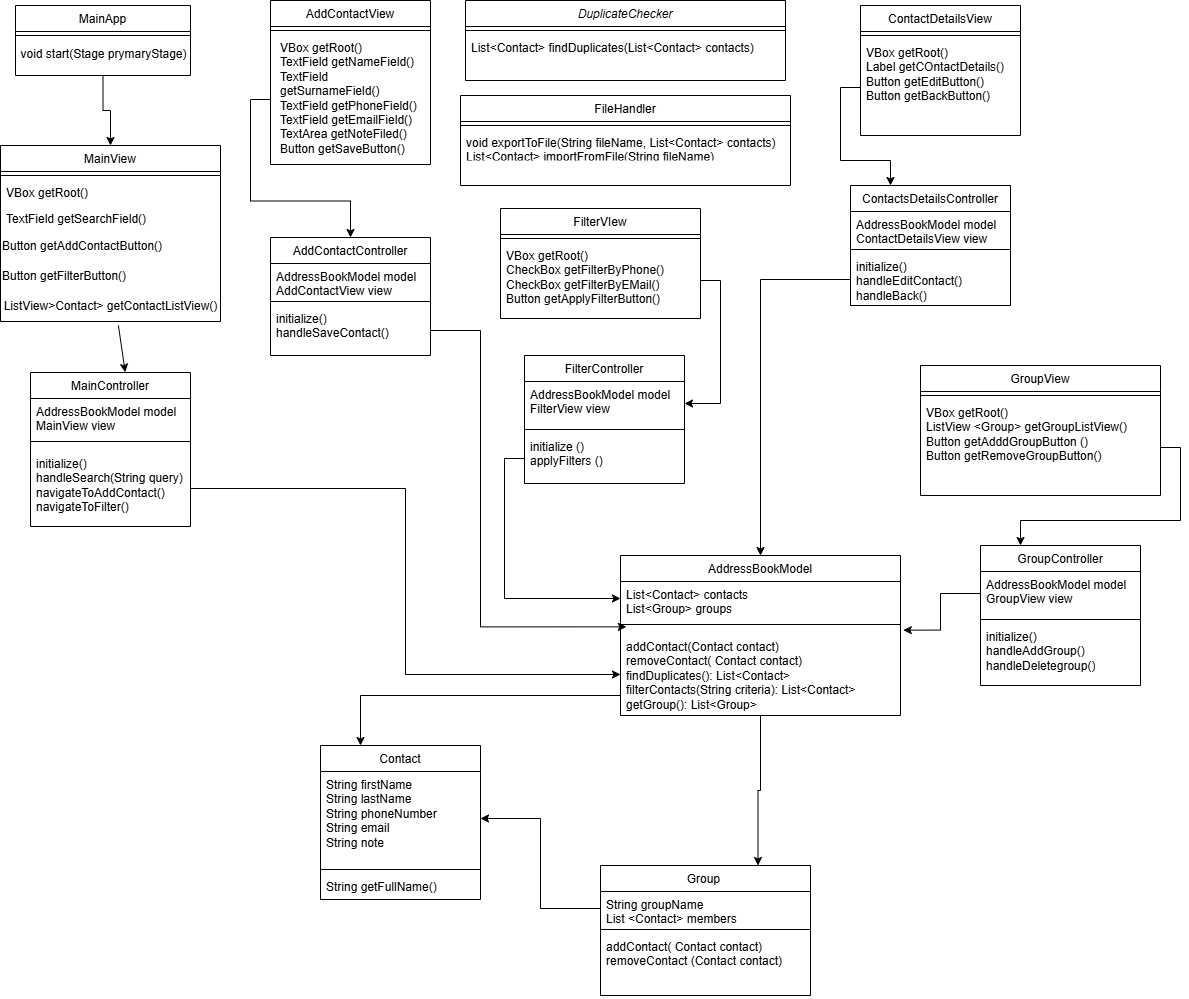
**Design**

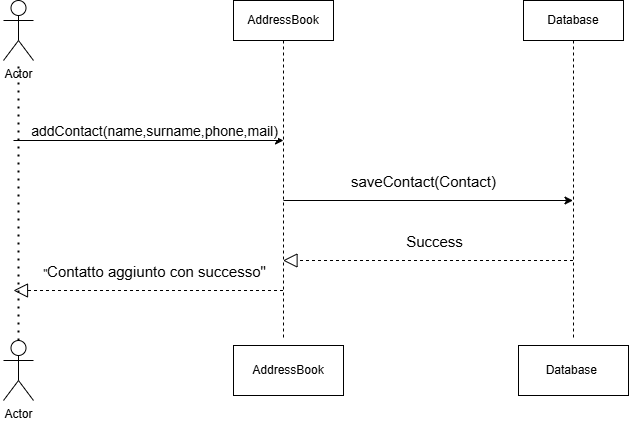
**Class diagrams**



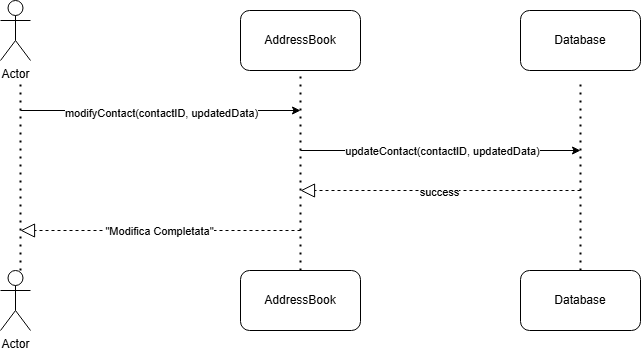
Il diagramma rappresenta un'applicazione per gestire una rubrica, organizzata secondo MVC (Model-View-Controller).Il modello gestisce i dati dell'applicazione, come contatti e gruppi, offrendo metodi per aggiungere, rimuovere, filtrare e trovare duplicati. Le viste rappresentano l'interfaccia utente, fornendo elementi grafici per visualizzare, aggiungere, filtrare o modificare i contatti e gestire i gruppi. I controller mediano tra le viste e il modello, gestendo la logica applicativa, come il salvataggio di un contatto o l'applicazione di un filtro.Inoltre, ci sono classi di utilità per la gestione di duplicati e l'import/export dei dati. L'applicazione viene avviata dalla classe principale, che collega la vista principale al suo controller.Nel diagramma, le classi mostrano un'alta coesione