DATA VALIDATION SYSTEM REPORT

Advanced Programming

Mmakgotso Thejane

09 October 2024

Table of Contents

1. Introduction

2. System Design and Architecture

- 2.1 Overview

- 2.2 Components

3. Data Validation Mechanisms

- 3.1 Types of Validation

- 3.2 Implementation Techniques

4. Implementation Details

- 4.1 Technology Stack

- 4.2 Code Structure

5. Testing and Debugging

- 5.1 Testing Strategies

- 5.2 Debugging Techniques

6. Conclusion

7. References

1. Introduction

This report outlines the design, architecture, and implementation of a Data Validation System (DVS). A Data Validation System is a system implemented to help organisations/ individuals validate the data in possession. A Data Validation System is crucial for ensuring data integrity, quality, and reliability in applications that handle significant amounts of data.

2. System Design and Architecture

2.1 Overview

The Data Validation System is designed to validate received data against predefined rules to ensure it meets specific criteria before it is processed or stored. This architecture emphasizes on the segmental, scalability, and maintainability of data

2.2 Components

- **Data Ingestion component**: Responsible for receiving and preprocessing incoming data.

- **Validation Engine**: Core component that applies validation rules, ensuring that the data receives adheres to the requirements.

- **Error Handling Component**: highlights invalid data and reporting mechanisms.

3. Data Validation Mechanisms

3.1 Types of Validation

- **Format Validation**: Ensures that the data entered adheres to the formats set.

- **Range Validation**: validates if the data entered for specific fields adheres to the range set.

- **Type Validation**: This type of validation speaks to the data types defined for each text field.

- **Uniqueness Validation**: Ensures that records are unique where required.

3.2 Implementation Techniques

- **Regular Expressions**: Used for format validation.

- **Boundary Checks**: Implemented for range validation.

- **Data Type Checks**: Using built-in functions or libraries for type validation.

- **Crossfield Checks**: Custom logic for consistency and uniqueness checks.

4. Implementation Details

4.1 Technology Stack

- Programming Language: Java
- Framework: NetBeans for API development
- Testing: NetBeans console for testing and reviewing

4.2 Code Structure

- **src**: Refers to the main application code
  - **Validation**: Validation rules and core components of the system
  - **Error handling**: Logic for handling validation errors
  - **Reporting**: Reporting functionalities
- **Tests**: Unit tests and integration tests
- **Config**: Configuration files for the systems environment

5. Testing and Debugging

5.1 Testing Strategies

- **Unit Testing:** Each component is tested independently using NetBeans console
- **Integration Testing**: Tests the interaction between different components of the system.
- **End-to-End Testing**: Simulates real-world scenarios to validate the system as a whole.

5.2 Debugging Techniques

- **Logging**: Implement comprehensive logging to track validation processes and identify issues.
- **Interactive Debugging**: Using tools like IDE debuggers to step through code.

- **Code Review**: Conducted regular reviews to identify potential issues before deployment.

6. Conclusion

The Data Validation System provides a robust solution for ensuring data quality and integrity. By implementing various validation methods and a well-structured architecture, it enhances data reliability in applications. The implementation details, along with regular and focussed testing and debugging strategies, ensure that the system operates efficiently and effectively.

7. References

-https://stackoverflow.com/questions/64298538/config-file-for-a-java-application

https://en.wikipedia.org/wiki/Software_engine#:~:text=A%20software%20engine%20is%20a,widespread%20in%20the%20software%20industry.

https://corporatefinanceinstitute.com/resources/data-science/data-validation/#:~:text=Data%20validation%20refers%20to%20the,of%20input%20and%20stored%20data.