

UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II
SCUOLA POLITECNICA E DELLE SCIENZE DI BASE
DIPARTIMENTO DI INGEGNERIA ELETTRICA E TECNOLOGIE DELL'INFORMAZIONE



CORSO DI LAUREA IN INFORMATICA INSEGNAMENTO
DI INGEGNERIA DEL SOFTWARE I ANNO ACCADEMICO
2020/2021

Specifiche, progettazione, implementazione e
validazione del Sistema Informativo
“CineMates20”

Autori:

GRUPPO INGSW 2021_S_08

Aldo Di GIOVANNI

MATRICOLA N86/2319

ald.digiovanni@studenti.unina.it

Docenti:

Prof. Sergio Di MARTINO

Dott. Luigi L. L. STARACE

Versioning del documento

Di seguito si riportano le modifiche apportate al documento nel corso dell'intero progetto. Ogni record della tabella sottostante indica una modifica e consta dei seguenti attributi: versione del documento (l'ultimo record ne specifica la versione corrente), data della modifica, autore e descrizione della modifica.

Versione	Data	Autore	Descrizione
0.1	27/08/2021	Aldo Di Giovanni	Creazione formato base del documento.
0.2	30/08/2021	Aldo Di Giovanni	Indice, descrizione progetto, requisiti funzionali e non.
0.3	06/09/2021	Aldo Di Giovanni	Use Case Diagram, Mockup e tabelle di Cockburn
0.4	02/09/2022	Aldo Di Giovanni	Analysis Diagrams, Design Diagrams, Test Plan & Cases
0.5	04/09/2022	Aldo Di Giovanni	Gantt, xUnit test, revision

Indice

1.	Descrizione del Progetto	4
2.	Requisiti Software	5
2.1	Modello funzionale	5
2.1.1	Requisiti funzionali	5
2.1.2	Requisiti non funzionali	8
2.1.3	Use case Diagram	9
2.1.4	Mockup App Mobile	11
2.1.5	Tabelle di Cockburn	35
2.1.6	Diagramma di Gantt	41
2.2	Modelli di Dominio	42
2.2.1	Class Diagram	42
2.2.2	Sequence Diagram	46
2.2.3	Activity Diagram	52
2.2.4	State Diagrams	54
3.	Design del Sistema	57
3.1	Analisi dell'architettura:	57
3.2	Design Class Diagram	58
3.3	Design Sequence Diagrams	59
3.4	Design State Chart Diagram	62
3.5	CRC Cards	63
4.	Testing del sistema	65
4.1	Test Planning	65
4.2	Unit Test Cases	65

1. Descrizione del Progetto

CineMates20 è un sistema complesso e distribuito finalizzato ad offrire un moderno *social network* multi-piattaforma per appassionati di cinema.

Il sistema consiste in un back-end sicuro, performante e scalabile, e in un Client mobile attraverso il quale gli utenti possono fruire delle funzionalità del sistema in modo intuitivo, rapido e piacevole.

Le funzionalità offerte da *CineMates20* sono indicate di seguito:

1. Registrazione\Autenticazione degli utenti. È presente la possibilità di autenticarsi utilizzando account su altre piattaforme come Google o Facebook.
2. Effettuare ricerche di film, utilizzando anche le API offerte dal servizio esterno The Movie DataBase (TMDb).
3. Inserire una recensione di un film visto, indicando una valutazione e una descrizione testuale.
4. Inviare\Ricevere richieste di collegamento a\da altri utenti.
5. Visualizzare il profilo di un amico, che riporta le sue attività più recenti (e.g.: recensioni inserite, film inseriti in una lista, liste create, commenti lasciati).

2. Requisiti Software

2.1 Modello funzionale

2.1.1 Requisiti funzionali

REQ1	
Nome	Visualizzazione dei film presenti
Descrizione	<p>L'applicativo deve dare possibilità a qualsiasi utente di visualizzare i film presenti sulla piattaforma.</p> <p>Per ciascun film, sarà necessaria la presenza di informazioni come:</p> <ul style="list-style-type: none">• Nome del film• Valutazione• Recensioni• Presenza del film in una delle liste dell'utente (in caso di utente autenticato)• Anno d'uscita• Copertina del film

REQ2	
Nome	Ricerca di un film
Descrizione	<p>L'applicativo deve dare la possibilità, a qualsiasi utente, di ricercare un film o un gruppo di film in base a criteri di ricerca.</p> <p>Il criterio di ricerca è quindi definito in base al nome del film.</p> <p>Nel caso in cui sia digitato un criterio di ricerca che non ha alcun riscontro con i film disponibili nell'applicativo allora non verrà visualizzato nessun film nella pagina di ricerca</p>

REQ3	
Nome	Ricerca degli utenti tra quelli registrati alla piattaforma
Descrizione	<p>L'applicativo deve dare la possibilità, a qualsiasi utente autenticato, di ricercare uno o più utenti che hanno in comune un criterio di ricerca.</p> <p>Il criterio di ricerca è quindi definito in base all' username dell'utente.</p> <p>Nel caso in cui sia digitato un username non presente nel database non verrà visualizzato nessun utente nella pagina di ricerca</p>

REQ4	
Nome	Collegamento tra utenti
Descrizione	<p>L'applicativo deve dare la possibilità, a qualsiasi utente autenticato, di collegarsi con altri utenti.</p> <p>Il collegamento tra utenti deve avvenire tramite delle richieste che vengono inviate/ricevute.</p> <p>La ricezione di una nuova richiesta dovrà essere notificata nell'apposita sezione di notifica e potrà essere accettata o respinta</p>

REQ5	
Nome	Accesso dall'applicativo alla piattaforma
Descrizione	<p>L'applicativo deve offrire agli utenti la possibilità di accedere alla piattaforma in modo da consentire a quest'ultimi di poter recensire un film, aggiungerli ad una delle due liste o inviare/ricevere richieste di collegamento con altri utenti.</p>

REQ6	
Nome	Registrazione dall'applicativo alla piattaforma
Descrizione	<p>L'applicativo deve offrire agli utenti la possibilità di registrarsi alla piattaforma in modo da consentire a quest'ultimi di poter recensire un film, aggiungerli ad una delle due liste o inviare/ricevere richieste di collegamento con altri utenti. Per realizzare ciò, è necessario che l'applicativo dia la possibilità, ad ogni utente ospite, di effettuare una registrazione collezionando eventuali dati quali:</p> <ul style="list-style-type: none"> • Nome e cognome • Username • Email e password di accesso

REQ7	
Nome	Aggiunta di una nuova recensione
Descrizione	<p>L'applicativo deve consentire a ciascun utente autenticato di aggiungere una nuova recensione relativa ad un film. In particolare, in fase di stesura della recensione, verranno richiesti:</p> <ul style="list-style-type: none"> • Valutazione da attribuire al film • Titolo della recensione • Corpo della recensione <p>Una volta inviata la recensione, sarà presente fin da subito sulla pagina del film relativo ad essa.</p>

REQ8	
Nome	Visualizzazione profilo di un amico
Descrizione	<p>L'applicativo deve consentire a ciascun utente autenticato di visualizzare il profilo di un altro utente, aggiunto come amico, e di conseguenza le sue attività più recenti (e.g.: recensioni inserite, film inseriti in una lista, liste create, commenti lasciati).</p>

