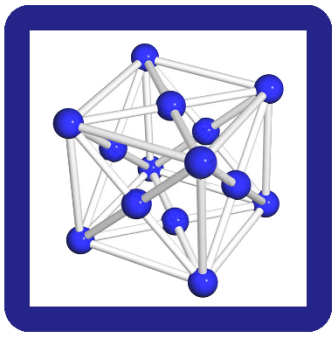
UNIVERSITÀ DEGLI STUDI DI SALERNO

UNIVERSITÀ DEGLI STUDI DI SALERNO
DIPARTIMENTO DI INFORMATICA



earth minerals
EMC
EUROPEAN METALS CORPORATION

CORSO DI INGEGNERIA DEL SOFTWARE PROF. A. DE LUCIA
PROGETTO EMC
TEST PLAN
2020/2021

PARTECIPANTI	MATRICOLA
ALESSANDRA POTESTA’	**06188**
ROSARIO ANNUNZIATA	**05810**
GIOVANNI TAVOLO	**05912**

Sommario

INTRODUZIONE

Lo scopo di questo documento è quello di gestire lo sviluppo e le attività di test riguardanti il sito web “European Metals Corporation”. Saranno identificati: gli elementi e le funzionalità da testare, le strategie di testing e gli strumenti utilizzati, il personale responsabile dei test, le risorse e le attività richieste per completare i test e i rischi associati al piano. Lo scopo del testing è quello di rilevare errori in maniera pianificata all'interno del codice realizzato. L’obiettivo del testing consiste nell’evitare che gli errori si presentino durante l’utilizzo del sistema dell'utente finale. I risultati prodotti dai test saranno utilizzati per comprendere dove intervenire per correggere gli errori o apportare modifiche per il migliorare il sistema. In questo documento verranno analizzate, in particolar modo, le seguenti attività:

- Gestione Utente
- Gestione Categoria
- Gestione Prodotto

Relazioni Con Altri Documenti

Il Test Plan ha una stretta relazione con i documenti prodotti finora, dato che il sistema è stato pianificato nelle precedenti documentazioni. Per verificare il corretto funzionamento del sito web “European Metals Corporation” saranno usati i test cases individuati e documentati precedentemente nel processo di sviluppo del sistema. I test cases sono basati sulle funzionalità individuate nel documento di raccolta ed analisi dei requisiti.

Relazioni Con Il Documento Di Analisi DeiRequisiti (Rad)

La relazione tra test plan e RAD (Requirement Analysis Document) riguarda in particolare i requisiti funzionali e non funzionali del sistema visto che i test saranno eseguiti su quelle funzionalità tenendo conto delle specifiche espresse nel documento precedente. In particolare, il RAD contiene lo scopo del sistema, l’ambito del sistema e gli obiettivi, evidenziando una panoramica di requisiti funzionali, requisiti non funzionali, scenari, casi d’uso, diagrammi e mock-up del sistema.

Relazioni Con Il System Design Document (Sdd)

Tramite il System Design Document sarà possibile definire i sottosistemi e i servizi da porre a Testing.

Relazioni Con L’object Design Document (Odd)

Tramite l’Object Design Document sarà possibile definire le classi e le componenti del

sistema da porre a Testing.

Funzionalita' Da Testare E Da Non Testare

Di seguito saranno elencate le funzionalità introdotte nel sistema che saranno sottoposte a test, suddivise per ogni gestione del sistema.

Feature da testare:

Gestione Utente

- Registrazione utente
- Login utente
- Modifica dati utenti

Gestione Categoria

- Aggiungi categoria
- Modifica categoria

Gestione Prodotto

- Aggiungi prodotto
- Modifica prodotto

Feature da non testare:

Le funzionalità escluse dal testing riguardano i requisiti funzionali di bassa e media priorità o i requisiti per cui non è necessario creare dei casi di test perché non accettano input utente particolari.

