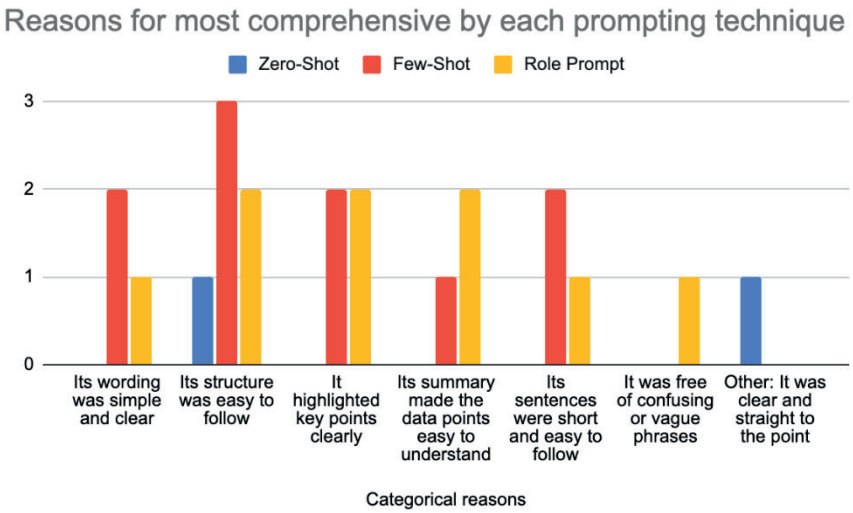


Fig. 17, Bar chart displaying the selected reasons to why the chosen prompting technique was the most comprehensive response for Nudge B.

Displayed in Fig. 17, the reasons selected to why the Few-Shot gave the most comprehensive response for Nudge B was: “Its wording was simple and clear” (2 votes), “Its structure was easy to follow” (3 votes), “It highlighted key points clearly” (2 votes), “Its summary made the data points easy to understand” (1 vote), and “Its sentences were short and easy to follow” (2 votes).

The reasons selected to why Role Prompting gave the most comprehensive response for Nudge B was: “Its wording was simple and clear” (1 vote), “Its structure was easy to follow” (2 votes), “It highlighted key points clearly” (2 votes), “Its summary made the data points easy to understand” (2 votes), “Its sentences were short and easy to follow” (1 vote), and “It was free of confusing or vague phrases” (1 vote).

The reasons selected to why the Zero-Shot gave the most comprehensive response for Nudge B was: “Its structure was easy to follow” (1 vote) and “Other: It was clear and straight to the point” (1 vote).

4.2.2. Most Helpful Prompting Technique

Nudge A

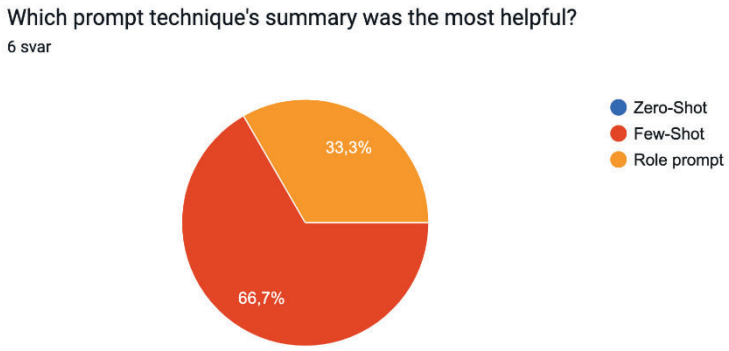


Fig. 18, Pie chart of responses for the most helpful prompting technique for Nudge A.

Fig. 18 shows that four participants selected Few-Shot to have been the prompting technique that gave the most helpful response which corresponds to 66,7%. Also, two participants selected Role

Prompting to have given the most helpful response, equaling 33,3 %. No participant answered that Zero-Shot to be the most helpful.

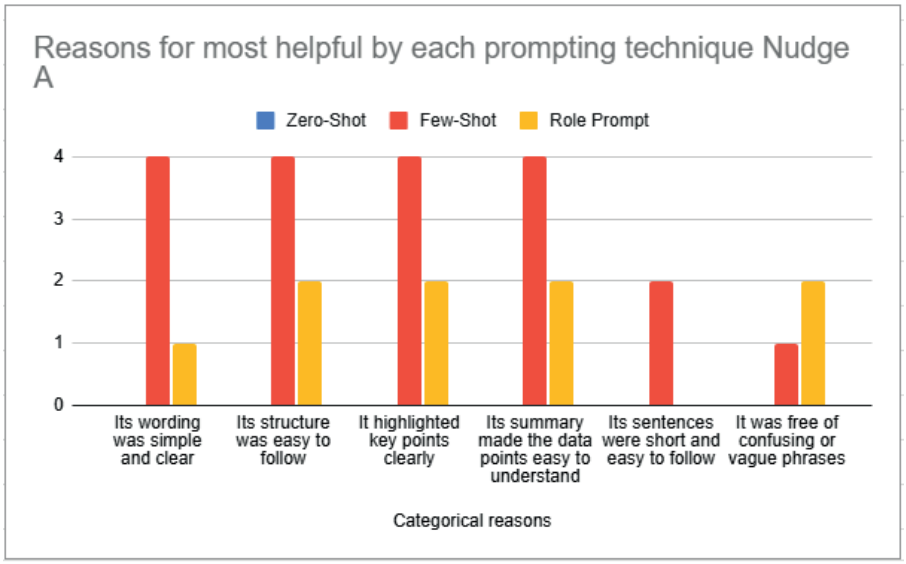


Fig. 19, Bar chart displaying the selected reasons to why the chosen prompting technique was the most helpful response for Nudge A.

Displayed in Fig. 19 the reasons selected to why the response of Few-Shot was the most helpful: “Its wording was simple and clear” (4 votes), “Its structure was easy to follow” (4 votes), “It highlighted key points clearly” (4 votes), “Its summary made the data points easy to understand” (4 votes), “Its sentences were short and easy to follow” (2 votes), and “It was free of confusing or vague phrases” (1 vote).