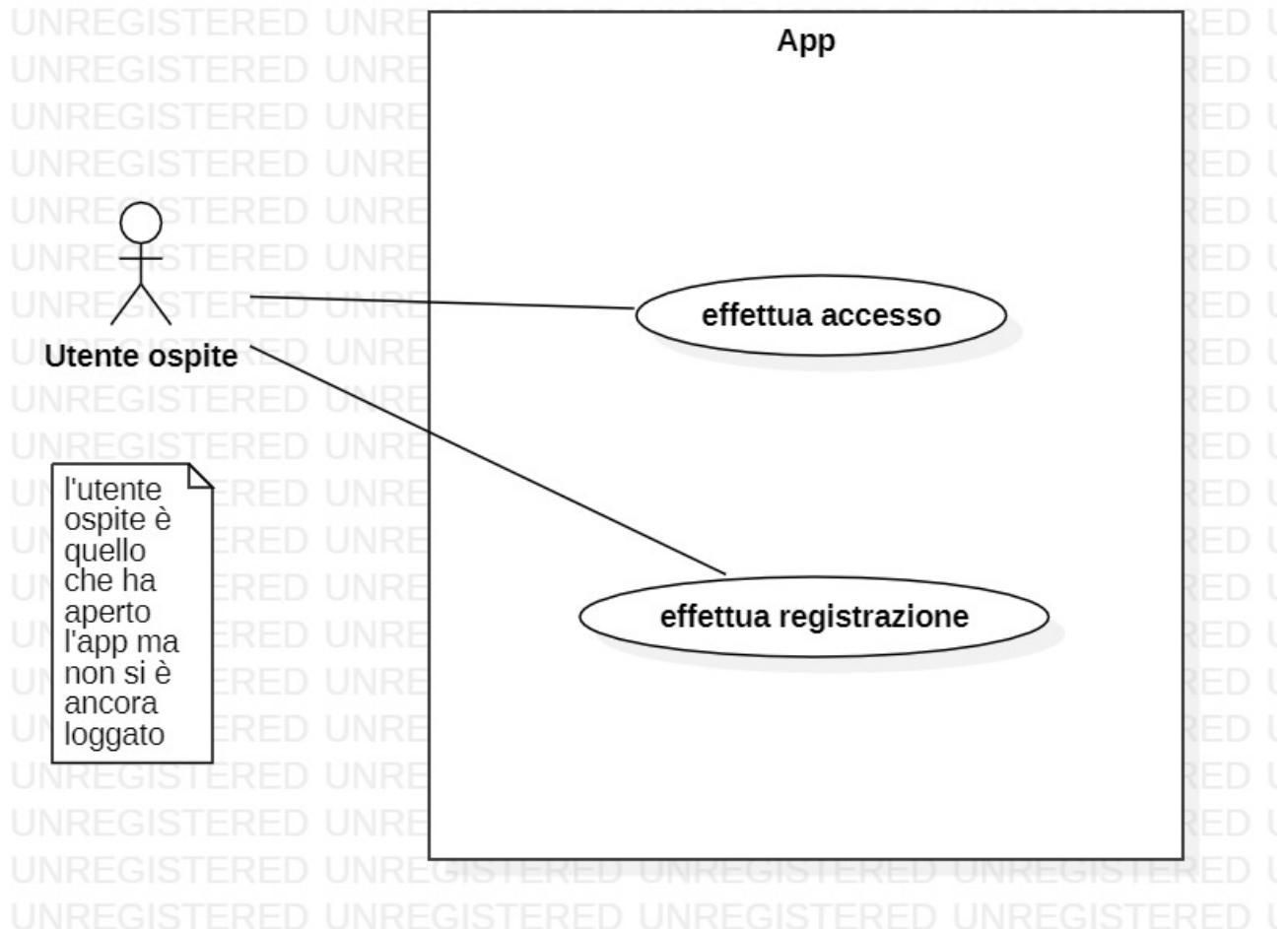
2.1.2 Requisiti non funzionali

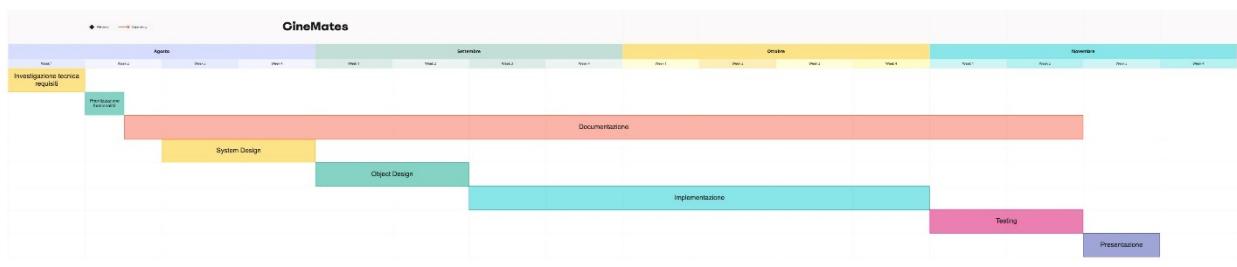
REQNF01	
Nome	Consentire una navigazione agevole ed user-friendly per l'intero sistema
Descrizione	L'utente o l'amministratore deve poter svolgere la propria attività in maniera semplice ed efficiente. L'utente deve poter ricercare i film e gli utenti a cui è interessato con semplicità e deve poter ritornare alla home page o alla pagina precedente in qualsiasi momento.

REQNF02	
Nome	Back-end scalabile
Descrizione	il back-end sia messo in opera utilizzando tecnologie allo stato dell'arte quali ad esempio servizi di public Cloud Computing come Azure o AWS, al fine di massimizzare la scalabilità del sistema in vista di un possibile repentino aumento del numero degli utenti nelle fasi iniziali di rilascio al pubblico.

2.1.3 Use case Diagram

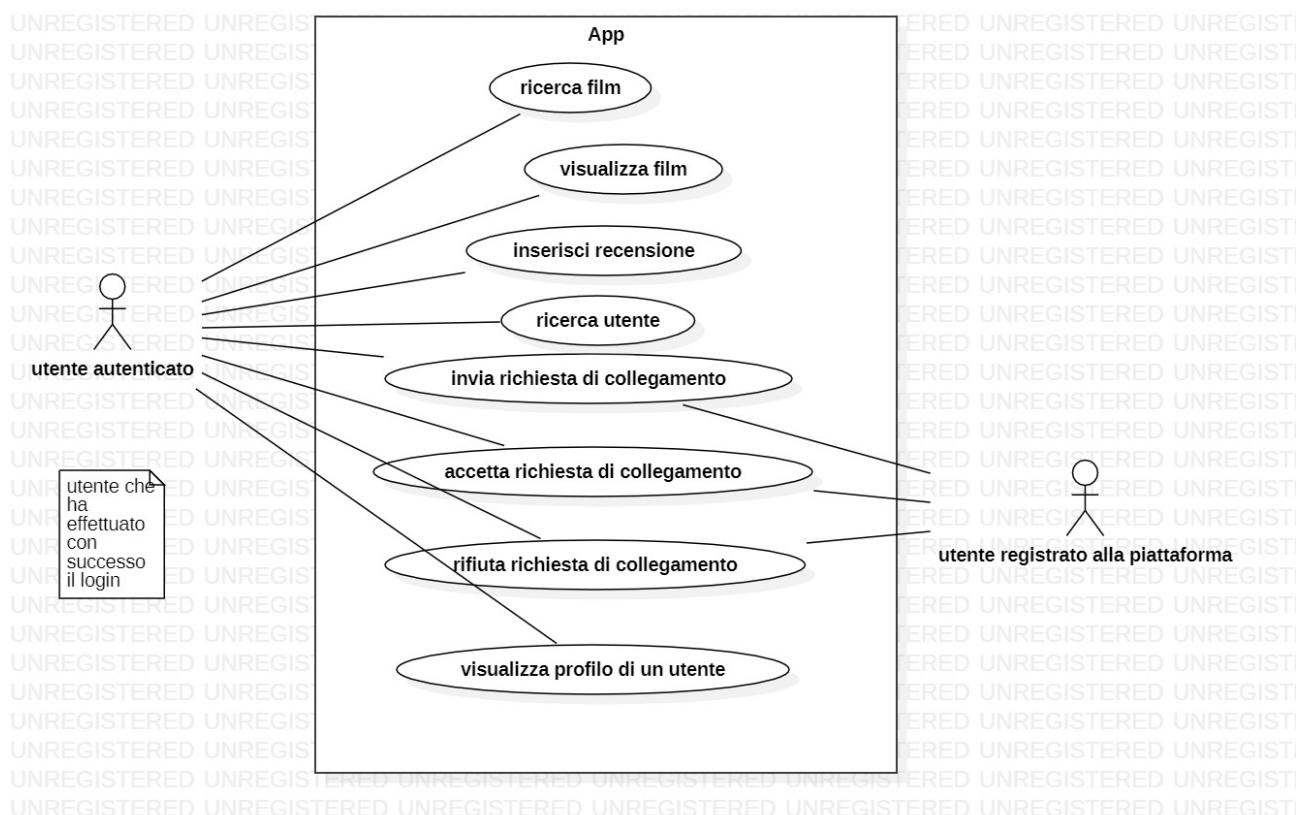
Utente ospite (che non ha ancora effettuato l'accesso):





miro

Utente che ha già effettuato il login:

