****

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **T**est **P**lan  **D**ocument  Easy Pass   |  |  | | --- | --- | | Riferimento |  | | Versione | 1.1 | | Data | 14/1/2022 | | Destinatario | Prof. C. Gravino | | Presentato da | Montefusco Alberto  Mulino Martina  Rinaldi Viviana  Spina Gennaro | | Approvato da |  | |

Sommario

[Sommario 2](#_Toc530825397)

[RevisionHistory 3](#_Toc530825396)

1. [Introduzione 4](#_Toc530825398)

1.1 Descrizione del Documento 4

1. Relazioni con altri Documenti 4
2. Panoramica del Sistema 4
3. Funzionalità da testare e no 4

4.1 Matrice di tracciabilità del Testing 5

1. Criteri di successo/fallimento 6
2. Approccio 6

6.1 Testing di Unità 6

6.2 Testing di Integrazione 7

6.3 Testing di Sistema 7

1. [Strumenti per il testing (Hardware/Software) 7](#_Toc530825398)
2. [Test Cases 8](#_Toc530825398)
3. Specifica dei Test Cases 9

9.1 Registrazione di un Docente 9

9.2 Selezione di un numero di studenti da validare 13

9.3 Invio di un Green Pass 14

9.4 Ricerca dei report 15

9.5 Selezione del formato dei report 19

1. Pianificazione del testing e Assegnazione dei ruoli 21

RevisionHistory

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versione** | **Descrizione** | **Autori** |
| 5/12/2021 | 0.1 | Prima stesura: aggiunta capitoli 1 – 2 – 3 – 4 | Alberto Montefusco |
| 6/12/2021 | 0.2 | Aggiunta capitoli da 5 – 8 | Alberto Montefusco  Viviana Rinaldi |
| 10/12/2021 | 0.3 | Aggiunta capitoli 6 – 7 – 10 | Martina Mulino |
| 11/12/2021 | 0.4 | Aggiunta Specifica dei Test Cases – capitolo 9 | Alberto Montefusco  Viviana Rinaldi  Martina Mulino  Gennaro Spina |
| 13/12/2021 | 1.0 | Revisione finale | Alberto Montefusco  Viviana Rinaldi  Martina Mulino  Gennaro Spina |
| 14/1/2022 | 1.1 | Correzioni | Alberto Montefusco  Viviana Rinaldi  Martina Mulino  Gennaro Spina |

1. Introduzione
   1. Descrizione del Documento

Il Test Plan del progetto Easy Pass presenta la pianificazione e la specifica di quelli che sono i Test Case, ovvero l’insieme di input e di risultati attesi che servono a testare una componente del Sistema. Nel caso in cui delle attività di testing evidenziassero degli errori che possano causare comportamenti diversi da quelli attesi e che possano compromettere il buon utilizzo del Sistema da parte degli utenti, quest’ultimo può essere sottoposto ad un processo di correzione degli errori individuati, per garantire agli utenti finali un prodotto software che rispecchi tutte le specifiche finora stabilite nelle precedenti fasi di sviluppo.

1. Relazioni con altri Documenti

Il Test Plan presenta diversi punti di correlazione con i documenti stilati durante le fasi precedenti dello sviluppo di Easy Pass. In particolare, il presente Documento fa riferimento ai casi d’uso descritti nel Requirement Analysis Document (RAD).

1. Panoramica del Sistema

Il Sistema Easy Pass fornisce tutte le sue funzionalità attraverso una Web Application. Per assicurarsi che ciascuna si comporti come previsto, bisogna quindi assicurarsi che vengano testate tutte le funzionalità offerte dal Sistema. Nel caso in cui queste funzionalità si comportino nella maniera prevista, si può ritenere che il Sistema Easy Pass soddisfi gli obiettivi che si erano prefissati.