The reasons selected to why Role Prompting’s response was the most helpful: “Its wording was simple and clear” (1 votes), “Its structure was easy to follow” (2 votes), “It highlighted key points clearly” (2 votes), “Its summary made the data points easy to understand” (2 votes), and “It was free of confusing or vague phrases” (2 vote). There were no reasons selected for Zero-Shot giving the most helpful response.

Nudge B

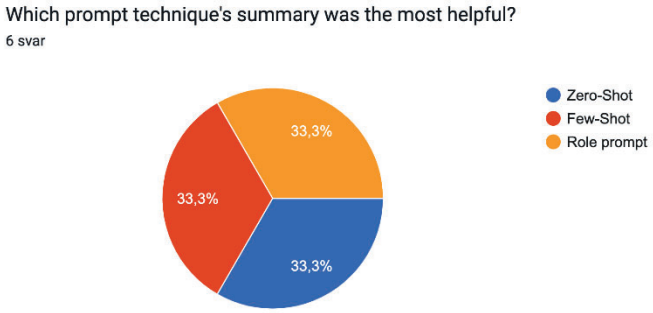


Fig. 20, Pie chart of responses for which prompting technique that gave the most helpful response for Nudge B.

Shown in Fig. 20 is the choices from the six participants from the usability test for Nudge B. All three prompting techniques: Zero-Shot, Few-Shot and Role Prompting each received 33.3% (2 votes), for which prompting techniques response was the most helpful.

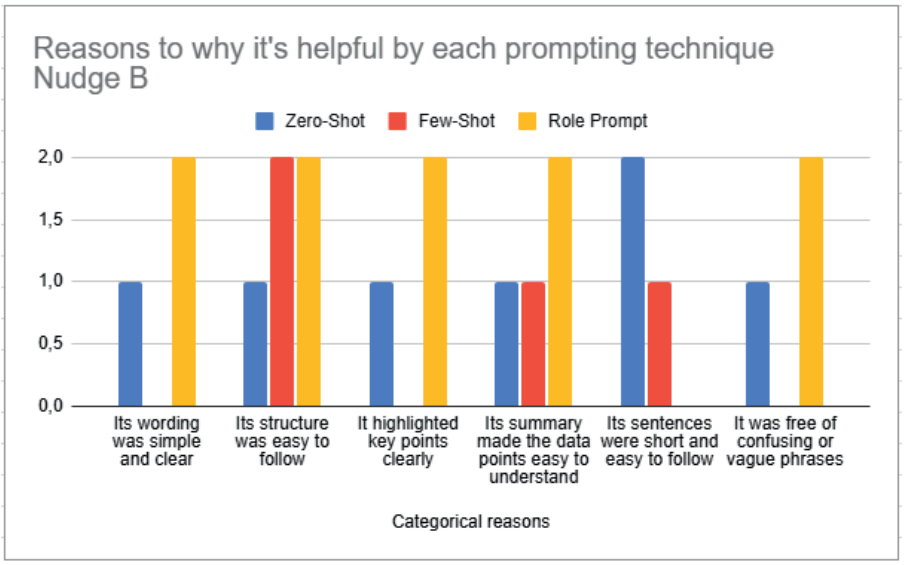


Fig. 21, Bar chart displaying the selected reasons to why the chosen prompting technique was the most helpful response for Nudge B.

Displayed in Fig. 21 the reasons selected to why Role Prompting gave the most helpful response for Nudge B was: “Its wording was

simple and clear" (2 votes), "Its structure was easy to follow" (2 votes), "It highlighted key points clearly" (2 votes), "Its summary made the data points easy to understand" (2 votes), and "It was free of confusing or vague phrases" (2 votes).

The reasons selected to why the Few-Shot gave the most helpful response for Nudge B was: "Its structure was easy to follow" (2 votes), "Its summary made the data points easy to understand" (1 vote), and "Its sentences were short and easy to follow" (1 votes).

The reasons selected to why the Zero-Shot gave the most helpful response for Nudge B was: "Its wording was simple and clear" (1 vote), "Its structure was easy to follow" (1 vote). "It highlighted key points clearly" (1 vote), "Its summary made the data points easy to understand" (1 vote), "Its sentences were short and easy to follow" (2 votes) and "It was free of confusing or vague phrases" (1 vote).

4.2.3. Most Decision Helping Prompting Technique

Nudge A

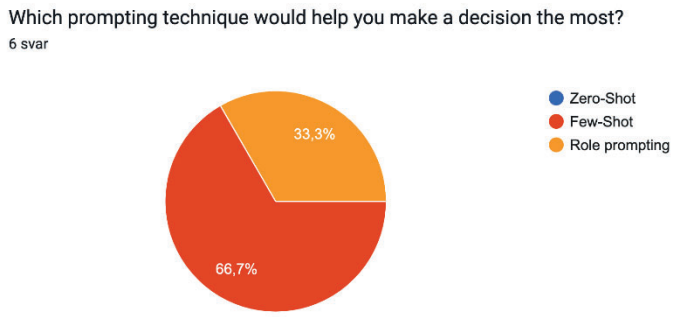


Fig. 22, Pie chart of responses for which prompting technique that gave the most decision helping response for Nudge A.

Fig. 22 shows that 66,7% of six participants selected Few-Shot to have given the most decision helping response. In addition, 33,3% of participants selected Role Prompting's response to have been the most decision helping response. No participants selected Zero-Shot to be the most decision helping response.

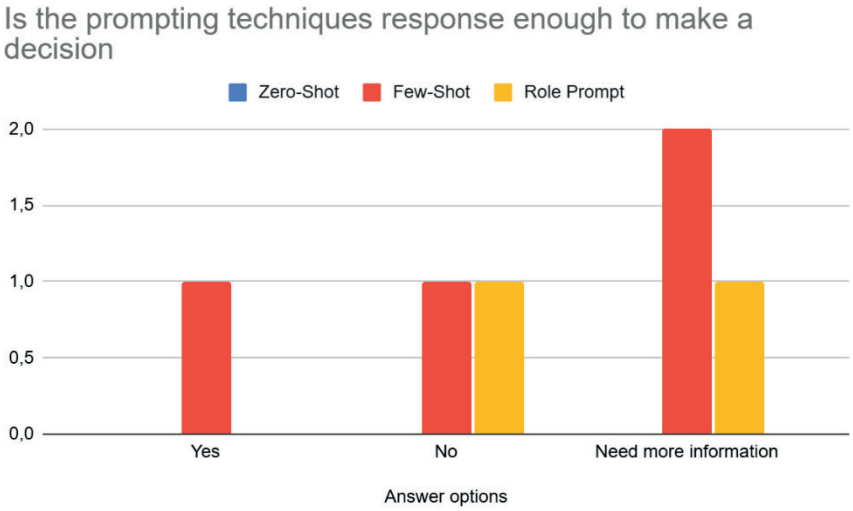


Fig. 23, Pie chart of responses of if the LLM response could be used to make a decision for Nudge A.

