MyContacts

Implementazione e testing

Indice

Introduzione

In questo documento vengono descritti i principali test effettuati sul programma, con particolare attenzione all'utilizzo di strumenti come **JUnit 5**, che permette di eseguire test automatizzati sui singoli metodi delle classi. Parallelamente, sono state eseguite anche verifiche manuali su alcune delle classi principali e sui controller, sfruttando il **debugger** integrato nell'ambiente di sviluppo. Queste verifiche hanno consentito di identificare e correggere tempestivamente errori di natura superficiale, anticipando possibili problematiche rispetto alla fase formale di testing.

Test Plan

Il **piano di test** ha coinvolto sia test automatizzati tramite **JUnit** sia test manuali, in base alla natura delle classi e dei componenti interessati:

1. **Unit Testing (JUnit 5):**
   Sono state selezionate le seguenti classi per l'**Unit Testing** automatizzato, in quanto le operazioni svolte da queste classi risultano facilmente verificabili in maniera automatica:
   - **Rubrica**
   - **RubricaPreferiti**
   - **Contatto**

   La scelta di focalizzarsi su queste classi è motivata dalla loro centralità nel funzionamento del sistema e dalla necessità di garantire che le operazioni principali, come l'aggiunta, la rimozione e la gestione dei contatti e dei preferiti, funzionino correttamente.

2. **Test di Integrazione e Test Funzionali:**
   Per le **classi rimanenti** e i componenti più complessi, come i controller e i servizi, è stato adottato un approccio basato sui **Test di Integrazione** e **Test Funzionali**.
   - **Testing dell'interfaccia grafica (GUI):**
     Per la GUI, non sono stati utilizzati strumenti di testing automatizzati, ma ci siamo concentrati su:
     - **Test di usabilità**, per verificare la semplicità e intuitività dell'interfaccia per l'utente finale.
     - **Test di integrazione**, per assicurarci che la GUI interagisca correttamente con le logiche di backend.
   - Questa scelta deriva dalla natura della GUI, che è stata progettata per essere essenziale e intuitiva. Pertanto, il rischio di errori gravi è stato considerato limitato, e non si è ritenuto necessario introdurre strumenti automatizzati per il suo testing.

3. **Validazione dei dati:**
   Un aspetto cruciale del piano di test è stato garantire che l'utente **non possa inserire dati non validi** durante operazioni critiche, come l'aggiunta di contatti.
   - Ad esempio, il campo dedicato al **numero di telefono** è stato implementato in modo da accettare **esclusivamente cifre numeriche**, impedendo l'inserimento di caratteri non validi.
   - Questa misura preventiva ha ridotto la necessità di controlli aggiuntivi e ha limitato il rischio di errori, alleggerendo al contempo il carico di lavoro del sistema.

Casi di Test

In questa sezione vengono approfonditi tutti gli aspetti relativi ai **Casi di Test**, che sono stati strutturati per garantire la copertura completa delle funzionalità e la verifica delle condizioni limite.

1. **Classi di Equivalenza:**
   Le Classi di Equivalenza sono state presentate attraverso **tabelle strutturate**, che includono i vettori di test necessari per coprire ciascuna classe identificata.

   Questo approccio ha consentito di ottimizzare il numero di test eseguiti, mantenendo al contempo una copertura completa delle possibili situazioni operative.

2. **Unit Test:**
   Gli **Unit Test** sono stati organizzati e schematizzati per ciascuna delle classi testate, riportando:
   - **Input:** Parametri o valori utilizzati per ciascun metodo.
   - **Operazioni:** La descrizione dettagliata delle operazioni eseguite sui metodi delle classi.
   - **Risultati attesi:** I valori di ritorno, gli stati finali delle classi e i comportamenti attesi durante l'esecuzione del test.

3. **Test Funzionali dell'Interfaccia Utente:**

   I test dell'interfaccia grafica (GUI) sono stati condotti come **Test Funzionali**, con l'obiettivo di verificare il corretto comportamento delle funzionalità offerte all'utente. Ogni test è stato schematizzato in tabelle che descrivono chiaramente:

   - **Condizioni iniziali**
   - **Flusso degli eventi**
   - **Condizioni finali**

Test Classe Contatto

Poiché un contatto è composto da diversi campi, ci siamo concentrati sui casi in cui i nomi sono vuoti o differiscono, in quanto questi rappresentano gli elementi principali nella tabella della rubrica e sono fondamentali per l'identificazione univoca dei contatti.

