### **Scala Project Documentation**

This document provides an overview and documentation for the functional programming project developed for the retail store's rule engine. The project is implemented in Scala, adhering to functional programming principles. The core logic of the rule engine is designed to calculate discounts for orders based on various qualifying rules. The project also includes database integration and logging functionality.

#### **Project Overview**

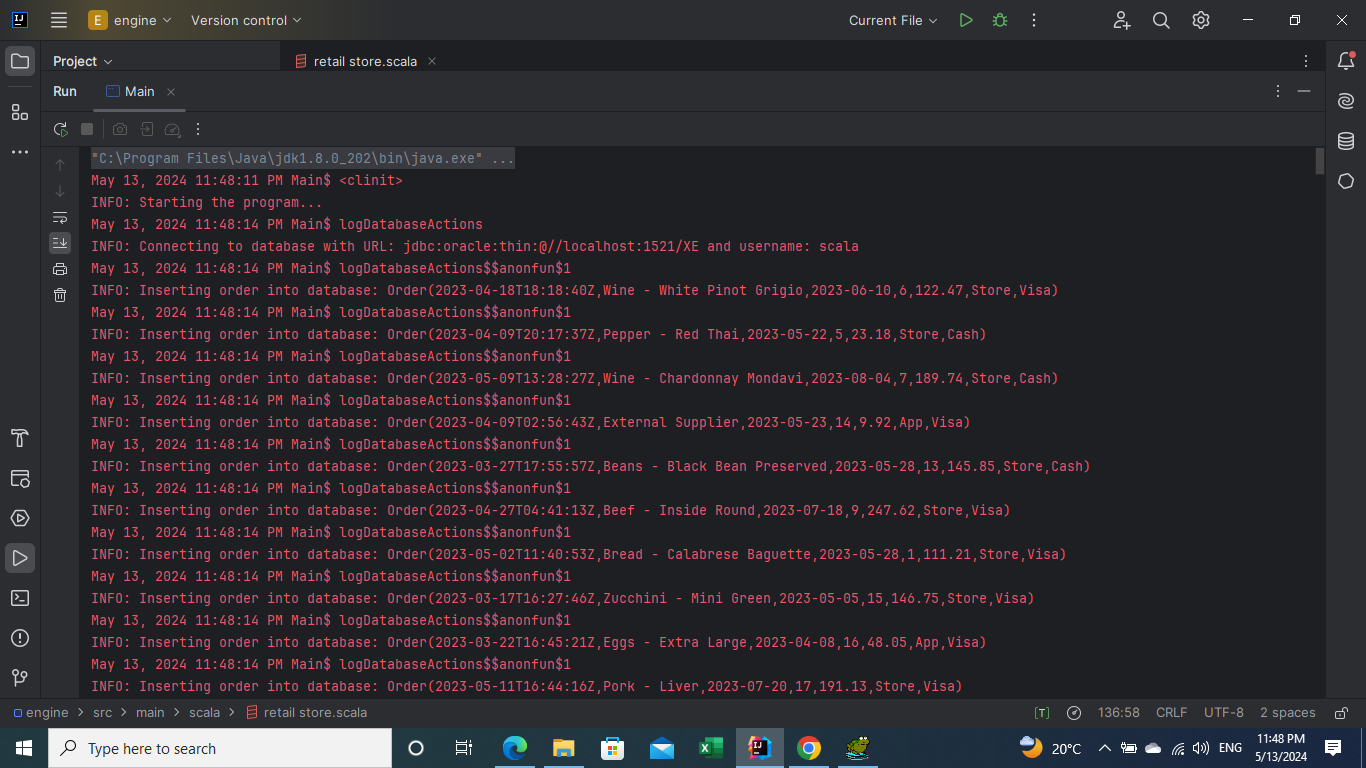
The project implements a rule engine for qualifying orders' transactions to discounts based on a set of rules. The discounts are calculated based on the remaining days for the product to expire, product type (cheese and wine), special dates (23rd of March), quantity purchased, payment method (Visa card), and channel (App). The project ensures that transactions qualifying for multiple discounts receive the top two discounts averaged. The final price is calculated for each order, and the results are logged in a database table.