poiché ogni classe ha una responsabilità chiara e ben definita. L'accoppiamento è ben gestito perché le viste interagiscono solo con i rispettivi controller e non direttamente con il modello, riducendo le dipendenze.

**Sequence diagrams**

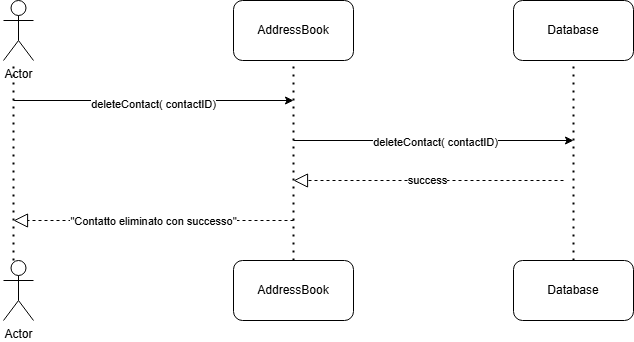
1. **Aggiungi un contatto (SD1)**

****

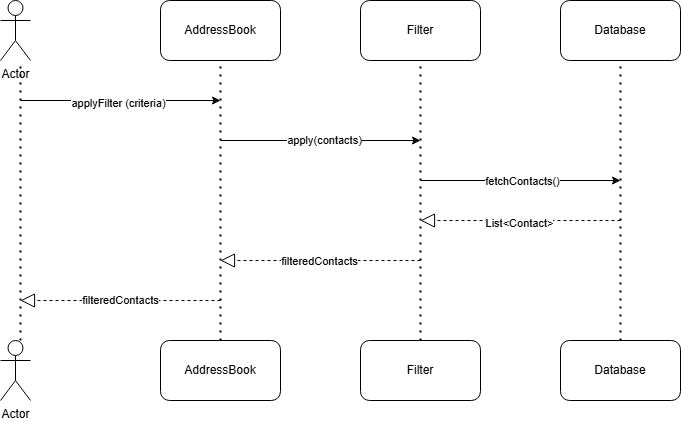
Questo diagramma di sequenza mostra il processo di aggiunta si un contatto, abbiamo fatto in modo di evidenziare una chiara separazione di responsabilità tra le componenti, in modo tale da garantire un’alta coesione e un basso accoppiamento poiché le classi hanno un ruolo ben specifico

1. **Modifica un contatto**

Il diagramma mostra il processo di modifica di un contatto, dove l'utente interagisce con la classe AddressBook che gestisce la logica principale e coordina con il database per aggiornare le informazioni. La coesione è alta, poiché ogni componente ha un ruolo ben definito.