Shown in Fig. 23 the response for whether or not the Few-Shot response was enough to make a decision: “Yes” (1 vote), “No” (1 vote), and “Need more information” (2 votes). Additionally, the responses for whether or not the response from Role Prompting was enough was: “No” (1 vote), and “Need more information” (1 vote). No responses for Zero-Shot regarding this question were collect from the usability test.

Nudge B

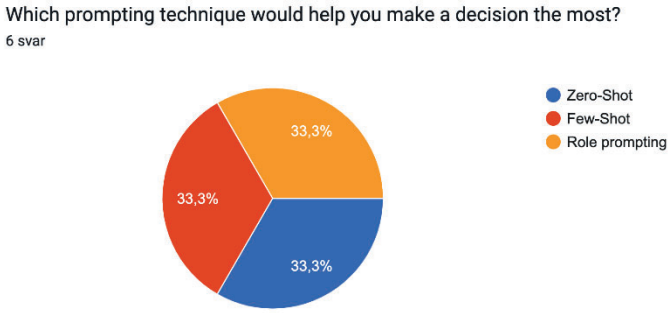


Fig. 24, Pie chart of responses for which prompting technique that gave the most decision helping response for Nudge B.

The pie chart in Fig. 24 shows that the prompting techniques: Zero-Shot, Few-Shot and Role Prompting, each received 33.3% (2 votes), for which technique would help the most in making a decision.

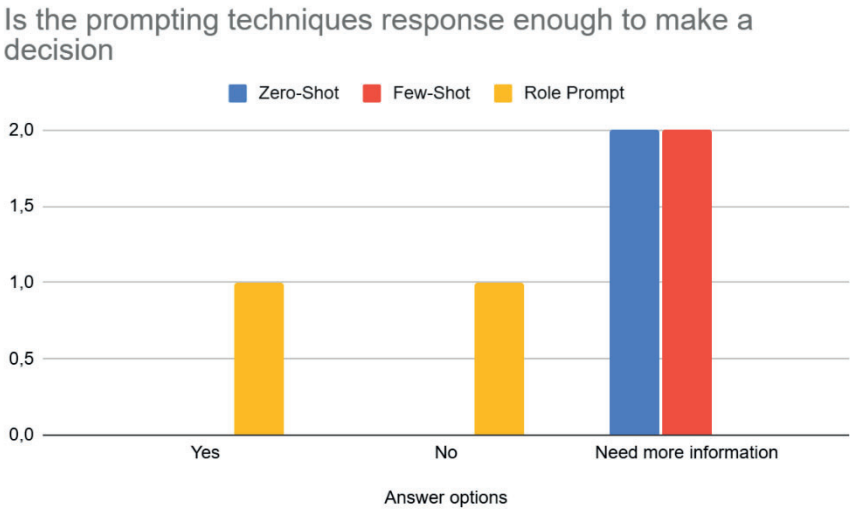


Fig. 25, Pie chart of responses of if the LLM response could be used to make a decision for Nudge A.

In Fig. 25 it is displayed that for both Zero-Shot and Few-Shot (4 votes, 2 for each) the participants chose “Need more information” for

if they felt like the prompting techniques response would be enough to make a decision. For Role Prompting both “Yes” (1 vote) and “No” (1 vote) were chosen for if the response would be enough to make a decision.

5. Discussion

This chapter contains the discussion of the results, the final prompt templates, and how the portal integration went. Additionally, it goes into the interpretation of the results from the usability testing and reflections over the chosen methodology used for this thesis. Discussions are made surrounding the limitations of the thesis, future work and reflections of some ethical aspects in regard to AI.

5.1. Interpretation of Results

This section will describe the interpretation of the results. Specifically, the implementation results, such as the prompt templates and the portal integration. It will then discuss the interpretation of the usability test results.

5.1.1. Interpretation of Implementation Results

This section will present the authors interpretations and discussions of the implementation results, which includes the developed prompt templates and the portal integration.

Prompt Templates

Prompt templates were developed to implement the three chosen prompting techniques: Zero-Shot, Few-Shot and Role Prompting. The implementations were integrated into the self-service portal, resulting in a prototype of the portal capable of displaying LLM-generated summaries of the nudge experiment result. The displayed response showed that the prompt templates fulfilled their purpose of enabling LLM-based summarizations. In addition, the templates gave varying responses differing in clarity and consistency.

Throughout the evaluation of the responses, potential refinements were identified that would improve the prompts templates. For the Few-shot technique, a more thorough investigation into the optimal number of provided examples to the LLM could be carried out. To find a balance between example quantity and response quality.

A notable challenge was encountered during the development of the Role Prompting template was the calibration of the specified tone. The tone specified was “happy, helpful and enthusiastic, in the style

of IKEA.". The LLM exhibited significant sensitivity for word "enthusiastic". Without it, the response became remarkably flat while with it the response became at times overly enthusiastic. Further investigations in regard to the tone, represents a clear avenue for improvement.

Regarding the Zero-Shot prompt, it generally performed good. However, its responses showed a certain degree of inconsistency, suggesting that further investigation in the clarity of instructions provided could lead to more consistent outputs.

The first creation of the prompt templates was carried out with ease, but as instructions were added it became clear that the LLM needed structure and clearer instructions. Testing with varying levels of directive language, ranging from ("you should", "you must" and "you MUST always"), showed a trend of more consistency of the LLMs ability to follow the instruction. At times it still deviated from to the given instruction which warrants further investigation into which words the LLM prioritizes when following instructions. It is worth noting that when an instruction was added that detailed the data, it was observed that the LLM were less due to misinterpret the data. This was shown for both the Few-Shot and Role Prompting techniques, which both contained a detailed explanation of the data.