Classi di Equivalenza

Classi di equivalenza: costruttore e getter	
Tutti i campi del costruttore definiti	Contatto("Mario", "Rossi", numeri, email, true)
Campi nome e cognome presenti, campi numeri e email null	Contatto("Mario", "Rossi", null, null, true)
Campo cognome vuoto	Contatto("Mario", "", numeri, mail, true)
Campo nome vuoto	Contatto("", "Rossi", numeri, mail, true)
Campi nome e cognome vuoti	Contatto("", "", numeri, mailull, true)

Classi di equivalenza: equals	
Contatti con stessi dati	Contatto("Mario", "Rossi", numeri, email, true) Contatto("Mario", "Rossi", numeri, email, true)
Contatti con stessi dati tranne che per preferito	Contatto("Mario", "Rossi", numeri, email, true) Contatto("Mario", "Rossi", numeri, email, false)
Contatti differiscono di almeno un parametro che non sia preferito	Contatto("Mario", "Rossi", numeri, email, true) Contatto("Mario", "Verdi", numeri, email, true)

Classi di equivalenza: compareTo	
Contatti con cognome che inizia con lettera diversa	Contatto("Mario", "Rossi", numeri, email, true) Contatto("Luigi", "Verdi", numeri, email, true)
Contatti con stesso cognome, ma nome che inizia con lettera diversa	Contatto("Mario", "Rossi", numeri, email, true) Contatto("Luigi", "Rossi", numeri, email, true)
Contatti con cognome con stessa lettera iniziale, ma seconda lettera diversa	Contatto("Mario", "Rossi", numeri, email, true) Contatto("Luigi", "Ricco", numeri, email, true)
Contatti che hanno solo nome con lettera iniziale diversa	Contatto("Mario", "", numeri, email, true) Contatto("Luigi", "", numeri, email, true)
Contatti che hanno solo nome con stessa lettera iniziale, ma seconda lettera diversa	Contatto("Mario", "", numeri, email, true) Contatto("Mino", "", numeri, email, true)
Contatti con stesso nome e cognome	Contatto("Mario", "Rossi", numeri, email, true) Contatto("Mario", "Rossi", numeri, email, true)

Unit Test

Test Case ID	testContatto1 TC-1
Input	Nome: “Mario” Cognome: “Rossi” Set numeri: {“3331234567”, “0817654321”, “4569868424”} Set email: {“mario.rossi@example.com”, “rossimario@gmail.com”, “tonyeffe@unisa.it”} Preferito: true
Operazione	Creazione di un contatto valido con tutti i campi inseriti
Risultato Atteso	Oggetto Contatto istanziato non nullo e correttamente popolato con i numeri e le email specificate
Dettagli Test	Verifica che il contatto venga creato correttamente con tutti i campi validi. Il contatto deve essere istanziato correttamente senza eccezioni.

Test Case ID	testContatto2 TC-2
Input	Nome: “Mario” Cognome: “Rossi” Set numeri: null Set email: null Preferito: true
Operazione	Creazione di un contatto con numeri ed email nulli
Risultato Atteso	Oggetto Contatto istanziato non nullo e correttamente popolato anche con numeri ed email nulli
Dettagli Test	Verifica che il contatto venga creato correttamente anche se numeri ed email sono nulli. La creazione del contatto non deve generare eccezioni.

Test Case ID	testContatto3 TC-3
Input	Nome: “Mario” Cognome: "" Set numeri: null Set email: null Preferito: true
Operazione	Creazione di un contatto con cognome vuoto
Risultato Atteso	Oggetto Contatto istanziato non nullo, ma con il cognome vuoto
Dettagli Test	Verifica che il contatto venga creato anche con un cognome vuoto. Non devono esserci eccezioni, ma il cognome dovrà essere gestito come una stringa vuota.

Test Case ID	testContatto4 TC-4
Input	Nome: "" Cognome: “Rossi” Set numeri: null Set email: null Preferito: true
Operazione	Creazione di un contatto con nome vuoto
Risultato Atteso	Oggetto Contatto istanziato non nullo, ma con il nome vuoto
Dettagli Test	Verifica che il contatto venga creato anche con un nome vuoto. La creazione del contatto non deve generare eccezioni e il nome sarà gestito come una stringa vuota.

Test Case ID	(boundary test) testContatto5 TC-5
Input	Nome: "" Cognome: "" Set numeri: null Set email: null Preferito: true
Operazione	Creazione di un contatto con nome e cognome vuoti
Risultato Atteso	Eccezione InvalidContactException sollevata
Dettagli Test	Verifica che la creazione di un contatto con nome e cognome vuoti generi un'eccezione InvalidContactException

Test Case ID	testGetNome1 TGN-1
Input	Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Chiamata del metodo getNome() (nome valido)
Risultato Atteso	Nome: "Mario"
Dettagli Test	Verifica che il metodo getNome() restituisca correttamente il nome del contatto quando il nome è valido.

Test Case ID	testGetNome2 TGN-2
Input	Contatto("", "Rossi", numeri, email, true)
Operazione	Chiamata del metodo getNome() (nome vuoto)
Risultato Atteso	Nome: "" (vuoto)
Dettagli Test	Verifica che il metodo getNome() restituisca una stringa vuota quando il nome del contatto è vuoto.

Test Case ID	testGetCognome1 TGC-1
Input	Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Chiamata del metodo getCognome() (cognome valido)
Risultato Atteso	Cognome: "Rossi"
Dettagli Test	Verifica che il metodo getCognome() restituisca correttamente il cognome del contatto quando il cognome è valido.

Test Case ID	testGetCognome2 TGC-2
Input	Contatto("Mario", "", numeri, email, true)
Operazione	Chiamata del metodo getCognome() (cognome vuoto)
Risultato Atteso	Cognome: "" (vuoto)
Dettagli Test	Verifica che il metodo getCognome() restituisca una stringa vuota quando il cognome del contatto è vuoto.

