Lab 2 – CueCode Specification

Software Requirements Specification Outline & Expectations

Kobe Franssen

09 March 2025

Version 2

Table of Contents

Table of Figures

1.1 Purpose

This is an SRS document for CueCode, taking the form of a technical reference for system implementation later down the development process. Developers will be able to use this document to see layout of core functions with intended inputs and outputs and how they together construct CueCode.

1.2 Scope

CueCode steps in to provide a service to go from natural language to business actions such as making appointments or obtaining user specific information. A plug and play application which requires little to no changes in the business logic and infrastructure. Allowing businesses to quickly pick up on Artificial Intelligence and its benefits.

1.3 Definitions, Acronyms, and Abbreviations

API Payload (informal): Information that is sent together with an API request or response. This data, which can be organized in JSON or XML forms, usually includes the details needed by the client to comprehend the answer or by the server to carry out an action.

CueCode Developer Portal: A web-based platform that allows easy API creation with NLP-generated requests and gives developers access to CueCode's tools, API configuration, and integration workflow management.

HTTP Header: Additional metadata, such as the content type, authentication information, or caching instructions, are transmitted with HTTP requests and answers. Headers give context, which improves communication.

HTTP (Hypertext Transfer Protocol): The protocol that specifies the format and transmission of messages between web clients and servers. The type of request is determined by the HTTP methods (GET, POST, etc.).

Representational State Transfer (REST): A set of design guidelines for networked apps that use stateless, cacheable, and consistent HTTP processes to facilitate interaction. Through the use of common HTTP techniques, REST allows clients to communicate with servers by modifying resources that match an expected structure.

URL (Uniform Resource Locator): A web address that indicates where a resource is located on the internet. Protocol (such as HTTP/HTTPS), domain, and resource path are all included in URLs. They are necessary in order to access and consult internet resources.

1.4 References

[1] K. Franssen, "Lab 1," [2025], [https://john-hix.github.io/cuecode-website/written-report.html#kobe-franssen-submission].

About continuous integration with GitHub Actions. (n.d.). GitHub Docs. Retrieved October 22, 2024, from https://docs.github.com/en/actions/about-github-actions/about-continuous-integration-with-github-actions

About Git. (n.d.). GitHub Docs. Retrieved October 22, 2024, from https://docs.github.com/en/get-started/using-git/about-git

Against LLM maximalism · Explosion. (2023, May 18). https://explosion.ai/blog/explosion.ai

AppDirect | Developer Portal. (2024). Appdirect.com. https://developer.appdirect.com/

Au-Yeung, J. (2020, March 2). Best practices for REST API design. Stack Overflow Blog. https://stackoverflow.blog/2020/03/02/best-practices-for-rest-api-design/

Baker, S. (2024). Paragonsean/ChatBotAsync [Python]. https://github.com/paragonsean/ChatBotAsync (Original work published 2024)

Cloud Natural Language. (n.d.). Google Cloud. Retrieved September 26, 2024, from https://cloud.google.com/natural-language

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

HTTP headers - HTTP | MDN. (n.d.). Developer.mozilla.org. https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers

Learn Data with Mark (Director). (2023, July 26). Returning consistent/valid JSON with OpenAI/GPT [Video recording]. https://www.youtube.com/watch?v=lJJkBaO15Po

Lorica, B. (2024, January 25). Expanding AI Horizons: The Rise of Function Calling in LLMs. Gradient Flow. https://gradientflow.com/expanding-ai-horizons-the-rise-of-function-calling-in-llms/

Merritt, R. (2023, November 15). What Is Retrieval-Augmented Generation aka RAG? NVIDIA Blog. https://blogs.nvidia.com/blog/what-is-retrieval-augmented-generation/

Microsoft/prompt-engine. (2024). [TypeScript]. Microsoft. https://github.com/microsoft/prompt-engine (Original work published 2022)

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

OpenAI Platform. (n.d.-a). Retrieved September 10, 2024, from https://platform.openai.com

