**Capstone Project Documentation: Code Reviewer**

**Overview**

The objective of this project is to build a Generative AI-based code reviewer that assists development teams by automatically providing the following features:

**Key Features**

* **Code Quality Assessment**: Detects code issues or bugs.
* **Code Optimization**: Optimizes code and ensures compliance with best practices.
* **Auto Code Documentation**: Automatically generates documentation for the code.
* **Code Reframing**: Generates different code snippets to meet requirements and reduce complexity.

**Architecture**

**Data Layer / Data Lake**

* **Historical Code Review Data**: Stores past code reviews and version control data.
* **Version Control Integration**: Connects with platforms like GitHub for real-time data.

**Data Processing**

* **Python Platform**: Used for preprocessing and feature extraction.
* **Machine Learning Algorithms**: Applied for code quality assessment and bug detection.

**AI Components**

* **Gen AI Models**: Employed for code documentation and reframing.
* **Langchain Platform**: Supports structured documentation and code transformation.

**Implementation Steps**

1. **Data Collection**: Gather historical code review data from version control systems.
2. **Data Preprocessing**: Process data using Python for analysis.
3. **Quality Assessment**: Use machine learning models to detect code issues.
4. **Documentation Generation**: Automatically create documentation for the code.
5. **Code Reframing**: Generate optimized and reframed code snippets.

**Tools and Technologies**

* **Python**: For data preprocessing and feature extraction.
* **Machine Learning Libraries**: Used for bug detection and optimization.
* **Generative AI Frameworks**: For code documentation and reframing tasks.

**Evaluation Metrics**

* **Bug Detection Accuracy**: Measure the precision of bug identification.
* **Optimization Effectiveness**: Evaluate the improvement in code quality.
* **Documentation Clarity**: Assess the usefulness of generated documentation.

**Tentative Data Schema for Code Reviewer Project**

This schema serves as a guideline and can be adjusted based on specific project needs.

1. **Code Review Table**
   * **review\_id**: Unique identifier for each code review (Primary Key)
   * **timestamp**: Date and time of the code review
   * **code\_snippet**: The specific code being reviewed
   * **issues\_detected**: List of detected issues or bugs
   * **recommendations**: Suggested improvements or fixes
2. **Version Control Table**
   * **commit\_id**: Unique identifier for each commit (Primary Key)
   * **author**: Name of the developer who made the commit
   * **timestamp**: Date and time of the commit
   * **changes**: Summary of the code changes
3. **Documentation Table**
   * **doc\_id**: Unique identifier for each documentation entry (Primary Key)
   * **review\_id**: Associated code review identifier (Foreign Key)
   * **content**: Generated documentation content
   * **timestamp**: Date and time when documentation was created
4. **Optimization Table**
   * **opt\_id**: Unique identifier for each optimization entry (Primary Key)
   * **review\_id**: Associated code review identifier (Foreign Key)
   * **optimized\_code**: Optimized version of the code
   * **performance\_metrics**: Metrics indicating the improvement in code performance

**Note**: This data schema is tentative and can be adapted to fit different requirements or data sources. Graduates are encouraged to modify it as needed to address specific challenges in the project.

A diagram of a computer

AI-generated content may be incorrect.

*The shared architecture flow diagram is intended to serve as a reference for building the solution. It can be adjusted or accommodated as needed, and some elements may require further exploration and adaptation. The instructions in the capstone projects act as guidance to support your development process.*