Gestione Utente

- Visualizzazione pagine informazioni (chi Siamo, About, sedi)
- Visualizzazione ordine

Gestione Categoria

- Visualizzazione categoria

Gestione Prodotto

- Visualizzazione prodotto

Criteri Pass/Fail Testing

Lo scopo del testing è quello di trovare delle failure durante l'esecuzione del sistema. Il testing ha successo se l'output osservato (finale) è diverso dall'output atteso: ciò significa che la fase di testing avrà successo se individua una failure. Nel caso verrà riscontrata una failure, bisognerà verificare a che tipo di fault è legata, se di tipo meccanico o algoritmico. Altermine dell'individuazione del fault si procederà alla sua correzione. Sarà infine iterata la fase di testing per verificare che la modifica non abbia impattato su altri componenti del sistema. La failure quindi è uno stato di condizione nel quale non si trova l'output desiderato.

Approccio

L'approccio alla fase di testing si compone di tre fasi, eseguiti nell'ordine inverso e progettati secondo il seguente ordine:

- Testing di Sistema
- Testing di Integrazione
- Testing di Unità

La progettazione dei casi di test di sistema e di integrazione avverrà prima della fase di implementazione del sistema e, inoltre, verranno raffinati durante la loro esecuzione. La progettazione dei casi di test di unità avverrà durante la fase di implementazione. La loro esecuzione avverrà durante la stesura del codice, in modo parallelo, così da avere un riscontro immediato degli errori di implementazione. Ogni volta che verrà introdotta una nuova funzionalità verranno eseguiti i test ad essa legata e, inoltre, i test di integrazione che la coinvolgono. Ad ogni modifica delle componenti, verranno eseguiti i test di unità e di integrazione delle componenti direttamente dipendenti da quelle modificate.

Testing Di Sistema

Il testing di sistema è stato realizzato tramite il tool Selenium. Questo permette di registrare le azioni che un utente svolge sul browser, in modo tale da implementare ed eseguire i test case di sistema. Per il testing, il server sarà deployato in localhost.

Testing Di Integrazione

L'approccio che verrà utilizzato è il bottom-up, ritenuto il più adatto per i software basati sul paradigma Object Oriented. Si farà utilizzo del framework JUnit per la definizione dei test case in Java, mentre Mockito sarà utilizzato per il mocking delle componenti.

Testing Di Unità

Per testing di unità si intende il testing delle singole unità software del sistema ovvero si andranno a testare ogni metodo di ciascuna classe del sistema. Il test verrà effettuato con il framework JUnit insieme alla libreria Mockito. Per Individuare i test da effettuare si utilizzerà

la tecnica Black-Box. Gli unit testing possono essere divisi principalmente in due tipologie: whitebox testing e blackbox testing. Il blackbox testing guarda solo l'input e l'output senza curare il codice, il whitebox testing invece cura solo il codice e la struttura senza guardare l'input e output.

Sospensione E Ripresa

Criteri di sospensione

La fase di testing sarà sospesa nel caso verrà rilevato un difetto che può limitare il processo di test. La sospensione del processo dovrà incidere il meno possibile sulle risorse disponibili. La fase di testing può essere sospesa qualora si raggiungeranno gli obiettivi dichiarati, rispettando i tempi fissati.

Criteri di ripresa

La fase di testing riprenderà quando il difetto verrà risolto con successo. I test verranno ripetuti per controllare se le modifiche non hanno generato nuovi errori.

Materiale Per Il Testing

Gli strumenti necessari per svolgere le attività di testing sono:

- WebServer Apache Tomcat 9 in locale che gira sul sistema.
- Un DBMS mySQL che gestisce l'utilizzo del database.
- Selenium IDE per il test di integrazione.
- JUnit per il test di unità.