Portal Integration

Integrating the LLM service with the portal was successful. The prototype automatically uses the selected prompting technique to send in a prompt to Vertex AI and displays the generated response. This shows that the developed LLM service can be integrated to the existing self-service portal, contributing to enhanced user experience by automating and summarizing complex data. The portal integration included both frontend and backend development, which was successful. However, there are certain areas that still offer room for improvements.

One of the improvement areas relates to caching the response to avoid inconsistency with the frontend. As the thesis wanted to explore different prompting techniques and compare the responses, the response was not cached to the page. If the LLM service were to be fully implemented caching the response for the session would be suitable. To ensure that the user only needs to read the LLM

summarization once and reduce the risk of the LLM responding differently while the user is coming to a conclusion.

Integrating the LLM summarization service into the portal using controllers or services enabled quick development and individual development of each part individually. This enables future altercation to the service or replacing the service used without impacting the functionality of the summarization.

The integration went mostly smoothly but a few problems occurred with the Google Cloud SDK along the way, this occurred even after trying to mitigate errors by performing the set up in advance. These issues were solved by reconfiguring the Google Cloud SDK but could be further investigated to find the underlying causation.

5.1.2. Interpretation of Usability Test Results

This section will present the authors interpretations of the results from the usability test. Discussing the results for which prompting technique is the most comprehensive, helpful and decision helping based on the usability test results.

Most Comprehensive Prompting Technique

The results from the usability test shows that Few-Shot prompting was perceived as the most comprehensive technique. The technique was selected by 66,7% of the participants for Nudge A and by 50% for Nudge B, indicating it's perceived as the most comprehensive choice. This suggests that including examples within the prompt template help generate a more comprehensive response. In comparison, 33,3 % of participants selected Role Prompting to have given the most comprehensive response for Nudge A and 33,3% for Nudge B, while Zero-Shot prompting received no votes for Nudge A and 16,7% for Nudge B (see Section 4.2.1, Most Comprehensive Prompting Technique). This could imply that the prompting techniques that lack contextual examples proves to be less effective in generating a comprehensive response.

To further interpret the results, it is needed to consider the participants selected reasons to why they perceived a prompting technique as the most comprehensive. The results shows that the participants selected all the listed reasons for why Few-Shot gave the

most comprehensive response (see Section 4.2.1, Most Comprehensive Prompting Technique). The reason with the most votes being “Its structure was easy to follow”, which was selected 7 times across Nudge A and Nudge B. Other common reasons selected included “It highlighted key points clearly” (6 votes combining Nudge A and Nudge B and “It’s summary made the data points easy to understand” (6 votes combining Nudge A and Nudge B). This suggests that a prompting technique that focuses on contextual examples aids comprehensiveness by allowing for a more structured and data point focused response.

In comparison, the reasons selected for Role Prompting being the most comprehensive technique were fewer as it was not the most comprehensive technique. However, it still showed some strengths on the results it received. The participants voted the reason “It highlighted key points clearly” (4 votes across Nudge A and B) and “Its summary made the data points easy to understand” (4 votes across Nudge A and B), (see Section 4.2.1, Most Comprehensive Prompting Technique). Relative to the most picked reason for Few-Shot being “Its structure was easy to follow” which was one of the lesser picked reasons for Role Prompting, having only been selected as a reason 3 times. This implies that even if they data points were easier to understand and better highlighted, the role prompting-generated response lacked structure which could be due to its lack of contextual example. Additionally, Zero-Shot was the least selected technique for comprehensiveness. This further suggests that a prompting technique lacking contextual examples may lead to a less clear response.

To summarize, from the usability test Few-Shot was picked as the option generating the most comprehensive response by the participants. The main reasoning being it provided a response with good structure. In contrast, Role Prompting showed potential by being better in data interpretation and highlighting key points. Zero-shot was the least selected, emphasizing the importance on including example-based input in the prompt.

Most Helpfulness Prompting Technique

The results from the usability test shows that the Few-Shot prompting was mostly perceived as the most helpful technique. The technique was selected by 66.7% for Nudge A but only 33.3% for

Nudge B for a total of 6 votes across both nudges, (see Section 4.2.2, Most Helpful Prompting Technique). This shows that it could be perceived as the most helpful technique. With the most common reasons selected being “Its structure was easy to follow”. To quote from the usability test “The text was split in good sized parts that made sense.”, supporting that the structure given as an example to the LLM made sense. This could be indicating that having a structure that easily can be followed is most beneficial in the response helpfulness.

For Nudge A, Few-Shot all 4 participants that chose Few-shot all selected the same reasons in: “Its wording was simple and clear”, “Its structure was easy to follow”, “It highlighted key points clearly” and “Its summary made the data points easy to understand”, (see Section 4.2.2, Most Helpful Prompting Technique). In contrast to Nudge B where Few-Shot did not receive as many selected reasons to why it was helpful, which could indicate the LLMs inconsistency in the output, even with a provided output example. Which in turn could support the idea of including additional output examples for the prompt to further make the LLM be more consistent with the output.

It is worth noting that Role Prompting technique received a combined 4 votes across both nudges, 33.3% (2 votes) in both nudges (see Section 4.2.2, Most Helpful Prompting Technique). In contrast to Few-Shot the most selected reason was “It highlighted key points clearly”. This indicates that even if the structure may not be as easy to follow as the Few-Shot technique, having clearly highlighted the key points makes the response helpful. What is worth noting is that the Role Prompting received total of 4 votes (2 for each nudge), for the reason “It was free of confusing or vague phrases”, compared to the 1 vote for the Few-Shot technique. This could be interpreted as that Role Prompting being freer in its choice of structuring the output, which could be generating fewer confusing phrases. It could also be interpreted as that the output example for the Few-Shot technique is not providing a good example to generate clear phrases. Which also supports the idea of potentially giving the Few-Shot technique a few more examples to work with.

