

Università degli Studi di Modena e Reggio Emilia
Dipartimento di Ingegneria “Enzo Ferrari”

Corso di Laurea Triennale in Ingegneria Informatica

Test di carico e stress su piattaforma clinica

Relatore:
Prof. Sonia Bergamaschi

Laureando:
Naima El Khattabi

Anno Accademico 2020-2021



SOLUZIONI SETTORI STRUMENTI AZIENDA CLIENTI BLOG CONTATTI

PRENOTA UNA DEMO

Aiutiamo le aziende a prendere le decisioni migliori sfruttando al massimo la potenza dei dati

Guarda come le imprese più innovative stanno migliorando i processi aziendali grazie a DataRiver

DECISIONI STRATEGICHE

PROCESSI PRODUTTIVI

PERFORMANCE MAGAZZINO

Ho svolto un periodo di tirocinio presso l'azienda **DataRiver SRL**, che tramite l'analisi dei Big Data permette alle imprese di imparare dall'esperienza e di ottimizzare i processi decisionali e produttivi.

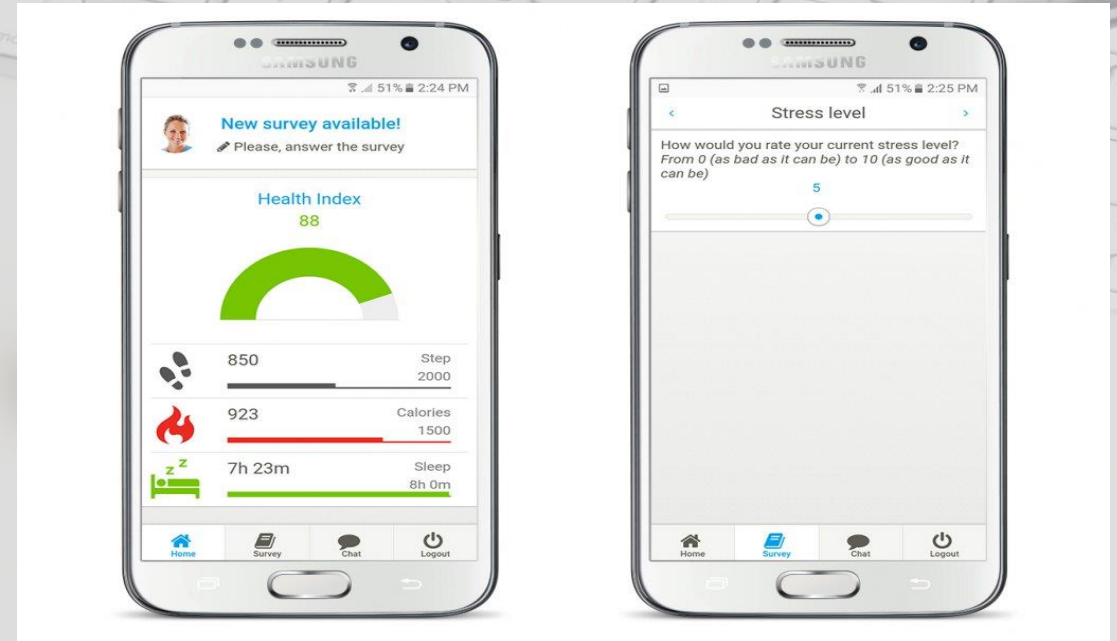
Lo scopo del mio tirocinio è stato quello di individuare una procedura automatizzata per inserire dati in blocco sulla piattaforma **MyHealth** e verificare l'andamento delle prestazioni e il comportamento dell'applicazione a regime.

Piattaforma per il clinical data management MyHealth

MyHealth è una piattaforma Web e Mobile progettata per garantire una interazione efficace tra staff medico e paziente che offre nuove opportunità per il monitoraggio continuo della salute e della qualità di vita dei pazienti.