OpenAI Platform. (n.d.-b). Retrieved October 24, 2024, from https://platform.openai.com

OpenAPI Specification—Version 3.1.0 | Swagger. (n.d.). Retrieved September 10, 2024, from https://swagger.io/specification/

OpenAPITools/openapi-generator. (2024). [Java]. OpenAPI Tools. https://github.com/OpenAPITools/openapi-generator (Original work published 2018)

piembsystech. (2023, October 2). Dynamic Binding in Python Language. PiEmbSysTech. https://piembsystech.com/dynamic-binding-in-python-language/

Scarpati, J. (n.d.). What is a URL (Uniform Resource Locator)? SearchNetworking. https://www.techtarget.com/searchnetworking/definition/URL

SpaCy · Industrial-strength Natural Language Processing in Python. (n.d.). Retrieved September 26, 2024, from https://spacy.io/

Stanfordnlp/dspy. (2024). [Python]. Stanford NLP. https://github.com/stanfordnlp/dspy (Original work published 2023)

Su, Y., Awadallah, A. H., Khabsa, M., Pantel, P., Gamon, M., & Encarnacion, M. (2017). Building Natural Language Interfaces to Web APIs. Proceedings of the 2017 ACM on Conference on Information and Knowledge Management, 177–186. https://doi.org/10.1145/3132847.3133009

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

Tutorial: ChatGPT Over Your Data. (2023, February 6). LangChain Blog. https://blog.langchain.dev/tutorial-chatgpt-over-your-data/

Wei, J., Wang, X., Schuurmans, D., Bosma, M., Ichter, B., Xia, F., Chi, E., Le, Q., & Zhou, D. (2023). Chain-of-Thought Prompting Elicits Reasoning in Large Language Models (arXiv:2201.11903). arXiv. http://arxiv.org/abs/2201.11903

What Is NLP (Natural Language Processing)? | IBM. (2021, September 23). https://www.ibm.com/topics/natural-language-processing

What is Representational State Transfer (Rest) API? - Ampcontrol. (2024). Ampcontrol.io. https://www.ampcontrol.io/ev-terminology/what-is-rest-api

Why Visual Studio Code? (n.d.). Retrieved October 22, 2024, from https://code.visualstudio.com/docs/editor/whyvscode

W3Schools. (n.d.). HTTP Methods GET vs POST. W3schools.com. https://www.w3schools.com/tags/ref_httpmethods.asp

Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., & Cao, Y. (2023). ReAct: Synergizing Reasoning and Acting in Language Models (arXiv:2210.03629). arXiv. http://arxiv.org/abs/2210.03629

Zafin, E. at. (2023, August 15). Bridging the Gap: Exploring use of Natural Language to interact with Complex Systems. Engineering at Zafin. https://medium.com/engineering-zafin/bridging-the-gap-exploring-using-natural-language-to-interact-with-complex-systems-11c1b056cc19

1.5 Overview

The overall description will cover how CueCode works, focusing on individual features and how users interact with them, important constraints and assumptions with each their dependencies needed to implement CueCode.

2 Overall Description

2.1 Product Perspective

CueCode is an innovative framework and service designed to implement Natural Language Processing (NLP) capabilities, enabling the understanding and processing of natural language inputs. The system will offer a user-friendly interface through API client libraries, making it accessible for developers to integrate into their projects. Additionally, CueCode will provide a developer portal web application where users can upload OpenAPI specifications for their APIs and configure their CueCode service, streamlining the setup process.

The significance of CueCode lies in its unique approach to transforming natural language into Web API payloads, a functionality that is currently unparalleled for arbitrary REST APIs. This innovative solution is designed to work seamlessly with any REST API defined using an OpenAPI specification, offering unprecedented versatility and adaptability. By enabling non-technical staff and customers to interact with complex APIs using simple language, CueCode dramatically improves access to technology and enhances the overall user experience.