1. Funzionalità da testare e no

Le attività di testing previste per il Sistema Easy Pass prevedono di verificare il corretto funzionamento della maggior parte delle funzionalità del Sistema. Quelle che saranno oggetto dell’attività di testing sono:

1. Registrazione di un Docente;
2. Selezione di un numero di studenti da validare;
3. Ricerca dei report tramite filtri;
4. Selezione di un formato dei report;
5. Invio di un Green Pass.
   1. Matrice di tracciabilità del Testing

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Requisiti** | **Casi di Test** | | | | |
| **TC\_1** | **TC\_2** | **TC\_3** | **TC\_4** | **TC\_5** |
| **RF[1]** | **🗸** |  |  |  |  |
| **RF[2]** |  |  |  |  |  |
| **RF[3]** |  | **🗸** |  |  |  |
| **RF[4]** |  |  | **🗸** |  |  |
| **RF[5]** |  |  |  |  |  |
| **RF[6]** |  |  |  |  |  |
| **RF[7]** |  |  |  |  |  |
| **RF[8]** |  |  |  |  |  |
| **RF[9]** |  |  |  |  |  |
| **RF[10]** |  |  |  |  |  |
| **RF[11]** |  |  |  |  |  |
| **RF[12]** |  |  |  | **🗸** |  |
| **RF[13]** |  |  |  |  | **🗸** |

1. Criteri di successo/fallimento

Un caso di test ha esito positivo se l’output osservato è differente dal risultato previsto dall’oracolo; al contrario, un caso di test ha esito negativo se l’output osservato coincide con il risultato previsto dall’oracolo. Pertanto, le attività di test hanno successo nei casi in cui riescono ad individuare dei comportamenti anomali nell’esecuzione delle funzionalità del Sistema. Nel caso in cui uno o più casi di testi riscuotono successo, è possibile attuare un’opportuna procedura di correzione del difetto riscontrato e, successivamente, ricorrere ad un test di regressione per testare nuovamente la funzionalità modificata ed accertarsi che il problema sia stato risolto.

1. Approccio

L’attività di Testing del Sistema è stata organizzata in tre categorie principali:

* Testing di Unità, che si occupa di testare le singole componenti del Sistema;
* Testing di Integrazione, che si occupa di integrare le unità testate in modo isolato, così da controllare che il sistema funzioni correttamente quando più unità collaborano;
* Testing di Sistema, che include l’attività di Testing Funzionale, la quale si occupa di testare le funzionalità del Sistema definite dai requisiti funzionali. Il Testing di Sistema include anche il Testing di Performance e il Testing Pilota, che però non verranno effettuate poiché il budget a disposizione per il progetto non copre l’ammontare di risorse necessarie.
  1. Testing di Unità

L’attività di Testing di Unità consiste nell’isolare un’unità del Sistema e provare che funzioni correttamente. La strategia prevista per affrontare tale attività consiste nel testare i metodi delle classi del sistema. In particolare, vengono esclusi dal testing le seguenti cose: …… , mentre sono oggetto di test le seguenti cose… . I casi di test saranno definiti attraverso un approccio black-box attraverso l’uso del framework JUnit. Inoltre, per isolare le componenti testate è stato utilizzato Mockito, mentre per il calcolo della branch coverage del testing è stato sfruttato JaCoCo.

* 1. Testing di Integrazione

L’attività di Testing di Integrazione consiste nell’integrare le componenti testate singolarmente in sottosistemi più grandi, così da rilevare i bug che non sono stati evidenziati con il Testing di Unità. Per questa attività è stata scelta la strategia di testing Bottom-Up, in quanto è ritenuta migliore per i sistemi software Object-Oriented, paradigma utilizzato per Easy Pass. In particolare, sono stati testati tutti i metodi delle classi Bean che richiamano i metodi delle classi DAO. Escludendo qualcosa? Le tecnologie utilizzate sono le stesse di quelle del Testing di Unità, dunque JUnit, JaCoCo e Mockito.

* 1. Testing di Sistema

L’attività di Testing Funzionale, inclusa in quella di Testing di Sistema, consiste nell’individuare i possibili faults generati dagli input degli utenti. L’approccio utilizzato per testare le funzionalità del Sistema, definite a partire dai requisiti funzionali e dai casi d’uso esposti nel documento “RAD”, è quello del “Black-Box Testing”. La tecnica utilizzata per individuare i casi di test e per descriverli è la “Category Partition”, in cui sfruttiamo le classi di equivalenza per partizionare l’insieme dei possibili input dell’utente.