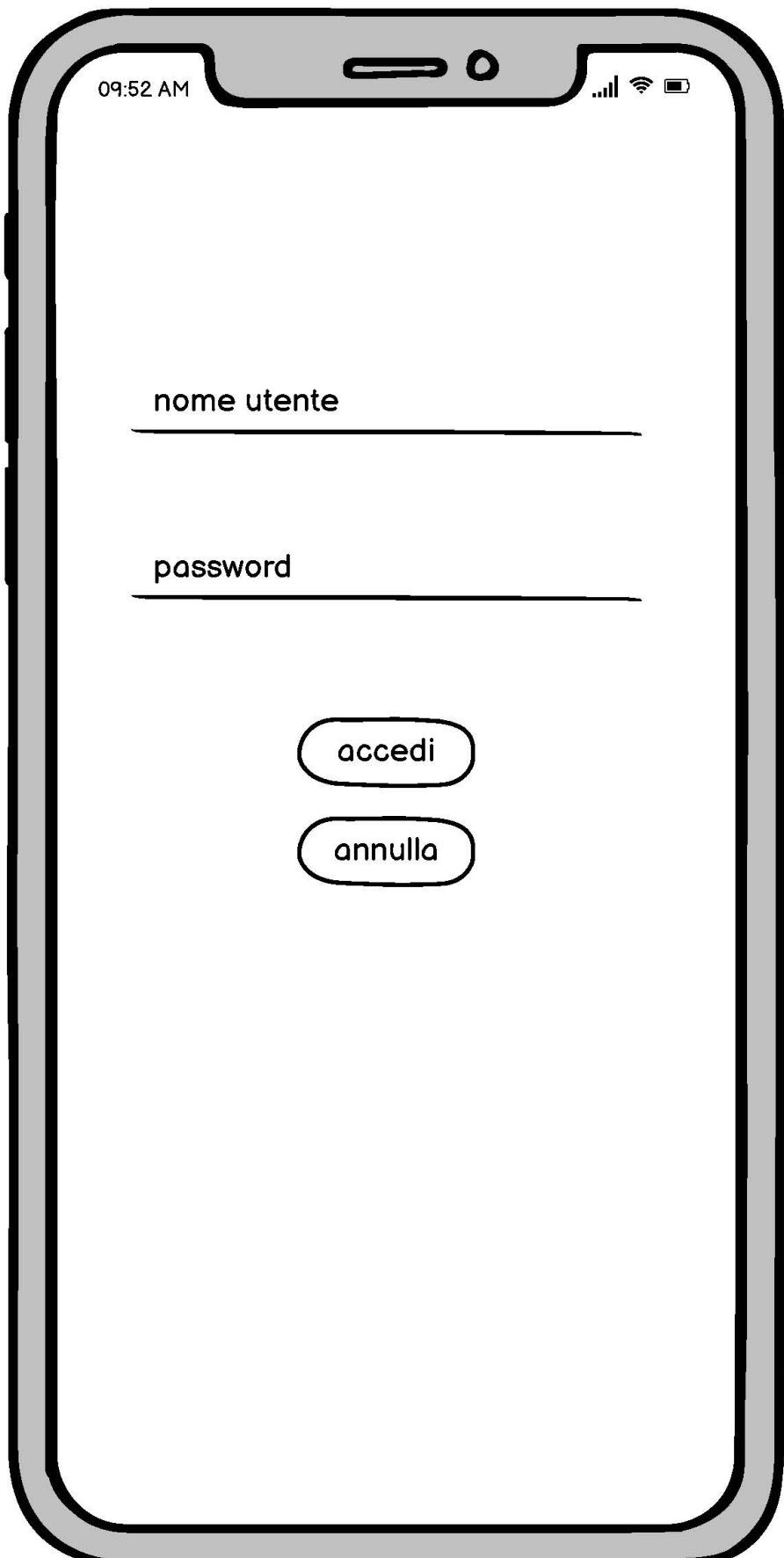


2.1.4 Mockup App Mobile

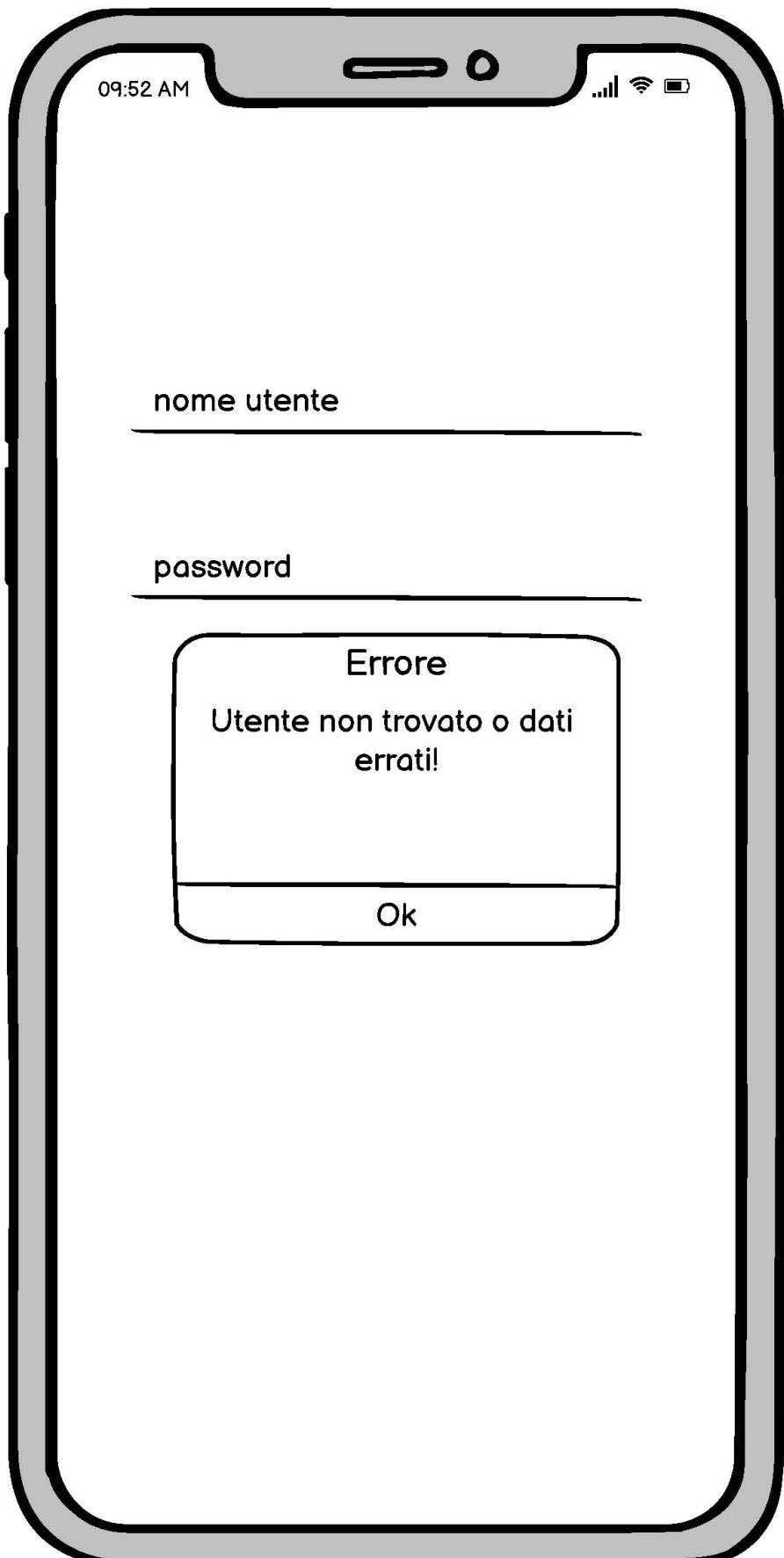
I wireframe che seguono sono stati creati usando il software Balsamiq.
Indicano le funzioni disponibili nell'app e forniscono un'idea
dell'implementazione delle stesse.