One of the key accomplishments of CueCode is its ability to increase efficiency in REST API interactions. By allowing quick and accurate API calls based on simple language inputs, organizations can respond to customer requests more rapidly, leading to enhanced service levels. This improved efficiency not only streamlines operations but also allows client service representatives to focus more on customer engagement rather than getting bogged down in technical details, ultimately resulting in higher customer satisfaction.

CueCode addresses a significant problem in the tech industry by making NLP and Large Language Model (LLM) generation capabilities accessible to non-specialist developers. It applies these advanced technologies to the specific challenge of converting natural language into REST API payloads. By abstracting the technical complexity involved in generating API payloads, CueCode effectively bridges the gap between human input and the technical execution of requests via REST API calls. This solution empowers a wider range of users to leverage powerful API functionalities without requiring in-depth technical knowledge, thereby democratizing access to advanced technological capabilities.

2.1.1 Major Functional Components

CueCode will require hardware capable of running the following systems separately (at most one at a time):

- Ollama 3.1, 70 billion parameter model
- Spacy.io (CPU or GPU)

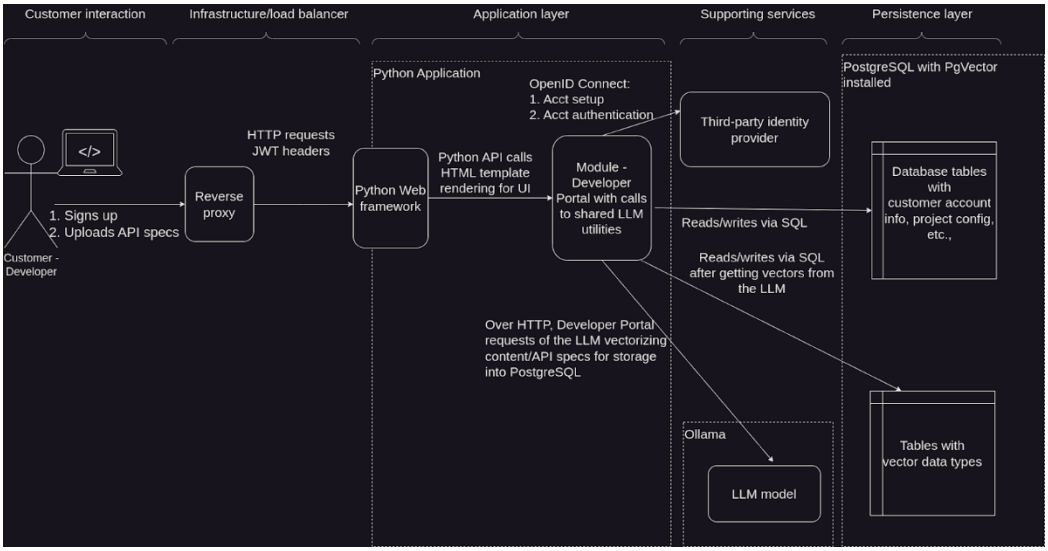

Figure 1 - Multi Functional Component Diagram

2.1.1.1 Application Layer

CueCode will have its input be on the developers website, and will then communicate with the backend using http requests and JWT headers. The CueCode webhook framework will then

trigger the requested function including authentication to a third party identity provider in Supported Services.

2.1.1.2 Supporting services

These are services which need to also be provided by the developer and include of:

- Ollama 3.1, 70 billion parameter model
- Spacio.io (CPU or GPU based with a preference to GPU)
- Third party authentication

And are required to make CueCode work as intended, most especially the LLM backend.

2.1.1.3 Persistence Layer

The persistence Layer is where all the user and company data is stored persistently. The functions from the Application Layer will interact with the persistence layer such as databases, querying the database to get required information before then querying the LLM model for a response which will be parsed and presented to the end user.