Test Case ID	testGetNumeri1 TGNm-1
Input	Set numeri: {"3331234567", "0817654321", "4569868424"} Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Chiamata del metodo getNumeri() su c1 (numeri validi)
Risultato Atteso	Set numeri: {"3331234567", "0817654321", "4569868424"}
Dettagli Test	Verifica che il metodo getNumeri() restituisca correttamente l'insieme dei numeri del contatto quando sono validi.

Test Case ID	TestGetNumeri2 TGNm-2
Input	Contatto("Mario", "Rossi", null, email, true)
Operazione	Chiamata del metodo getNumeri() (numeri nulli)
Risultato Atteso	Numeri: {} (insieme vuoto)
Dettagli Test	Verifica che il metodo getNumeri() restituisca un insieme vuoto quando i numeri del contatto sono nulli.

Test Case ID	testGetEmail1 TGE-1
Input	Set email: {"mario.rossi@example.com", "rossimario@gmail.com", "tonyeffe@unisa.it"} Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Chiamata del metodo getEmail() su c1 (email valide)
Risultato Atteso	Set email: {"mario.rossi@example.com", "rossimario@gmail.com", "tonyeffe@unisa.it"}
Dettagli Test	Verifica che il metodo getEmail() restituisca correttamente l'insieme delle email del contatto quando sono valide.

Test Case ID	testGetEmail2 TGE-2
Input	Contatto("Mario", "Rossi", numeri, null, true)
Operazione	Chiamata del metodo getEmail() (email nulle)
Risultato Atteso	Email: {} (insieme vuoto)
Dettagli Test	Verifica che il metodo getEmail() restituisca un insieme vuoto quando le email del contatto sono nulle.

Test Case ID	testIsPreferito1 TIP-1
Input	Contatto("Mario", "Rossi", numeri, email, false)
Operazione	Chiamata del metodo isPreferito() (contatto non preferito)
Risultato Atteso	Preferito: false
Dettagli Test	Verifica che il metodo isPreferito() restituisca false quando il contatto non è preferito.

Test Case ID	testIsPreferito2 TIP-2
Input	Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Chiamata del metodo isPreferito() (contatto preferito)
Risultato Atteso	Preferito: true
Dettagli Test	Verifica che il metodo isPreferito() restituisca true quando il contatto è preferito.

Test Case ID	testEquals1 TE-1
Input	Contatto c1 = new Contatto("Mario", "Rossi", null, null, true) Contatto c2 = new Contatto("Mario", "Rossi", null, null, true)
Operazione	Chiamata del metodo equals() su c1 e c2
Risultato Atteso	true
Dettagli Test	Verifica che il metodo equals() restituisca true quando due contatti con lo stesso nome, cognome e stato sono confrontati.

Test Case ID	testEquals2 TE-2
Input	Contatto c1 = new Contatto("Mario", "Rossi", null, null, true) Contatto c2 = new Contatto("Mario", "Rossi", null, null, false)
Operazione	Chiamata del metodo equals() su c1 e c2
Risultato Atteso	true
Dettagli Test	Verifica che il metodo equals() restituisca true quando due contatti con lo stesso nome, cognome, numeri e email, ma stato preferito diverso, sono confrontati.

Test Case ID	testEquals3 TE-3
Input	Contatto c1 = new Contatto("Mario", "Rossi", null, null, true) Contatto c2 = new Contatto("Mario", "Verdi", null, null, true)
Operazione	Chiamata del metodo equals() su c1 e c2
Risultato Atteso	false
Dettagli Test	Verifica che il metodo equals() restituisca false quando due contatti che differiscono di almeno un'informazione che non sia lo stato preferito sono confrontati.

Test Case ID	testCompareTo1 TCt-1
Input	Contatto c1 = new Contatto("Mario", "Rossi", null, null, true) Contatto c2 = new Contatto("Luigi", "Verdi", null, null, false)
Operazione	Chiamata del metodo compareTo() su c1 e c2
Risultato Atteso	c2.compareTo(c1) < 0
Dettagli Test	Verifica che il metodo compareTo() restituisca un valore negativo quando il contatto c2 ha un nome e cognome precedenti a quelli di c1.

Test Case ID	testCompareTo2 TCt-2
Input	Contatto c1 = new Contatto("Mario", "Rossi", null, null, true) Contatto c2 = new Contatto("Luigi", "Rossi", null, null, false)
Operazione	Chiamata del metodo compareTo() su c1 e c2
Risultato Atteso	c2.compareTo(c1) < 0
Dettagli Test	Verifica che il metodo compareTo() restituisca un valore negativo quando il cognome è lo stesso, ma il nome di c2 è prima di quello di c1.

Test Case ID	testCompareTo3 TCt-3
Input	Contatto c1 = new Contatto("Mario", "Rossi", null, null, true) Contatto c2 = new Contatto("Luigi", "Ricco", null, null, false)
Operazione	Chiamata del metodo compareTo() su c1 e c2
Risultato Atteso	c2.compareTo(c1) < 0
Dettagli Test	Verifica che il metodo compareTo() restituisca un valore negativo quando il nome di c1 è successivo a quello di c2, ma il cognome è diverso.

