Laurea Magistrale in informatica-Università di Salerno
Corso di *Ingegneria del Software* - Prof.ssa F. Ferrucci, Prof. F. Palomba

Test Plan

Environmental Intelligence for Agriculture

Riferimento	
Versione	1.0
Data	10/12/2022
Destinatario	Prof.ssa Filomena Ferrucci Prof. Fabio Palomba
Presentato da	C04
Approvato da	Carmine Laudato, Pierluigi Lambiase

Revision History

Data	Versione	Descrizione	Autori
07/12/2022	0.1	Aggiunta Introduzione, Panoramica del sistema	Carmine Laudato Pierluigi Lambiase
08/12/2022	0.3	Aggiunta Funzionalità da testare e Funzionalità da non testare	Carmine Laudato Pierluigi Lambiase
09/12/2022	0.5	Aggiunta Approccio, Criteria Pass/Fail, Criteri di Sospensione e Ripresa, Materiale per il testing	Carmine Laudato Pierluigi Lambiase
10/12/2022	0.9	Aggiunta Test Cases, Testing Schedule	Maria Lombardi Benedetto Scala Francesco Maria Puca Gerardo Frino Francesco Fattorusso
10/12/2022	1.0	Revisione Test Plan	Maria Lombardi Gerardo Frino Francesco Maria Puca

Sommario

1. Introduzione

In questo documento saranno scelte le linee guida e le attività di testing attuate per la piattaforma EnIA. Inoltre, saranno identificate le principali funzionalità che dovranno essere testate e quelle che non verranno testate. Lo scopo principale del testing è quello di individuare e prevedere quanti più fault possibili esistono in una funzionalità e di conseguenza nel sistema. Infatti, potremmo dire che le attività di testing hanno avuto successo se identificheremo il maggior numero di fault e failure presenti nel sistema.

1.1. Definizioni, Acronimi e Abbreviazioni

Definizioni:

- Branch Coverage: copertura da parte dei test di tutti i rami o stati condizionali del sistema;
- Failure: mancata prestazione di un servizio atteso;
- Fault: causa di un failure, insieme di istruzioni che nel momento in cui vengono eseguite generano un fallimento;
- Three Tier: modello architetturale che viene utilizzato per dividere il prodotto software in tre blocchi ognuno con una funzionalità ben distinta.

Acronimi e Abbreviazioni:

- RAD: Abbreviazione utilizzata per indicare il Requirement Analysis Document;
- SDD: Abbreviazione utilizzata per indicare il System Design Document;
- ODD: Abbreviazione utilizzata per indicare il Object Design Document;
- SOW: Abbreviazione utilizzata per indicare lo Statement Of Work;
- BC: Abbreviazione utilizzata per indicare il Business Case;
- TP: Abbreviazione utilizzata per indicare il Test Plan;
- STC: Abbreviazione utilizzata per indicare il System Test Case;
- DB: Abbreviazione utilizzata per indicare il Database;
- TC: Abbreviazione utilizzata per indicare il Test Case

1.2. Riferimenti ad altri documenti

Per la corretta individuazione dei casi di test si fa riferimenti ai seguenti documenti:

- Relazione con il Requirements Analysis Document (RAD): Tale relazione di fonda sul fatto che i test case verranno selezionati sulla base dei requisiti funzionali e non funzionali espressi nel documento sopracitato.
- Relazione con il System Design Document (SDD): Tale relazione comprende la suddivisione in sottosistemi effettuata in fase di system design e che dovrà essere preservata durante la fase di pianificazione del testing.
- Relazione con l'Object Design Document (ODD): In questo documento (ancora non sviluppato al momento del rilascio dell'C04_TP_Vers.1.0) sono contenuti i package e le classi del sistema.
- Relazione con lo Statement of Work (SOW): Tale relazione comprende la suddivisione in sottosistemi effettuata in fase di system design e che dovrà essere preservata durante la fase di pianificazione del testing.

2. Panoramica del sistema

Lo scopo del progetto EnIA è quello di offrire ai propri clienti una piattaforma web per il supporto nelle decisioni relative ad attività agroindustriali. Gli obiettivi principali di EnIA sono:

- ridurre al minimo i consumi delle risorse idriche, diminuendo al contempo l'impatto ambientale;
- fornire un tracciamento del livello di inquinamento ed esposizione ambientale della coltivazione;
- localizzazione e gestione dei vari terreni.

