Project Name: CueCode

Tagline: "Safely humanize data entry, without the headache"

John Hicks

Old Dominion University

CS410

Dr. Sumaya Sanober

September 14, 2024

Problem Description

Have you ever felt like your carefully designed and elegantly implemented user interface became little more than a necessary annoyance to your end users? You're not alone.

Filling out forms and clicking buttons is a very limiting way for humans to communicate with computer systems, which is why your users will be asking for AI features, if they aren't already.

What if a natural language processing (NLP) system could generate requests to an existing Web application based on a user's natural language description of actions within an application's problem domain?

To make a Web application responsive to human language input (i.e., take meaningful action on input, such as data entry) means that the part of the system that interprets language should be aware of the structure of the Web application's data and the actions available to act upon that data. For example, the language interpretation part should be aware of what fields an API's data structures contain, also how the data structures are related to each other.

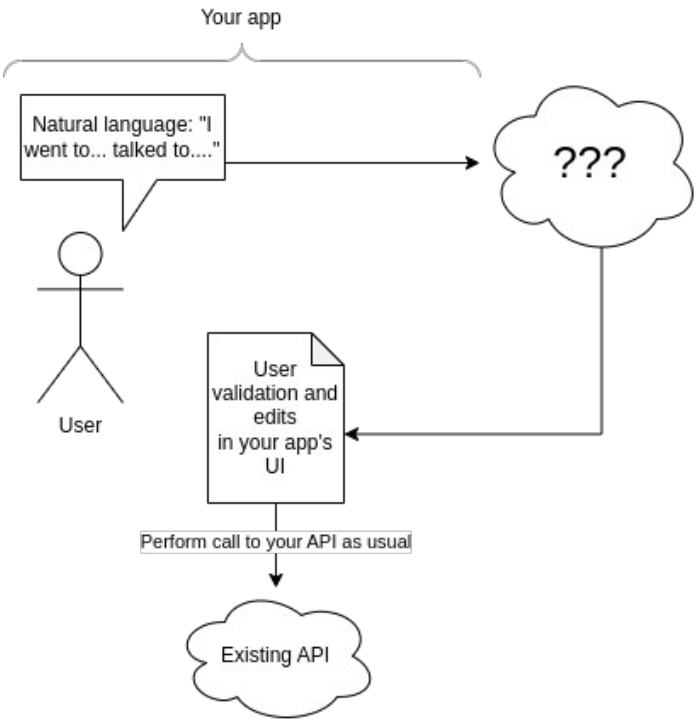

Figure 1: No application exists for rapid development of NLP features for this app.

To generate requests for an application, a Large Language Model (LLM) is the tool of choice, given the need to create content.

However, current techniques for making LLMs aware of an application's expected output structure are one-off, defined per application, lack a clearly defined concept of entity relationships, and require heavy development effort and awareness of prompt engineering and other more complex AI techniques. This is not practical for the fullstack or frontend software developer who needs to make an application act on natural language input provided by the user.

Further, LLM development tooling is failing to leverage existing machine-readable information about how to integrate with Web applications. Application Programming

CueCode – Safely humanize data entry, without the headache – Easy Language to API interface for Web developers

Interface (API) specifications provide excellent, machine-readable context to guide an LLM toward generating API requests for any given API following the specification standard.

Use of API specifications is a natural choice for developers bringing in text recognition for creating API actions to humanize their apps. However, on review, there are no tools that leverage OpenAPI or GraphQL specifications, the two most common ways to describe API capabilities.

The current state of LLM development tooling would have the developer throw away this information and start from scratch. This is an as-yet unleveraged opportunity for developer productivity when building LLM-based product features.

Solution

What if Web, frontend, and fullstack developers could make their apps understand and act on human language, without spending tons of time or money doing it?

CueCode will give developers the tools they need to turn user-generated natural language text into structured and context-aware API requests in minutes.

This is accomplished by CueCode's ability to:

1. Understand common API specifications at training time based on algorithms for parsing the information, then load that information into a vector store to augment an LLM's generation at runtime (figure 2).