Test Case ID	testCompareTo4 TCt-4
Input	Contatto c1 = new Contatto("Mario", "", null, null, true) Contatto c2 = new Contatto("Luigi", "", null, null, false)
Operazione	Chiamata del metodo compareTo() su c1 e c2
Risultato Atteso	c2.compareTo(c1) < 0
Dettagli Test	Verifica che il metodo compareTo() restituisca un valore negativo quando il nome di c1 è successivo a quello di c2, e il cognome di c1 e c2 sono vuoti.

Test Case ID	testCompareTo5 TCt-5
Input	Contatto c1 = new Contatto("Mario", "Rossi", null, null, true) Contatto c2 = new Contatto("Mario", "Rossi", null, null, true)
Operazione	Chiamata del metodo compareTo() su c1 e c2
Risultato Atteso	c1.compareTo(c2) == 0
Dettagli Test	Verifica che il metodo compareTo() restituisca 0 quando due contatti con nome e cognome uguali vengono confrontati.

Classe Rubrica

Alcuni casi, come l'aggiunta di un contatto nullo o la rimozione di un contatto non presente in rubrica, sono stati deliberatamente omessi, poiché tali operazioni non sono accessibili all'utente tramite la nostra interfaccia grafica (GUI). Abbiamo scelto di concentrarci principalmente su operazioni più complesse e rilevanti, come la ricerca dei contatti, che riflettono i principali scenari di utilizzo e interazione dell'utente.

Classi di Equivalenza

Classi di equivalenza: getter	
Rubrica vuota	Rubrica{}
Rubrica contenente almeno un contatto	Rubrica{contatto}
Rubrica preferiti vuota	RubricaPreferiti{}
Rubrica preferiti contenente almeno un contatto	RubricaPreferiti{contatto}

Classi di equivalenza: ricercaContatti	
Esempio Rubrica	Rubrica: {Mario Rossi, Luigi Verdi, …}
Ricerca senza aver inserito caratteri	“”
Ricerca per nome intero di un contatto presente in rubrica	“Mario”
Ricerca per cognome intero di un contatto presente in rubrica	“Rossi”
Ricerca per lettera iniziale del nome di un contatto presente in rubrica	“M”
Ricerca per lettera iniziale del cognome di un contatto presente in rubrica	“R”
Ricerca a partire da caratteri che non corrispondono alle iniziali di nome e cognome di alcun contatto in rubrica	“aaa”

Classi di equivalenza: metodi di gestione della rubrica	
Esempio Rubrica	Rubrica: {Mario Rossi, Umberto Zorro, …} Rubrica preferiti: {Umberto Verdi, …}
Contatti presenti in rubrica	“Mario Rossi”, “Umberto Zorro”
Contatti non presenti in rubrica	“Andrea Sangi”, “Mick Naddeo”
Contatti presenti in rubrica preferiti	“Mario Rossi”
Contatti non presenti in rubrica preferiti	“Umberto Zorro”, “Andrea Sangi”
Contatti che hanno campo preferito “true”	Contatto("Mario", "Rossi", numeri, email, true)
Contatti che hanno campo preferito “false”	Contatto("Umberto", "Zorro", numeri, email, false)

Unit Test

Test Case ID	testGetElenco1 TGE-1
Input	rubrica = new Rubrica()
Operazione	Chiamata la funzione getElenco() su rubrica
Risultato Atteso	L'elenco dei contatti è vuoto, quindi la dimensione dell'elenco è 0
Dettagli Test	Verifica che la funzione getElenco() restituisca esattamente il set vuoto

Test Case ID	testGetElenco2 TGE-2
Input	rubrica = new Rubrica() c = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Aggiunta del contatto c alla rubrica e chiamata la funzione getElenco() su rubrica
Risultato Atteso	L'elenco dei contatti contiene un solo contatto, cioè c. La dimensione dell'elenco deve essere 1 e il contatto deve essere presente nell'elenco
Dettagli Test	Verifica che la funzione getElenco() restituisca esattamente il set contenente c

Test Case ID	testGetElencoPreferiti1 TGEP-1
Input	rubrica = new Rubrica()
Operazione	Chiamata la funzione getElencoPreferiti() su rubrica
Risultato Atteso	L'elenco dei preferiti è vuoto, quindi la dimensione dell'elenco preferiti è 0
Dettagli Test	Verifica che la funzione getElencoPreferiti() restituisca esattamente il set vuoto

Test Case ID	testGetElencoPreferiti2 TGEP-2
Input	rubrica = new Rubrica() c = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Aggiunta del contatto c ai preferiti e chiamata la funzione getElenco() su rubrica
Risultato Atteso	L'elenco dei preferiti contiene un solo contatto, cioè c. La dimensione dell'elenco preferiti deve essere 1 e il contatto deve essere presente nell'elenco preferiti.
Dettagli Test	Verifica che la funzione getElencoPreferito() restituisca esattamente il set contenente c