Tali obiettivi saranno raggiunti tramite le seguenti implementazioni di funzionalità:

- la localizzazione e gestione dei vari terreni;
- le previsioni microclimatiche e delle precipitazioni per i luoghi di interesse provenienti da open source intelligence;
- MODULO FIA: sviluppo di un modulo di intelligenza artificiale per supportare l'agricoltore nella fase decisionale tramite gli insight della piattaforma;
- informazioni inerenti ai livelli di inquinamento dei luoghi di interesse;
- una fase di analisi delle previsioni meteo con lo scopo di gestire gli impianti di irrigazione;
- una fase di analisi dell'esposizione ambientale delle varie coltivazioni.

L'architettura utilizzata è di tipo three tier. In questo tipo di architettura i sottosistemi che compongono il software sono suddivisi in tre livelli: data tier, application tier e presentation tier.

3. Funzionalità da testare

Nell'ambito del testing di sistema si andranno a verificare le principali funzionalità per ciascuna gestione funzionale individuata.

Le funzionalità selezionate da testare sono:

- Gestione Ambiente Agricolo
  - Aggiunta Ambiente Agricolo
  - Modifica Ambiente Agricolo
  - Eliminazione Ambiente Agricolo
  - Visualizzazione Ambiente Agricolo
  - Visualizzazione Meteo
- Gestione Inquinamento
  - Visualizzazione Agenti Inquinanti
- Gestione Utente
  - Registrazione Farmer
  - Gestione account utente
  - Visualizzazione area utente
  - Modifica dati utenti
- Gestione Azienda Agricola
  - Visualizzazione degli utenti
  - Aggiunta di utente e assegnazione dei ruoli
  - Rimozione utente
- Gestione degli insight
  - Decision Inteligence
  - Storico Eventi

4. Funzionalità da non testare

Le funzionalità escluse dal testing riguardano i requisisti funzionali di bassa e media priorità che verranno implementati nelle successive release.

Le funzionalità non selezionate per il testing sono:

- Gestione Ambiente Agricolo
    - Assegnazione priorità ambiente agricolo
    - Gestione irrigazione
- Gestione Inquinamento
    - Download Report Agenti Inquinanti
- Gestione degli insight
    - Sistema di alert
    - Rilevamento eventi estremi

5. Approccio

La fase di test dell'intero sistema si divide in tre fasi:

- Unit testing;
- Integration testing;
- System testing.

I test rispettivi saranno eseguiti in quest'ordine, e progettati in ordine inverso.

La progettazione dei casi di system test sarà effettuata prima della fase di implementazione del sistema. Tali casi saranno poi migliorati e perfezionati durante la loro fase di esecuzione. Il codice sarà sottoposto a periodiche attività di revisione durante tutto lo sviluppo.

Il testing sarà effettuato sfruttando un approccio di tipo Bottom-up, partendo dal testing e dall'integrazione delle singole componenti del livello più basso, per poi passare all'application tier, fino a raggiungere le interfacce degli utenti.

Le tecnologie utilizzate durante la fase di testing sono:

- **PyUnit**: utilizzata per il testing di unità e di sistema;
- **Coverage.Py**: utilizzata per effettuare l'analisi del code coverage e valutare l'efficacia dei test;
- **Selenium**: utilizzata per testare l'interazione dell'utente con la piattaforma.

Unit testing

Durante questa fase verranno testate le singole unità software attraverso un controllo delle varie classi e metodi al fine di ricercare eventuali condizioni di fallimento.

Lo Unit testing sarà effettuato dal team di sviluppo tramite l'utilizzo di classi di test, sfruttando il framework PyUnit (sarà creata una relativa classe PyUnit per ogni test case).

Integration testing

L'Integration testing sarà effettuato per assicurare il corretto funzionamento di componenti, i quali funzionano bene individualmente , nel momento in cui vengono associati ad altri componenti.

Lo scopo è quello di individuare eventuali problemi inerenti alla loro interfaccia.

Verrà applicato un approccio di tipo bottom-up, e saranno creati dei driver utili a simulare le componenti più ad alto livello non ancora testate.

System testing

il System testing serve a verificare che l'intero sistema sia coerente con quanto richiesto nei requisiti funzionali e non funzionali e quindi non richiede il codice sorgente del sistema.

Per questa fase verrà adottato un approccio di tipo black-box per la definizione dei suites test, più precisamente verrà utilizzata la metodologia Category Partition, la quale permette di ottenere un buon livello di dettaglio nel tempo a nostra disposizione.

