Валидация, Верификация и Тестови план

Students Log Filter

Ревизионен лист

Релийз No.	Дата	Описание
Rev. 0	06/03/2022	Първоначална имплементация на системата за филтриране и изчислпване на excel таблицата.
Rev. 1	07/03/2022	Добавяне на unit тестове
Rev. 2	13/03/2022	Изготвяне и добавяне на Матрица проследяване на изискванията
Rev. 3	01.04/2022	Финилизиране на проекта

Валидация, Верификация и Тестови план

СЪДЪРЖАНИЕ

Page #

1.0 ОБЩА ИНФОРМАЦИЯ

1.1 Цел

Целта на системата е четене на данни от Excel-ска таблица и тестване на определени фукнционалности:

- *Абсоултна и относителна честота на филтрираните данни*
- *Средната им стойност*
- *Стандартното отклонение*

Целта на валидацията, верификацията и тестването е да осъществи, валидира и тества правилно отношение на системата, общото покритие от тестовете осъществява покритие и правилна работа на всички случаи, особено ‚ръбовете' (Edge cases)

1.2 Обхват

Идеята е функционалността на системата да се покрие от всеки вид тестове (unit, integration) и така осигури правилната работа на изчислените и филтрираните данни.

1.3 Общ преглед на системата

*Системата е написана на Java Spring Boot Framework, за система на изграждане и управление на dependency-тата е използван Gradle. Използвани са Стандартни Spring-ски библиотеки за вдигане на системата в уеб (**Spring Boot starter web**). **Lombok** е използван за минимизиране/премахване на стандартния код. (boilerplate code). За четене на Excel-ската таблица е използвана **'Poiji'** библиотека. За писане на унит тестове използван е **Junit 4.** Системата съдържа унит и интеграционни тестови както и Postman колекция.*

1.5 Съкращения

Не са използвани съкращения.

2.0 ОЦЕНКА НА ТЕСТВАНЕТО

2.1 Матрица за проследяване на изискванията (Requirements Traceability Matrix)

Requirement Traceability Matrix.xlsx се намира в пратения файл.

2.2 Критерии за оценка на тест

Тестовете трябва да покриват всички възможни случаи и всички да минават успешно

Пример на API тестове направени през Postman: При изпратена GET заявка системата връща обратно отговор с определените изчисления.

- *filter/count/absolute – връща абсолютна честота*
- *filter/count/relative – връща относителна честота*
- *filter/count/average– връща средна стойност*
- *filter/count/deviation – върща стандартно отклонение на филтрираните данни*

Полето Test Result трябва да е зелено, т.е всички тестове да са минали.

3.0 ОПИСАНИЕ НА ТЕСТОВИТЕ СЛУЧАИ

3.1 Функционални тестове

3.1.1 [Тест за изчисляване на броя на данните изчислени според филтър: “редактирано (updated) Wiki: (Component -> Wiki)’]

*Описание - тестът сравнява дали метода **getFilteredData()** връща резултат равен на очакавания. Елементи: ResponseEntity (Response със изчисления резултат), value(изчислената данни), метод за сравняване **assertEquals()**.:*

- ***Тестови / функционални връзки***

TC#001 Test Uploads excel file and filteres and shows a data based on given filter

- ***Средства за контрол***

Тестът се изпълнява полу-автоматично.(изисква се самото пускане на теста – не е автоматизиран).

- ***Входни данни***

Файл: Logs_Course A_StudentsActivities.xls

- ***Изходни данни***

Филтрира и изчислява броя на данните филтрирани според зададения филтер (updated) Wiki: (Component -> Wiki).

- ***Процедура за тестване***

*1. Създаване на тест с име **shouldTestCountTotalEntitiesOf302** ().*
*2. Правим **expected** response с резултат който очакваме да покрие реалния подход.*
3. Извикаме правия method и записваме резултата в actual response.
*4. Извикваме метода assertEquals(), в който подаваме като параметри очаквания резултат и резултата, който ще ни върне методът **getFilteredData** () – очакваме 302.*
5. Стартираме теста и резултата трябва да е успешен ако expected и actual са същи.