2. Generate API calls at runtime, using the API specification information from the vector store (figure 3).

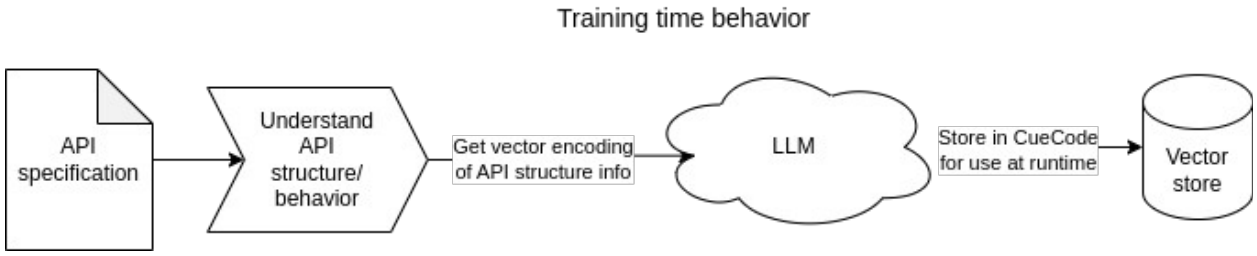

Figure 2: CueCode training behavior - done in minutes

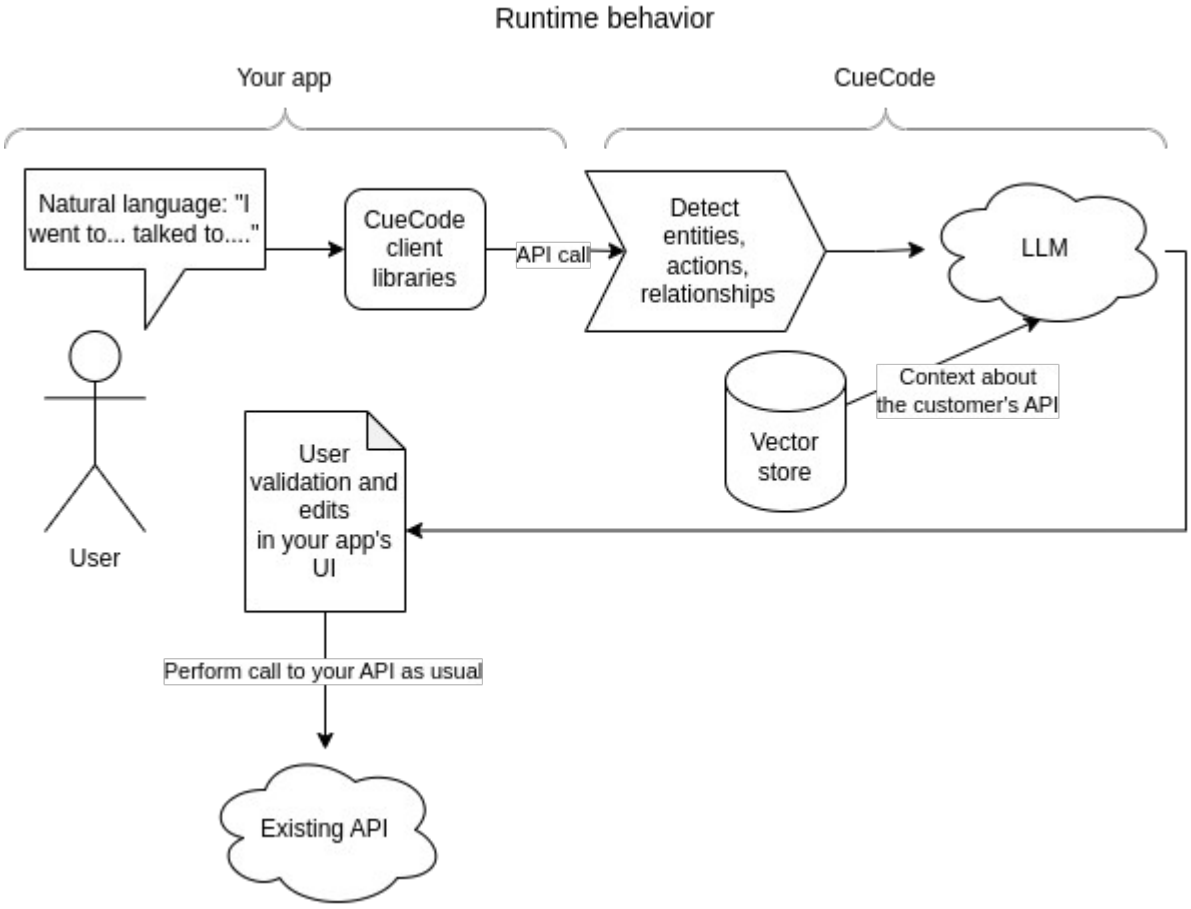

Figure 3: CueCode runtime behavior with a user-in-the-loop implementation

Key Features

- **Generate API calls based on natural language:** Web service which allows existing frontend applications to automatically to create API calls for creating, reading, and updating data in existing services from natural language descriptions.