An interesting observation with Nudge B compared to Nudge A, is that the Zero-Shot prompting technique was chosen twice (33.3% of the votes) compared to 0% for Nudge A. The reasons selected for the Zero-Shot response for Nudge B was the reason “Its sentences

were short and easy to follow" (2 votes), suggesting that keeping the response short and concise could result in a more helpful response.

To summarize Few-Shot could be perceived as the most helpful prompting technique mostly due to its structure while providing a decent summary. It is worth noting that it cannot be certain as it did not get a majority for Nudge B. In comparison Role Prompting showed potential in its helpfulness due to its perceived strength in keeping both data points and phrases clearer. Zero-Shot was the least selected, which can conclude that providing more information about the data to the LLM proves beneficial for the LLM to generate a helpful response.

Most Decision Helping Prompting Technique

The findings from the usability test showed which prompting technique the participants selected as most helpful. 66.7% of participants selected Few-Shot for Nudge A and 33.3% for Nudge B. In addition, Role Prompting was selected 33.3% for both Nudge A and B, and Zero-Shot was not selected once for Nudge A but selected 33.3% for Nudge B, (see Section 4.2.3, Most Decision Helping Prompting Technique). Overall, the results from Nudge A indicates that the most helpful prompting technique was Few-Shot, however due to the even split between votes for Nudge B makes it difficult to draw a clear conclusion.

To elaborate on the result, when looking across the results from Nudge A and B, no prompting technique is clearly leading in terms of decision helping. This may imply that the selection of which prompting technique that is the most decision helping, is dependent on the nudge data. In specific, it could be due to the complexity of the data metric. The prompting techniques may not be suitable for the data and fails to include clear enough instructions to the LLM on the complexity of the metrics.

In Fig. 23 and Fig 25 (see Section 4.2.3, Most Decision Helping Prompting Technique), the most selected answer was "Need more information" for all prompting techniques, regarding if the prompting technique generated a response enough to support a decision. This further suggests that the prompting techniques and templates are not sufficient enough in helping the user to make a decision.

5.2. Reflection of Methodology

Reflection of the methodology used to carry out this thesis include emphasis on the structured workflow cycle: pre-studies, local development, portal integration, development of prompt templates, usability testing and evaluation.

5.2.1. Overall Workflow

Having a defined workflow structure with lined out phases in logical order helped make the thesis work be feasible and structured. The defined phases being: pre-studies, local development, portal integration, usability testing and evaluation, made the workload structured and more manageable. Reflecting back on the phases, one key point was having the local development phase before portal integration. It is generally common practice to make a most viable product outside the original code base, as it allows for early error detection. Most importantly, it helps in reducing potential risks and technical difficulties when integrating to the existing system's codebase. In this thesis having local development before portal integration helped in reducing authentication issues within the self-service codebase. Moreover, it allowed portal integration phase to specifically focus on frontend and backend development for the LLM service.

5.2.2. Reflection of Development

This section will present the reflections made during the different development phases, including the local development and the portal integration.

Local Development

Working in a local development for the initial phase enabled testing of a minimal viable product of the LLM summarization feature. Which yielded valuable insights of the structure of the response coming from Optimizely. While simultaneously mitigating potential technical difficulties associated with the Google Cloud SDK and Vertex AI API authentication, to enable a smoother development during the portal integration phase. Reflecting back on the local test environment it could be stated that doing the implementation in the portal's codebase, would be beneficial. Had the local test

development been developed in TypeScript as well it would have offered an additional opportunity for code refinement while going from local to the portal integration.

Portal Integration

The portal integration phase focused on developing the backend service and frontend components for the LLM summarization feature. This included developing a LLM controller and service for the backend to handle and send requests with the prompt to Vertex AI. For frontend this meant developing frontend components to display the LLM-generated response.

Overall, this phase was conducted successfully with help from the local development preparation. Additionally, the Nudging Framework team was a big help in solving any technical difficulties within the codebase. The development for the LLM service itself was fairly effective as the challenge was connecting methods and attributes within the codebase. Notably, creating the LLM backend before the frontend components seemed to have been an effective approach. Enabling usage of the different prompt templates on the portal was implemented effectively. For further reflection upon the prompt templates (see Section 5.1.1, Interpretation of Implementation Results).

5.2.3. Reflection of Usability Testing and Evaluation

This section will present the reflections of the usability test and the reflections surrounding the evaluation of the LLM response.

Reflections of Usability Testing

Overall, the usability tests went well but there are some improvements that could be made to improve the test and also the survey used. For the usability test specifically, it was noted that the 30-minute time was a bit short. Setting the usability test session to 60 minutes instead, would have given the participants a more time to answer the survey and look at the responses more.

As for the overall structure of the usability test it went well, but more tests performed would have been beneficial. Unfortunately, that was not possible due to the time constraints. An improvement could be made to the survey to avoid having to restructure the answers to be

able to display them for the selected reasons for why it was comprehensive or helpful. Further testing on more nudges could prove useful but due to the time constraint and limited nudges this was not possible.

Evaluation of LLM Response

During the evaluation of the response from the LLM it became clear that the LLM was inconsistent in its interpretation of the data. This occurred most frequently with the percentage value for the increase or decrease of a metric. Within the prompt it was informed to the LLM that the data was represented as 0.9 for 90%, which it in the majority of cases it adhered to. Once it did occur in the response it seemed to have a higher chance of reoccurring in the following responses within the same day. This error occurred more frequently during one session of the usability test, which reduced the confidence in their belief of the LLMs summary. Part of the participant quote was “The answers given by the LLM did not inspire confidence since the numbers were often all-over-the-place.”, which confirms further investigation in reducing misinterpretation of the data.

Another implication with the data and the LLM response occurred when the LLM encountered multiple metrics with the same name. This made the LLM have a hard time to differentiate between them and at times coupled them together or mixed results between the metrics with the same name. This did not occur as often once some columns were added to the table that aided the LLM to differentiate each metric more, but it still happened that it combined or mixed up the data at times.

5.3. Limitations of the thesis

Several limitations were encountered during the course of this thesis. The first limitation was the limited variety of nudges available that could be subjected to the usability testing and the cross-referencing. This limited the breadth and depth of the evaluation of the LLM responses. Access to a wider more diverse set of nudges would have enabled a more comprehensive usability testing and further testing of the LLM response.