Test cases

Per il test cases si utilizzerà la tecnica BLACK-BOX. Con il BLACK-BOX testing ci focalizzeremo sul comportamento dell'input/output delle singole componenti senza tener conto della loro struttura interna. A causa della mancanza di fattibilità di effettuare un test esaustivo per l'ingente quantitativo di dati di input, verrà utilizzata la strategia del Category Partition, che consente di decomporre lo spazio di input in categorie per poi partizionare le "categorie" in classi di equivalenza chiamate "scelte". Al termine saranno specificate le "combinazioni" delle scelte da testare creando delle istanze di casi di test specificando i valori dei dati effettivi per ciascuna scelta e determinare i risultati corrispondenti. Mediante il Category Partition otterremo, quindi, un test efficiente e privo di ridondanze.

Test Case Registrazione Utente

TC_RegistrazioneUtente

Parametro: e-mail Formato: ^\\w+([\\.-]?\\w+)*@\\w+([\\.-]?\\w+)*(\\.\\w+)+$	
CATEGORIE	SCELTE
Lunghezza e-mail - LE	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza > 1 - [property LE_OK]
Formato e-mail - FE	1. Non rispetta il formato [if LE_OK] [error] 2. Rispetta il formato [if LE_OK] [property FE_OK]
Esiste email - EE	1. Esiste nel DB [if LE_OK AND FE_OK] [error] 2. Non esiste nel DB [if LE_OK AND FE_OK] [property EE_OK]

Parametro: nome Formato: ^[a-zA-Z\u00C0-\u00ff]+$	
CATEGORIE	SCELTE
Lunghezza nome - LN	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >1 - [property LN_OK]
Formato nome - FN	1. Non rispetta il formato [if LN_OK AND FN_OK] [error] 2. Rispetta il formato [if LN_OK AND FN_OK] [Property FN_OK]

Parametro: username Formato: ^[0-9a-zA-Z]+$	
CATEGORIE	SCELTE
Lunghezza username - LU	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >0 and <6 - [error] 3. Lunghezza >=6 - [property LU_OK]
Formato username - FU	1. Non rispetta il formato [if LU_OK][error] 2. Rispetta il formato [if LU_OK][property FU_OK]
Esiste username - EU	1. Esiste nel DB [if LU_OK AND FU_OK] [error] 2. Non esiste nel DB [if LU_OK AND FU_OK] [property EU_OK]

Parametro: password Formato: .*[0-9].*	
CATEGORIE	SCELTE
Lunghezza password - LP	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >0 and <7 - [error] 3. Lunghezza >7 [property LP_OK]
Formato password - FP	1. Non rispetta il formato [if LP_OK] [error] 2. Rispetta il formato [if LP_OK] [property LP_OK]

Corrispondenza password - CP	1. Lunghezza ConfermaPassword = 0 - [error] 2. ConfermaPassword non corrisponde a password [if LP_OK AND FP_OK] [error] 3. ConfermaPassword corrisponde a password [if LP_OK AND FP_OK] [property CP_OK]

CODICE	COMBINAZIONE	ESITO
TC_RegistrazioneUtente_1	LN1	**Errore:campo email vuoto**
TC_RegistrazioneUtente_2	LN2, FN1	**Errore:formato non valido**
TC_RegistrazioneUtente_3	LN2, FN2, EE1	**Errore:email esistente**
TC_RegistrazioneUtente_4	LN2, FN2, EE2, LN1	**Errore:campo nome vuoto**
TC_RegistrazioneUtente_5	LN2, FN2, EE2, LN2, FN1	**Errore:formato non valido**
TC_RegistrazioneUtente_6	LN2, FN2, EE2, LN2, FN2, LU1,	**Errore:campo username vuoto**
TC_RegistrazioneUtente_7	LN2, FN2, EE2, LN2, FN2, LU2,	**Errore:lunghezza username non valida**
TC_RegistrazioneUtente_8	LN2, FN2, EE2, LN2, FN2, LU3, FU1	**Errore:formato non valido**
TC_RegistrazioneUtente_9	LN2, FN2, EE2, LN2, FN2, LU3, FU2, EU1	**Errore:username esistente**
TC_RegistrazioneUtente_10	LN2, FN2, EE2, LN2, FN2, LU3, FU2, EU2, LP1	**Errore:campo password vuoto**
TC_RegistrazioneUtente_11	LN2, FN2, EE2, LN2, FN2, LU3, FU2, EU2, LP2	**Errore:lunghezza password non valida**
TC_RegistrazioneUtente_12	LN2, FN2, EE2, LN2, FN2, LU3, FU2, EU2, LP3, FP1	**Errore:formato non valido**
TC_RegistrazioneUtente_13	LN2, FN2, EE2, LN2, FN2, LU3, FU2, EU2, LP3, FP2, CP1	**Errore:corrispondenza password vuoto**