Test Case ID	TestRicercaContatti1 TRC-1
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c2 = new Contatto("Umberto", "Zorro", numeri2, email2, false)
Operazione	Ricerca contatti con stringa vuota ("")
Risultato Atteso	La ricerca restituisce tutti i contatti nella rubrica, quindi il risultato deve contenere entrambi i contatti (c1 e c2) con una dimensione di 2
Dettagli Test	Verifica che una ricerca vuota restituisca l'intera rubrica. L'elenco deve contenere entrambi i contatti aggiunti

Test Case ID	testRicercaContatti2 TRC-2
Operazione	Ricerca contatti con stringa "Mario"
Risultato Atteso	La ricerca restituisce solo il contatto c1, quindi il risultato deve contenere solo c1 con una dimensione di 1
Dettagli Test	Verifica che la ricerca per nome ("Mario") restituisca solo il contatto corrispondente

Test Case ID	testRicercaContatti3 TRC-3
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c2 = new Contatto("Umberto", "Zorro", numeri2, email2, false)
Operazione	Ricerca contatti con stringa "Rossi"
Risultato Atteso	La ricerca restituisce solo il contatto c1, quindi il risultato deve contenere solo c1 con una dimensione di 1
Dettagli Test	Verifica che la ricerca per cognome ("Rossi") restituisca solo il contatto corrispondente

Test Case ID	TestRicercaContatti4 TRC-4
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c2 = new Contatto("Umberto", "Zorro", numeri2, email2, false)
Operazione	Ricerca contatti con stringa "R"
Risultato Atteso	La ricerca restituisce solo il contatto c1, quindi il risultato deve contenere solo c1 con una dimensione di 1
Dettagli Test	Verifica che la ricerca per iniziale del cognome ("R") restituisca solo il contatto c1

Test Case ID	testRicercaContattiPreferiti5 TRCP-5
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c2 = new Contatto("Umberto", "Zorro", numeri2, email2, false)
Operazione	Ricerca contatti con stringa "M"
Risultato Atteso	La ricerca restituisce solo il contatto c1, quindi il risultato deve contenere solo c1 con una dimensione di 1
Dettagli Test	Verifica che la ricerca per iniziale del nome ("M") restituisca solo il contatto c1

Test Case ID	testRicercaContattiPreferiti6 TRCP-6
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c2 = new Contatto("Umberto", "Zorro", numeri2, email2, false)
Operazione	Ricerca contatti con stringa "aaa"
Risultato Atteso	La ricerca non restituisce alcun contatto, quindi il risultato deve contenere 0 elementi
Dettagli Test	Verifica che la ricerca per un nome inesistente ("aaa") non restituisca alcun contatto

Test Case ID	testAggiungiContatto TAC-1
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Aggiunta del contatto c1 alla rubrica
Risultato Atteso	L'elenco dei contatti deve contenere c1 e la sua dimensione deve essere 1
Dettagli Test	Verifica che un contatto venga aggiunto correttamente alla rubrica

Test Case ID	testRimuoviContatto TRC-1
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Il contatto c1 viene dapprima aggiunto alla rubrica correttamente e successivamente rimosso dalla rubrica
Risultato Atteso	L'elenco dei contatti deve essere vuoto o non contenere c1
Dettagli Test	Verifica che un contatto venga rimosso correttamente dalla rubrica

Test Case ID	testResetRubrica TRR-1
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Il contatto c1 viene dapprima aggiunto alla rubrica correttamente e successivamente viene effettuata un reset della rubrica
Risultato Atteso	L'elenco dei contatti e l'elenco dei preferiti devono essere vuoti
Dettagli Test	Verifica che il reset della rubrica rimuova tutti i contatti e i preferiti

Test Case ID	testIsRubricaVuota1 TIRV-1
Input	rubrica = new Rubrica()
Operazione	Verifica se la rubrica è vuota utilizzando il metodo isRubricaVuota()
Risultato Atteso	La rubrica è vuota, quindi il metodo restituisce true
Dettagli Test	Verifica che la rubrica sia vuota e il metodo isRubricaVuota() restituisca true

Test Case ID	testIsRubricaVuota2 TIRV-2
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Verifica se la rubrica è vuota utilizzando il metodo isRubricaVuota() dopo aver aggiunto il contatto c1 alla rubrica
Risultato Atteso	La rubrica non è vuota, quindi il metodo restituisce false
Dettagli Test	Verifica che il metodo isRubricaVuota() restituisca false dopo aver aggiunto un contatto alla rubrica

Test Case ID	testAggiungiAPreferiti1 TAP-1
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Aggiunta del contatto c1 ai preferiti
Risultato Atteso	Il contatto c1 è presente nell'elenco dei preferiti
Dettagli Test	Verifica che un contatto con stato preferito true venga aggiunto correttamente all'elenco dei preferiti.