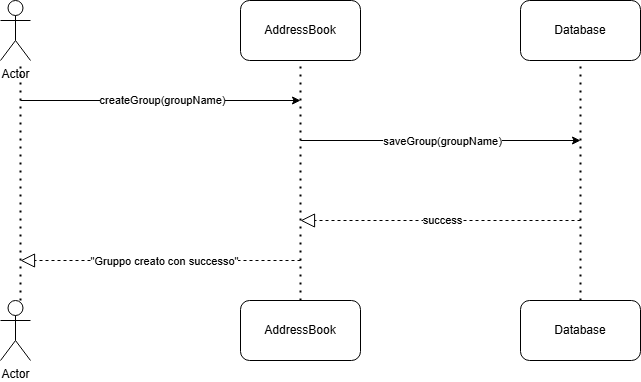
1. **Elimina contatto**

Il diagramma descrive il processo di eliminazione di un contatto, dove l'utente invia la richiesta a AddressBook specificando l'ID del contatto da eliminare. Il diagramma mostra alta coesione poiché ogni componente ha un ruolo ben specifico e basso accoppiamento perché l’utente non interagisce direttamente con il database.

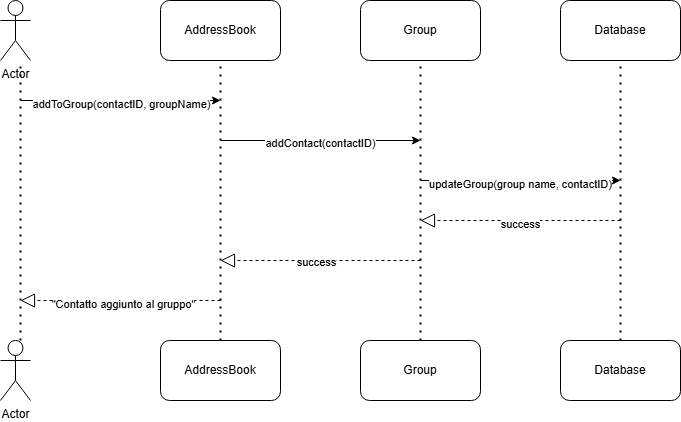
1. **Applica un filtro personalizzato** 

Il diagramma illustra il processo di applicazione di un filtro ai contatti. L'utente invia una richiesta ad AddressBook specificando i criteri di filtro. Il diagramma evidenzia una buona coesione, con ogni componente dedicato a una specifica responsabilità.

1. **Creazione di un gruppo**

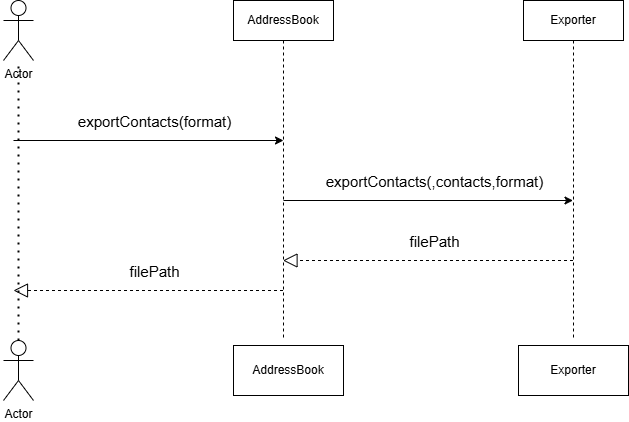
****

Il diagramma rappresenta il processo di creazione di un gruppo, con un'interazione chiara tra utente, AddressBook e Database. La AddressBook gestisce la logica applicativa e delega al Database la memorizzazione dei dati, garantendo una separazione dei ruoli che migliora la coesione.

