Lab 1 – CueCode: A Natural Language to REST API Framework

Diya Patel

Old Dominion University

CS411W

Sumaya Sanober

02/28/2025

Lab 1 Version 1

Table of Contents

List of Figures

List of Tables

1 Introduction

In today's quickly changing software development landscape, seamless integration of human language and machine-executable procedures has never been more important. Large Language Models (LLMs) and artificial intelligence (AI) have allowed developers to create and modify code dynamically. A major gap still exists, nevertheless, despite these developments, there is no standardized framework for directly translating natural language into structured API payloads for RESTful services. This absence eventually slows innovation, resulting in longer development times, greater employee costs, and possible inconsistent API implementation.

As part of the CS410 and CS411W coursework, Team RED proposes CueCode, an LLM-powered solution that aims to overcome this gap. CueCode greatly speeds up the development process by using advanced natural language processing (NLP) to interpret user instructions and produce API requests. CueCode combines human validation checks and business rule enforcement before executing API calls, ensuring dependability and security in contrast to current AI-driven technologies that solely focus on code generation.This issue has a significant impact on society. APIs are used by companies in a variety of industries to link apps, automate operations, and improve user experiences. However, creating API calls by hand is a difficult and error-prone process for both non-technical users and even developers. According to a 2020 Stripe study, ineffective API integration costs companies more than $140 billion a year in lost productivity. The ability to smoothly translate human intent into machine-executable API requests will become a critical innovation as the need for AI-assisted development increases.

CueCode is intended to use a prototype technique to show demonstrate its effectiveness. This prototype will demonstrate how LLMs may create API payloads, reliably parse commands, and integrate practical business logic. CueCode has the ability to speed up digital transformation

while maintaining control, accuracy, and compliance in API-driven applications by lowering reliance on manual API development.

CueCode hopes to establish a new benchmark for LLM-assisted software development with this strategy, opening the door for more user-friendly and effective API integration frameworks.

2 Product Description

CueCode is an advanced tool that converts natural language into structured REST API payloads, making it simple for web and full-stack developers to include large language models (LLMs) into their applications. Since enforcing consistent business logic and human oversight in API interactions can be difficult, many firms find it difficult to include AI-generated text in their processes.

CueCode provides an organized method for creating AI-powered API payloads, enabling developers to use LLMs while keeping command over data processing and validation. The platform is built with accessibility in mind, so developers without specific AI experience can still benefit from smart API integration. CueCode helps businesses to responsibly embrace AI by emphasizing human oversight and business logic in the building of API payloads. This reduces risks and gets applications ready for future developments in AI-driven automation.

2.1 Key Product Features and Capabilities

A web application and developer tool called CueCode converts text in natural language into structured REST API payloads. It offers a supporting service that suggests API calls depending on user input, as well as a developer site for setting up API connections. To incorporate AI-powered API generation into their apps, developers can use the CueCode client

library after uploading their OpenAPI specifications and configuring CueCode's interface with their APIs.

The first full-service framework created to translate natural language into Web API payloads for any REST API is called CueCode. CueCode guarantees structured, appropriately sequenced API calls that are in line with business logic, in contrast to other AI-powered products that focus on broad text production. It is a risk-aware approach to AI-driven automation since it also includes human oversight and validation. Furthermore, developers may incorporate AI-driven automation without the need for specialist machine learning knowledge thanks to its support for OpenAPI guidelines, which guarantees connectivity with a variety of APIs.

CueCode removes the need for developers to manually arrange and evaluate API payloads, making it easier to integrate large language models (LLMs) into business applications. It makes it possible for developers to easily integrate natural language processing into their processes, which lowers errors, boosts productivity, and makes it possible for non-experts to use AI for API interactions. CueCode assists developers in creating more dependable and secure applications by making sure API calls are properly organized and evaluated.

Many firms struggle with AI-powered automation because current LLMs do not automatically enforce structured, logical API interactions. The specific knowledge required to ensure that LLMs generate precise Web API payloads is frequently lacking among developers. By offering an easy-to-use, structured interface for converting natural language into executable API calls, CueCode fills this gap. It lowers the possibility of mistakes and security flaws by ensuring that created API calls are appropriately formed, context-aware, and in line with business standards.

2.2 Major Components (Hardware/Software)

CueCode requires GPU-optimized computing resources to perform LLM inference and natural language processing. A third-party identity provider for authentication, an Ollama service, a PostgreSQL database with pgvector, and a backend Python program make up the system. On specialized GPU hardware, Ollama 3.1+ processes large-scale language models, while SpaCy manages entity recognition and text structuring. API-related embeddings can be efficiently stored and retrieved thanks to the PostgreSQL database with pgvector. A client library for smooth application interaction, a developer portal for set up, and a backend API are all features of CueCode's software. This modular solution maintains flexibility for various REST API specifications while guaranteeing safe, structured API call creation from natural language. CueCode makes API automation easier for developers with little experience with natural language processing by offering a risk-aware AI-driven solution.