3.1.2 [Тест за изчисляване на абсолютна честота на броя на “редактирано (updated) Wiki: (Component -> Wiki)’]

*Описание - тестът сравнява дали метода getAbsoluteCount() връща резултат равен на очакавания. Елементи: ResponseEntity (Response със изчисления резултат), value(изчислената абсолютна стойност), метод за сравняване **assertEquals()**.:*

- ***Тестови / функционални връзки***

TC#002 Test filteres the data and calculates the absolute value

- ***Средства за контрол***

Тестът се изпълнява полу-автоматично.(изисква се самото пускане на теста – не е автоматизиран).

- ***Входни данни***

Файл: Logs_Course A_StudentsActivities.xls

- ***Изходни данни***

Абсолютна честота на броя на филтрираните данни според даден филтер (updated) Wiki: (Component -> Wiki).

- ***Процедура за тестване***

*1. Създаване на тест с име **shouldTestAbsoluteMethodResponse** ().*
*2. Правим **expected** response с резултат който очакваме да покрие реалния подход.*
3. Извикаме правия method и записваме резултата в actual response.
*4. Извикваме метода assertEquals(), в който подаваме като параметри очаквания резултат и резултата, който ще ни върне методът **getAbsoluteCount** ().*
5. Стартираме теста и резултата трябва да е успешен ако expected и actual са същи.

3.1.3 [Тест за изчисляване на относителната честота на броя на “редактирано (updated) Wiki: (Component -> Wiki)’]

*Описание - тестът сравнява дали метода **getPercentCount**() връща резултат равен на очакавания. Елементи: ResponseEntity (Response със изчисления резултат), value(изчислената данни), метод за сравняване **assertEquals**().:*

- ***Тестови / функционални връзки***

TC#003 Test filteres the data and calculates the relative value

- ***Средства за контрол***

Тестът се изпълнява полу-автоматично.(изисква се самото пускане на теста – не е автоматизиран).

- ***Входни данни***

Файл: Logs_Course A_StudentsActivities.xls

- ***Изходни данни***

Филтрира и изчислява броя на данните филтрирани според зададения филтер (updated) Wiki: (Component -> Wiki).

- ***Процедура за тестване***

*1. Създаване на тест с име **shouldTestRelativeMethodResponse** ().*
*2. Правим **expected** response с резултат който очакваме да покрие реалния подход.*
3. Извикаме правия method и записваме резултата в actual response.
*4. Извикваме метода assertEquals(), в който подаваме като параметри очаквания резултат и резултата, който ще ни върне методът **getPercentCount** ().*
5. Стартираме теста и резултата трябва да е успешен ако expected и actual са същи.

3.1.4 [Тест за изчисляване на средната стойност на броя на “редактирано (updated) Wiki: (Component -> Wiki)’]

*Описание - тестът сравнява дали метода **getAverageCount()** връща резултат равен на очакавания. Елементи: ResponseEntity (Response със изчисления резултат), value(изчислената данни), метод за сравняване **assertEquals()**.:*

- ***Тестови / функционални връзки***

TC#004 Test filteres the data and calculates the average value

- ***Средства за контрол***

Тестът се изпълнява полу-автоматично.(изисква се самото пускане на теста – не е автоматизиран).

- ***Входни данни***

Файл: Logs_Course A_StudentsActivities.xls

- ***Изходни данни***

Филтрира и изчислява броя на данните филтрирани според зададения филтер (updated) Wiki: (Component -> Wiki).

- ***Процедура за тестване***

*1. Създаване на тест с име **shouldTestAverageCountMethodResponse** ().*
*2. Правим **expected** response с резултат който очакваме да покрие реалния подход.*
3. Извикаме правия method и записваме резултата в actual response.
*4. Извикваме метода assertEquals(), в който подаваме като параметри очаквания резултат и резултата, който ще ни върне методът **getAverageCount** ().*
5. Стартираме теста и резултата трябва да е успешен ако expected и actual са същи.

3.1.5 [Тест за изчисляване на стандартната девиация на броя на “редактирано (updated) Wiki: (Component -> Wiki)’]

*Описание - тестът сравнява дали метода **getDeviationCount()** връща резултат равен на очакавания. Елементи: ResponseEntity (Response със изчисления резултат), value(изчислената данни), метод за сравняване **assertEquals()**.:*

- ***Тестови / функционални връзки***

TC#005 Test filteres the data and calculates the standard deviation value

- ***Средства за контрол***

Тестът се изпълнява полу-автоматично.(изисква се самото пускане на теста – не е автоматизиран).