TC_RegistrazioneUtente_14	LN2, FN2, EE2, LN2, FN2, LU3, FU2, EU2, LP3, FP2, CP2	**Errore:corrispondenza password lunghezza non valida**
TC_RegistrazioneUtente_15	LN2, FN2, EE2, LN2, FN2, LU3, FU2, EU2, LP3, FP2, CP3	**Corretto, registrazione avvenuta**

Test Case Login Utente

TC_LoginUtente

Parametro: username Formato: ^[0-9a-zA-Z]+$	
CATEGORIE	SCELTE
Lunghezza username - LU	1. Lunghezza = 0 [error] 2. Lunghezza >=6 [property LU_OK]
Esiste username - EU	1. Esiste nel DB [if LU_OK] [error] 2. Non esiste nel DB [if LU_OK] [property EU_OK]

Parametro: password Formato: .*[0-9].*	
CATEGORIE	SCELTE
Lunghezza password - LP	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >= 8 - [LP_OK]

Corrispondenza password e-mail - CPE	1. Non c'è corrispondenza tra password ed e-mail nel database [if LP_OK] [error] 2. C'è corrispondenza tra password ed e-mail nel database [if LP_OK] [property CPE_OK]

CODICE	COMBINAZIONE	ESITO
TC_LoginUtente_1	LE1	**Errore: campo e-mail vuoto**
TC_LoginUtente_2	LE2, EE1	**Errore: e-mail esiste nel database**
TC_LoginUtente_3	LE2, EE2, LP1	**Errore: campo password vuoto**
TC_LoginUtente_4	LE2, EE2, LP2, CPE1	**Errore: corrispondenza non valida**
TC_LoginUtente_5	LE, EE2, LP2, CPE2	**Corretto: login avvenuto**

Test Case Modifica Dati Utenti

TC_ModificaDatiUtenti

Parametro: username Formato: ^[0-9a-zA-Z]+$	
CATEGORIE	SCELTE
Lunghezza username - LU	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >0 AND <6 - [error] 3. Lunghezza >=6 - [property LU_OK]

Formato username - FU	1. Non rispetta il formato [if LU_OK] [error] 2. Rispetta il formato [if LU_OK] [property FU_OK]
Esiste username - EU	1. Esiste nel DB [if LU_OK AND FU_OK] [error] 2. Non esiste nel DB [if LU_OK AND FU_OK] [property EU_OK]

Parametro: nome Formato: ^[a-zA-Z\u00C0-\u00ff]+$	
CATEGORIE	SCELTE
Lunghezza nome - LN	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >=1 - [property LN_OK]
Formato nome - FN	1. Non rispetta il formato [if LN_OK] [error] 2. Rispetta il formato [if LN_OK] [property FN_OK]

Parametro: e-mail Formato: ^\\w+([\\.-]?\\w+)*@\\w+([\\.-]?\\w+)*(\\.\\w+)+$	
CATEGORIE	SCELTE
Lunghezza e-mail - LE	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >= 1 - [property LE_OK]

Formato e-mail - FE	1. Non rispetta il formato [if LE_OK] [error] 2. Rispetta il formato [if LE_OK] [property FE_OK]
Esiste email - EE	1. Esiste nel DB [if LE_OK AND FE_OK] [error] 2. Non esiste nel DB [if LE_OK AND FE_OK] [property EE_OK]