- **Configure the generation model in minutes with an OpenAPI or GraphQL spec:** A good API specification and a few key questions are all the model needs to start generating API requests for your service. No access to your API is required for initial setup.

- **Human feedback loop:** First-class support for human-in-the-loop workflows that remove the risk of having the LLM decide when to perform data mutations. Allow users to review generated structured data objects in the customer's existing user interface, giving developers complete control over presentation. Users can modify generated data objects via the existing app's UI and, optionally, send CueCode their corrected version of

the data to fine-tune the model against the existing app's real-world usage. Whether for simple confirmation or continual training, CueCode makes it easy to put a human in the loop.

- **Integration libraries:** Leverage an augmented OpenAPI CodeGen experience to integrate client applications with the CueCode API using popular Web frameworks and libraries.

Market Potential

Who are the users?

CueCode appeals to two kinds of users:

1. Frontend and full-stack developers who need to integrate NLP into their application's data entry feature-set, but who either lack the specialized skills to implement an NLP system or do not want to take the time to do so. These are users of the development libraries and administrative interface for CueCode.

2. Any user who uses natural language to interact with a developer's application is an end user of CueCode's NLP capabilities.

Who are the customers?

CueCode's direct customers are the developers as described above. These customers need to implement natural language processing in their apps. Implementation methods include live interaction with users or calls to the CueCode API.

Natural language processing is expected to enjoy an exponentially growing market, according to Fortune Business Insights (*Natural Language Processing [NLP] Market Size | Growth, 2032*, n.d.).

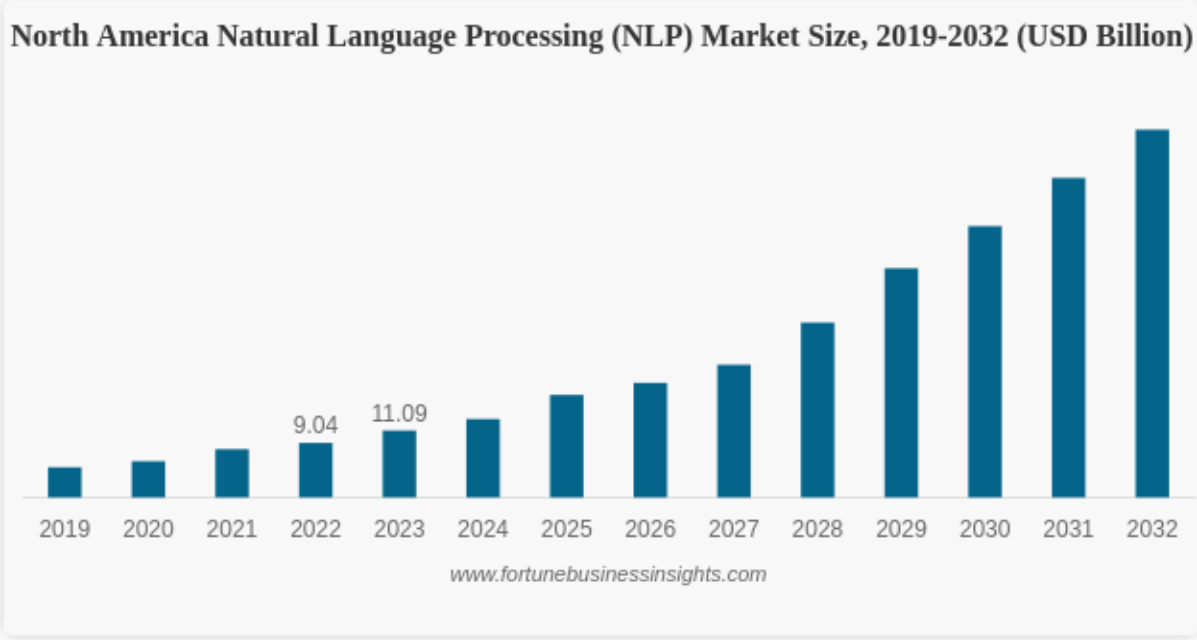

Figure 4: NLP Market Size, 2019-2032 (USD Billion)

Who are the stakeholders?

- End users who have their text translated are stakeholders, as the quality and availability of the CueCode service directly impacts them. It is also possible their natural language data will be sent to CueCode, depending on whether their app's developer has enabled the feedback loop feature.