1. **Aggiunta di un contatto a un gruppo** 

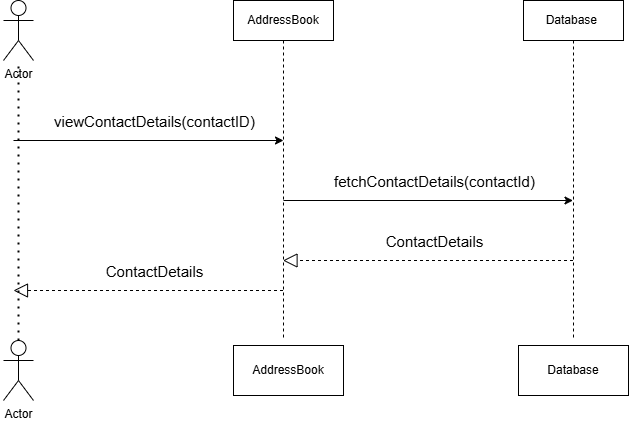
Il diagramma illustra il processo di aggiunta di un contatto a un gruppo. L'utente invia una richiesta alla AddressBook, che delega l'operazione alla classe Group, responsabile della gestione dei contatti nei gruppi. Il diagramma mantiene una chiara separazione dei ruoli, garantendo coesione e un basso accoppiamento tra le componenti, facilitando la manutenibilità.

1. **Esportazione di un contatto**



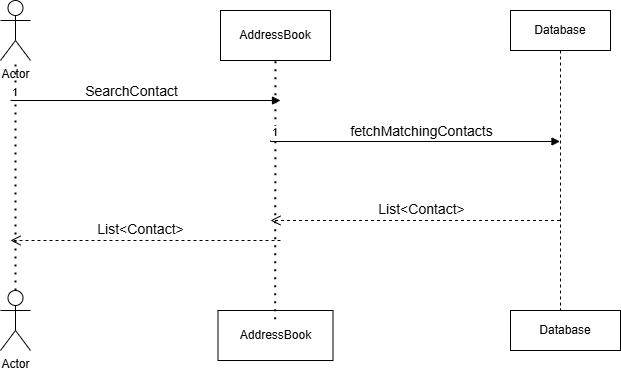
Il diagramma descrive il processo di esportazione dei contatti. L'utente invia una richiesta ad AddressBook, specificando il formato desiderato. La AddressBook delega l'operazione alla classe Exporter. Questo approccio mantiene coesione tra le classi, e riduce l'accoppiamento.

1. **Visualizzazione dei dettagli di un contatto**

****

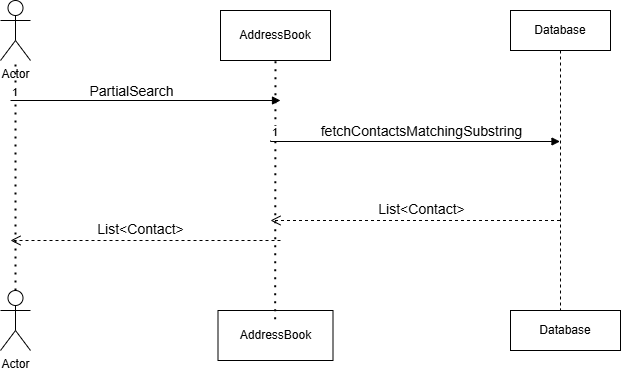
Il diagramma rappresenta il processo di visualizzazione dei dettagli di un contatto. L'utente richiede alla AddressBook i dettagli di un contatto specifico.La AddressBook delega la ricerca al Database, che restituisce le informazioni richieste, mantenendo coesione e un basso accoppiamento tra i componenti.

1. **Cerca un contatto**

****

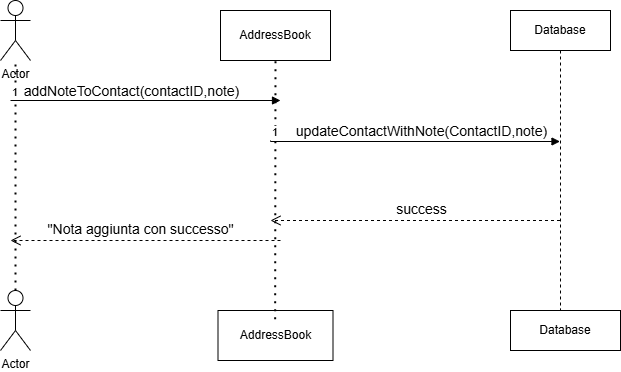
Il sequence diagram mostra come un utente cerchi un contatto fornendo un criterio ad AddressBook, che delega al Database la ricerca dei contatti corrispondenti. Il database restituisce una lista di contatti trovati, che viene inoltrata all'utente. L’accoppiamento è mantenuto basso poiché AddressBook interagisce con Database solo attraverso metodi ben definiti.