The second limitations of the thesis was the limitation of only evaluating based on a single LLM. While this let the thesis have a

more focused approaching for the chosen model's capabilities, it did limit the potential evaluation of the responses from other LLMs. Exploring and comparing different models could prove useful and provide a more nuanced and comprehensive evaluation of the summarization task. Future research of different models could be beneficial.

The third limitation was the extent of human evaluation performed on the available data. The limited number of nudges and time constraint the possibility of evaluating the response. Further to acknowledge would be that the human evaluation was performed by the thesis authors and it could be beneficial to have more participants performing the evaluation. A larger data set and more extensive evaluations could prove beneficial to understanding the possibilities of LLM-generated summaries.

The fourth limitation occurred during the usability testing phase, in terms of both the number and background of the participants. Ideally, it would have been beneficial to perform the usability test with the actual end-user of the nudging framework and could provide more valid insights and potential improvements. However, due to the logistical constraints of time and availability it was needed to conduct the usability tests with a smaller group participant. Future studies should strive to include a wider sample of participants.

5.4. Reflection of Ethical aspects

This part discusses the ethical aspects of handling confidential information and integrity with Large Language Models. The discussion goes into the human responsibility of using LLMs and how to be cautious with sensitive data.

5.4.1. Confidential Information and AI

When reflecting on the confidential information and how it may share or be used in regards with AI. First step is to define what confidential information would represent in this context. In this reflection confidential information will contain user data and algorithms (or other code as intellectual property) and values and analysis in regard to data analysis. When it comes to these kinds of confidential information it is important to acknowledge how the AI will be using the information once it has been prompted with it. Will

the information be used to train the AI further or if it will not store the information at all.

It is important to note that confidential information can have various degrees of sensitivity. An example can be algorithms, in itself the algorithm or the function within it might not be confidential by its own nature. But given the context of what it is use case it could make it confidential. This can be important for when algorithms are provided to the AI to look for improvements or faults within the algorithm. When it comes to the sensitivity of user data it important to note that a single data point would rarely connect something to a specific user but giving an extended amount of information could be used to pinpoint a specific individual. For user data it is also important to ensure that it follows the regulations for user data protection such as GDPR. One way to ensure this would be anonymize the data to ensure that it cannot be connected to specific individual.

Reflecting on how the AI then interacts with the confidential information is key. Does the AI work locally and interacts with a database containing the information or is the AI a service provided externally and with that how is the information handled then. If the AI is an externally provided service, it can be agreed upon the provided and user how the information given to the AI will be used.

From an ethical standpoint it is important to think how what information you share to the AI and how it will be used. In regard to this thesis the data provided did not include any user data nor any code algorithms about the framework or the summarization function implemented. The only data given to the LLM could only be used to determine what kind of nudge was created, and the confidentiality could be that of the increase or decrease in performance of certain metrics.

5.4.2. LLM Responsibility – Risks of Misinterpretation

When automating and enabling summarization of data with an LLM it is important to acknowledge the model's limitations. In addition, it is also important be aware of the potential risk of the model misinterpreting the task and data. In this thesis the risk of the LLM misinterpreting the data and generating inaccurate summaries was observed during testing. Despite, developing structured prompt

templates with different techniques there's no guarantee that the LLM will always understand the data as desired and needed for the system.

From an ethical standpoint, the risk of the LLM misinterpretation is relevant when the response is used for decision-making. Inconsistent, vague, and inaccurate summaries could mislead the users to make wrong business decisions. This reinforces the importance of human oversight.

Moreover, transparency for the user about how the response is generated is needed. Making the user aware that the response is LLM-generated and adding disclaimers could be a way to ethically handle the issue.

5.5. Future Work

There are multiple areas for improvement for this thesis. One of the key ones would be broadening the usability testing by including more nudges and participants. Future studies could include usability testing on actual end-users of the self-service portal, such as e-commerce and marketing workers. This would allow for a more applicable response. Higher number of answers would enable more accurate conclusions in evaluating the different prompting techniques.

One area that could be further researched is the usage of different models for the summarization service. This could enable the usage of a model more specifically trained for summarization tasks, instead of relying on a more general LLM. A fine-tuned model specifically tuned for understanding the experiment result metrics could also be explored. Research of different models would allow for further comparisons between responses, which could further showcase improvements in the prompt templates and potentially improve the benefit of the summarization function.

Another area that could be further researched is other prompting techniques that could enhance the prompt template. Such prompting techniques could be Retrieval Augmented Generation (RAG), Prompt Chaining and more. Could be researched to explore more ways an LLM could be handling the data and generating a summary of it.

Another area that could be improved on is caching the responses on the site until it is refreshed. This would prevent repeated API calls and recurrent responses from the LLM.

6. Conclusions

This chapter goes into the conclusions that can be drawn from the research and development conducted during this thesis. This includes answering the researching topics and presenting a summary of the thesis.

6.1. Summary of the Thesis

This thesis explored how Large Language Models could be leveraged to enable data summarization to enhance user experience. In specific, summarizing nudge experiment results for IKEA's internal systems. The goal was to develop a prototype that could generate summarized responses that could help improve the user's interpretation of the data. The prompting techniques Zero-Shot, Few-Shot and Role Prompting were implemented into different prompt templates. Furthermore, the implemented prompting techniques were evaluated through a usability test.

6.2. Answer Research Questions

This section will go over and answer the research questions that were stated in section 1.5, Problems to Research, based on the results and finding of the thesis.

6.2.1. How can LLMs be Leveraged to Summarize Experiment Result Metrics?

LLMs can be leveraged to summarize experiment result metrics through different strategies, one of which is prompting engineering which is explored within this thesis. This thesis primarily explored the strategy of prompt engineering, by implementing three prompting techniques that would shape the LLM's output. The three developed prompt templates included the techniques: Zero-Shot, Few-Shot and Role Prompting, (see Section 3.4.7, Developing Prompt Templates). These implementations showcased how the clarity of instructions is crucial for tailoring the summary to the end-user. Not only is the clarity important but so is the way data is defined and presented to the LLM, alongside the instruction or output example was structured.