- ***Входни данни***

Файл: Logs_Course A_StudentsActivities.xls

- ***Изходни данни***

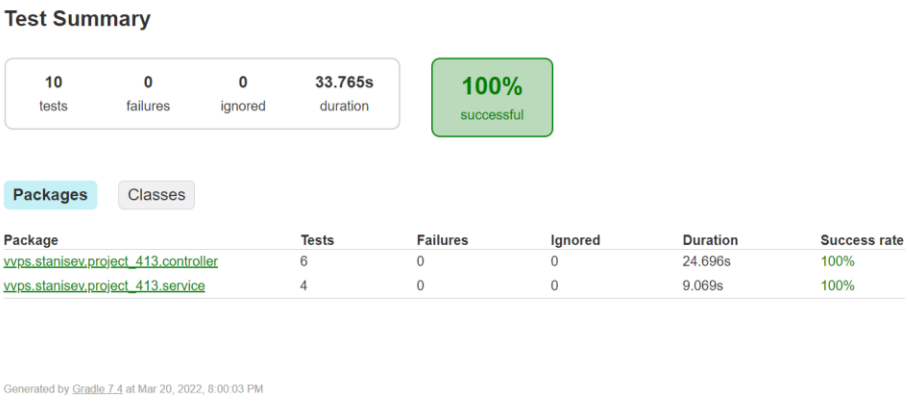
Филтрира и изчислява броя на данните филтрирани според зададения филтер (updated) Wiki: (Component -> Wiki).