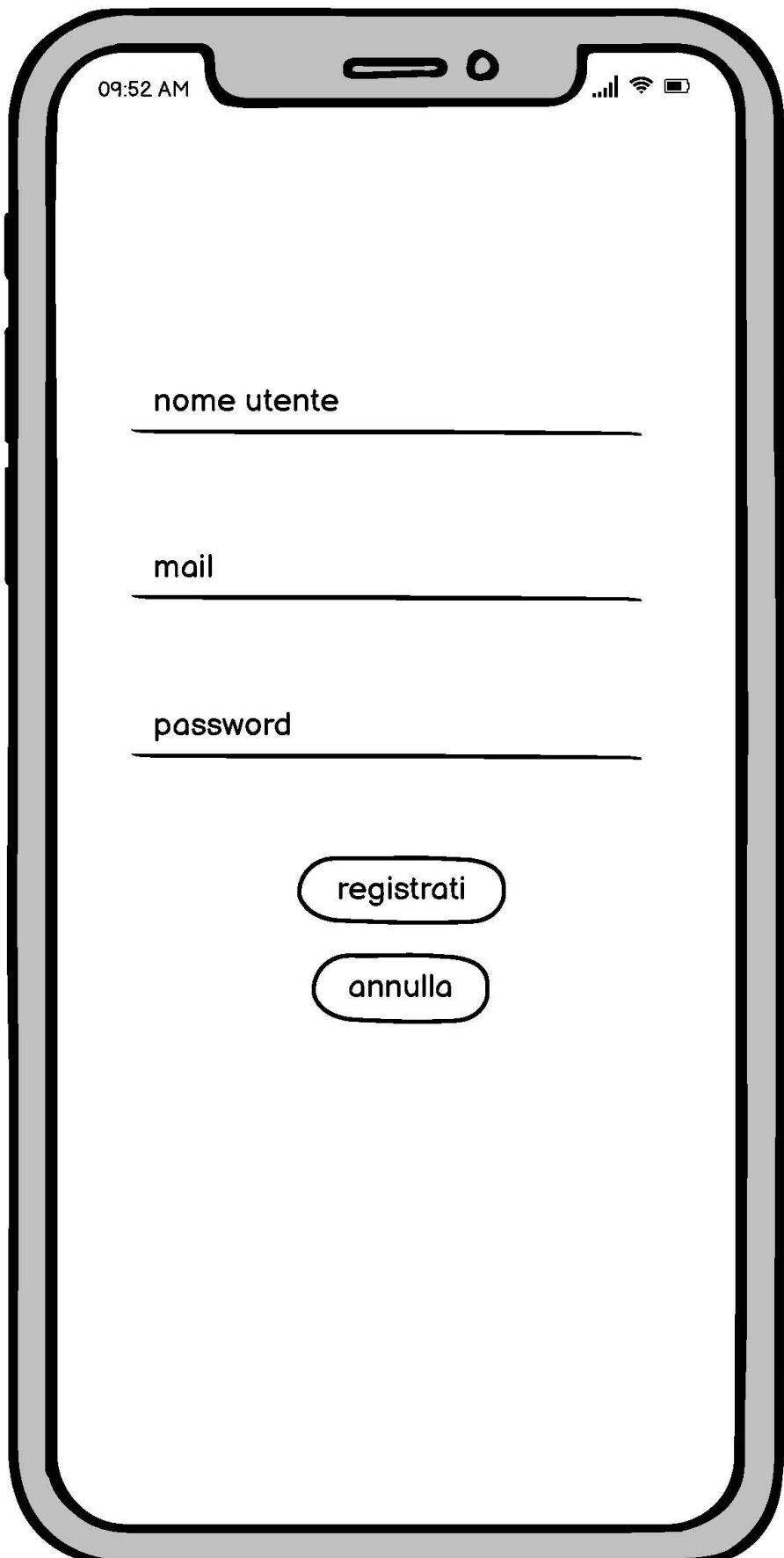
M1



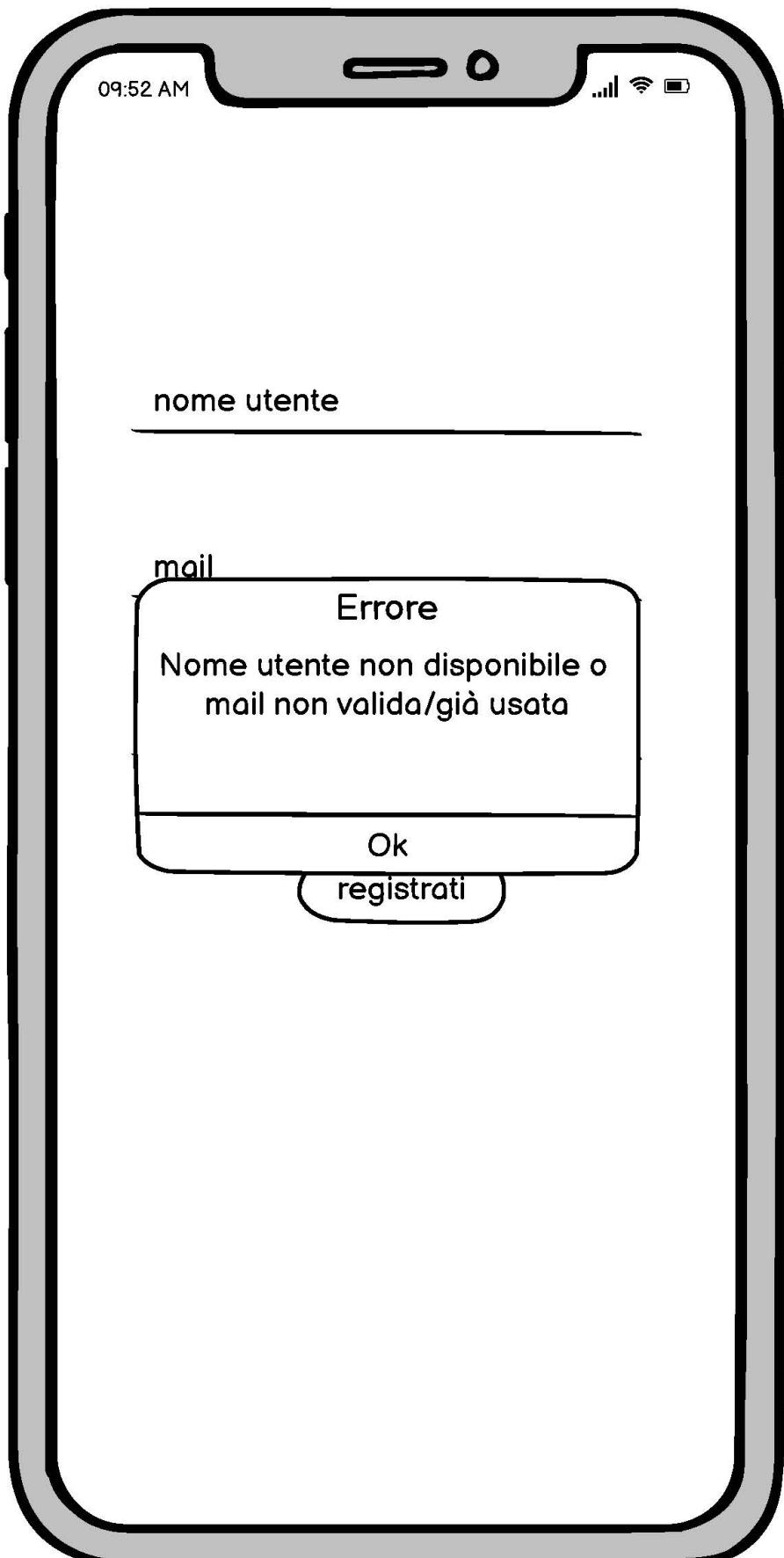
M2



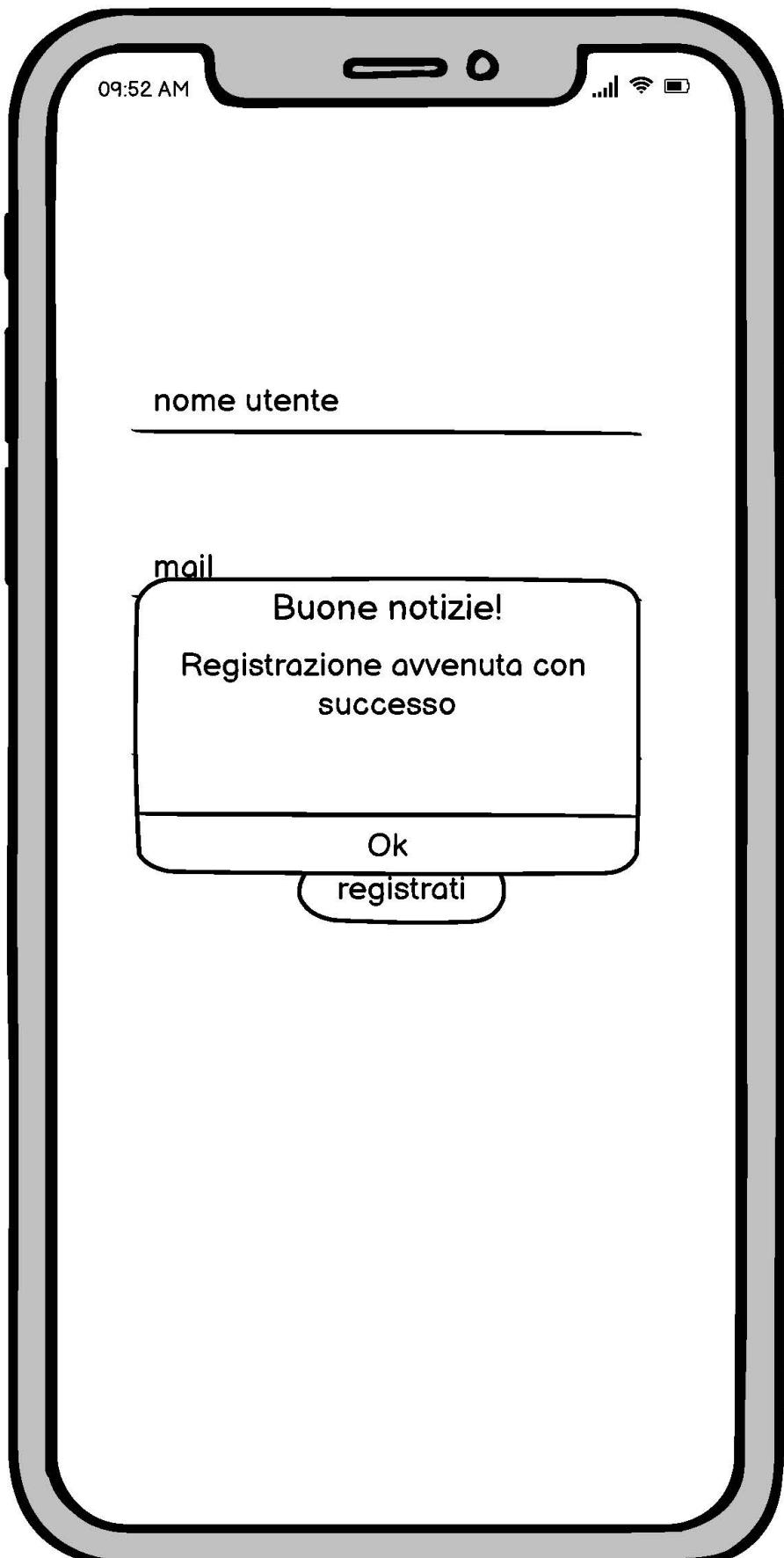
M3



M4



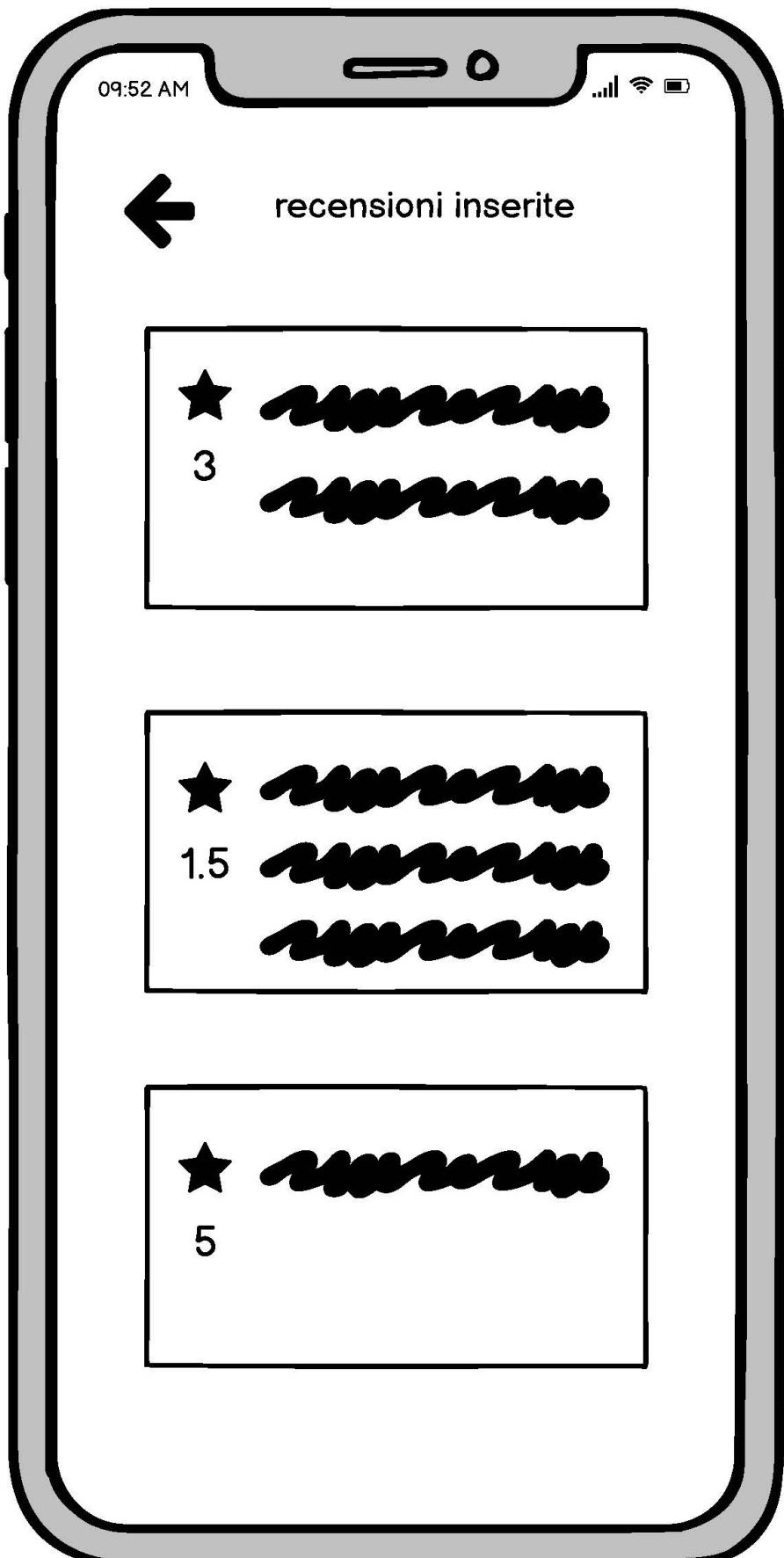
M5



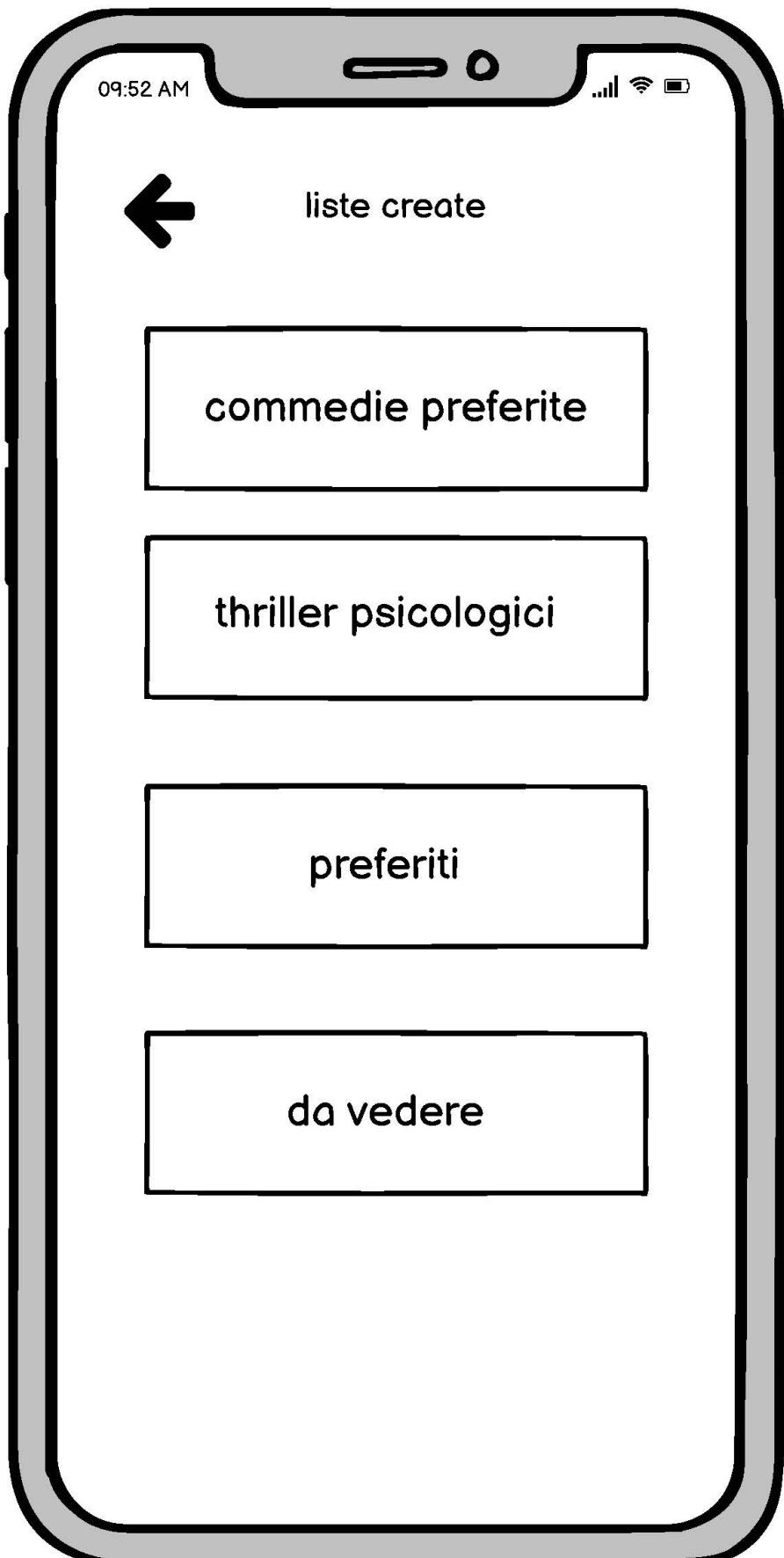
M6



M7



M8



M9



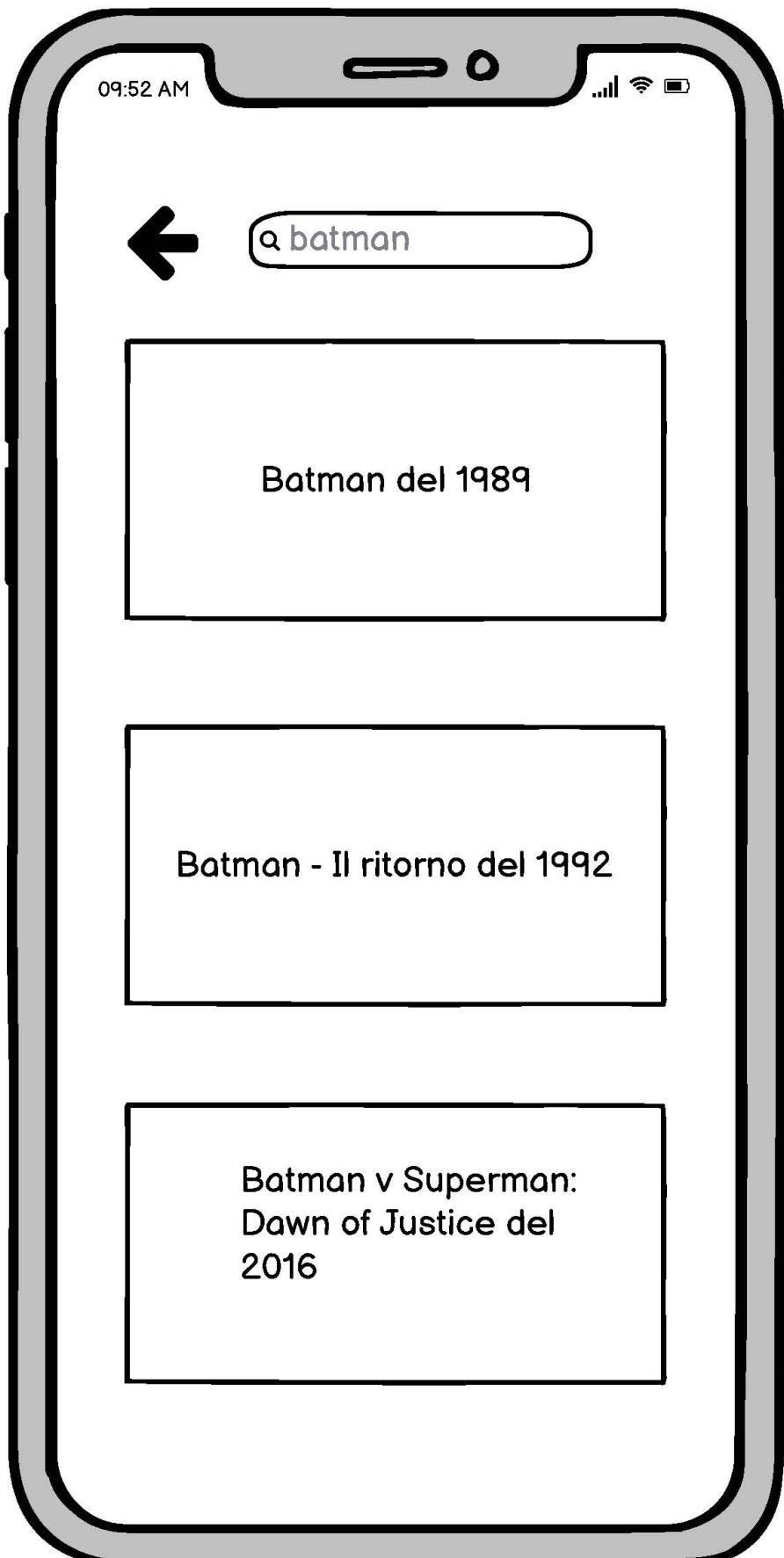
M10



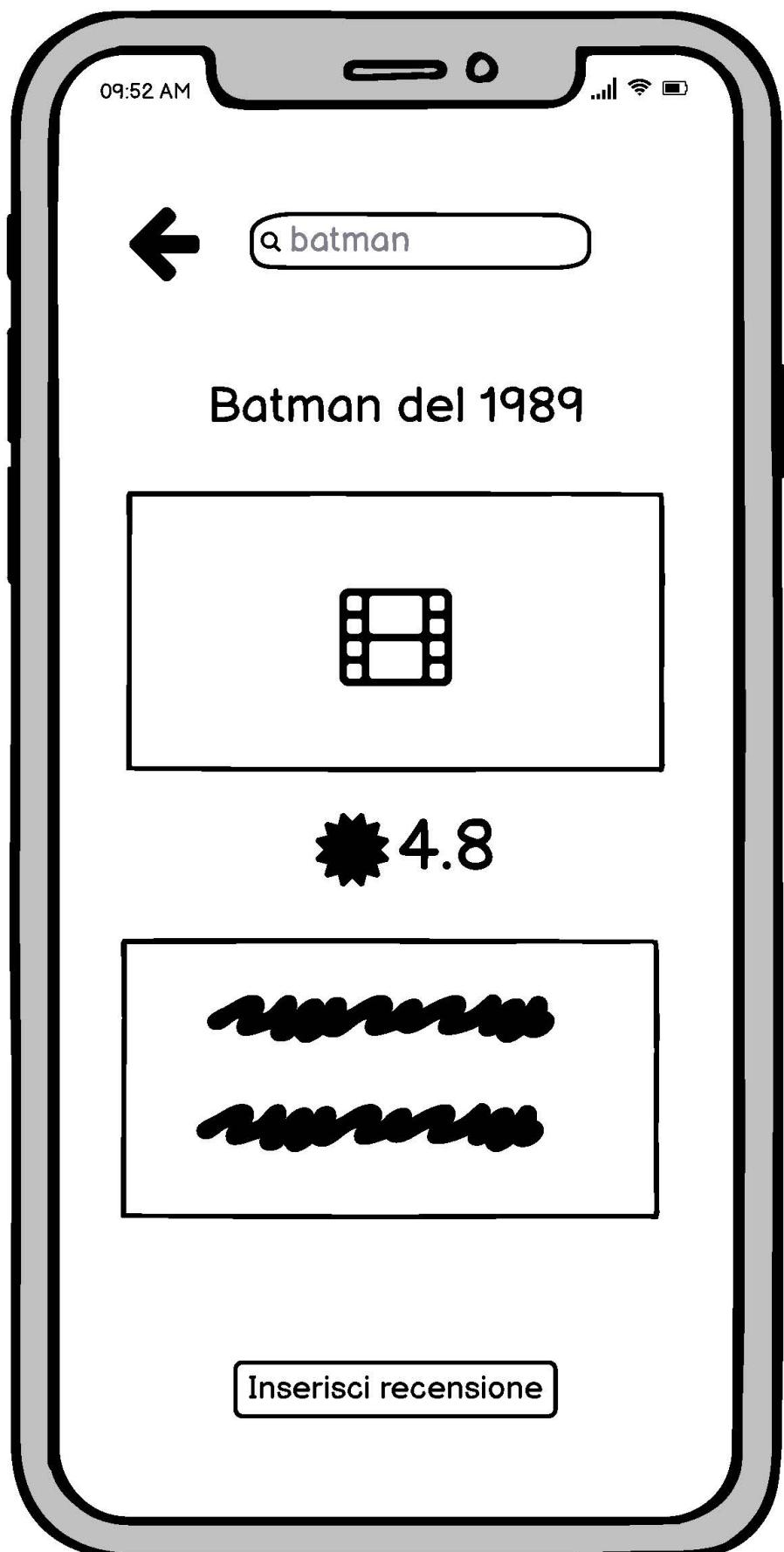
M11



M12



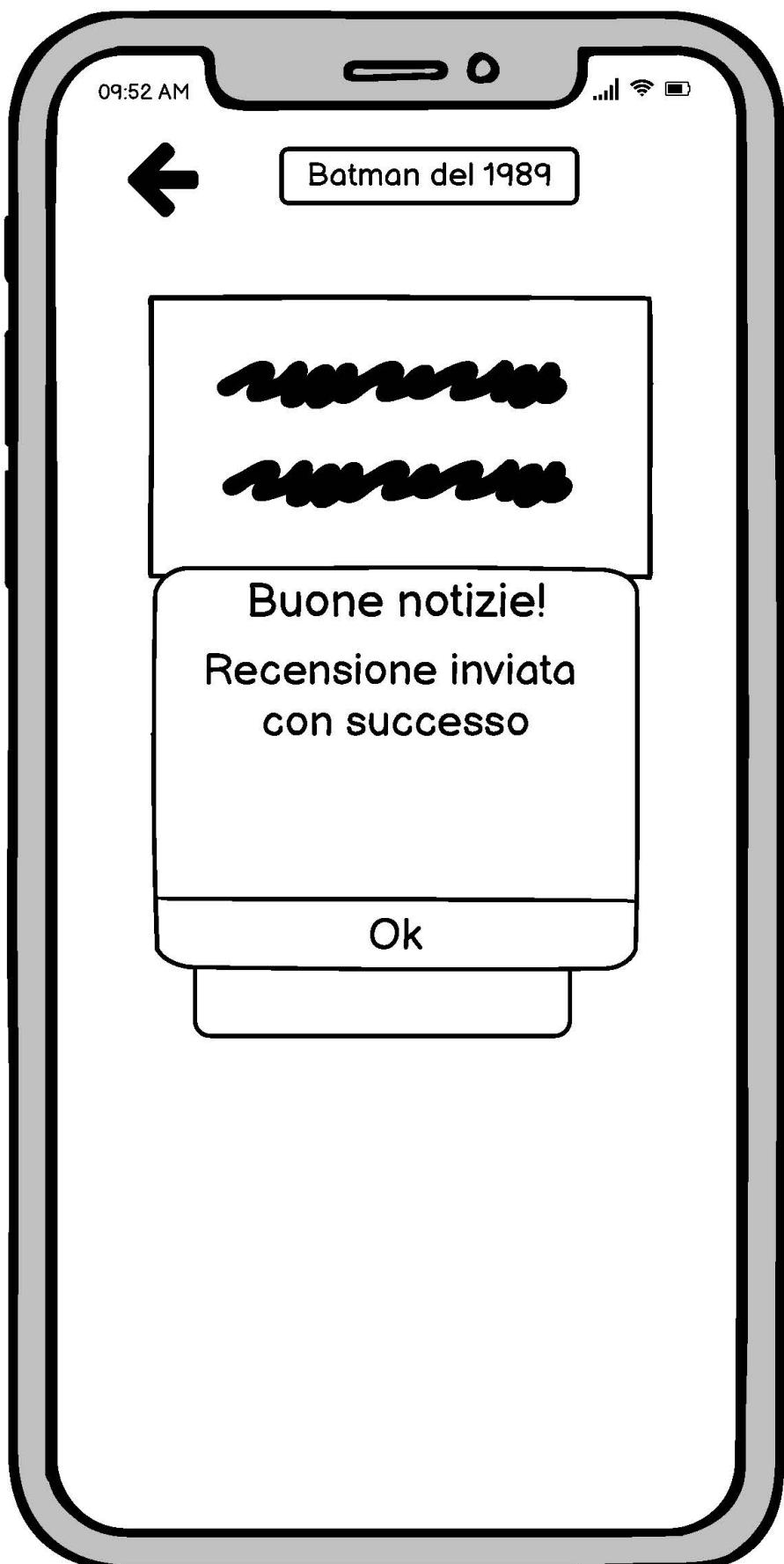
M13



M14



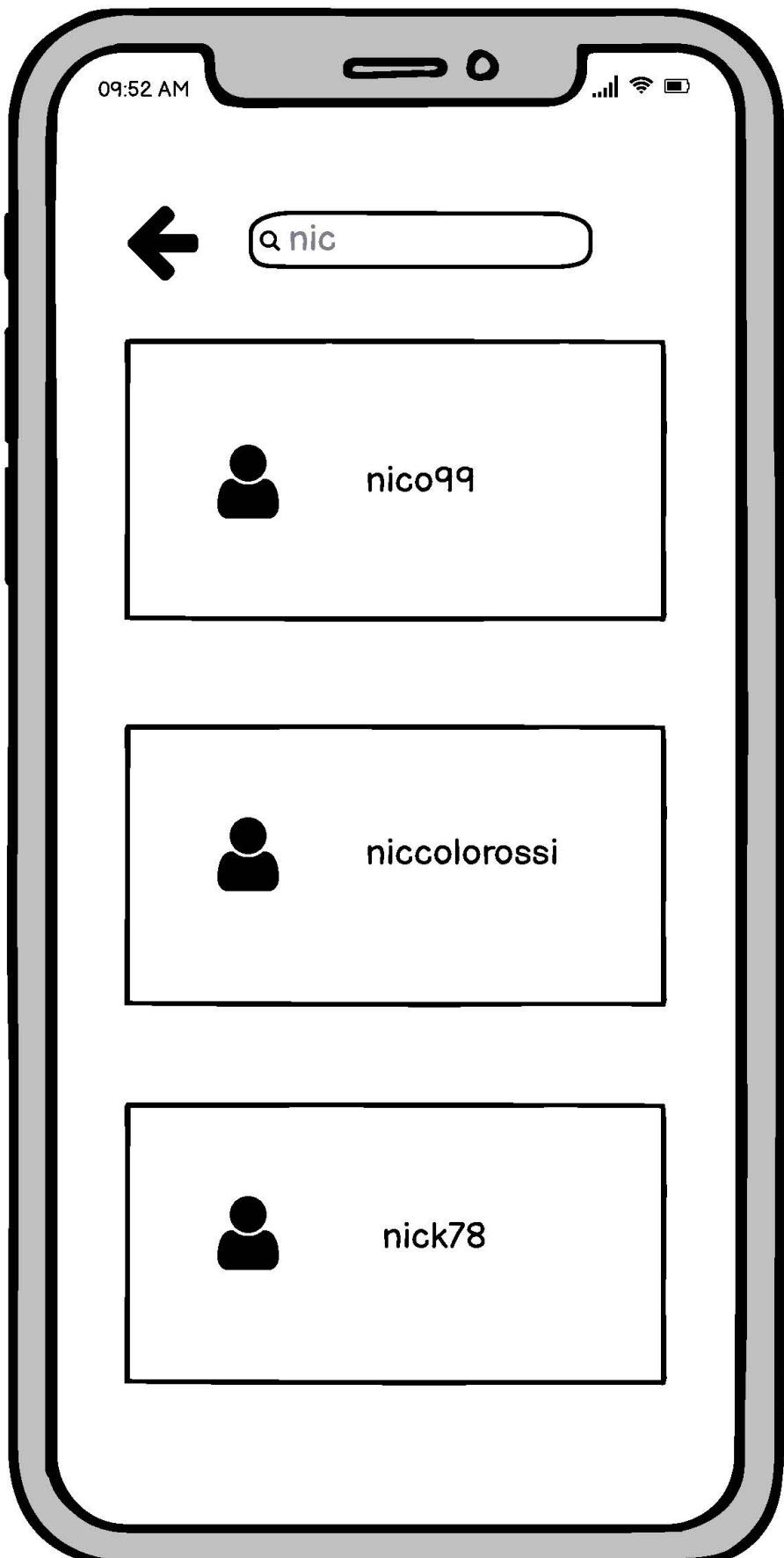
M15



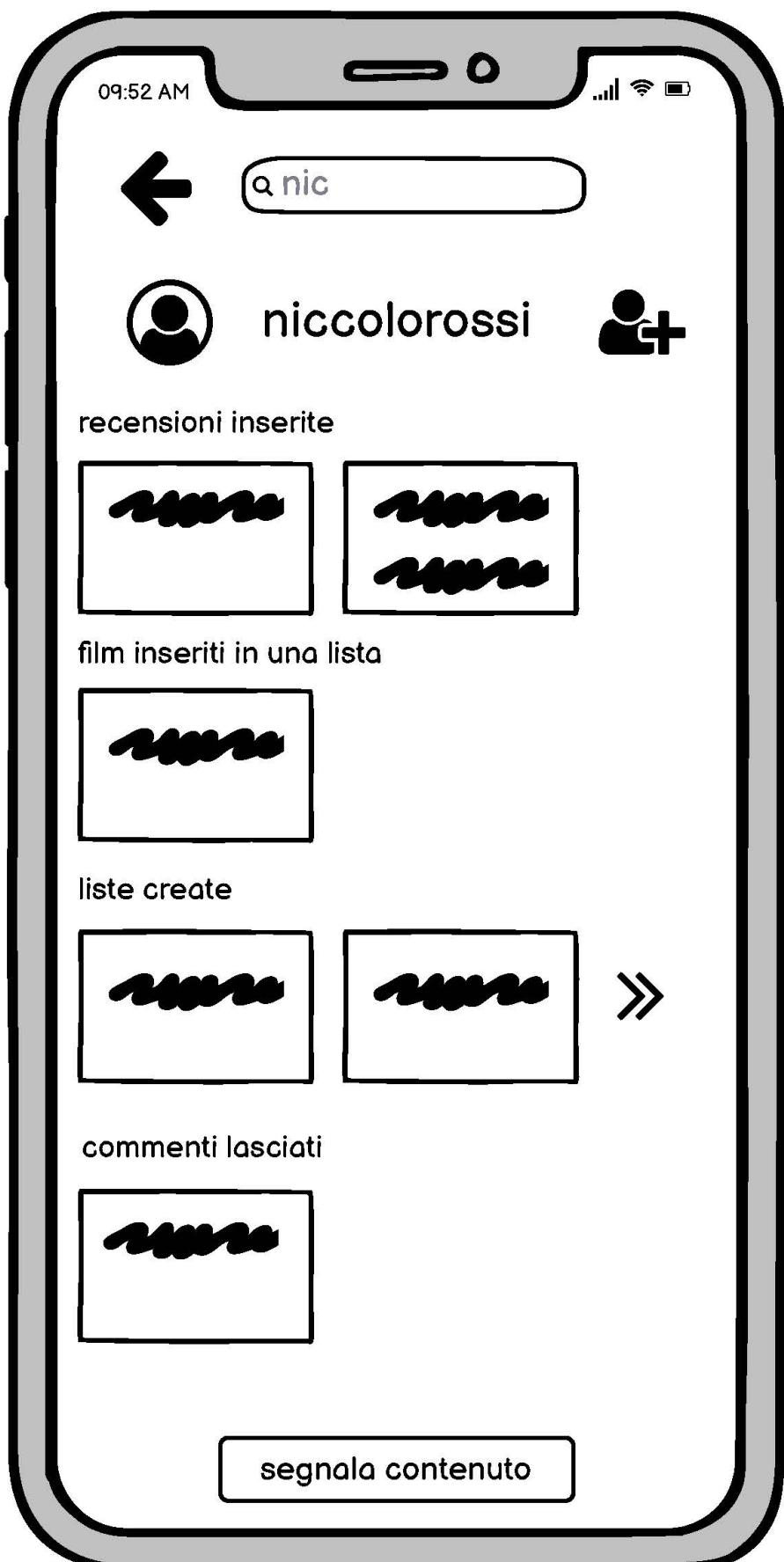
M16



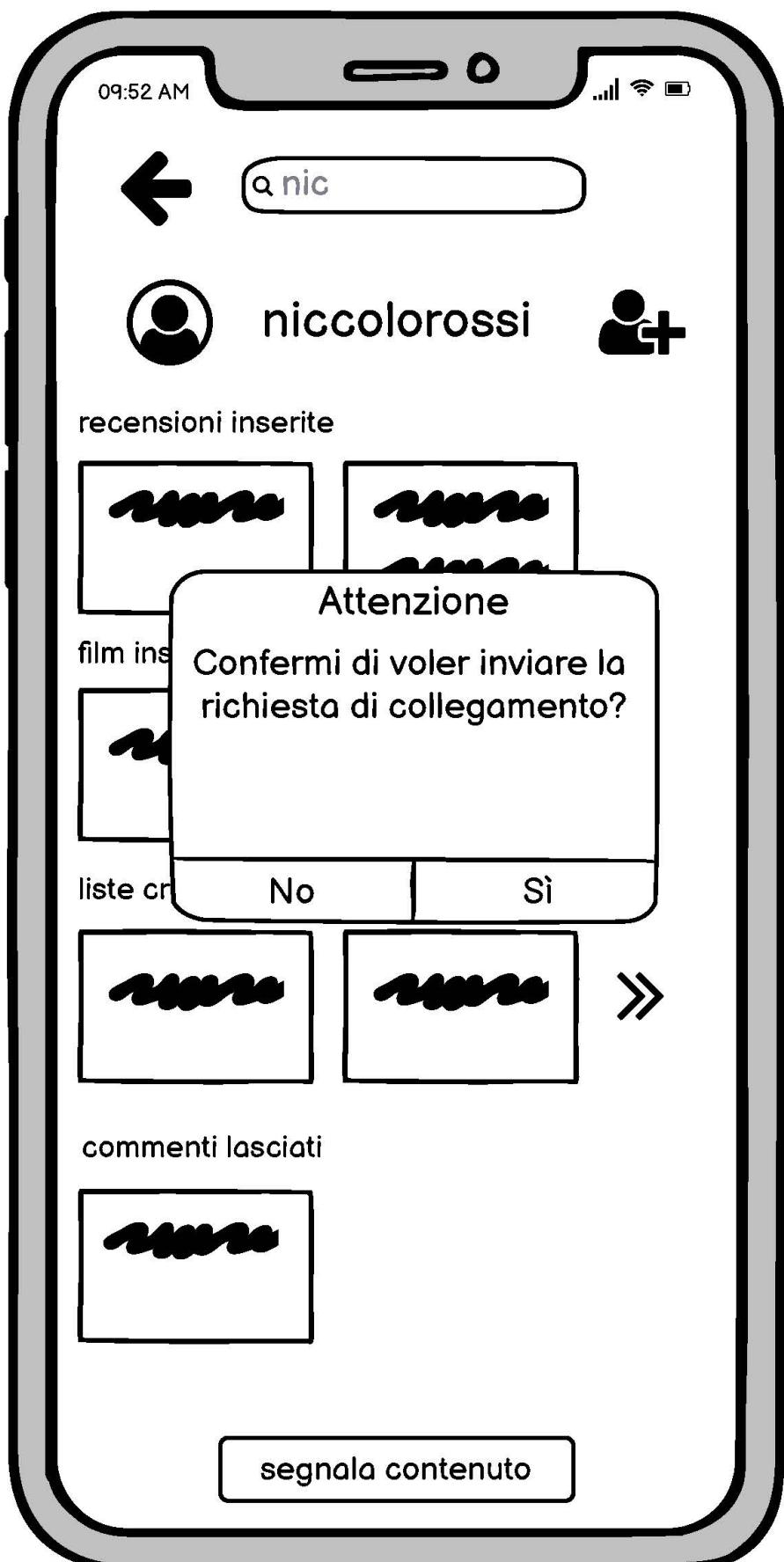
M17



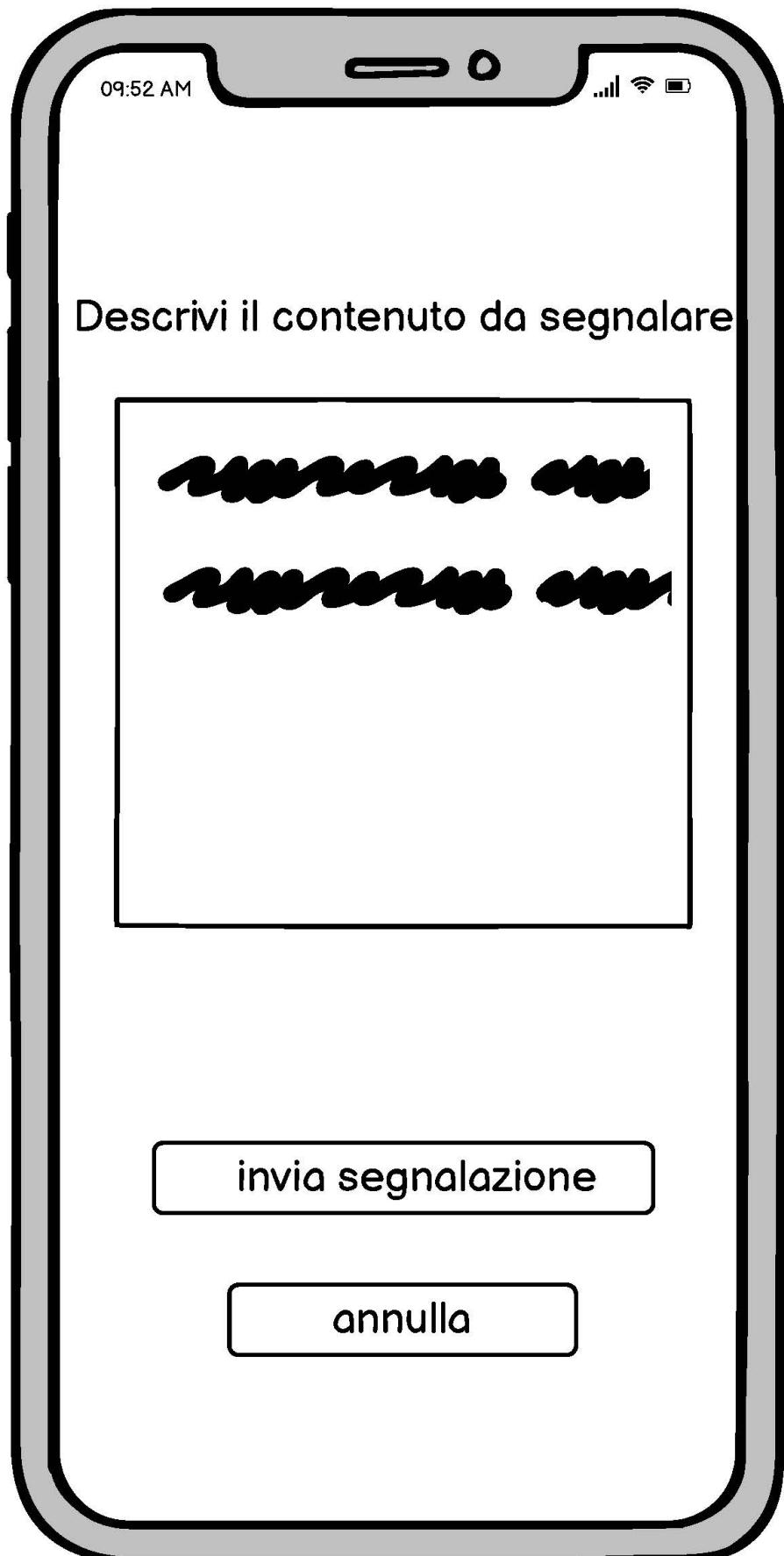
M18



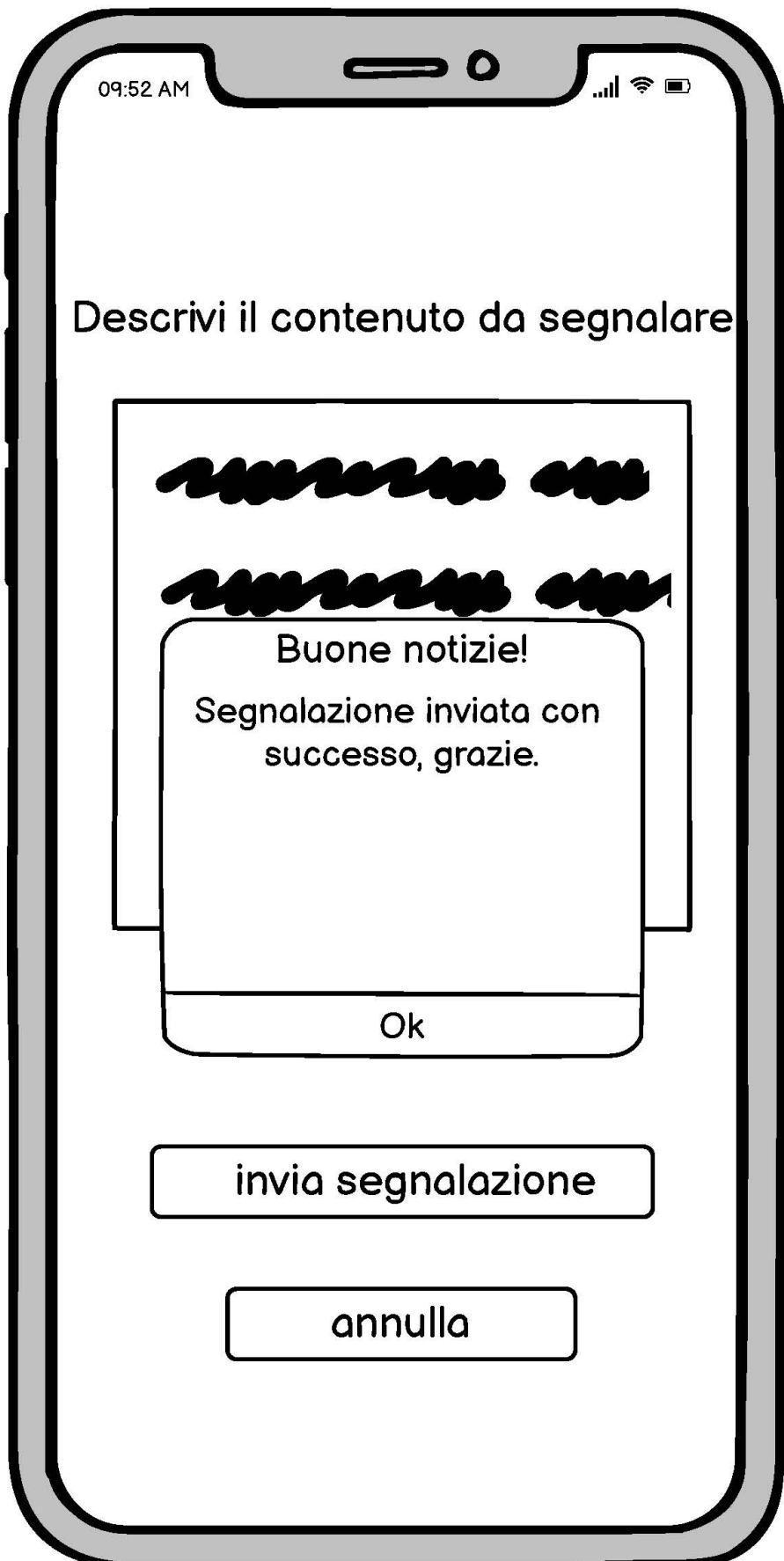
M19



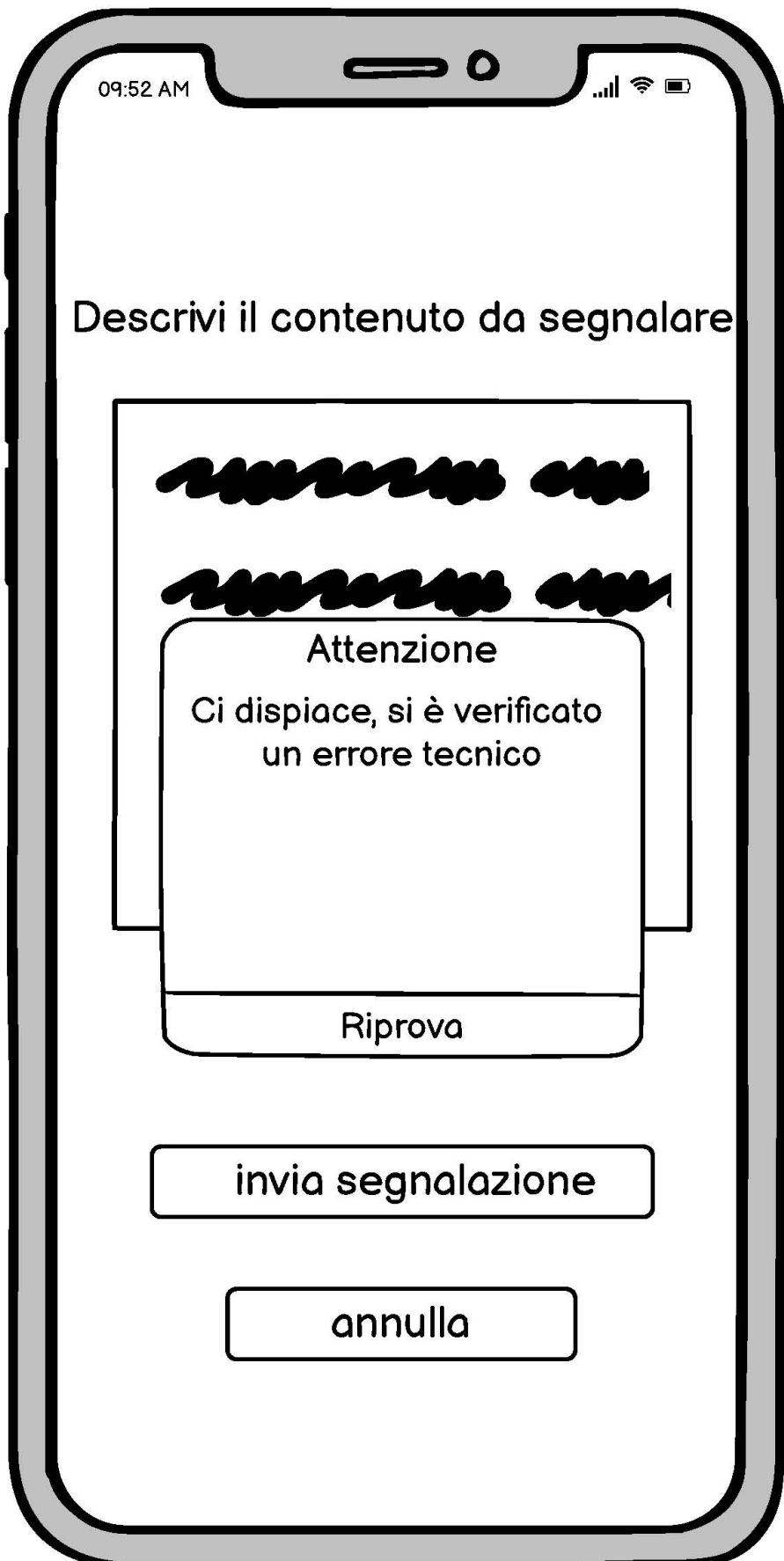
M20



M21



M22



M23

2.1.5 Tabelle di Cockburn

UC01	Effettua accesso		
Goal in context	Un utente non autenticato effettua l'accesso all'applicazione mobile		
Preconditions	L'utente deve essere non autenticato all'applicazione		
Success End Condition	L'utente effettua l'accesso alla piattaforma tramite applicazione mobile		
Failed End Condition	L'utente non esegue l'accesso		
Primary Actor	Utente non autenticato		
Trigger	L'utente preme su uno dei button di login in M1		
Description	Step n°	Utente non autenticato	System
	1	Azione Trigger	
	2		Mostra M2
	3	Inserisce le credenziali	
	4	Preme "accedi"	
	5		Mostra M7
Extension N°1 username o password non sono corretti	Step n°	Utente non autenticato	System
	4.1		Mostra M3
	FAILED END CONDITION		