Test Case ID	testAggiungiAPreferiti2 TAP-2
Input	rubrica = new Rubrica() c2 = new Contatto("Umberto", "Zorro", numeri2, email2, false)
Operazione	Tentativo di aggiungere il contatto c2 ai preferiti
Risultato Atteso	Il contatto c2 non è presente nell'elenco dei preferiti
Dettagli Test	Verifica che un contatto con stato preferito false non venga aggiunto ai preferiti se non è già presente nella rubrica

Test Case ID	testRimuoviDaPreferiti TRDP-1
Input	rubrica = new Rubrica() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Il contatto c1 viene dapprima aggiunto alla rubrica preferiti correttamente e successivamente viene rimosso
Risultato Atteso	Il contatto c1 non è più presente nell'elenco dei preferiti
Dettagli Test	Verifica che un contatto possa essere rimosso correttamente dall'elenco dei preferiti

Classe RubricaPreferiti

I test effettuati su **RubricaPreferiti** sono stati tra i più semplici e rapidi da implementare e verificare, poiché si tratta di una classe autonoma che gestisce un set di contatti tramite operazioni semplici ed efficienti. Analogamente a quanto avvenuto con la classe **Rubrica**, alcuni scenari potenzialmente problematici sono stati trascurati, in quanto tali operazioni non sono accessibili all'utente attraverso la nostra interfaccia grafica (GUI).

Classi di Equivalenza

Classi di equivalenza: getter	
Rubrica preferiti vuota	RubricaPreferiti{}
Rubrica preferiti contenente almeno un contatto	RubricaPreferiti{contatto, ...}

Classi di equivalenza: metodi di gestione della rubrica	
Esempio Rubrica Preferiti	Rubrica Preferiti: {Mario Rossi, Umberto Zorro, ...}
Contatti presenti in rubrica preferiti	“Mario Rossi”, “Umberto Zorro”
Contatti non presenti in rubrica preferiti	“Andrea Sangi”, “Luigi Verdi”
Contatti che hanno campo preferito “true”	Contatto("Mario", "Rossi", numeri, email, true)
Contatti che hanno campo preferito “false”	Contatto("Umberto", "Zorro", numeri, email, false)

Unit Test

Test Case ID	testAddContattoPreferito1 TAPC-1
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Aggiunta di c1 ai preferiti utilizzando il metodo addContattoPreferito()
Risultato Atteso	Il contatto c1 è presente nell'elenco dei preferiti
Dettagli Test	Verifica che il contatto c1 con stato preferito true venga correttamente aggiunto all'elenco dei preferiti

Test Case ID	testAddContattoPreferito2 TACP-2
Input	rubrica = new RubricaPreferiti() c2 = new Contatto("Umberto","Zorro",numeri2,email2,false)
Operazione	Tentativo di aggiungere c2 ai preferiti utilizzando il metodo addContattoPreferito()
Risultato Atteso	Il contatto c2 non è presente nell'elenco dei preferiti
Dettagli Test	Verifica che un contatto che non è preferito non venga aggiunto all'elenco dei preferiti

Test Case ID	testRemoveContattoPreferito TRCP-1
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Dopo aver aggiunto correttamente ai preferiti c1, viene rimosso dai preferiti utilizzando il metodo removeContattoPreferito()
Risultato Atteso	Il contatto c1 non è più presente nell'elenco dei preferiti
Dettagli Test	Verifica che un contatto possa essere rimosso correttamente dall'elenco dei preferiti

Test Case ID	testResetRubricaPreferiti TRRP-1
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Dopo aver aggiunto correttamente ai preferiti c1, viene effettuato un reset della rubrica dei preferiti utilizzando il metodo resetRubricaPreferiti()
Risultato Atteso	L'elenco dei preferiti è vuoto dopo il reset
Dettagli Test	Verifica che tutti i contatti vengano rimossi dalla rubrica dei preferiti dopo il reset

Test Case ID	testGetElencoPreferiti1 TGEP-1
Input	rubrica = new RubricaPreferiti()
Operazione	Recupero dell'elenco dei preferiti tramite il metodo getElencoPreferiti()
Risultato Atteso	La dimensione dell'elenco dei preferiti è 0
Dettagli Test	Verifica che l'elenco dei preferiti ottenuto sia vuoto

Test Case ID	testGetElencoPreferiti2 TGEP-2
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true)
Operazione	Recupero dell'elenco dei preferiti dopo aver aggiunto c1 tramite il metodo getElencoPreferiti()
Risultato Atteso	La dimensione dell'elenco dei preferiti è 1 e c1 è presente
Dettagli Test	Verifica che l'elenco dei preferiti contenga c1 dopo l'aggiunta del contatto

Test Case ID	testRicercaContattiPreferiti1 TRCP-1
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c3 = new Contatto("Umberto","Zorro",numeri2,email2,true)
Operazione	Dopo aver aggiunto i due contatti alla rubrica si effettua una ricerca nell'elenco dei preferiti con campo vuoto ("")
Risultato Atteso	La ricerca restituisce entrambi i contatti (c1 e c3), con dimensione 2
Dettagli Test	Verifica che la ricerca con campo vuoto restituisca tutti i contatti preferiti