1. Strumenti per il testing (Hardware/Software)

Lo strumento hardware utilizzato per l’attività di testing è il computer e, poiché il Sistema non è stato ancora rilasciato, non necessariamente deve essere connesso ad Internet.

Come anticipato nei precedenti paragrafi, gli strumenti software utilizzati per le attività di testing sono:

* JUnit: framework open source usato per scrivere ed eseguire Testing di Unità per Java.
* JaCoCo: libreria per il Code Coverage in ambito Java.
* Mockito: framework utilizzato per la realizzazione di oggetti mock durante il Testing di Unità;
* Selenium: tool open source utilizzato per la gestione automatizzata dei browser, utilizzato come framework di testing. Esso permette di registrare le azioni che un utente può effettuare nell’interazione con il Sistema, così da poter eseguire i casi di test.

1. Test Cases

Per sviluppare i Test Cases sarà utilizzato il metodo del Category Partition. Questo metodo consiste nell’identificare per ogni funzionalità da testare dei parametri; per ogni parametro verranno individuate delle categorie, le quali poi saranno suddivise in scelte. Alle scelte verrà assegnato un valore. I Test Cases verranno definiti nel documento di Test Cases Specification (TCS).

1. Specifica dei Test Cases
   1. Registrazione di un Docente

**TC\_1 Registrazione Docente**

|  |  |
| --- | --- |
| **Parametro**: Nome  **Formato**: ^[a-zA-Z .']+$ | |
| **Categorie** | **Possibili Scenari** |
| **Campo vuoto cvn** | 1. Il campo non è stato compilato [**errore**] 2. Il campo è stato compilato   [property **campoVuotoCVNok**] |
| **Lunghezza ln** | 1. Lunghezza <= 0 [if **campoVuotoCVNok**] [**errore**] 2. Lunghezza > 30 [if **campoVuotoCVNok**] [**errore**] 3. 0 < Lunghezza <= 30 [if **campoVuotoCVNok**]   [property **lunghezzaLNok**] |
| **Formato fn** | 1. Non rispetta il formato   [if **campoVuotoCVNok** and l**unghezzaLNok**] [**errore**]   1. Rispetta il formato [if **campoVuotoCVNok** and **lunghezzaLNok**]   [property **formatoFNok**] |

|  |  |
| --- | --- |
| **Parametro:** Cognome  **Formato:** ^[a-zA-Z .']+$ | |
| **Categorie** | **Scelte** |
| **Campo vuoto cvc** | 1. Il campo non è stato compilato [**errore**] 2. Il campo è stato compilato [property **campoVuotoCVCok**] |
| **Lunghezza lc** | 1. Lunghezza <= 0 [if **campoVuotoCVCok**] [**errore**] 2. Lunghezza > 30 [if **campoVuotoCVCok**] [**errore**] 3. 0 < Lunghezza <= 30 [if **campoVuotoCVCok**]   [property **lunghezzaLCok**] |
| **Formato fc** | 1. Non rispetta il formato   [if **campoVuotoCVCok** and **lunghezzaLCok**] [**errore**]   1. Rispetta il formato [if **campoVuotoCVCok** and **lunghezzaLCok**] [property **formatoFCok**] |

|  |  |
| --- | --- |
| **Parametro:** Dipartimento (Dropdown menù)  **Formato:** - | |
| **Categorie** | **Scelte** |
| **Scelta sd** | 1. Scelta non effettuata [**errore**] 2. Scelta effettuata [property **sceltaSDok**] |

|  |  |
| --- | --- |
| **Parametro:** E-mail  **Formato:** ^[a-zA-Z0-9\_.]{3,}@(?:(?:[a-zA-Z0-9-]+\.)?[a-zA-Z]+\.)?(unisa)\.it$ | |
| **Categorie** | **Scelte** |
| **Campo vuoto cve** | 1. Il campo non è stato compilato [**errore**] 2. Il campo è stato compilato [property **campoVuotoCVEok**] |
| **Formato fe** | 1. Non rispetta il formato [if **campoVuotoCVEok**] [**errore**] 2. Rispetta il formato [if **campoVuotoCVEok**] [property **formatoFEok**] |
| **Esiste ee** | 1. Esiste nel database [if **campoVuotoCVEok** and **formatoFEok**] [**errore**] 2. Non esiste nel database [if **campoVuotoCVEok** and **formatoFEok**] [property **esisteEEok**] |