UC02	Effettua registrazione		
Goal in context	Un utente non autenticato si registra alla piattaforma		
Preconditions	L'utente deve essere non autenticato all'applicazione		
Success End Condition	L'utente si registra alla piattaforma		
Failed End Condition	L'utente non si registra alla piattaforma		
Primary Actor	Utente non autenticato		
Trigger	Un utente preme "Registrati con mail" in M1		
Description	Step n°	Utente non autenticato	System
	1	Azione Trigger	
	2		Mostra M4
	3	Compila i campi	

	4	Preme su "Registrati" in M4	
	5		Mostra M6
Extension N° 1 Uno o più campi non validi	Step n°	Utente non autenticato	System
	4.1		Mostra M5
	Failed End Condition		

UC03	Ricerca film			
Goal in context	Un utente ricerca un film in base al criterio di ricerca			
Preconditions	L'utente è già loggato			
Success End Condition	Il sistema visualizza l'insieme dei film che corrispondono alla parola chiave digitata dall'utente			
Failed End Condition	Il sistema non trova corrispondenze			
Primary Actor	Utente autenticato			
Trigger	L'utente preme sul bottone di ricerca in M7			
Description	Step n°	Utente autenticato	System	TMDB API
	1	Azione trigger		
	2			Fetch dei film
	3		Mostra M13	
Extension N°1 La ricerca non ha prodotto risultati	1.1		Mostra M12	
FAILED END CONDITION				
Subvariation n° 1 L'utente clicca su "Recensioni inserite" in M7	Step n°	Utente autenticato	System	
	1.1	Azione Trigger		
	1.2		Mostra M8	