CODICE	COMBINAZIONE	ESITO
TC_ModificaDatiPersonali_1	LU1	**Errore: campo username vuoto**
TC_ModificaDatiPersonali_2	LU2	**Errore: lunghezza username non valida**
TC_ModificaDatiPersonali_3	LU3, FU1	**Errore: formato username non valido**
TC_ModificaDatiPersonali_4	LU3, FU2, EU1	**Errore: username esiste nel database**
TC_ModificaDatiPersonali_5	LU3, FU2, EU2, LN1	**Errore: campo nome vuoto**
TC_ModificaDatiPersonali_6	LU3, FU2, EU2, LN2, FN1	**Errore: formato nome non valido**
TC_ModificaDatiPersonali_7	LU3, FU2, EU2, LN2, FN2, LE1	**Errore: campo e-mail vuoto**
TC_ModificaDatiPersonali_8	LU3, FU2, EU2, LN2, FN2, LE2, FE1	**Errore: formato e-mail non valido**
TC_ModificaDatiPersonali_9	LU3, FU2, EU2, LN2, FN2, LE2, FE2, EE1	**Errore: e-mail esiste nel database**
TC_ModificaDatiPersonali_10	LU3, FU2, EU2, LN2, FN2, LE2, FE2, EE2	**Corretto: modifica dati personali avvenuta con successo**

Test Case Aggiunta Categoria

TC_AggiuntaCategoria

Parametro: nome	
CATEGORIE	SCELTE
Lunghezza nome - LN	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >=1 - [property LN_OK]
Esiste nome - EN	1. Esiste nel DB [if LN_OK] [error] 2. Non esiste nel DB [if LN_OK] [property EN_OK]

Parametro: descrizione	
CATEGORIE	SCELTE
Lunghezza descrizione - LD	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >=1 - [property LD_OK]

CODICE	COMBINAZIONE	ESITO
TC_AggiuntaCategoria_1	LN1	Errore: campo nome vuoto
TC_AggiuntaCategoria_2	LN2, EN1	Errore: nome esiste nel database
TC_AggiuntaCategoria_3	LN2, EN2, LD1	Errore: campo descrizione vuoto
TC_AggiuntaCategoria_4	LN2, EN2, LD2	Corretto: aggiunta categoria avvenuta con successo

Test Case Modifica Categoria

TC_ModificaCategoria

Parametro: nome	
CATEGORIE	SCELTE
Lunghezza nome - LN	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >=1 - [property LN_OK]

Parametro: nome e descrizione	
CATEGORIE	SCELTE
Esiste categoria - EC	1. Esiste nel DB [error] 2. Non esiste nel DB [property EC_OK]

Parametro: descrizione	
CATEGORIE	SCELTE
Lunghezza descrizione - LD	3. Lunghezza = 0 - campo vuoto [error] 4. Lunghezza >=1 - [property LD_OK]

CODICE	COMBINAZIONE	ESITO
TC_ModificaCategoria_1	LN1	Errore: campo nome vuoto
TC_ModificaCategoria_2	LN2, EC1	Errore: categoria esiste nel database
TC_ModificaCategoria_3	LN2, EC2, LD1	Errore: campo descrizione vuoto
TC_ModificaCategoria_4	LN2, EC2, LD2	Corretto: modifica categoria avvenuta con successo

Test Case Aggiunta Prodotto

TC_AggiuntaProdotto

Parametro: categoria	
CATEGORIE	SCELTE
Selezione categoria - SC	1. Selezione = false [error] 2. Selezione = true - [property SC_OK]

Parametro: nome	
CATEGORIE	SCELTE
Lunghezza nome - LN	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >=1 - [property LN_OK]
Esiste nome - EN	1. Esiste nel DB [if LN_OK] [error] 2. Non esiste nel DB [if LN_OK] [property EN_OK]

Parametro: descrizione	
CATEGORIE	SCELTE
Lunghezza descrizione - LD	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >=1 - [property LD_OK]

Parametro: prezzo	
CATEGORIE	SCELTE
Inserimento prezzo - IP	1. Inserimento = false [error] 2. Inserimento = true - [property IP_OK]