|  |  |
| --- | --- |
| **Parametro:** Password  **Formato:** ^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*\d)(?=.\*[=^ ì{}+çò°àù§èé#@$!%€\*?&:,;'.\_<>|-])[A-Za-z\d=^ ì{}+çò°àù§èé#@$!%€\*?&:,;'.\_<>|-]{8,}$ | |
| **Categorie** | **Scelte** |
| **Campo vuoto cvp** | 1. Il campo non è stato compilato [**errore**] 2. Il campo è stato compilato [property **campoVuotoCVPok**] |
| **Lunghezza lp** | 1. Lunghezza < 8 [if **campoVuotoCVPok**] [**errore**] 2. Lunghezza >= 8 [if **campoVuotoCVPok**]   [property  **lunghezzaLPok**] |
| **Formato fp** | 1. Non rispetta il formato [if **campoVuotoCVPok** and **lunghezzaLPok**] [**errore**] 2. Rispetta il formato [if **campoVuotoCVPok** and **lunghezzaLPok**] [property **formatoFPok**] |

|  |  |  |
| --- | --- | --- |
| **Codice** | **Combinazione** | **Esito** |
| TC\_1\_01 | cvn1 | FAIL |
| TC\_1\_02 | cvn2.ln1 | FAIL |
| TC\_1\_03 | cvn2.ln2 | FAIL |
| TC\_1\_04 | cvn2.ln3.fn1 | FAIL |
| TC\_1\_05 | cvn2.ln3.fn2.cvc1 | FAIL |
| TC\_1\_06 | cvn2.ln3.fn2.cvc2.lc1 | FAIL |
| TC\_1\_07 | cvn2.ln3.fn2.cvc2.lc2 | FAIL |
| TC\_1\_08 | cvn2.ln3.fn2.cvc2.lc3.fc1 | FAIL |
| TC\_1\_09 | cvn2.ln3.fn2.cvc2.lc3.fc2.sd1 | FAIL |
| TC\_1\_10 | cvn2.ln3.fn2.cvc2.lc3.fc2.sd2.cve1 | FAIL |
| TC\_1\_11 | cvn2.ln3.fn2.cvc2.lc3.fc2.sd2.cve2.fe1 | FAIL |
| TC\_1\_12 | cvn2.ln3.fn2.cvc2.lc3.fc2.sd2.cve2.fe2.ee1 | FAIL |
| TC\_1\_13 | cvn2.ln3.fn2.cvc2.lc3.fc2.sd2.cve2.fe2.ee2.cvp1 | FAIL |
| TC\_1\_14 | cvn2.ln3.fn2.cvc2.lc3.fc2.sd2.cve2.fe2.ee2.cvp2.lp1 | FAIL |
| TC\_1\_15 | cvn2.ln3.fn2.cvc2.lc3.fc2.sd2.cve2.fe2.ee2.cvp2.lp2.fp1 | FAIL |
| TC\_1\_16 | cvn2.ln3.fn2.cvc2.lc3.fc2.sd2.cve2.fe2.ee2.cvp2.lp2.fp2 | PASS |

* 1. Selezione di un numero di studenti da validare

**TC\_2** **Inserimento numero studenti**

|  |  |
| --- | --- |
| **Parametro:** Numero Studenti  **Formato:** ^[1-9]+[0-9]\*$ | |
| **Categorie** | **Scelte** |
| **Campo vuoto cvns** | 1. Il campo non è stato compilato [**errore**] 2. Il campo è stato [property **campoVuotoCVNSok**] |
| **Formato fns** | 1. Non rispetta il formato [if **campoVuotoCVNSok**] [**errore**] 2. Rispetta il formato [if **campoVuotoCVNSok**] [property **formatoFNSok**] |
| **Valore vns** | 1. Valore <= 0 [if **campoVuotoCVNSok** and **formatoFNSok**] [**errore**] 2. Valore > 0 [if **campoVuotoCVNSok** and **formatoFNSok**] [property **valoreVNSok**] |

|  |  |  |
| --- | --- | --- |
| **Codice** | **Combinazione** | **Esito** |
| TC\_2\_01 | cvns1 | FAIL |
| TC\_2\_02 | cvns2.fns1 | FAIL |
| TC\_2\_03 | cvns2.fns2.vns1 | FAIL |
| TC\_2\_04 | cvns2.fns2.vns2 | PASS |