2.2 Product Functions

2.2.1 Real World Product Table

Category	Feature	Real-world product	Prototype
Portal	Interactive Login / Authentication	Web	CLI
	Account creation / deletion	✓	
Config	REST API definition management	Spec file + Web	Spec file
	Upload OpenAPI specifications	✓	✓
	REST API configuration wizard	✓	
	Allow only a subset of valid OpenAPI specs be used	✓	✓
Runtime	Process natural language and turn it into REST API payloads	✓	✓
	Map natural language to customer's data entities via search or API call	✓	
Client libraries	Integrate application with CueCode's NLP API	✓	✓
NLP monitoring	Trace, debug, and report on translation requests in CueCode Web UI	✓	
Marketplace	Share CueCode API configurations with other users	✓	

CueCode 13

Figure 2- Real World Product Table

2.2.2 Portal

The portal is for the developers to access their CueCode configuration and make any modifications required. This would come with an interactive login which triggers account creation and deletion so a team can all together work on CueCode with individual accounts for auditing purposes.

2.2.3 Config

The config is what the developer can modify in the developer portal to fine tune CueCode to their needs. This includes uploading the OpenAPI specification for the developers application, modify how CueCode should query their API server including API keys.

The “only allow subset of OpenAPI spec to be used” is a functionality to limit the scope of actions CueCode could perform. This provides a security safeguard to prevent CueCode from taking unwanted actions or the customer from doing things they would not be able to do from the developers website.

2.2.4 Runtime

This is where CueCode receives the raw user input, filters out any fillers and uses pattern recognition and the LLM to match their requests to the OpenAPI specification previously provided by the developer in the portal.

2.2.5 Client Libraries

The client library is what the developer will be using to implement CueCode into their application. It will be a library they call actions upon from their frontend and receive responses from CueCode. This will be in the format of API queries or potentially a python library depending on the developers interests.

2.2.6 NLP monitoring

A part of the portal where the developer can view recent actions performed by CueCode including trace error logging for any exceptions.

2.2.7 Marketplace

Part of what previously mentioned in client libraries where users can work in team, the marketplace will be where they can share their OpenAPI specifications and CueCode configurations for development purposes.

2.3 User Characteristics

2.3.1 Owner

The “Owner” is the individual or team developer which will manage the CueCode installation into their own environment. They need to have great knowledge into their own environment and how to assign their developers into optimizing their OpenAPI specifications to meet user requests.

2.3.2 Developers

They will be the individual developers logging into CueCode’s portal, uploading the OpenAPI specification, tracing CueCode logs and producing the applications OpenAPI specification. Producing the OpenAPI specification includes ensuring that all actions a customer should be able to make is an action CueCode can take through the API with appropriate tagging for triggers.

2.3.3 Customers

Customers are the end users who log into the developers portal and interact with CueCode. They will never visually see CueCode but will receive responses through the developers application. They will be querying CueCode using natural language explaining their wants.

2.4 Constraints

2.4.1 LLM Unpredictability

LLM’s are unpredictable and can therefore produce undesired results. As CueCode is taking action on the provided OpenAPI specification, it can perform destructive actions if the scope is not limited. We do have the feature for agents to “review” before allowing CueCode to commit actions but with this out of the loop

2.5 Assumptions and Dependencies

2.5.1 Frontend

The environment needs to have a frontend to communicate with users, including triggers for user response and a way to display CueCode responses in forms of text or success (Boolean).
This should also include authorization as CueCode can perform actions depending on the users status to the environment such as “worker” and “customer” but is optional with a default.

2.5.2 API server

An API server following the provided OpenAPI specification should be available at all times for CueCode to interact with and perform business logic. CueCode would need an API key or preferred method of authentication at that time to authenticate with the server and have no API rate limits.

2.5.3 LLM hosting

As with the current specification, CueCode purely provides the program and not the LLM model. This therefore requires for the backend to have its own or accessibility to an LLM model for CueCode to use with no limitations. CueCode is designed on ollama 3.8b and is recommended to be used with.