Il system testing verrà effettuato nelle fasi finali di progettazione tramite l'ausilio di Selenium, al fine di testare in modo completo l'interfaccia utente.

6. Criteria Pass/Fail

Un case test avrà successo (pass) se la sua esecuzione porterà ad un risultato uguale a quello atteso.

Viceversa, sarà considerato fallito (fail) se il risultato ottenuto sarà differente da quello atteso.

Tutto il testing sarà considerato valido se verranno rispettate tutte le seguenti condizioni:

- Testare tutti i requisiti da testare indicati precedentemente;
- Effettuare test di regressione ogni qualvolta verranno effettuate le modifiche alle componenti del sistema;
- Ottenere un branch coverage non inferiore al 75%.

7. iteri di Sospensione e Ripresa

Si andranno a definire i criteri di sospensione del testing. Inoltre, si specificano i criteri per riprendere le attività di testing in caso di sospensione.

Criteri di sospensione

La fase di testing verrà interrotta quando verranno raggiunti i risultati attesi in accordo con in tempi e i costi allocati alle attività di testing.

Criteri di ripresa

Le attività di testing riprenderanno in seguito a delle modifiche che potrebbero introdurre nuovi errori all'interno del sistema.

8. Materiale per il testing

L'hardware necessario per l'attività di testing sarà un pc, non necessariamente avente connessione ad Internet, in quanto il sistema verrà testato in locale.

9. Test cases

La definizione dei test case sarà effettuata attraverso l'utlizzo del category partition. Per definire l'output atteso si userà un oracolo.

9.1. Gestione Utente

9.1.1. Registrazione con codice di accesso

Parametro: E-Mail	
FORMATO: **[a-zA-Z0-9_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]{2,3}**	
Nome Categoria	**Scelte per la categoria**
Formato [FE]	1. Formato valido = false [error] 2. Formato valido = true [PROPERTY FE_OK]
Parametro: Password	
FORMATO: (?=.*[0-9])(?=.*[a-z])(?=.*[A-Z])(?=.*[@#$%^&-+=()])(?=\\S+$).{8, 20}	
Formato [FP]	1. Formato valido = false [error] 2. Formato valido = true [PROPERTY FP_OK]
Parametro: Conferma Password	
Nome Categoria	**Scelte per la categoria**
Match [MCP]	1. Match con parametro password = false [error] 2. Match con parametro password = false [PROPERTY MCP_OK]
Parametro: Nome	
FORMATO: **[A-zÀ-ù '-]{2,30}**	
Nome Categoria	**Scelte per la categoria**
Formato [FNO]	1. Formato valido = false [error] 2. Formato valido = true [PROPERTY FNO_OK]
Parametro: Cognome	
FORMATO: **[A-zÀ-ù '-]{2,30}**	

Formato [FCN]	1. Formato valido = false [error] 2. Formato valido = true [PROPERTY FCN_OK]
Parametro: Numero Di Telefono	
FORMATO: ^[\+]?[(]?[0-9]{3}[)]?[-\s\.]?[0-9]{3}[-\s\.]?[0-9]{4,6}$	
Nome Categoria	**Scelte per la categoria**
Formato [FNT]	1. Formato valido = false [error] 2. Formato valido = true [PROPERTY FNT_OK]
Parametro: Codice Di Accesso	
Nome Categoria	**Scelte per la categoria**
Lunghezza [LCA]	1. Lunghezza != 16 [error] 2. Lunghezza = 16[PROPERTY LCA_OK]
MatchDB [DBCA]	1. Presente nel sistema = false [error] 2. Presente nel sistema = true [PROPERTY DBCA_OK]
Non Usato [NUCA]	1. Codice usato = true [error] 2. Codice usato = false [PROPERTY NUCA_OK]
Parametro: Indirizzo	
FORMATO: ^[A-Za-zÀ-ù0-9 ,'-]+$	
Formato [FI]	1. Formato valido = false [error] 2. Formato valido = true [PROPERTY FI_OK]
Esistenza [EI]	1. Indirizzo esistente = false [error] 2. Indirizzo esistente = true [PROPERTY EI_OK]