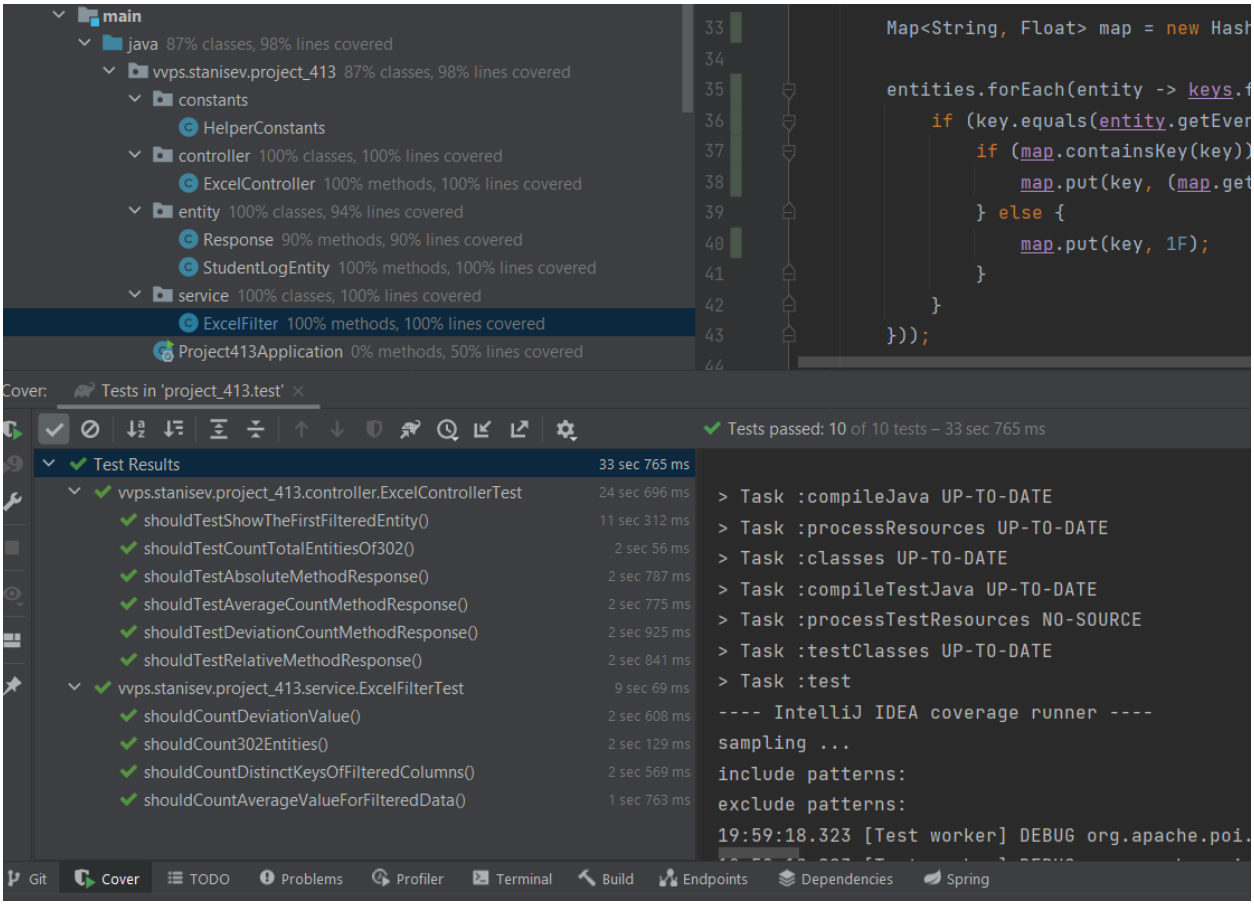
- ***Процедура за тестване***

1. Създаване на тест с име ***shouldTestDeviationCountMethodResponse*** *().*
2. Правим ***expected*** *response с резултат който очакваме да покрие реалния подход.*
3. Извикаме правия method и записваме резултата в actual response.
4. Извикваме метода assertEquals(), в който подаваме като параметри очаквания резултат и резултата, който ще ни върне методът ***getDeviationCount*** *().*
5. Стартираме теста и резултата трябва да е успешен ако expected и actual са същи.

3.2 Модулни (Unit) тестове

Всички описания на моодулните тестове се намират в файлта RTM - Requirement Traceability Matrix




main
java 87% classes, 98% lines covered
vvps.stanisev.project_413 87% classes, 98% lines covered
constants
HelperConstants
controller 100% classes, 100% lines covered
ExcelController 100% methods, 100% lines covered
entity 100% classes, 94% lines covered
Response 90% methods, 90% lines covered
StudentLogEntity 100% methods, 100% lines covered
service 100% classes, 100% lines covered
ExcelFilter 100% methods, 100% lines covered
Project413Application 0% methods, 50% lines covered
Map<String, Float> map = new Hash
entities.forEach(entity -> keys.f
if (key.equals(entity.getEver
if (map.containsKey(key))
map.put(key, (map.get
} else {
map.put(key, 1F);
}
}
}));
Cover:
Tests in 'project_413.test'
Tests passed: 10 of 10 tests – 33 sec 765 ms
Test Results 33 sec 765 ms
vvps.stanisev.project_413.controller.ExcelControllerTest 24 sec 696 ms
shouldTestShowTheFirstFilteredEntity() 11 sec 312 ms
shouldTestCountTotalEntitiesOf302() 2 sec 56 ms
shouldTestAbsoluteMethodResponse() 2 sec 787 ms
shouldTestAverageCountMethodResponse() 2 sec 775 ms
shouldTestDeviationCountMethodResponse() 2 sec 925 ms
shouldTestRelativeMethodResponse() 2 sec 841 ms
vvps.stanisev.project_413.service.ExcelFilterTest 9 sec 69 ms
shouldCountDeviationValue() 2 sec 608 ms
shouldCount302Entities() 2 sec 129 ms
shouldCountDistinctKeysOfFilteredColumns() 2 sec 569 ms
shouldCountAverageValueForFilteredData() 1 sec 763 ms
> Task :compileJava UP-TO-DATE
> Task :processResources UP-TO-DATE
> Task :classes UP-TO-DATE
> Task :compileTestJava UP-TO-DATE
> Task :processTestResources NO-SOURCE
> Task :testClasses UP-TO-DATE
> Task :test
---- IntelliJ IDEA coverage runner ----
sampling ...
include patterns:
exclude patterns:
19:59:18.323 [Test worker] DEBUG org.apache.poi.
Git
Cover
TODO
Problems
Profiler
Terminal
Build
Endpoints
Dependencies
Spring