#### **Core Functions**

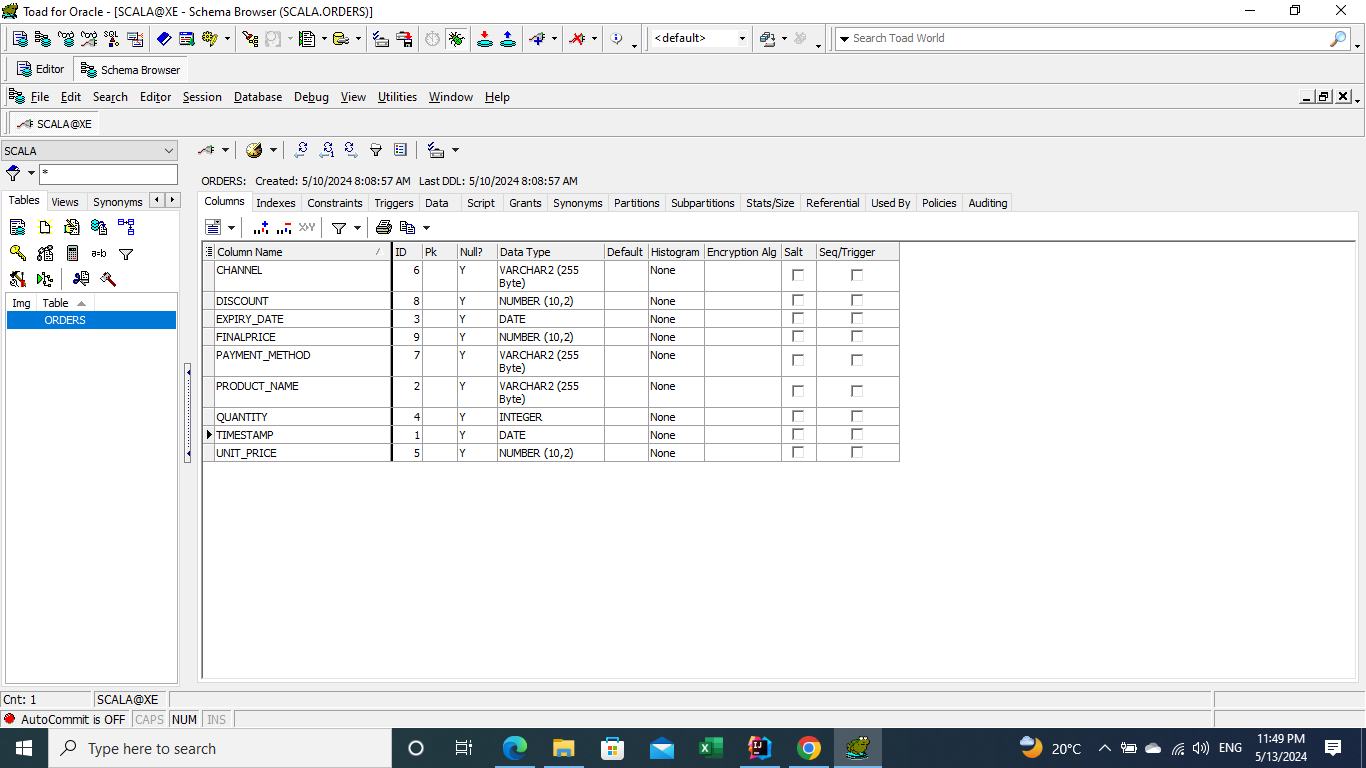
1. calculateExpiryDateDiscount(order: Order): Double
   * Calculates the discount based on the remaining days for the product to expire.
2. cheeseAndWineDiscount(order: Order): Double
   * Calculates the discount for cheese and wine products.
3. specialDate(timestamp: String): Boolean
   * Checks if a given date is a special date (23rd of March).
4. marchSpecialDate(order: Order): Double
   * Calculates the discount for products sold on the special date (23rd of March).
5. boughtMoreQuantity(order: Order): Double
   * Calculates the discount based on the quantity purchased.
6. visadiscount(order: Order): Double
   * Calculates the discount for transactions made using Visa cards.
7. appdiscount(order: Order): Double
   * Calculates the discount for transactions made through the App.
8. applydiscount(order: Order): Double
   * Applies all discount rules to an order and returns the final discount.
9. finaloutput(order: Order): String
   * Formats the order details along with the discount and final price.

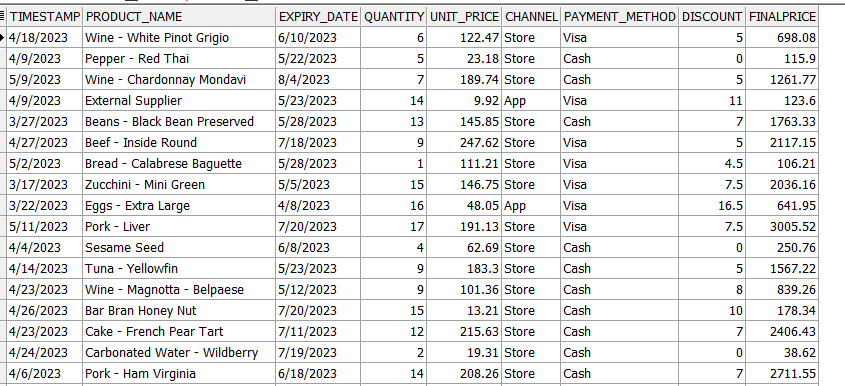
#### **Additional Functions**

* Logging: The project logs events in a log file "rules\_engine.log" using Java's logging framework.



* Database Integration: The project loads the calculated discounts and final prices into a database table using JDBC.





#### **Usage**

* The project reads order data from a CSV file and processes each order to calculate discounts and final prices.
* The calculated results are written to an output CSV file and inserted into a database table.

#### **Conclusion**

This project provides a scalable and maintainable solution for the retail store's discount calculation requirements. It adheres to functional programming principles, ensuring purity, immutability, and predictability in the core logic. The project can be further enhanced with additional discount rules and improved error handling.