In this thesis the summarization feature was implemented by using an API-based integration. This approach is practical for integrating LLMs into existing platforms. In addition, the thesis explored the implementation utilizing the Vertex AI API service, to enable an LLM to generate summaries. Implementing the summarization feature using a services and controllers' architecture within the codebase provided great modularity within the code base for future investigations regarding other AI models or prompting techniques being used (see Section 5.5, Future Work).

Acknowledging and integrating user feedback is crucial when leveraging LLMs for summarization tasks. Such feedback is essential for enhancing the LLM's ability to generate comprehensive and decision helping responses for the end user. Through the usability test conducted (see Section 4.2, Usability Test Results and 5.1.2, Interpretation of Usability Test Results), it was concluded that the developed LLM summarization feature, overall was perceived as helpful and comprehensible. Although it is noted that the implemented prompting templates were not deemed sufficient for making a decision in most cases, and that the users would need more information. This shows that an iterative process for further development would be beneficial in refining the developed prompt templates.

To summarize, leveraging LLMs for summarization of experiment metrics is not a one-time setup but a feature that needs to be further improved. It requires careful instructions to the LLM through prompting, a functional service within the portal and users to help further development on the summary. This thesis explored the possibility of using Vertex AI API and various prompting techniques to enable summarization, while stating some key factors in what is needed to be able to leverage a LLM for summarizations.

6.2.2. What Is Needed to Develop a Prompt Template?

Developing a prompt template to enable LLM-generated summarizes of data requires in depth explanation of the task and data. This thesis identified that the phrasing and clarity of the task instruction was vital, as there was potential risk for the LLM to 73 misunderstand the instructions, leading to inconsistent and hallucinated summaries (see Section 5.1.1, Interpretation of

Implementation Results). To prevent this risk prompt instructions, needs to be define with precision and assertive language.

Another point that the thesis identified was that the format of the data impacted the LLM's generated response. During the phase of local development phase (see Section 3.3, Local Development), a parsed table format of the experiment metric data was implemented to replace the raw JSON data from the Optimizely API. The parsed table format only included the necessary data to avoid hallucination, and to ensure that the data followed a structured format. This structured format allowed for better interpretation of the data by the LLM. This step was key as it addressed the input size limitations and simplified the data input.

Another aspect that was identified in what is needed to develop a prompt template was defining the desired output format and structure. By including specified formats for wording and dividing up the response in sections (e.g. hypothesis, summary, conclusion), responses resulted in being more consistent and structured. Furthermore, consistent and structured responses was deemed to be an important factor in determining the effectiveness of the prompt template shown in the usability testing, (see Section 5.1.2, Interpretation of Usability Test Results).

Another point that was identified in what is needed to develop a prompting template is how the data is defined to the LLM. Providing the LLM with clear and detailed descriptions of the data proved to significantly benefit the LLMs capability of generating a comprehensive and helpful summary. This conclusion is supported by that Few-Shot and Role Prompting receiving more votes during the usability tests, (see Section 4.2, Usability Test Results and Section 5.1.2, Interpretation of Usability Test Results). Both these prompting templates included a more detailed description (see Section 4.1.1, Prompt Templates) of the data compared to Zero-Shot.

To summarize the response generated by the LLM was strongly correlated to how the prompt was implemented. Key factors being the formatting of input data, the wording and clarity of instructions, and the clarified description of the desired output response.

6.2.3. Which Prompting Technique Enhances the User Experience the Most?

Which of the prompting techniques: Zero-Shot, Few-Shot and Role Prompting that enhanced the user experience the most. Few-Shot prompting was the most selected technique in the usability test across the categories of comprehension, helpfulness, and decision support. This indicates that it was the prompting technique that was the most effective in enchaining user experience, (see Section 4.2, Usability Test Results and Section 5.1.2, Interpretation of Usability Test Results). The main reasons selected in the usability test for its selection were its clear structure, simple wording, and ability highlight key data point.

In comparison, Zero-Shot was the least preferred prompting technique, often due to inconsistent summaries and not being comprehensive enough. This was likely due to it not having a structured formative prompt and lacking examples for the LLM.

Role prompting showed strengths in its ability to highlight data points but was occasionally exaggerated and overly enthusiastic in its tone, (see Section 5.1.2, Interpretation of Usability Test Results).

In conclusion, the results from the usability test shows that prompting techniques with contextual examples such as Few-Shot prompting, generates the more effective response to help the user experience. These findings emphasize the value of providing structured contextual examples and defined instructions in the prompt to enhance the comprehensiveness, helpfulness and decision support of LLM-generated summaries.

6.3. How the results contribute to the purpose

The purpose of this thesis was to investigate the usage of Large Language Model to generate summarized descriptions of nudge experiment result metrics and integrate it into the IKEA nudge self-service portal. The successful implementation of the LLM service and its integration into the self-service portal directly contributes to the purpose by displaying LLM-generated summaries. Thus, enabling

further development and a foundation for the Nudging Framework Team's evaluation of a potential summarization service.

6.4. How the results will be used

The findings of this thesis can be used for both practical applications and further research. For the Nudging Framework Team the research can be used as a foundation for their evaluation of the benefits of a LLM summarization feature for the nudging framework. The summarization feature, as implemented within the portal is available to the Nudging Framework team as a starting point for further development. Which could be further research of prompting techniques or models to tailor the summarization to their requirements for the nudging framework.

As for academic research this thesis can be utilized for further investigations regarding LLM summarization for metric data. Where the prompting techniques and the results obtained from the usability tests could be used for comparisons or as a foundation for further testing of different models or prompting techniques.

7. References

[1] IBM, "What is an AI model?," *IBM*, [Online]. Available: https://www.ibm.com/think/topics/ai-model. [Accessed: Mar. 19, 2025].

[2] IBM, "What are large language models (LLMs)?," *IBM*, [Online]. Available: https://www.ibm.com/think/topics/large-language-models. [Accessed: May 12, 2025].

[3] A. Ronthal and P. Sonderegger, *Generative AI and LLMs for Dummies*, Snowflake, 2024. [Online]. Available: https://www.snowflake.com/resource/generative-ai-and-llms-for-dummies/ . [Accessed: Feb. 5, 2025].