1. **Ricerca Parziale**

****

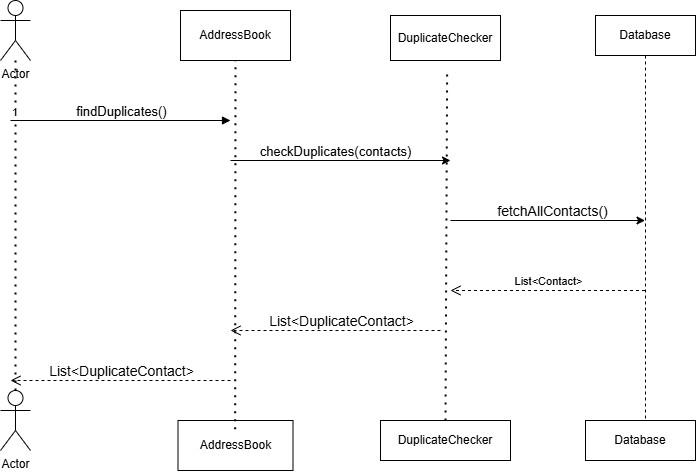
Nel diagramma per la ricerca parziale, la coesione è alta perché ogni elemento è responsabile di un compito preciso: l'attore User invia una richiesta specifica a AddressBook, che delega al Database l'effettivo recupero dei contatti che corrispondono ai criteri parziali.

1. **Aggiungere una nota a un contatto**

****

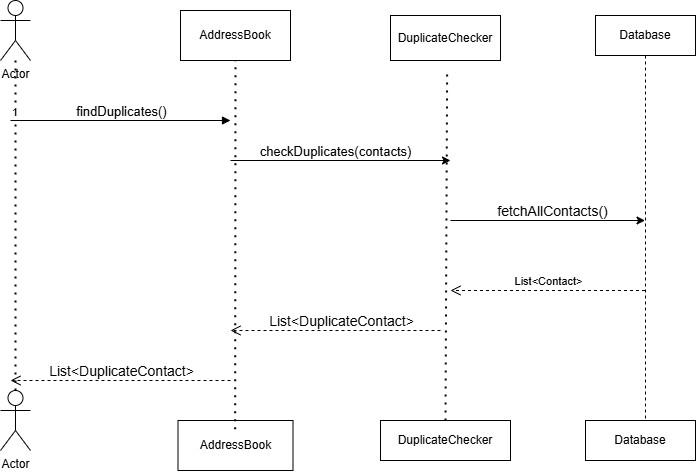
Nel diagramma della funzionalità aggiungere una nota a un contatto, la coesione è mantenuta alta poiché ogni elemento ha un compito ben definito: l'attore User invia la richiesta a AddressBook, che funge da mediatore per aggiornare il contatto nel Database con la nota aggiuntiva. Questo garantisce che ogni componente si occupi di un aspetto specifico del processo.L’accoppiamento è basso, poiché AddressBook interagisce con Database tramite un’operazione specifica.