- Developers and organizations creating and maintaining software using CueCode are also stakeholders, as they are the direct customers.

- Third-party vendors for large language technology, such as OpenAI, are also stakeholders.

- Due to the use and modification of open-source libraries such as the OpenAPI CodeGen project. open source project maintainers are stakeholders, though less strongly than the others listed above.

Competitive Landscape and Challenges

Competitors

- Tool calls in Large Language Models (LLMs) like ChatGPT allow the LLM to specify function calls that should be made in situations described to the model in prompts (*Function Calling*, n.d.).

- Google Firebase Genkit is a framework provides tools for developing and deploying backend applications that use LLMs (*Firebase Genkit*, n.d.).

- LangChain, a leading open source LLM backend development framework

What makes CueCode better?

Tools in LLMs and LangChain: While it is common now for LLMs like ChatGPT to have "tools" that allow them to collect data to augment a response or even specify a function to call with given parameters, extensive backend development work is needed to deliver this functionality to a Web frontend, much less to generate JSON payloads ready to send to an API (*Function Calling*, n.d.; *Tool/Function Calling | LangChain*, n.d.).

Further, using a Foundation model like GPT without additional context will not yield high quality results specific to the customer's API, as compared with the Retrieval Augmented Generation (RAG) approach that CueCode takes - and simplifies!

Google Firebase Genkit supports defining and enforcing expected outputs from the LLM as Python classes (*Evaluation | Genkit*, n.d.), but this is not as general-purpose or decoupled as CueCode's service-based approach is, making CueCode a plug-and-play option ready for service-oriented or microservices architectures. Using Genkit would require also require some specialized LLM knowledge on the developer's part.

LangChain, a leading open source LLM backend development framework, also supports enforcing the format of responses received from an LLM; however, this is a far cry from CodeCue's value proposition: rapid natural language integration with applications based on an API specification. Use of LangChain would require extensive knowledge of LLMs on the developer's part.

CueCode takes the non-trivial tasks of context-aware API request generation and validation and makes it easy. Even CueCode's closest competitor can only provide backend tooling, which would require significant developer effort. CueCode brings this functionality right to the frontend via easy to use client libraries.

Challenges to overcome

- Parsing API specifications into LLM vectors in a way that makes the LLM perform well at request generation given a number of factors such as relationships between entities.

- GraphQL API signatures are designed to change often (*Schemas and Types | GraphQL*, 2024), particularly with regard to field additions. So, the vector store that helps the LLM understand the "shape" of the API may need to be updated frequently and without the kind of inter-team notice that is suggested by the use of OpenAPI specifications.

- Because of the above, a practical GraphQL implementation may require understanding a GraphQL API's schema may require GraphQL introspection (*Introspection | GraphQL*, 2024; *Schemas and Types | GraphQL*, 2024), which would

require calling the API and potentially authentication issues associated with it. A minimal viable product would only need to operate on a GraphQL schema, not requiring Introspection.

- Ensure that the generated API requests remain specific to each customer, i.e., that vector stores and other data remain accessible only to their respective customers.

- Availability of online LLMs or hardware with GPUs for ODU student use during development.

Conclusion

CueCode provides the right application of developer tooling and service delivery to elevate your application's user experience from blunt form entry to a natural and delightful interaction.

This project presents the opportunity to work in an emerging set of technologies. I hope this presentation has been informative and whets your appetite to join the project. Thank you for your time and attention.

References

Evaluation | Genkit. (n.d.). Firebase. Retrieved September 14, 2024, from https://firebase.google.com/docs/genkit/evaluation

Firebase Genkit. (n.d.). Retrieved September 14, 2024, from https://firebase.google.com/docs/genkit

Function Calling. (n.d.). Retrieved September 14, 2024, from https://platform.openai.com/docs/guides/function-calling

Introspection | GraphQL. (2024, August 15). https://graphql.org/learn/introspection/

Natural Language Processing [NLP] Market Size | Growth, 2032. (n.d.). Retrieved September 14, 2024, from https://www.fortunebusinessinsights.com/industry-reports/natural-language-processing-nlp-market-101933

Schemas and Types | GraphQL. (2024, August 15). https://graphql.org/learn/schema/

Tool/function calling | LangChain. (n.d.). Retrieved September 14, 2024, from https://python.langchain.com/v0.1/docs/modules/model_io/chat/function_calling/