Web Tool for Phonemes Technical Design Document

1. Tech Stack

Language

Python

BackEnd

1. FastApi - A web framework used by python for API development.
2. Flask - A microservice framework used by python for developing Web Applications

Front End

1. React - An open-source javascript library for building user interfaces based on components.
2. HTML & CSS

Database

Apache Solr - Solr contains features for full-text search and real time indexing which is very crucial for this project as we have to search for phonemes. Solr has NoSQL features.

Source Code Control

GitHub

APIS

1. We will scrape data from Wiktionary
2. NLTK for phoneme data
3. WordNet, SemCor , BabelNet, FrameNet APIs for categories & semantics

IDE

IntelliJ

We don't need a specific version for any techstack mentioned above anything that's basic should be fine

2. Accounts and Infrastructure

2.1 Development

Infrastructure: For the development phase, each developer will use their local machines for developing and testing the functionalities. This allows for a flexible and personalized development environment.

Machine specifications: Local machines with a minimum of 8 GB RAM, and a quad-core processor.
Operating Systems: Windows/Mac/Linux
Access Control: We will have access to the central GitHub repository where the source code is maintained.

URLs/Endpoints for API Testing: Localhost endpoints will be used for internal testing. Tools like Postman can be utilized for API endpoint testing.

2.2 Production

Depending on the availability of the production servers we will either deploy it onto AWS or IBM servers

Data Sources, Models, Timing

This section outlines the sources of data for the project, their origins, and the methods for obtaining and maintaining them, the data models

1.1 Data Sources

Graphemes: Data for graphemes, which represent the written form of words, is obtained through web scraping from Wiktionary using tools like BeautifulSoup or Scrapy. The data is subsequently cleaned and processed for our analysis.

https://en.wiktionary.org/wiki/university

Phoneme Data: Phoneme data, consisting of approximately 100,000 words, is accessed from the Natural Language Toolkit (NLTK). NLTK serves as a comprehensive linguistic data repository, enriching our phonetic analysis.

https://www.nltk.org/

Articulation Rules: Articulation data follows predefined rules and guidelines. These rules are utilized to generate articulation data as required for the project. Additionally, an external dataset is available from the ipa-data repository.

https://github.com/AdamSteffanick/ipa-data/blob/master/guid-o-matic/ipa-data/ipa-data.csv

Categories: Data related to word categories is sourced from various linguistic resources, including WordNet, SemCor, BabelNet, and FrameNet. These datasets provide valuable insights into the semantic categorization of words.

http://globalwordnet.github.io/schemas/
http://wordnetweb.princeton.edu/perl/webwn

Semantics: The project leverages WordNet for semantic-related data. WordNet is a valuable resource for exploring the semantic relationships between words and is also used to identify the parts of speech.

http://wordnetweb.princeton.edu/perl/webwn

These data sources collectively contribute to the project's functionality, content, and linguistic analysis. Data extraction methods, maintenance, and integration are vital considerations in ensuring the project's success.