Test Case ID	testRicercaContattiPreferiti2 TRCP-2
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c3 = new Contatto("Umberto","Zorro",numeri2,email2,true)
Operazione	Dopo aver aggiunto i due contatti alla rubrica si effettua una ricerca nell'elenco dei preferiti per nome "Mario"
Risultato Atteso	La ricerca restituisce solo c1, con dimensione 1
Dettagli Test	Verifica che la ricerca per nome restituisca correttamente il contatto c1

Test Case ID	testRicercaContattiPreferiti3 TRCP-3
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c3 = new Contatto("Umberto","Zorro",numeri2,email2,true)
Operazione	Dopo aver aggiunto i due contatti alla rubrica si effettua una ricerca nell'elenco dei preferiti per cognome "Rossi"
Risultato Atteso	La ricerca restituisce solo c1, con dimensione 1
Dettagli Test	Verifica che la ricerca per cognome restituisca correttamente il contatto c1

Test Case ID	testRicercaContattiPreferiti4 TRCP-4
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c3 = new Contatto("Umberto","Zorro",numeri2,email2,true)
Operazione	Dopo aver aggiunto i due contatti alla rubrica si effettua una ricerca nell'elenco dei preferiti per iniziale del cognome "R"
Risultato Atteso	La ricerca restituisce solo c1, con dimensione 1
Dettagli Test	Verifica che la ricerca per iniziale del cognome restituisca correttamente il contatto c1

Test Case ID	testRicercaContattiPreferiti5 TRCP-5
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c3 = new Contatto("Umberto","Zorro",numeri2,email2,true)
Operazione	Dopo aver aggiunto i due contatti alla rubrica si effettua una ricerca nell'elenco dei preferiti per iniziale del nome "M"
Risultato Atteso	La ricerca restituisce solo c1, con dimensione 1
Dettagli Test	Verifica che la ricerca per iniziale del nome restituisca correttamente il contatto c1.

Test Case ID	testRicercaContattiPreferiti6 TRCP-6
Input	rubrica = new RubricaPreferiti() c1 = new Contatto("Mario", "Rossi", numeri, email, true) c3 = new Contatto("Umberto","Zorro",numeri2,email2,true)
Operazione	Dopo aver aggiunto i due contatti alla rubrica si effettua una ricerca nell'elenco dei preferiti inserendo una stringa casuale che non coincida con iniziali di alcun nome e cognome “aaa”
Risultato Atteso	La ricerca restituisce 0 risultati
Dettagli Test	Verifica che la ricerca per nome inesistente non restituisca alcun contatto

Test Funzionali

	Aggiunta di un contatto [TF-1.1]
Entry Condition	La GUI è aperta e la rubrica è in esecuzione.
Flow of Events	1. L'utente clicca su **"Aggiungi"** nella barra superiore. 2. Compilare i campi richiesti (nome, cognome, numero, email, ecc.). 3. Confermare l'operazione tramite **"Conferma"**.
Exit Condition	Il contatto viene correttamente aggiunto alla rubrica.

	Modifica di un contatto [TF-1.2]
Entry Condition	La GUI è aperta e almeno un contatto è selezionato nella rubrica.
Flow of Events	1. L'utente seleziona un contatto dalla rubrica. 2. Cliccare su **"Modifica"** nella barra superiore. 3. Aggiornare i campi desiderati (nome, cognome, ecc.). 4. Confermare tramite **"Conferma"**.
Exit Condition	Le modifiche vengono salvate e aggiornano il contatto nella rubrica.

	Eliminazione di un contatto [TF-1.3]
Entry Condition	La GUI è aperta e almeno un contatto è selezionato nella rubrica.
Flow of Events	1. L'utente seleziona un contatto dalla lista della rubrica. 2. Cliccare su **"Elimina"** nella barra superiore.
Exit Condition	Il contatto viene rimosso dalla rubrica.

	Aggiunta di un contatto ai preferiti [TF-1.4]
Entry Condition	La GUI è aperta e almeno un contatto è presente nella rubrica.
Flow of Events	1. L'utente seleziona un contatto dalla lista della rubrica. 2. Cliccare su **"Modifica"** nella barra superiore. 3. Spuntare l'opzione **"Preferito"** (checkbox). 4. Confermare tramite il pulsante **"Conferma"**.
Exit Condition	Il contatto viene aggiunto correttamente alla lista dei preferiti.

	Rimozione di un contatto dai preferiti [TF-1.5]
Entry Condition	La GUI è aperta e almeno un contatto è presente nella lista dei preferiti.
Flow of Events	1. L'utente seleziona un contatto dalla lista dei preferiti. 2. Cliccare su **"Modifica"** nella barra superiore. 3. Deselezionare l'opzione **"Preferito"** (checkbox). 4. Confermare tramite il pulsante **"Conferma"**.
Exit Condition	Il contatto viene rimosso correttamente dalla lista dei preferiti.

	Ricerca contatto nella rubrica [TF-1.6]
Entry Condition	La GUI è aperta e almeno un contatto è presente nella rubrica.
Flow of Events	1. L'utente inserisce una stringa nel campo **"Ricerca contatti"**.
Exit Condition	La lista della rubrica mostra solo i contatti corrispondenti alla ricerca.