Test Case ID	Test Frame	Esito
TC_1.1_1	FE1	Errore: Formato e-mail non corretto
TC_1.1_2	FE2, FP1	Errore: Formato password non valido
TC_1.1_3	FE2, FP2, MCP1	Errore: Conferma password non corrisponde a password
TC_1.1_4	FE2, FP2, MCP2, FNO1	Errore: Nome non valido
TC_1.1_5	FE2, FP2, MCP2, FNO2, FCN1	Errore: Cognome non valido
TC_1.1_6	FE2, FP2, MCP2, FNO2, FCN2, FNT1	Errore: Formato numero di telefono non valido
TC_1.1_7	FE2, FP2, MCP2, FNO2, FCN2, FNT2, LCA1	Errore: Lunghezza codice di accesso non valida
TC_1.1_8	FE2, FP2, MCP2, FNO2, FCN2, FNT2, LCA2, DBCA1	Errore: Il codice di accesso non è presente nel sistema
TC_1.1_9	FE2, FP2, MCP2, FNO2, FCN2, FNT2, LCA2, DBCA2, NUCA1	Errore: Il codice di accesso è già stato usato da un altro utente
TC_1.1_10	FE2, FP2, MCP2, FNO2, FCN2, FNT2, LCA2, DBCA2, NUCA2, FI1	Errore: Il formato dell'indirizzo non è valido

TC_1.1_11	FE2, FP2, MCP2, FNO2, FCN2, FNT2, LCA2, DBCA2, NUCA2, FI2, EI1	Errore: L'indirizzo non esiste
TC_1.1_12	FE2, FP2, MCP2, FNO2, FCN2, FNT2, LCA2, DBCA2, NUCA2, FI2, EI2	Esito Valido

9.2. Gestione Ambiente Agricolo

9.2.1. Inserimento Terreno

Parametro: Nome	
Formato: ^[a-zA-Z0-9]{4,50}$	
Nome Categoria	**Scelte per la categoria**
Formato [FN]	1. Formato non valido = false[errore] 2. Formato valido = true [PROPERTY FN_OK]
Esistenza [EN]	1. Nome non inserito = false [errore] 2. Nome inserito = true [PROPERTY_EN_OK]
Parametro: Coltura	
Formato: ^[a-zA-Z0-9]{4,50}$	
Nome Categoria	**Scelte per la categoria**
Esistenza [EC]	1. Coltura non Inserita = false [errore] 2. Coltura inserit = true [PROPERTY_EC_OK]
Formato [FC]	1. Formato non valido = false[errore] 2. Formato valido = true [PROPERTY FC_OK]
Lunghezza [LC]	1. Lunghezza > 30 OR Lunghezza < 0 = false [errore] 2. Lunghezza <= 30 AND Lunghezza > 0 = true [PROPERTY_LC_OK]
Parametro: Dimensione Terreno	
Formato: m^2	
Nome Categoria	**Scelte per la categoria**

Formato[FT]	1. Formato non valido = false[errore] 2. Formato valido = true [PROPERTY FT_OK]
Esistenza [ET]	1. Dimensione non inserita = false [errore] 2. Dimensione inserita = true [PROPERTY_ET_OK]
Parametro: Posizione sulla mappa	
Formato: GEOJSON • **Polygon**	
Nome Categoria	**Scelte per la categoria**
Esistenza [EP]	1. Geojson Non Inserito = false [errore] 2. Geojson Inserito = true [PROPERTY_EP_OK]
Formato[FP]	1. Formato non valido = false[errore] 2. Formato valido = true [PROPERTY FP_OK]
Parametro: Priorità	
Formato: Boolean	
Nome Categoria	**Scelte per la categoria**
Formato[FPR]	1. Formato non valido = false[errore] 2. Formato valido = true [PROPERTY FPR_OK]

Test Case ID	Test frame	Esito
TC_2.1_1	FN1	Errato: Formato nome non valido
TC_2.1_2	FN2, EN1	Errato: Nome non inserito
TC_2.1_3	FN2, EN2, EC1	Errato: Coltura non inserita

TC_2.1_4	FN2, EN2, EC2, FC1	Errato:Formato coltura non valido
TC_2.1_5	FN2, EN2, EC2, FC2, LC1	Errato: Lunghezza errata
TC_2.1_6	FN2, EN2, EC2, FC2, LC2, FT1	Errato: Formato dimensione Terreno errata
TC_2.1_7	FN2, EN2, EC2, FC2, LC2, FT2, ET1	Errato: Dimensione terreno non inserita
TC_2.1_8	FN2, EN2, EC2, FC2, LC2, FT2, ET2, EP1	Errato: Geojson non inserito
TC_2.1_9	FN2, EN2, EC2, FC2, LC2, FT2, ET2, EP2, FP1	Errato: Formato Geojson non valido
TC_2.1_10	FN2, EN2, EC2, FC2, LC2, FT2, ET2, EP2, FP2, FPR1	Errato: formato non valido
TC_2.1_11	FN2, EN2, EC2, FC2, LC2, FT2, ET2, EP2, FP2, FPR2	Corretto