Consente :

- la raccolta dei dati sull'attività fisica svolta dai pazienti ed il monitoraggio tramite medical and wearable devices dei parametri fisiologici
- la gestione della somministrazione di questionari al paziente
- la gestione di cartelle cliniche e registri clinici e la gestione della terapia del paziente
- schede CRF contenenti dati raccolti direttamente dal paziente tramite la gestione di questionari paziente e la raccolta dati da wearable and medical devices



I Dati raccolti vengono elaborati per effettuare un monitoraggio multidimensionale dei pazienti, supportare una corretta comprensione delle manifestazioni cliniche, pianificare di percorsi di cura personalizzati e programmi di prevenzione.

GOALS



CARICARE DATI DIRETTAMENTE SUL
DATABASE DELLA PIATTAFORMA 'MY
HEALTH' UTILIZZANDO DELLE PROCEDURE
AUTOMATICHE



ANALIZZARE E IMPLEMENTARE UN METODO
PER TESTARE GLI ACCESSI CONCORRENTI
ALLA PIATTAFORMA



TESTARE I TEMPI DI RISPOSTA
DELL'APPLICAZIONE IN CASI DI STRESS
UTILIZZANDO LO STRUMENTO 'VAADIN TEST
BENCH'

The screenshot shows the Eclipse IDE interface. The top part displays the menu bar and toolbar. Below that is the Project Explorer on the left, showing a tree view of the project structure. The main editor window shows the source code for SystemLoadTest.java, with line numbers 280 to 302 visible. The code includes comments and logic for generating a label code. The bottom part of the IDE shows the Console window with the following output:

```
Run application [Maven Build] C:\Program Files\Java\jdk1.8.0_281\bin\javaw.exe (26 mag 2021, 19:11:09)
la lingua e : null
prefisso : MHS-02-
suffisso :
la stringa finale : MHS-02-07
Sto entrando nella funzione utente
Sono dentro
vediamoMHS-02-07
MHS - Modena
it
it 2021-05-26 19:11:05:500 INFO [id=1] it.datariver.module.loadtest.backend.timertask.SystemLoadTest($Action
```

Analisi inserimento dati

Le opzioni erano due :

- Tramite Db
- Implementazione direttamente nella piattaforma per poter sfruttare la sua già esistente connessione al Db e utilizzare dei thread per permettere di implementare più inserimenti concorrenti (uno degli obiettivi)

E' stata scelta la seconda opzione in quanto più efficiente in questo caso specifico.

Data generation

Per la generazione dei dati si è seguito un approccio sequenziale per ogni paziente, in particolare :

1) Creazione paziente

2) Generazione 'phase group' → anagrafica dei gruppi associati alle visite (es. General, Body part treatment, End of study, ecc.)

3) Generazione 'phase' → visite del paziente

4) Generazione 'form schedule' → rappresentazione di tutti i form (questionari) presenti per ogni utente con i relativi valori associati

Per quanto riguarda i valori utilizzati, si è deciso di randomizzarli in base al loro tipo, per simulare il comportamento di un vero paziente.

FIRST STEP: Generation of data with values placed in the code

The screenshot displays an IDE interface with a Package Explorer on the left and a code editor on the right. The Package Explorer shows a project structure with several modules under 'it.datariver'. The code editor shows the following Java code in `*SystemLoadTest.java`:

```
88
89
90 ProjectCenterPhaseValueBean projectCenterPhaseValueBean = new ProjectCenterPhaseValueBean();
91 PhaseBean phaseBean = new PhaseBean();
92 phaseBean.setPhaseId((long) 7); //7
93 PhaseGroupBean phaseGroupBean = new PhaseGroupBean();
94 phaseGroupBean.setPhaseGroupId((long) 1);
95 ProjectCenterPhaseGroupValueBean projectCenterPhaseGroupValueBean = new ProjectCenterPhaseGroupValueBean();
96 projectCenterPhaseGroupValueBean.setPhaseGroupBean(phaseGroupBean);
97 projectCenterPhaseGroupValueBean.setPcpGroupValueId((long)74);
98
99 projectCenterPhaseValueBean.setPhaseBean(phaseBean);
100 projectCenterPhaseValueBean.setProjectCenterPhaseGroupValueBean(projectCenterPhaseGroupValueBean);
101 projectCenterPhaseValueBean.setImageTagName(null);
102
103 projectCenterPhaseValueBean.setProjectCenterId((long) 1);
104 projectCenterPhaseValueBean.setTargetUserAccountId((long) 12); //12
105 projectCenterPhaseValueBean.setDateCreated(now);
106 //projectCenterPhaseValueBean.setDateUpdated(now);
107 projectCenterPhaseValueBean.setUserCreatedId((long) 1);
108 projectCenterPhaseValueBean.setStatusLabel(StatusType.STARTED.getCode());
109 projectCenterPhaseValueBean.setLocked(false);
110 projectCenterPhaseValueBean.setRepetitionIndex(1);
111
```