1. **Segnalare contatti duplicati**

****

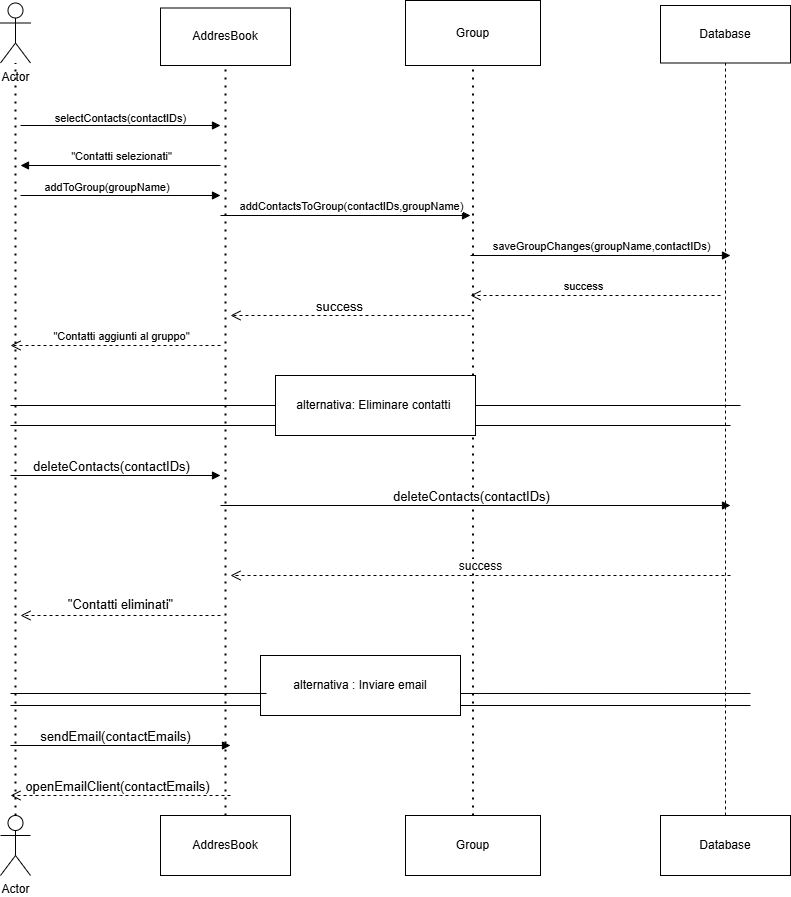
Nel diagramma per la funzionalità segnalare contatti duplicati, la coesione è mantenuta poiché ogni componente è responsabile di un compito specifico: l'utente User inizia il processo tramite AddressBook, che delega il controllo dei duplicati a DuplicateChecker. Questo componente si occupa di analizzare i contatti forniti dal Database e restituisce la lista dei duplicati trovati.

1. **Visualizzazione ordinata dei contatti**

****

Nel diagramma per la funzionalità segnalare contatti duplicati, la coesione è mantenuta poiché ogni componente è responsabile di un compito specifico: l'utente User inizia il processo tramite AddressBook, che delega il controllo dei duplicati a DuplicateChecker. Questo componente si occupa di analizzare i contatti forniti dal Database e restituisce la lista dei duplicati trovati.

1. **Gestione contatti avanzati**

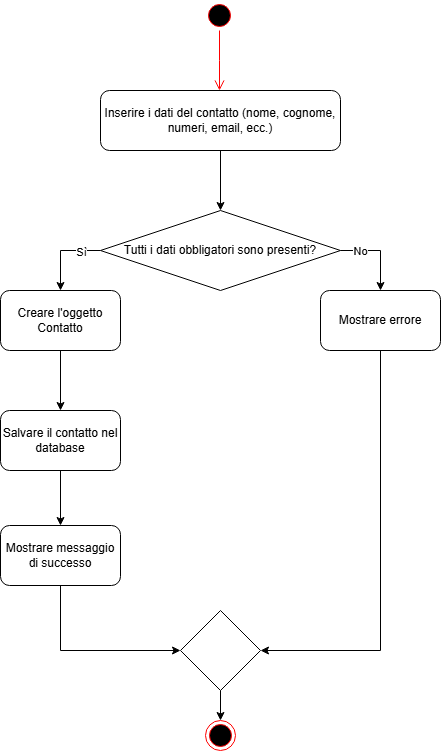
****

Nel diagramma per la gestione avanzata dei contatti, l'interazione si concentra su operazioni multiple come l'aggiunta a un gruppo, l'eliminazione e l'invio di email. La coesione è ben mantenuta poiché ogni componente ha un ruolo ben definito.L’accoppiamento è mantenuto basso attraverso la separazione delle responsabilità.