	Importazione contatti [TF-1.7]
Entry Condition	La GUI è aperta e la rubrica è vuota o contiene alcuni contatti.
Flow of Events	1. L'utente clicca sull'opzione **"Importa"** nella barra superiore. 2. Selezionare un file valido contenente contatti. 3. Confermare l'importazione.
Exit Condition	I contatti presenti nel file vengono aggiunti alla rubrica.

	Esportazione contatti [TF-1.8]
Entry Condition	La GUI è aperta e almeno un contatto è presente nella rubrica.
Flow of Events	1. L'utente clicca sull'opzione **"Esporta"** nella barra superiore. 2. Scegliere la posizione e il formato del file di esportazione. 3. Confermare l'operazione.
Exit Condition	Il file viene generato e contiene tutti i contatti presenti nella rubrica.

	Reset della rubrica [TF-1.9]
Entry Condition	La GUI è aperta e la rubrica contiene almeno un contatto.
Flow of Events	1. L'utente clicca sull'opzione **"Reset"** nella barra superiore.
Exit Condition	Tutti i contatti vengono rimossi dalla rubrica.

	Visualizzazione dei contatti preferiti [TF-1.10]
Entry Condition	La GUI è aperta e il pulsante "Preferiti" è visibile.
Flow of Events	1. L'utente clicca sul pulsante **"Preferiti"** (toggle). 2. Vengono filtrati e visualizzati solo i contatti contrassegnati come preferiti.
Exit Condition	La lista della rubrica mostra esclusivamente i contatti preferiti.

	Esportazione contatti [TF-1.11]
Entry Condition	La GUI è aperta e almeno un contatto è presente nella rubrica.
Flow of Events	1. L'utente clicca su "Esporta". 2. Seleziona la destinazione e il formato del file. 3. Conferma l'operazione.
Exit Condition	I contatti vengono salvati correttamente nel file selezionato.

	Importazione contatti [TF-1.12]
Entry Condition	La GUI è aperta e la rubrica può essere vuota o contenere alcuni contatti.
Flow of Events	1. L'utente clicca su "Importa". 2. Seleziona un file valido contenente contatti. 3. Conferma l'importazione.
Exit Condition	I contatti vengono correttamente aggiunti alla rubrica.

Test Incident Reports e Test Report Summary

Durante l'esecuzione dei test automatizzati, abbiamo riscontrato vari fallimenti, principalmente legati alla classe *Rubrica*. La maggior parte degli errori erano dovuti a problemi di scrittura del codice e a fattori che inizialmente non avevamo preso in considerazione, come ad esempio l'aggiunta di contatti non preferiti alla rubrica dei contatti con stato preferito "false". Tuttavia, grazie alla strutturazione delle classi in base a criteri di coesione e accoppiamento, siamo riusciti a correggere facilmente questi errori, intervenendo esclusivamente sulle classi interessate, senza compromettere il resto del sistema.

In parallelo, i test funzionali si sono rivelati fondamentali per individuare errori difficilmente rilevabili durante i test automatizzati. Anche in questo caso, è stato necessario intervenire su alcune parti del codice, in particolare sulle classi controller, che gestiscono la GUI e le interazioni con l'utente. Un esempio significativo riguarda la gestione della *TableView* associata alla rubrica e l'aggiornamento dinamico della stessa durante le operazioni di aggiunta, modifica ed eliminazione dei contatti.

Una volta effettuate le necessarie modifiche, sono stati rieseguiti tutti gli unit test e i test funzionali. Al termine della fase di implementazione, tutti i test hanno dato esito positivo, confermando la correttezza del sistema.

Matrice di Tracciabilità

Requisito	Design	Codice	Test	Requisiti correlati
R-1	LeftViewController, RightView1Controller	LeftViewController, RightView1Controller, Rubrica, RubricaPreferiti	T.F 1.1, T.F 1.3, T.F 1.7, T.F 1.8	R-2, R-5, R-6
R-2	LeftViewController, RightView2Controller	LeftViewController, RightView2Controller, Rubrica, Contatto	T.F 1.10, T.F 1.2, T.F 1.7, T.F 1.8	R-1, R-3, R-6
R-3	RightView1Controlle, RightView2Controller	RightView1Controlle, RightView2Controller, Rubrica, Contatto	T.F1.1	R-2, R-4
R-4	RightView2Controller	RightView2Controller, Contatto	T.F 1.1, T.F 1.2	R2, R-3
R-5	RightView1Controlle, RightView2Controller	RightView1Controlle, RightView2Controller, Rubrica	T.F 1.10, T.F 1.2, T.F 1.7, T.F 1.8	R-1, R-3, R-6
R-6	LeftViewController	LeftViewController, Rubrica	T.F 1.10, T.F 1.7, T.F 1.8, TF-1.11, TF-1.12	R-1, R-6
R-7	LeftViewController	LeftViewController, Rubrica	T.F 1.10, T.F 1.2	R-1, R-3

Partecipanti

NOME	COGNOME	MATRICOLA
Gabriele	Lupo	0612707641
Michele	Naddeo	0612708622
Andrea	Sangineto	0612707424
Gianmarco	Vitolo	0612708376