* 1. Invio di un Green Pass

**TC\_3 Invio del Green Pass**

Questo test case verifica se la stringa decodificata dal QR Code dell’immagine inserita dallo Studente rappresenti un Green Pass (un Green Pass è definito tale se la stringa che lo rappresenta è in base 45).

|  |  |
| --- | --- |
| **Parametro:** Green Pass  **Formato:** REGEX | |
| **Categorie** | **Scelte** |
| **Campo vuoto cvgp** | 1. campo vuoto [**errore**] 2. campo non vuoto [property **campoVuotoCVGPok**] |
| **Formato fgp** | 1. input non corrispondente al formato [if **campoVuotoCVGPok**] [**errore**] 2. input corrispondente al formato [if **campoVuotoCVGPok**] [property **formatoFGPok**] |

|  |  |  |
| --- | --- | --- |
| **Codice** | **combinazione** | **Esito** |
| TC\_3\_01 | cvgp1 | FAIL |
| TC\_3\_02 | cvp2.fgp1 | FAIL |
| TC\_3\_03 | cvp2.fgp2 | PASS |

* 1. Ricerca dei report

**TC\_4 Ricerca dei Report**

Poiché il caso d’uso UC\_RicercaReport (vedi documento RAD) indica che la ricerca dei report può essere effettuata specificando o solo il Docente che ha generato i report, o il range di date entro cui vengono generati i report, o entrambi, il test case TC\_4 Ricerca dei Report è l’aggregazione di tre test case componenti:

1. TC\_4.1 RicercaSoloDocente;
2. TC\_4.2 RicercaSoloDate;
3. TC\_4.3 RicercaCompleta.

Il motivo di tale suddivisione dei test case è che il caso in cui venga specificato il Docente ma non le date non è un errore, in quanto è una funzionalità prevista dallo use case UC\_RicercaReport. Stessa cosa nel caso in cui vengano specificate le date ma non il Docente.

**TC\_4.1 RicercaSoloDocente**

|  |  |
| --- | --- |
| **Parametro:** Nome completo del Docente  **Formato:** ^[a-zA-Z .']+$ | |
| **Categorie** | **Scelte** |
| **Campo vuoto cvndoc** | 1. Il campo è vuoto [**errore**] 2. Il campo non è vuoto [property **campoVuotoCVNDOCok**] |
| **Esiste endoc** | 1. non esiste nel DB [if **campoVuotoCVNDOCok**] [**errore**] 2. esiste nel DB [if **campoVuotoCVNDOCok**] [property **esisteENDOCok**] |
| **Formato fndoc** | 1. input non corrispondente al formato [if **esisteENDOCok**] [**errore**] 2. input corrispondente al formato [if **esisteENDOCok**]   [property **formatoFNDOCok**] |

|  |  |  |
| --- | --- | --- |
| **Codice** | **Combinazione** | **Esito** |
| TC\_4.1\_01 | cvndoc1 | FAIL |
| TC\_4.1\_02 | cvndoc2.endoc1 | FAIL |
| TC\_4.1\_03 | cvndoc2.endoc2.fndoc1 | FAIL |
| TC\_4.1\_04 | cvndoc2.endoc2.fndoc2 | PASS |