Parametro: iva	
CATEGORIE	SCELTE
Lunghezza iva - LI	3. Lunghezza = 0 - campo vuoto [error] 4. Lunghezza >=1 - [property LI_OK]

CODICE	COMBINAZIONE	ESITO
TC_AggiuntaProdotto_1	SC1	Errore: Categoria non selezionata
TC_AggiuntaProdotto_2	SC2,LN1	Errore: Campo nome vuoto
TC_AggiuntaProdotto_3	SC2,LN2,EN1	Errore: Nome non trovato

TC_AggiuntaProdotto_4	SC2,LN2,EN2,LD1	Errore: Campo descrizione vuoto
TC_AggiuntaProdotto_5	SC2,LN2,EN2,LD2,IP1	Errore: Prezzo non inserito
TC_AggiuntaProdotto_6	SC2,LN2,EN2,LD2,IP2,LI1	Errore: Campo iva vuoto
TC_AggiuntaProdotto_7	SC2,LN2,EN2,LD2,IP2,LI2	Corretto: Prodotto aggiunto correttamente

Test Case Modifica Prodotto

TC_ModificaProdotto

Parametro: categoria	
CATEGORIE	SCELTE
Selezione categoria - SC	3. Selezione = false [error] 4. Selezione = true - [property SC_OK]

Parametro: nome	
CATEGORIE	SCELTE
Lunghezza nome - LN	3. Lunghezza = 0 - campo vuoto [error] 4. Lunghezza >=1 - [property LN_OK]

Parametro: categoria, nome, descrizione, prezzo, iva	
CATEGORIE	SCELTE
Esiste prodotto- EP	1. Esiste nel DB [error] 2. Non esiste nel DB [property EP_OK]

Parametro: descrizione	
CATEGORIE	SCELTE
Lunghezza descrizione - LD	5. Lunghezza = 0 - campo vuoto [error] 6. Lunghezza >=1 - [property LD_OK]

Parametro: prezzo	
CATEGORIE	SCELTE
Inserimento prezzo - IP	3. Inserimento = false [error] 4. Inserimento = true - [property IP_OK]

Parametro: iva	
CATEGORIE	SCELTE
Lunghezza iva - LI	7. Lunghezza = 0 - campo vuoto [error] 8. Lunghezza >=1 - [property LI_OK]

CODICE	COMBINAZIONE	ESITO
TC_ModificaProdotto_1	SC1	Errore: Categoria non selezionata
TC_ ModificaProdotto_2	SC2,LN1	Errore: Campo nome vuoto
TC_ ModificaProdotto_3	SC2,LN2,EP1	Errore: Prodotto non trovato
TC_ ModificaProdotto_4	SC2,LN2,EP2,LD1	Errore: Campo lunghezza vuoto

TC_ ModificaProdotto_5	SC2,LN2,EP2,LD2,IP1	Errore: Prezzo non inserito
TC_ ModificaProdotto_6	SC2,LN2,EP2,LD2,IP2,LI1	Errore: Campo iva vuoto
TC_ ModificaProdotto_7	SC2,LN2,EP2,LD2,IP2,LI2	Corretto: Prodotto modificato correttamente

Test Case Procedi Acquisto

TC_ProcediAcquisto

Parametro: indirizzo	
CATEGORIE	SCELTE
Lunghezza indirizzo - LI	1. Lunghezza = 0 - campo vuoto [error] 2. Lunghezza >=1 - [property LI_OK]

Parametro: utente	
CATEGORIE	SCELTE
Log utente - LU	1. Login = false [error] 2. Login = true - [property LU_OK]

CODICE	COMBINAZIONE	ESITO
TC_ ProcediAcquisto_1	LI1	Errore: Campo indirizzo vuoto
TC_ ProcediAcquisto_2	LI2,LU1	Errore: Utente non loggato
TC_ ProcediAcquisto_3	LI2,LU2	Corretto: Prodotto acquistato correttamente