[4] Google, "Vertex AI Platform," *Google Cloud*, [Online]. Available: https://cloud.google.com/vertex-ai?hl=en. [Accessed: May 12, 2025].

[5] Google, "Vertex AI documentation," *Google Cloud*, [Online]. Available: https://cloud.google.com/vertex-ai/docs. [Accessed: May 12, 2025].

[6] Google, "Method: projects.locations.endpoints.generateContent," *Google Cloud*, [Online]. Available: https://cloud.google.com/vertex-ai/docs/reference/rest/v1/projects.locations.endpoints/generateContent. [Accessed: May 12, 2025].

[7] Google, "Gemini," *DeepMind*, [Online]. Available: https://deepmind.google/technologies/gemini/. [Accessed: Feb. 25, 2025].

[8] Google, "Gemini 2.0 Flash-Lite," *DeepMind*, [Online]. Available: https://deepmind.google/technologies/gemini/flash-lite/. [Accessed: Feb. 24, 2025].

[9] Google, "What is prompt Engineering," *Google Cloud*, [Online]. Available: https://cloud.google.com/discover/what-is-prompt-engineering?hl=en. [Accessed: May 12, 2025].

[10] Microsoft, "Prompt Engineering Fundamentals," *GitHub*, [Online]. Available: https://github.com/microsoft/generative-ai-for-beginners/blob/main/04-prompt-engineering-fundamentals/README.md?WT.mc_id=academic-105485-koreyst. [Accessed: May 12, 2025].

[11] OpenAI, "Text Generation and Prompting," *OpenAI*, [Online]. Available: https://platform.openai.com/docs/guides/prompt-engineering. [Accessed: May 12, 2025].

[12] IBM. “What is Zero-Shot prompting” URL: https://www.ibm.com/think/topics/zero-shot-prompting [Accessed: 2025-05-12]

[13] Microsoft. “Zero-shot and few-shot learning”, URL: https://learn.microsoft.com/en-us/dotnet/ai/conceptual/zero-shot-learning [Accessed: 2025-05-12]

[14] IBM, “What is few shot prompting,” *IBM*, [Online]. Available: https://www.ibm.com/think/topics/few-shot-prompting. [Accessed: May 12, 2025].

[15] IBM, V. Windland and A. Gutowska, “What is role prompting?,” *IBM*, [Online]. Available: https://www.ibm.com/think/tutorials/using-role-prompting-with-watsonx-and-granite. [Accessed: May 12, 2025].

[16] Optimizely, “Get started with Experimentation REST API,” *Optimizely Developer Docs*, [Online]. Available: https://docs.developers.optimizely.com/web-experimentation/reference/overview. [Accessed: Mar. 19, 2025].

[17] Cloudomation, “What is a Local Development Environment?,” *Cloudomation*, [Online]. Available: https://cloudomation.com/glossary/what-is-a-local-development-environment/. [Accessed: May 12, 2025].

[18] S. Necula, “Exploring The Model-View-Controller (MVC) Architecture: A Broad Analysis of Market and Technological Applications,” *ResearchGate*, [Online]. Available: https://www.researchgate.net/publication/380197155_Exploring_The_Model-View-Controller_MVC_Architecture_A_Broad_Analysis_of_Market_and_Technological_Applications. [Accessed: Mar. 19, 2025].

[19] Zheng, L., “Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena”, *arXiv e-prints*, Art. no. arXiv:2306.05685, 2023. doi:10.48550/arXiv.2306.05685. [Online]. Available: https://arxiv.org/abs/2306.05685. [Accessed: May 12, 2025].

[20] W. Albert and T. Tullis, *Measuring the user experience: Collecting, analyzing, and presenting usability metrics*, 2nd ed. Elsevier, 2013. [Online]. Available: https://shop.elsevier.com/books/measuring-the-user-experience/albert/978-0-12-415781-1.

[21] M. Kuniavsky, *Observing the user experience: A practitioner's guide to user research*, Morgan Kaufmann, 2003. [Online]. Available: https://www.sciencedirect.com/book/9780123848697/observing-the-user-experience.

[22] MUI, “React modal component,” *MUI*, [Online]. Available: https://mui.com/material-ui/react-modal/. [Accessed: May 12, 2025].

[23] MUI, “Segmented controls with tabs in Joy UI,” *MUI*, [Online]. Available: https://mui.com/joy-ui/react-tabs/#segmented-controls. [Accessed: May 12, 2025].

[24] curl, “curl – Documentation,” *curl*, [Online]. Available: https://curl.se/docs/. [Accessed: May 12, 2025].

[25] React, “React – A JavaScript library for building user interfaces,” *React.dev*, [Online]. Available: https://react.dev/. [Accessed: May 12, 2025].

[26] TypeScript, “What is TypeScript?” *TypeScript*, [Online]. Available: https://www.typescriptlang.org/. [Accessed: May 12, 2025].

[27] TypeScript, “Interfaces – TypeScript handbook,” *TypeScript*, [Online]. Available: https://www.typescriptlang.org/docs/handbook/interfaces.html. [Accessed: May 12, 2025].

[28] Mozilla, “JavaScript,” *MDN Web Docs*, [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/JavaScript. [Accessed: May 12, 2025].

[29] JSON.org, “Introducing JSON,” *JSON.org*, [Online]. Available: https://www.json.org/json-en.html. [Accessed: May 12, 2025].

[30] npm, “About npm,” *npm Docs*, [Online]. Available: https://docs.npmjs.com/about-npm. [Accessed: May 12, 2025].

[31] Node.js, “Event loop, timers, and nextTick,” *Node.js*, [Online]. Available: https://nodejs.org/en/learn/asynchronous-work/event-loop-timers-and-nexttick. [Accessed: May 12, 2025].

[32] npm, “@google-cloud/vertexai package,” *npm*, [Online]. Available: https://www.npmjs.com/package/@google-cloud/vertexai. [Accessed: May 12, 2025].

Printed by Tryckeriet i E-huset, Lund 2025

LUND
UNIVERSITY

Series of Bachelor`s theses
Department of Electrical and Information Technology
LU/LTH-EIT 2025-1063
http://www.eit.lth.se