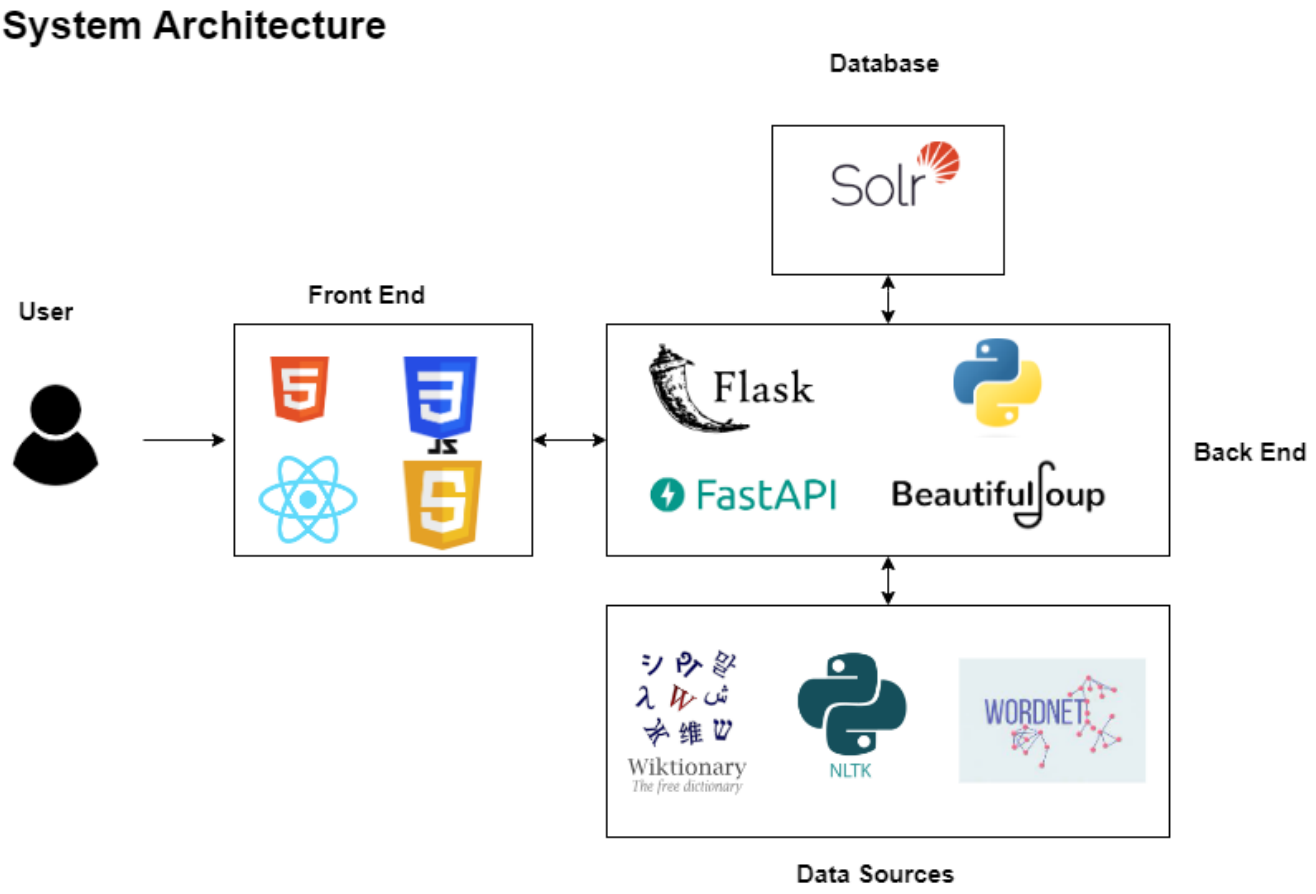
1.2 Data Models and Structure

We have numerous fields associated with a single word, including multi-valued fields. This prompts us to lean towards NoSQL instead of SQL. Our preference for Solr stems from its ability to facilitate faceted search and provide an extensive range of querying features, making it particularly effective for situations where fields contain multiple values.
While NoSQL databases offer flexibility and adaptability for managing data across various data models, Solr's strength lies in its specialized capabilities for text-based search and retrieval, particularly within multi-valued fields. Our primary objective is to conduct efficient and robust text-based searches within multi-valued fields, Solr often proves to be a more suitable choice compared to NoSQL databases. Therefore Solr is preferred in this case.
Solr excels when it comes to searching within multi-valued fields or arrays, especially in comparison to many NoSQL databases. Here are some benefits of opting for Solr over NoSQL databases when dealing with searches in multi-valued fields

1.3 Timing

Typically, we utilize data gathered from diverse sources, with each word possessing various attributes like phonetics, semantics, complexity, and more. Our usual practice involves writing and executing scripts to obtain information from either a single or multiple sources. As we move forward, our plan is to incorporate additional fields as required, ensuring that these fields are populated with relevant data from the appropriate source. We will then modify Solr's schema and reload the data into Solr. This data remains stored in Solr indefinitely, and we repeat this process whenever we need to introduce new fields and populate them with data in Solr.

System Architecture Diagram



Deployment Methodology

All the Data stored in the development server will be eventually moved to the production server. The production server details will be provided by professor at later stage