**TC\_4.2 RicercaSoloDate**

|  |  |
| --- | --- |
| **Parametro:** PrimaData, SecondaData  **Formato: ^**\d{4}\-(0?[1-9]|1[012])\-(0?[1-9]|[12][0-9]|3[01])$ | |
| **Categorie** | **Scelte** |
| **Campo vuoto cvfdat** | 1. campo prima data vuoto [**errore**] 2. campo prima data non vuoto [property **campoVuotoCVFDATok**] |
| **Formato ffdat** | 1. input non corrispondente al formato [if **campoVuotoCVFDATok**] [**errore**] 2. input corrispondente al formato [property **formatoFFDATok**] |
| **Campo vuoto cvsdat** | 1. campo vuoto [if **formatoFFDATok**] [**errore**] 2. campo non vuoto [if not **campoVuotoCVFDATok**] [**errore**] 3. campo non vuoto [if **formatoFFDATok**]   [property **campoVuotoCVSDATok**] |
| **Formato fsdat** | 1. input non corrispondente al formato [if **campoVuotoCVSDATok**] [**errore**] 2. input corrispondente al formato [if **campoVuotoCVSDATok**] [property **formatoFSDATok**] |
| **Confronto cdat** | 1. ffdat > fsdat [**formatoFFDATok** and **formatoFSDATok**] [**errore**] 2. ffdat <= fsdat [if **formatoFFDATok** and **formatoFSDATok** and **campoVuotoCVSDATok**] [property **confrontoCVDATok**] |

|  |  |  |
| --- | --- | --- |
| **Codice** | **Combinazione** | **Esito** |
| TC\_4.2\_01 | cvfdat1 | FAIL |
| TC\_4.2\_02 | cvfdat2.ffdat1 | FAIL |
| TC\_4.2\_03 | cvfdat2.ffdat2.cvsdat1 | FAIL |
| TC\_4.2\_04 | cvfdat2.ffdat2.cvsdat2 | FAIL |
| TC\_4.2\_05 | cvfdat2.ffdat2.cvsdat3.fsdat1 | FAIL |
| TC\_4.2\_06 | cvfdat2.ffdat2.cvsdat3.fsdat2.cdat1 | FAIL |
| TC\_4.2\_07 | cvfdat2.ffdat2.cvsdat3.fsdat2.cdat2 | PASS |

**TC\_4.3 RicercaCompleta**

|  |  |
| --- | --- |
| **Parametro:** Nome completo del Docente  **Formato:** ^[a-zA-Z .']+$ | |
| **Categorie** | **Scelte** |
| **Campo vuoto cvndoc** | 1. Il campo è vuoto [**errore**] 2. Il campo non è vuoto [property **campoVuotoCVNDOCok**] |
| **Esiste endoc** | 1. non esiste nel DB [if **campoVuotoCVNDOCok**] [**errore**] 2. esiste nel DB [if **campoVuotoCVNDOCok**]   [property **esisteENDOCok**] |
| **Formato fndoc** | 1. input non corrispondente al formato [if **esisteENDOCok**] [**errore**] 2. input corrispondente al formato [if **esisteENDOCok**]   [property **formatoFNDOCok**] |

|  |  |
| --- | --- |
| **Parametro:** PrimaData, SecondaData  **Formato: ^**\d{4}\-(0?[1-9]|1[012])\-(0?[1-9]|[12][0-9]|3[01])$ | |
| **Categorie** | **Scelte** |
| **Campo vuoto cvfdat** | 1. campo prima data vuoto [**errore**] 2. campo prima data non vuoto [property **campoVuotoCVFDATok**] |
| **Formato ffdat** | 1. input non corrispondente al formato [if **campoVuotoCVFDATok**] [**errore**] 2. input corrispondente al formato [property **formatoFFDATok**] |
| **Campo vuoto cvsdat** | 1. campo vuoto [if **formatoFFDATok**] [**errore**] 2. campo non vuoto [if not **campoVuotoCVFDATok**] [**errore**] 3. campo non vuoto [if **formatoFFDATok**]   [property **campoVuotoCVSDATok**] |
| **Formato fsdat** | 1. input non corrispondente al formato [if **campoVuotoCVSDATok**] [**errore**] 2. input corrispondente al formato [if **campoVuotoCVSDATok**] [property **formatoFSDATok**] |
| **Confronto cdat** | 1. ffdat > fsdat [**formatoFFDATok** and **formatoFSDATok**] [**errore**] 2. ffdat <= fsdat [if **formatoFFDATok** and **formatoFSDATok** and **campoVuotoCVSDATok**] [property **confrontoCVDATok**] |