Below the code editor, the console shows the following output:

```
Run application [Maven Build] C:\Program Files\Java\jdk1.8.0_281\bin\javaw.exe (28 apr 2021, 17:54:25)
iot 2021-04-28 17:59:31,240 INFO i.d.a.a.AccessControllerImpl/updateContent:
    1 is authorized? true
iot 2021-04-28 17:59:31,298 INFO i.d.a.b.d.ProjectCenterPhaseValueDaoImpl/mapSubjectGroupAndPhaseByFilters:
    query: SQLQueryImpl( SELECT uarpc.project_center_id, ua.id_user_account, ua.userr
params
language:it
roleType: subject
```

SECOND STEP: Data generation with random values generated ad hoc

```
LoadTestSer... GenericOQ.java CRF_OPEN.java ElementsApi... ConfirmDialo... QueryExecuto... OQTestVariab... AppService.java TestConfigu...
88
89     String key = formItemBean.getItemLabel();
90     String val = null;
91
92     String nome_label = formItemBean.getFormItemTypeLabel();
93     // System.out.println("nome label : "+nome_label);
94     ArrayList<String> formItemOptions;
95     switch(nome_label)
96     {
97     case "RADIOBUTTON":
98         List<String> list = ModuleFormService.getFormService().getFormItemOptionsByItemId(formItemBean.getFormItemId());
99         formItemOptions = new ArrayList<String>(list); val = String.valueOf(random.nextInt(form
100 // System.out.println("val : "+ val);
101
102 // System.out.println(" key : "+key);
103 // System.out.println(" val : "+val);
104 values.put(key, val);
105 break;
106 case "TIMEFIELD":
107 //formItemOptions = (ArrayList<String>) ModuleFormService.getFormService().getFormItemOptionsByItemId(formItemBean.getFormItemId());
108 val = AppStringUtils.convertTimeToString(now);
109 // System.out.println("val : "+ val);
110
111 // System.out.println(" key : "+key);
112 // System.out.println(" val : "+val);
113 values.put(key, val);
114 break;
115 case "DATEFIELD":
116 //formItemOptions = (ArrayList<String>) ModuleFormService.getFormService().getFormItemOptionsByItemId(formItemBean.getFormItemId());
117 val = generateDateDefaultValue();
118 //val = generateDateRandomValue();
119
120 // System.out.println(" key : "+key);
121 // System.out.println(" val : "+val);
122 values.put(key, val);
123 break;
124 case "CHECKBOX":
125
126 List<String> listCheckbox = ModuleFormService.getFormService().getFormItemOptionsByItemId(formItemBean.getFormItemId());
127 formItemOptions = new ArrayList<String>(listCheckbox);
128
129 val = String.valueOf(random.nextInt(formItemOptions.size()));
130 // System.out.println("val : "+ val);
```

Problems @ Javadoc Declaration Console Call Hierarchy JUnit Coverage

The screenshot shows the Eclipse IDE interface. The top menu bar includes 'Object', 'Tools', and 'Help'. Below it is a toolbar with various icons. The main window displays a SQL query in the 'Query Editor' for a PostgreSQL database. The query is:

```
1 SELECT * FROM public.test_configuration
2 ORDER BY test_configuration_id ASC
```

The 'Data Output' window shows the results of the query as a table:

test_configuration_id	test_key	test_value
1	dateMax	2021-12-31
2	dateMin	2001-01-01

A green notification box at the bottom right of the IDE indicates: 'Successfully run. Total query runtime...'. The status bar at the bottom shows 'Writable', 'Smart Insert', and '371 : 41 : 18544'.

