Lab 2 - CueCode: A Natural Language to REST API Framework

Diya Patel

Old Dominion University

CS411W

Sumaya Sanober

03/09/2025

Lab 2 Section 1&2

Table of Contents

List of Figures

1. Introduction

In today's fast-paced digital world, many interfaces often don't speak the user's language, creating a gap between how users engage with apps and how those apps actually operate. Users want regular use of natural language and seamless, intuitive experiences. However, while trying to include these functionalities in their apps, developers run into a number of challenges. CueCode is useful in this situation. With the help of this software, developers will be able to leverage the power of NLP and LLMs in a safer, more scalable manner, allowing AI applications to be designed with assurance and without needless risk. CueCode will provide nontechnical employees who can communicate with APIs in plain English.

1.1 Purpose

The CueCode system's functional requirements, technical specifications, and limitations are outlined in this Software Requirements Specification (SRS) paper. For developers, product managers, and stakeholders, it acts as an organized resource to guarantee system implementation that is feasible, consistent, and clear. The main architecture, system relationships, and both functional and non-functional needs are covered in detail in this document. Additionally, it offers instructions on how the system should behave and integrate with outside services.

1.2 Scope

In order to facilitate the conversion of natural language input into structured API payloads for RESTful services, CueCode was created. It allows developers to create API requests dynamically while maintaining adherence to OpenAPI and GraphQL guidelines by utilizing Large Language Models (LLMs). By automating the creation of API requests, avoiding errors, and lowering human labor, the system seeks to increase productivity.

CueCode's main objective is to automate the creation of API requests by transforming human-readable instructions into structured API payloads. This enhances uniformity in API implementation and cuts down on development time. The system also includes business rule enforcement and validation methods to guarantee the security and correctness of API calls prior to execution. Developers can tailor the system to meet their unique requirements, optimize NLP processing, and alter API mappings. CueCode can effectively manage multi-step API interactions because it also enables entity recognition and request scheduling.

Users will be able to fine-tune NLP mappings, modify API settings, and publish configurations to a shared marketplace through CueCode's web-based developer site. After processing input in plain language, the system will extract pertinent elements and produce API requests that meet user-specified requirements. To guarantee dependability, it will also validate API payloads and offer debugging assistance.

Nevertheless, CueCode is not intended to be a conversational AI or all-purpose chatbot. It won't take the place of conventional API documentation or do away with the requirement for clear API definitions. Furthermore, it won't host third-party APIs, offer application backend infrastructure, or manage authentication directly outside of interfacing with outside identity suppliers.

CueCode seeks to increase program stability, cut down on development time, and eliminate the inefficiencies of manual API development. It creates a methodical and effective framework for integrating AI-assisted APIs, facilitating quicker and more precise API deployment.

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

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/restfulapi/

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

Bergmann, Z. (2017, May 18). *Integrating Natural Language Parsing into a MEAN stack Application*. Medium. https://medium.com/@zbbergma/integratingnatural-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-theright-llm-api-call-in-feb-25-a2468aa6bb9a

1.5 Overview

This document is structured as follows:

- **Section 2**: Overall Description, including product perspective, functions, user roles, constraints, and dependencies.
- **Section 3 (Future Work)**: Specific system requirements and implementation details.

2. General Description

Web and full-stack developers can easily incorporate large language models (LLMs) into their applications using CueCode, an innovative tool that transforms natural language into structured REST API payloads. Many businesses find it challenging to integrate AI-generated text into their processes because it can be challenging to enforce consistent business logic and human oversight in API interactions.

Developers may employ LLMs while maintaining control over data processing and validation thanks to CueCode's structured approach to building AI-powered API payloads. Because the platform was designed with accessibility in mind, developers who lack specialized knowledge of AI can nevertheless take advantage of clever API integration. By placing a strong emphasis on human oversight and business logic while creating API payloads, CueCode assists companies in embracing AI safely.