Subvariation n° 2 L'utente clicca su "liste create" in M7	Step n°	Utente autenticato	System	
	1.1	Azione Trigger		
	1.2		Mostra M9	
Subvariation n° 3 L'utente clicca su "effettua logout" in M7	Step n°	Utente autenticato	System	
	1.1	Azione Trigger		
	1.2		Mostra M1	

UC04	Risponde alla richiesta di collegamento		
Goal in context	Un utente risponde alle eventuali richieste di collegamento		
Preconditions	L'utente è già loggato		
Success End Condition	L'utente risponde alle richieste di collegamento		
Failed End Condition			
Primary Actor	Utente autenticato		
Trigger	L'utente preme il pulsante "notifiche" in M7		
Description	Step n°	Utente autenticato	System
Subvariation n°1	1	L'utente accetta una richiesta di collegamento	
	Step n°	Utente autenticato	System
	1.1	L'utente rifiuta una richiesta di collegamento	

UC05	Visualizza Film		
Goal in context	Un utente generico visualizza il film desiderato		
Preconditions	L'utente è già loggato		
Success End Condition	L'utente visualizza il film		
Failed End Condition			
Primary Actor	Utente autenticato		
Trigger	L'utente clicca sul nome del film in M13		
Description	Step n°	Utente autenticato	System
	1	Azione trigger	
	2		Mostra M14

UC06	Recensisci film		
Goal in context	Un utente autenticato scrive una recensione e la invia		
Preconditions	L'utente è già loggato		

Success End Condition	La recensione viene inviata		
Failed End Condition	La recensione non viene inviata		
Primary Actor	Utente autenticato		