Dopo aver finito la generazione dati random e l'acquisizione dei dati dal Database per creare nuovi pazienti (con form, ecc), è stato necessario passare allo step successivo.

Come menzionato precedentemente, dal momento che tutti i dati vengono inseriti da codice, l'unica cosa che ci serve al momento è il range di date minime e massime.

Per convenienza è stata semplicemente creata una tabella di tipo "chiave-valore" che conterrà il range di date che si vuole utilizzare.

Questa soluzione è eccellente perché è configurabile; in poche parole, quando si vorranno eseguire altri test e sarà necessario recuperare dei dati dal Database, sarà semplicemente necessario aggiungere una colonna in questa tabella e modificare i file java di conseguenza.

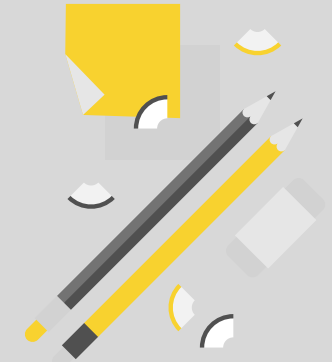
Validation → garantire il funzionamento

Validazione con lo strumento **Vaadin Test Bench** che simula il comportamento di un'utente dell'applicazione ed esegue le attività specificate nel codice Java nelle OQ.

Una OQ (Operational Qualification) è una verifica documentata che il sistema funziona come previsto, operando all'interno dei range stabiliti

Nel mio caso ho dovuto sviluppare un unico TestCase, dove le operazioni da seguire sono state:

- login
- click on CRF button
- choose patient
- choose phaseGroup
- choose form.



HOME CRFs

[Add new subject](#)
[Block subject](#)
[Remove subject](#)

[Add Treatment](#)
[Add visit](#)
[Add CRF](#)
[Change status](#)
[Lock visit](#)

[Open blank CRF](#)
[Open casebook](#)
[Download Audit Trail PDF](#)
[Study Design](#)

Filter on subject

- ▼ MHS-02-01
- ▼ General
 - V4 Confirmation**
 - End Of Study
 - Concomitant
- ▶ MHS-02-02
- ▶ MHS-02-03
- ▶ MHS-02-04

CRFs for event: V4 Confirmation

Visit Date	Completed	Edit	Print	Download
Concomitant Medications 2	Not started	Edit	Print	Download
Adverse Events 2	Not started	Edit	Print	Download
Physical Examination	Not started	Edit	Print	Download
Vital Signs	Not started	Edit	Print	Download
ECOG Status	Not started	Edit	Print	Download
EBDASI	Not started	Edit	Print	Download
Inclusion/Exclusion Criteria	Not started	Edit	Print	Download
Informed Consent	Not started	Edit	Print	Download
Haematology	Not started	Edit	Print	Download
Blood Chemistry	Not started	Edit	Print	Download
Electrocardiogram	Not started	Edit	Print	Download
B Cells Immunological Tests - Biopsy	Not started	Edit	Print	Download
B Cells Immunological Tests - Blood	Not started	Edit	Print	Download

Validation



Tutto questo è servito per testare il tempo impiegato per l'esecuzione del TestCase e verrà riportato come output nel report PDF generato da Vaadin.