**OTHER DIAGRAMS:**

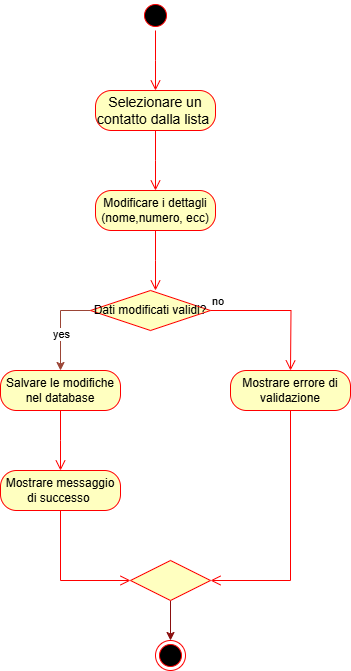
* **ACTIVITY DIAGRAMS:**

1. **Aggiungere un contatto**

****

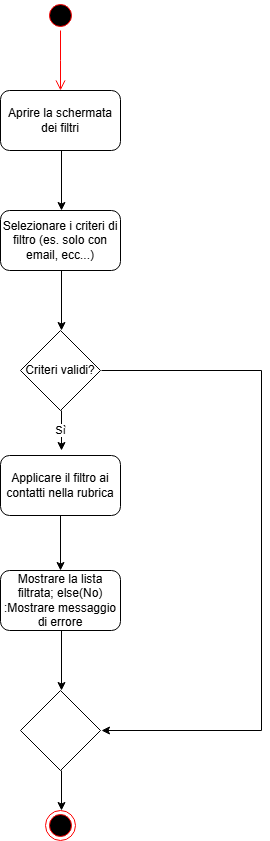
Questo diagramma di attività descrive il flusso per aggiungere un nuovo contatto. L'utente inserisce i dati del contatto, e il sistema verifica se tutti i dati obbligatori (ad esempio nome e cognome) sono presenti. Se invece i dati obbligatori mancano, viene mostrato un messaggio di errore, interrompendo il flusso.Il diagramma è ben coeso .Inoltre, la separazione delle operazioni di verifica, creazione e salvataggio riduce l'accoppiamento.

1. **Modificare un contatto**

****

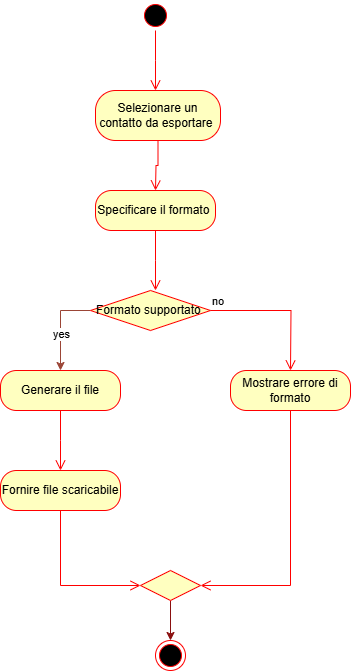
Questo diagramma di attività rappresenta il flusso per la modifica di un contatto. L'utente seleziona un contatto dalla lista e modifica i dettagli desiderati. Il sistema verifica la validità dei dati aggiornati. Se i dati sono validi, le modifiche vengono salvate nel database e un messaggio di successo viene mostrato all'utente.Il diagramma mantiene una chiara coesione, con ogni passaggio focalizzato sulla modifica del contatto.

1. **Applicare filtri personalizzati**

****

Il diagramma di attività descrive il flusso per applicare filtri personalizzati ai contatti. L'utente apre la schermata dei filtri e seleziona i criteri desiderati. Il diagramma è coeso con attività chiaramente definite per la selezione e l'applicazione dei filtri. La verifica dei criteri come passo separato riduce l'accoppiamento.

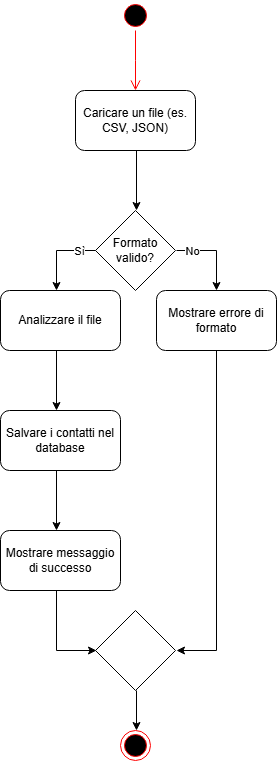
1. **Esportare i contatti**

****

Questo diagramma di attività rappresenta il flusso per esportare contatti dalla rubrica. L'utente seleziona i contatti desiderati e specifica un formato di esportazione.In caso positivo, genera il file e lo rende disponibile per il