9.3. Gestione Azienda Agricola

9.3.1. Emissione codice di accesso

Parametro: Ruolo	**FORMATO: [A-Za-z]**
Nome Categoria	Scelte per la categoria
Correttezza [CRR]	1. Correttezza != “Irrigation Manager” || “Pollution Analyst” [ERROR] 2. Correttezza == “Irrigation Manager” || “Pollution Analyst” [PROPERTY CRR_OK]

Test Case ID	Test Frame	Esito
TC_4.1_1	CRR1	Errore: Ruolo non valido
TC_4.1_2	CRR2	Esito Positivo

9.4. Gestione degli insight dell'AI

9.4.1. Controllo degli Alert

Parametro: Data **FORMATO:** **^ (0? [1-9] | [12] [0-9] |3[01]) [\/\-] (0?[1-9] |1[012]) [\/\-] \d {4} $**	
Nome Categoria	**Scelte per la categoria**
Presenza (PRD)	1. Data == null [error] 2. Data != null [PROPERTY_PRD_OK]
Formato (FD)	1. ControllaFormatoData == false [error] 2. ControllaFormatoData == true [PROPERTY_FD_OK]
Corrispondenza (CD)	1. EsistonoAlertinDatabase == false [error] 2. EsistonoAlertinInDatabase == true [PROPERTY_CD_OK]

Test Case ID	Test Frame	Esito
TC_5.1_1	PRD_1	Errore: Data nulla.
TC_5.1_2	PRD_2, FD_1	Errore: Formato data non valido.
TC_5.1_3	PRD_2, FD_2, CD_1	Errore: Impossibile caricare alert.
TC_5.1_4	PRD_2, FD_2, CD_2	Corretto

9.4.2. Controllo possibili danni a colture

Parametro: Nome terreno **FORMATO:** **^[a-zA-Z0-9] {4,50} $**	
Nome Categoria	**Scelte per la categoria**
Formato (FT)	1. ControllaFormatoNomeTerreno == false [error] 2. ControllaFormatoNomeTerreno = true [PROPERTY_FT_OK]
Corrispondenza (CT)	1. EsisteInDatabase == false [error] 2. EsisteInDatabase == true [PROPERTY_CT_OK]
Parametro: Data **FORMATO:** **^ (0? [1-9] | [12] [0-9] |3[01]) [\/\-] (0?[1-9] |1[012]) [\/\-] \d {4} $**	
Nome Categoria	**Scelte per la categoria**
Presenza (PRD)	1. Data == null [error] 2. Data != null [PROPERTY_PRD_OK]
Formato (FD)	1. ControllaFormatoData == false [error]

	2. ControllaFormatoData == true [PROPERTY_FD_OK]
Corrispondenza (CD)	1. EsistonoRilevamentiConDataInDatabase == false [error] 2. EsistonoRilevamentiConDataInDatabase == true [PROPERTY_CD_OK]

Test Case ID	Test Frame	Esito
TC_3.1_1	FT_1	Errore: Formato nome dell'ambiente agricolo non valido.
TC_3.1_2	FT_2, CT_1	Errore: Non esiste un ambiente agricolo con questo nome.
TC_3.1_3	FT_2, CT_2, PRD_1	Errore: Data nulla.
TC_3.1_4	FT_2, CT_2, PRD_2, FD_1	Errore: Formato data non valido.
TC_3.1_5	FT_2, CT_2, PRD_2, FD_2, CD_1	Errore: Impossibile stabilire rischio futuro danni alle colture.
TC_3.1_6	FT_2, CT_2, PRD_2, FD_2, CD_2	Corretto

10. Testing schedule

La pianificazione delle attività di testing sarà avviata successivamente alla fine della fase di progettazione dei test rispettivi.

Si prevede di eseguire i test prima e durante la fase di implementazione del sistema. Inoltre, alcuni test saranno ripetuti qualora vi fosse possibilità di regressione durante lo sviluppo del codice.

I Test Case prodotti verranno raffinati e rivisti durante la fase di scrittura del codice.