- Per aggiungere il tempo di esecuzione è stato necessario andare ad agire sul file `ConfirmDialogReport.java`
- Purtroppo in questo modo si riusciva a vedere solo il tempo totale e non il tempo per ogni test case; quest'ultimo è utile per capire qual è la fase che dà maggiori problemi nei 2 casi.
- Proprio per questo motivo, ho cambiato agendo sul file `GenericOQ` dove si trovano tutte le funzionalità che richiamo nel mio `CRF_OPEN`.
- In questo modo per ogni step avrò il tempo impiegato per ogni step e potrò avere un effettivo confronto tra più casi ed eventualmente individuare un possibile problema

Test durations a confronto

Selezionando il **paziente 1**, i tempi sono stati :

- 14667 secondi (PRIMO STEP)
- 14720 secondi (SECONDO STEP)
- 139886 secondi (TERZO STEP)

Per un totale di : 169,273 secondi, cioè **2,821216667** minuti.

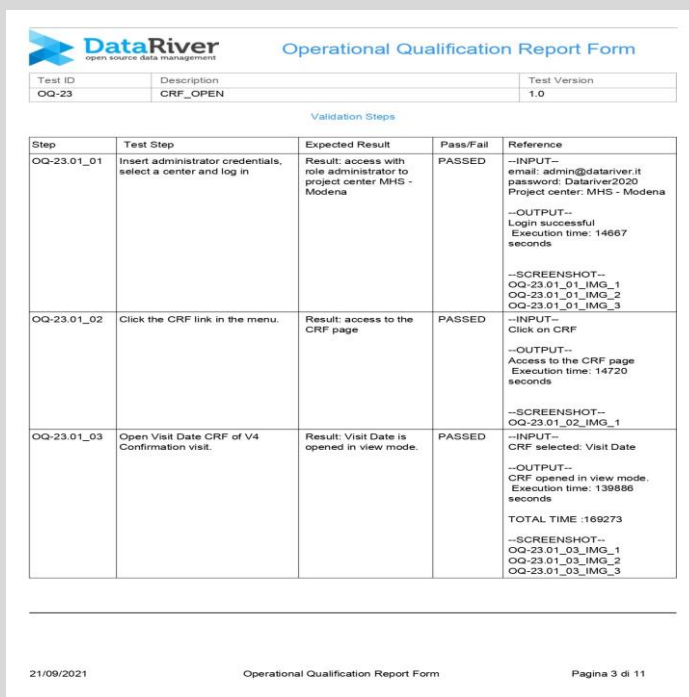
Selezionando il **paziente 305**, invece, i tempi sono stati :

- 13206 secondi (PRIMO STEP)
- 16192 secondi (SECONDO STEP)
- 262565 secondi (TERZO STEP)

Per un totale di 291,963 secondi, cioè **4,86605** minuti.

La cosa che salta più all'occhio è quella che accade nel **TERZO STEP**; come si legge dalla tabella, abbiamo una differenza di circa 2 minuti in più rispetto al primo test con il paziente 1.

Questo ci mostra che le operazioni svolte nel terzo step non sono performanti in presenza di più soggetti arruolati e quindi indica uno dei potenziali problemi su cui concentrarsi in futuro per poter migliorare le prestazioni e la fluidità della piattaforma.



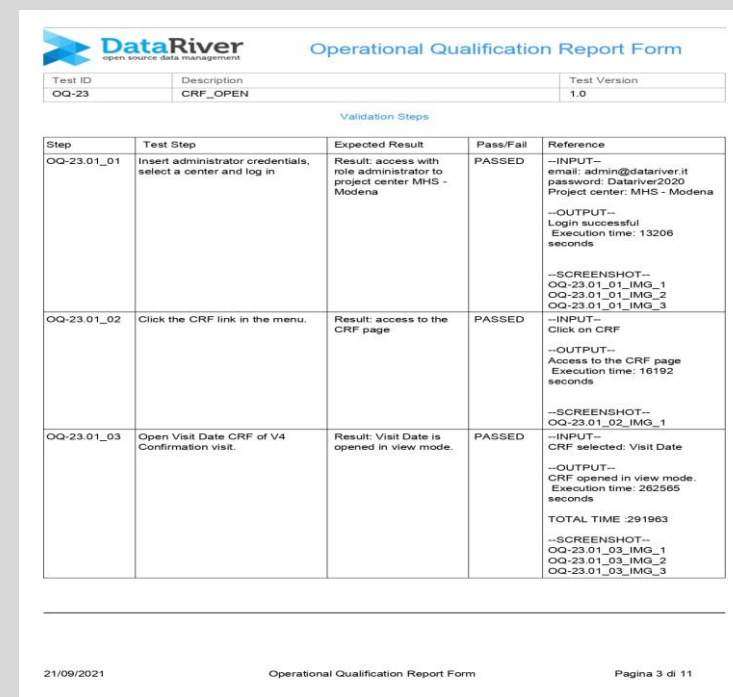
DataRiver open source data management
Operational Qualification Report Form