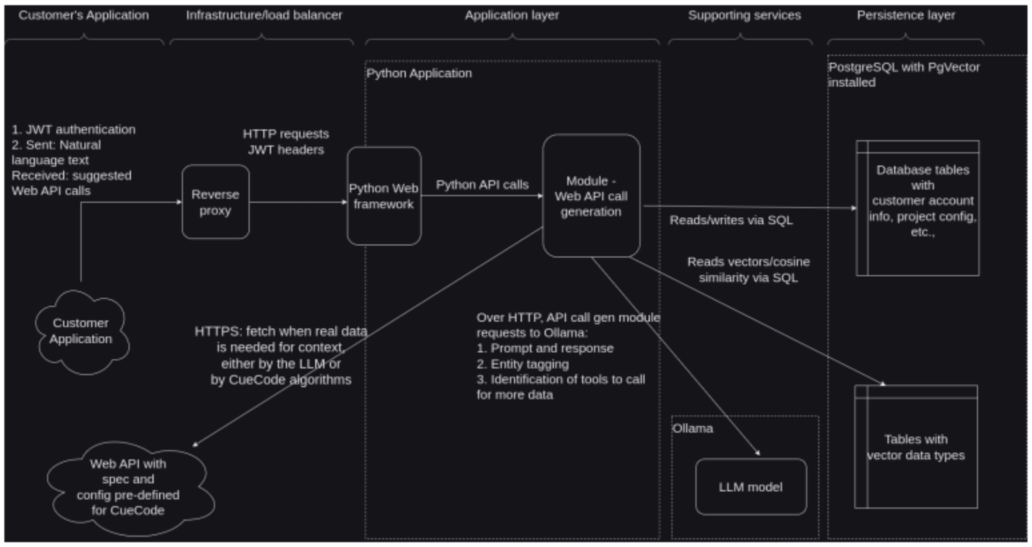
2.1 Product Perspective

CueCode acts as an intelligent middleware that bridges human natural language input with structured API calls. The system consists of the following core components:

- **Developer Portal**: A web-based interface for managing API configurations and NLP mappings.
- **Core NLP Engine**: Responsible for processing user input, recognizing entities, and generating structured API calls.
- **Reverse Proxy**: Terminates HTTPS, routes API requests, and provides load balancing if required.
- **External Identity Provider**: Manages authentication and user access control.
- **Monitoring and Analytics Module**: Tracks system performance, user queries, and API call success rates.

Figure 1

CueCode Prototype Major Functional Component Diagram


Customer's Application
Infrastructure/load balancer
Application layer
Supporting services
Persistence layer
Python Application
PostgreSQL with PgVector installed
1. JWT authentication
2. Sent: Natural language text
Received: suggested Web API calls
HTTP requests JWT headers
Reverse proxy
Python Web framework
Python API calls
Module - Web API call generation
Reads/writes via SQL
Database tables with customer account info, project config, etc.,
Reads vectors/cosine similarity via SQL
Customer Application
HTTPS: fetch when real data is needed for context, either by the LLM or by CueCode algorithms
Over HTTP, API call gen module requests to Ollama:
1. Prompt and response
2. Entity tagging
3. Identification of tools to call for more data
Ollama
LLM model
Tables with vector data types
Web API with spec and config pre-defined for CueCode

Major functional components diagram, focusing on the customer's application interacting with CueCode and components involved in the API payload generation process.

2.2 Product Functions

CueCode provides the following core functionalities:

- **Natural Language to API Mapping**: Converts user input into structured API calls based on OpenAPI/GraphQL specifications.
- **API Configuration Management**: Allows developers to fine-tune API mappings and optimize NLP processing.
- **Entity Recognition & Lookup**: Identifies relevant entities within input text and maps them to API parameters.
- **Automated API Request Sequencing**: Handles dependencies between API calls when multiple steps are required.
- **Response Validation & Error Handling**: Ensures API requests conform to expected formats and provides debugging support.
- **Marketplace Integration**: Enables sharing and discovery of pre-configured API mappings.

Implementation Status:

- **Fully Implemented**: Basic NLP parsing, API call generation, and developer portal UI.
- **Partially Implemented**: Entity recognition enhancements, intelligent error handling.
- **Planned for Future Development**: Advanced model fine-tuning, user-configurable NLP pipelines.

.

2.3 User Characteristics

CueCode serves multiple user roles:

- **Developers**: The primary users who integrate APIs and manage configurations.
- **Marketplace Publishers**: Users who share API configurations for public or private use.
- **CueCode Administrators**: System maintainers who oversee marketplace security and performance monitoring.
- **Customer End-Users**: Indirect users who interact with applications that utilize CueCode-generated API calls.

Users are expected to have a basic understanding of API development. Advanced users may configure NLP models and API workflows for optimized performance.

2.4 Constraints

CueCode operates under the following constraints:

- **Performance**: NLP request processing time should not exceed 500ms.
- **Security**: Authentication is outsourced to a third-party provider, and all data is encrypted in transit.
- **Infrastructure**: The system relies on cloud-based hosting services for deployment and scalability.
- **Compliance**: Must adhere to API specifications (OpenAPI, GraphQL) and applicable data privacy regulations.

2.5 Assumptions and Dependencies

CueCode assumes the following dependencies:

- **Third-Party APIs**: Users must provide valid OpenAPI/GraphQL specifications for proper operation.

- **Identity Provider**: Authentication requires integration with a third-party service (e.g., OAuth, SAML).
- **Hosting Environment**: The system is designed for deployment in a cloud-based infrastructure such as AWS, Azure, or Google Cloud.
- **External NLP Libraries**: Some functionalities may depend on third-party NLP libraries for entity recognition and language processing.