Contents

[Overview 1](#_Toc111463723)

[Input Source 1](#_Toc111463724)

[Output 1](#_Toc111463725)

[High Level Design 2](#_Toc111463726)

[Low Level Design 2](#_Toc111463727)

[Microservices and REST Patterns 3](#_Toc111463728)

[Technologies Used 3](#_Toc111463729)

[Taken Care of 3](#_Toc111463730)

**Expense Tracker**

## Overview

Analyze and provide a segregation of the expenses for each customer based on the category of the expense.

### Input Source

Queue and File are the input sources through which we receive the input.

Individual transactions (in JSON format) will be received through Queue. Transaction list (in JSON format) will be received via files (<workspace>\filequeue\input).

### Output

Rest service is provided to retrieve the expense of each customer based on category.

## High Level Design



## Low Level Design

Below shows only for one REST communicate to show on how naming server works.

Note: As of now we are not running multiple instances of any service but we can run it in future



## Microservices and REST Patterns

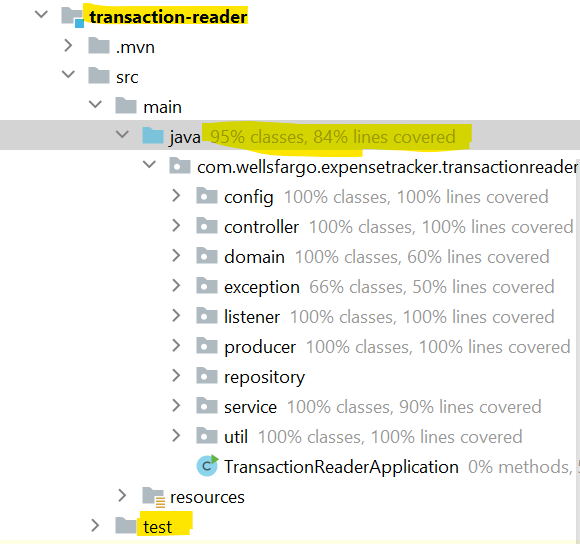
1. API Gateway
2. Naming server
3. Actuator
4. Swagger
5. HATEOS
6. Exception Handling
7. Circuit Breaker etc.

## Technologies Used

Java8, Springboot, Spring cloud, REST, Kafka, Hibernate, H2 DB

## Taken Care of

1. Unit testing for one service– Transaction-Reader



1. Resolved all the relevant static code analysis issues(in all services) which were reported by Sonar Lint