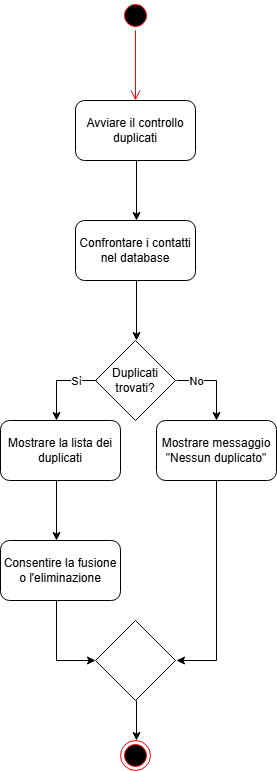
download. Se invece il formato non è valido, viene mostrato un messaggio di errore all'utente.Il flusso è progettato per garantire coesione, con ogni passo direttamente legato all'obiettivo dell'export.

1. **Importare i contatti**

****

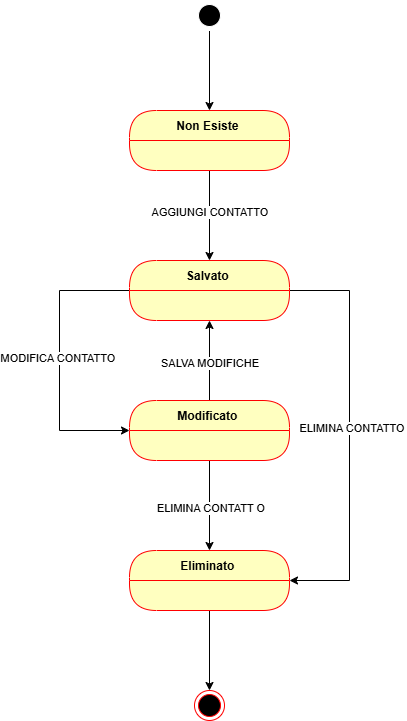
Questo diagramma di attività descrive il processo di importazione dei contatti. L'utente inizia caricando un file. Il sistema verifica la validità del formato: se il file è corretto, viene analizzato, i contatti vengono estratti e salvati nel database, e un messaggio di successo viene mostrato. In caso di errore nel formato, il sistema notifica l'utente con un messaggio di errore.

1. **Gestione duplicati**

****

Questo diagramma di attività rappresenta il processo per rilevare contatti duplicati. L'operazione inizia avviando il controllo sui dati salvati nel database. I contatti vengono confrontati tra loro per identificare eventuali duplicati. Se vengono trovati duplicati, il sistema presenta una lista, consentendo all'utente di scegliere tra opzioni come la fusione o l'eliminazione. In caso contrario, un messaggio informa che non sono stati rilevati duplicati. In questo modo garantiamo grande coesione e basso accoppiamento.

* **STATE MACHINE DIAGRAMS**



Nel nostro diagramma degli stati è altamente coeso ossia ogni stato riflette un momento specifico nel ciclo di vita di un contatto, mentre è privo di accoppiamento perché ogni transizione è causata da un’azione o un evento, rendendo il sistema flessibile.

**Tabella di tracciabilità**

| Requisito | Priorità | Use case | Design | Implementazione | Testing |
| --- | --- | --- | --- | --- | --- |
| RF01 | Alta | UC1 | SD1 |  |  |
| RF02 | Alta | UC2-UC3 | SD2-SD3 |  |  |
| RF03 | Alta | UC4 | SD9 |  |  |
| RF04 | Media | UC5 | SD5-SD6 |  |  |
| RF05 | Media | UC6-UC7 | SD7 |  |  |
| RF06 | Bassa | UC10 | SD10 |  |  |
| RF07 | Alta | UC14 | SD13 |  |  |
| RF08 | Media | UC8 | SD4 |  |  |
| RF09 | Bassa | UC9 | SD14 |  |  |
| RF10 | Media | UC13 | SD12 |  |  |
| RF11 | Bassa | UC12 | SD11 |  |  |
| RF12 | Alta | UC11 | SD8 |  |  |