Test ID	Description	Test Version
OO-23	CRF_OPEN	1.0

Validation Steps

Step	Test Step	Expected Result	Pass/Fail	Reference
OO-23.01_01	Insert administrator credentials, select a center and log in	Result: access with role administrator to project center MHS - Modena	PASSED	--INPUT-- email: admin@datariver.it password: Datariver2020 Project center: MHS - Modena --OUTPUT-- Login successful Execution time: 14667 seconds --SCREENSHOT-- OO-23.01_01_IMG_1 OO-23.01_01_IMG_2 OO-23.01_01_IMG_3
OO-23.01_02	Click the CRF link in the menu.	Result: access to the CRF page	PASSED	--INPUT-- Click on CRF --OUTPUT-- Access to the CRF page Execution time: 14720 seconds --SCREENSHOT-- OO-23.01_02_IMG_1
OO-23.01_03	Open Visit Date CRF of V4 Confirmation visit.	Result: Visit Date is opened in view mode.	PASSED	--INPUT-- CRF selected: Visit Date --OUTPUT-- CRF opened in view mode. Execution time: 139886 seconds TOTAL TIME :169273 --SCREENSHOT-- OO-23.01_03_IMG_1 OO-23.01_03_IMG_2 OO-23.01_03_IMG_3

21/09/2021 Operational Qualification Report Form Pagina 3 di 11



DataRiver open source data management
Operational Qualification Report Form

Test ID	Description	Test Version
OO-23	CRF_OPEN	1.0

Validation Steps

Step	Test Step	Expected Result	Pass/Fail	Reference
OO-23.01_01	Insert administrator credentials, select a center and log in	Result: access with role administrator to project center MHS - Modena	PASSED	--INPUT-- email: admin@datariver.it password: Datariver2020 Project center: MHS - Modena --OUTPUT-- Login successful Execution time: 13206 seconds --SCREENSHOT-- OO-23.01_01_IMG_1 OO-23.01_01_IMG_2 OO-23.01_01_IMG_3
OO-23.01_02	Click the CRF link in the menu.	Result: access to the CRF page	PASSED	--INPUT-- Click on CRF --OUTPUT-- Access to the CRF page Execution time: 16192 seconds --SCREENSHOT-- OO-23.01_02_IMG_1
OO-23.01_03	Open Visit Date CRF of V4 Confirmation visit.	Result: Visit Date is opened in view mode.	PASSED	--INPUT-- CRF selected: Visit Date --OUTPUT-- CRF opened in view mode. Execution time: 262565 seconds TOTAL TIME :291963 --SCREENSHOT-- OO-23.01_03_IMG_1 OO-23.01_03_IMG_2 OO-23.01_03_IMG_3

21/09/2021 Operational Qualification Report Form Pagina 3 di 11

POSSIBLE FUTURE EXTENSIONS:



La soluzione è stata implementata in modo flessibile per permettere future estensioni per migliorare la performance, il tempo e le informazioni della piattaforma.

Esempi estensioni future:

- Gestione multi-centro
- Test su alcuni tipi di visite



THANK YOU FOR
YOUR ATTENTION

-Naima El Khattabi