3 Identification of Case Study

For developers and end users who require a smooth method of converting natural language input into structured API payloads, CueCode was created. Among the main users are:

- Full-stack developers who must incorporate text-to-API capabilities without developing unique NLP solutions or mastering LLMs. CueCode provides an easy-to-implement solution that streamlines their workflow.
- Non-technical customers who use services driven by APIs but are not technically proficient enough to manually construct API requests. For instance, CueCode will structure a patient's basic text request for the system when they arrange a hospital appointment.

The initial test group is made up of:

- Engineers and software developers working on applications that need API interactions driven by natural language processing. Their comments will center on customisation, API compatibility, and simplicity of integration.
- End-users consumers of API-based services, including people who schedule appointments, place service orders, or obtain data. Their comments will focus on efficiency, accuracy, and usability.

CueCode has the potential to be extensively used in a variety of businesses in the future. Businesses that offer API-driven services, like financial institutions, airlines, and e-commerce platforms, could employ CueCode to enhance user experiences. Additionally, by using natural language commands, IT and automation experts might use it to expedite process automation. CueCode has the potential to become a vital tool in industries including corporate, government, and healthcare where automation and accessibility are crucial for efficiency and compliance.

Feedback will be gathered via structured interviews, user questionnaires, and practical testing sessions to guarantee a thorough assessment. Developers will be requested to share their integration experiences, any difficulties they have had, and recommendations for enhancing customization possibilities. End users will give their comments on usability, response accuracy, and the overall usefulness of CueCode in aiding their tasks.

Analysis of the received input will reveal recurring themes, problems, and opportunities for development. This information will direct future improvements, guaranteeing that CueCode develops to better satisfy user requirements. Based on user demand, the development team will rank the improvements in order of importance, concentrating on enhancing the user experience

overall, increasing compatibility, and optimizing performance. CueCode strives to become a reliable and essential tool for developers and end users alike by iterating constantly based on feedback from the real world.

4 Product Prototype Description

Figure 1

Product XYZ Major Functional Component Diagram

<Note the format of labeling the figure: figure/table number should be bold followed by a line break with the title of figure/table in italics>

[*This space intentionally left blank.*]

<Note: leaving blank space for formatting is acceptable>

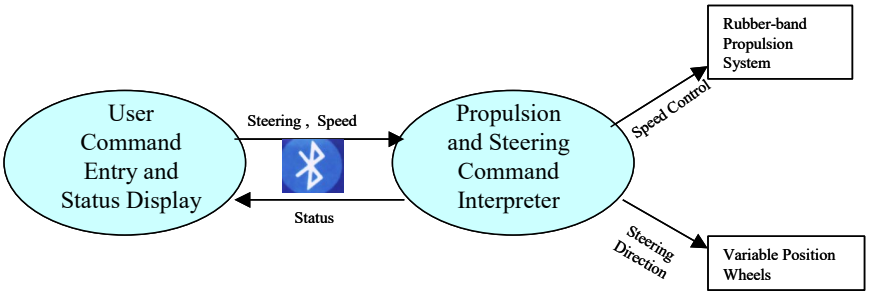

Table 1

Table of Comparison Between RWP and Prototype

Feature	Prototype	Real World Product
a propulsion system	Rubber band	Belt
fixed position wheels	Same as RWP	Same as Prototype
variable position wheels	Same as RWP	Same as Prototype
cockpit	Simulated using PC-based display and Bluetooth	Operator will use
Exhaust distribution element	Simulated on PC-display	Real combustion

<Note: A figure/table should be embedded within a section. There should be text before and after a table/figure. A section should neither start nor end with a figure/table.>

5 Glossary

- Definitions, acronyms, or abbreviations used in this document. This section should only include those terms or abbreviations that are not commonly known or are unique to your product.
- Terms should be bolded followed by a colon and the definition.

<Note: This must start at the top of a new page>

Example:

Radio Frequency Identification (RFID): an automatic identification method, relying on storing and remotely retrieving data using devices called RFID tags or transponders. An RFID tag is an object that can be attached to or incorporated into a product, animal, or person for the purpose of identification using radio waves.

6 References

Amazon Web Services. (n.d.). *What is RESTful API? - RESTful API Beginner's Guide - AWS*. Amazon Web Services, Inc. https://aws.amazon.com/what-is/restful-api/

API4AI. (2024, December 29). *API-Based Approach to Software Development | by API4AI | Medium*. Medium. https://medium.com/@API4AI/api-based-approach-to-software-development-cd51a28980eb

Bergmann, Z. (2017, May 18). *Integrating Natural Language Parsing into a MEAN stack Application*. Medium. https://medium.com/@zbbergma/integrating-natural-language-parsing-into-a-mean-stack-application-c4fe508a6794

From LLMs to LLM-based Agents for Software Engineering: A Survey of Current, Challenges and Future. (2020). Arxiv.org. https://arxiv.org/html/2408.02479v1

Haberlah, D. (2025, January 27). *Making the Right LLM API Call in Feb '25 - David Haberlah - Medium*. Medium. https://medium.com/@haberlah/making-the-right-llm-api-call-in-feb-25-a2468aa6bb9a