Trigger	Un utente clicca su “Inserisci recensione” in M14		
Description	Step n°	Utente autenticato	System
	1	Azione Trigger	
	2		Mostra M15
	3	Compila i campi	
	4	Clicca su “invia”	
Extension n°1 Uno o più campi non sono validi	Step n°	Utente autenticato	System
	4.1		Mostra M17
	Failed End Condition		

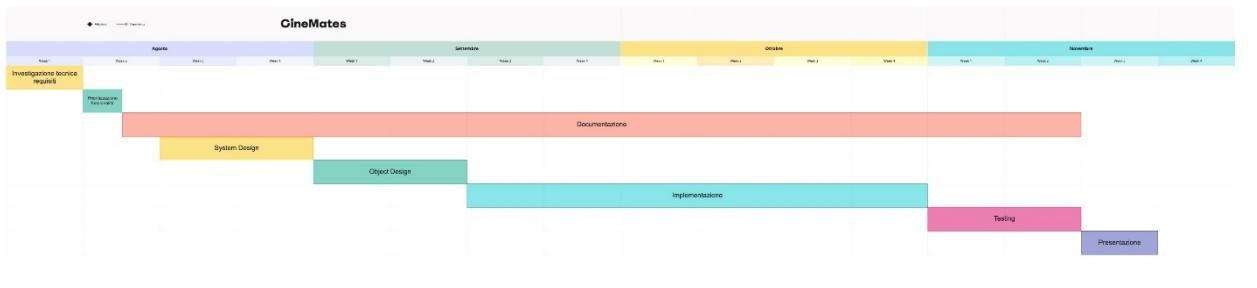
UC07	Ricerca utenti		
Goal in context	Un utente autenticato ricerca un utente o un gruppo di utenti in base al criterio di ricerca digitato		
Preconditions	L’utente è già loggato.		
Success End Condition	Il sistema visualizza l’insieme degli utenti che corrispondono alla parola chiave digitata dall’utente autenticato		
Failed End Condition	Il sistema non trova corrispondenze		
Primary Actor	Utente autenticato		
Trigger	L’utente inserisce la chiave di ricerca in M7		
Description	Step n°	Utente autenticato	System
	1	Azione trigger	
	2		Mostra M18
Extension n°1 La ricerca non ha prodotto risultati	Step n°	Utente autenticato	System
	2.1		Mostra M11
Failed End Condition			

UC08	Visualizza Profilo Utente		
Goal in context	Un utente generico visualizza il film desiderato		
Preconditions	L'utente è già loggato		
Success End Condition	L'utente visualizza il profilo		
Failed End Condition			
Primary Actor	Utente autenticato		
Trigger	L'utente clicca sul nome del profilo in M18		
Description	Step n°	Utente autenticato	System
	1	Azione trigger	
	2		Mostra M19

UC09	Invia richiesta di collegamento		