|  |  |  |
| --- | --- | --- |
| **Codice** | **Combinazione** | **Esito** |
| TC\_4.3\_01 | cvndoc1 | FAIL |
| TC\_4.3\_02 | cvndoc2.endoc1 | FAIL |
| TC\_4.3\_03 | cvndoc2.endoc2.fndoc1 | FAIL |
| TC\_4.3\_04 | cvndoc2.endoc2.fndoc2.cvfdat1 | FAIL |
| TC\_4.3\_05 | cvndoc2.endoc2.fndoc2.cvfdat2.ffdat1 | FAIL |
| TC\_4.3\_06 | cvndoc2.endoc2.fndoc2.cvfdat2.ffdat2.cvsdat1 | FAIL |
| TC\_4.3\_07 | cvndoc2.endoc2.fndoc2.cvfdat2.ffdat2.cvsdat2 | FAIL |
| TC\_4.3\_08 | cvndoc2.endoc2.fndoc2.cvfdat2.ffdat2.cvsdat3.fsdat1 | FAIL |
| TC\_4.3\_09 | cvndoc2.endoc2.fndoc2.cvfdat2.cvfdat2.ffdat2.cvsdat3.fsdat2.cdat1 | FAIL |
| TC\_4.3\_10 | cvndoc2.endoc2.fndoc2.cvfdat2.cvfdat2.ffdat2.cvsdat3.fsdat2.cdat2 | PASS |

* 1. Selezione del formato dei report

**Nota:** il Test Case seguente si occupa di testare l’input dell’utente fornito da checkbox opzionali. Pertanto, nelle scelte di ogni categoria, sarà evidente che non è considerato errore né il caso in cui la checkbox è stata selezionata né il caso in cui non è stata selezionata. Di conseguenza, i singoli parametri verranno inseriti nel Test Case solo per evidenziare gli errori risultanti dalla combinazione degli stessi

(es: non si può selezionare la checkbox della data di nascita se non è stata selezionata quella dell’anagrafica).

**TC\_5 Selezione formato Report**

|  |  |
| --- | --- |
| **Parametro:** Anagrafica - checkbox  **Formato:** - | |
| **Categorie** | **Scelte** |
| **FlagAnagrafica fan** | 1. flag selezionato [property **flagAnagraficaFANok**] 2. flag non selezionato [property **flagAnagraficaFANok**] |

|  |  |
| --- | --- |
| **Parametro:** Data di nascita - checkbox  **Formato:** - | |
| **Categorie** | **Scelte** |
| **FlagDDN fdn** | 1. flag selezionato [property **flagDDN\_FDNok**] 2. flag non selezionato [property **flagDDN\_FDNok**] |

|  |  |
| --- | --- |
| **Parametro:** Numero di GP controllati - checkbox  **Formato:** - | |
| **Categorie** | **Scelte** |
| **FlagNumGP fgp** | 1. flag selezionato [property **flagNumGP\_FGPok**] 2. flag non selezionato [property **flagNumGP\_FGPok**] |

|  |  |
| --- | --- |
| **Parametro:** Esito - checkbox  **Formato:** - | |
| **Categorie** | **Scelte** |
| **FlagEsito fes** | 1. flag selezionato [property **flagEsitoFESok**] 2. flag non selezionato [property **flagEsitoFESok**] |

|  |  |  |
| --- | --- | --- |
| **Codice** | **Combinazione** | **Esito** |
| TC\_5\_01 | fan2.fdn1 | FAIL |
| TC\_5\_02 | fan2.fdn2.fgp2.fes2 | FAIL |

**Nota**: per questo Test Case si è deciso di indicare solamente i casi di errore (visto che i casi di successo sono determinati dalle restanti combinazioni, e che pertanto non necessitano di testing).

1. Pianificazione del Testing e Assegnazione dei ruoli

Per documentare l’organizzazione adottata, in merito alle attività di pianificazione del testing e di assegnazione dei ruoli, presentiamo la tabella sottostante che funge da matrice di tracciabilità per facilitare l’identificazione delle informazioni. In particolare, le attività di testing inizieranno con il Testing Funzionale, quindi con l’individuazione dei Test Cases, e a seguire saranno realizzate tutte le altre categorie di testing descritte.

|  |  |
| --- | --- |
| **Test Case** | **Autore** |
| TC\_01 | Martina Mulino |
| TC\_02 | Martina Mulino |
| TC\_03 | Viviana Rinaldi |
| TC\_04 | Alberto Montefusco |
| TC\_05 | Gennaro Spina |