Goal in context	L'utente invia una richiesta di collegamento		
Preconditions	L'utente è già loggato		
Success End Condition	L'utente invia una richiesta di collegamento		
Failed End Condition	L'utente non invia una richiesta di collegamento		
Primary Actor	Utente autenticato		
Trigger	L'utente preme il pulsante con l'icona della persona col + in M19		
Description	Step n°	Utente autenticato	System
	1	Azione trigger	
	2		Mostra M20
Subvariation n° 1 L'utente conferma di non voler inviare una richiesta di collegamento in M20	Step n°	Utente autenticato	System
	3.1	L'utente conferma di non voler inviare una richiesta di collegamento	

UC10	Invia segnalazione		
Goal in context	Un utente autenticato scrive una segnalazione e la invia		
Preconditions	L'utente è già loggato		
Success End Condition	La segnalazione viene inviata		
Failed End Condition	La segnalazione non viene inviata		
Primary Actor	Utente autenticato		
Trigger	Un utente clicca su “segnala contenuto” in M19		
Description	Step n°	Utente autenticato	System
	1	Azione Trigger	
	2		Mostra M21
	3	Compila i campi	
	4	Clicca su “invia segnalazione”	
	5		Mostra M22
Extension n°1 La segnalazione non viene inviata per un errore inaspettato	Step n°	Utente autenticato	System
	4.1		Mostra M23
Failed End Condition			
Subvariation n° 1 L'utente clicca su “annulla” in M21	Step n°	Utente autenticato	System
	1.1	L'utente clicca su “annulla”	
	1.2		Mostra M19

2.1.6 Diagramma di Gantt

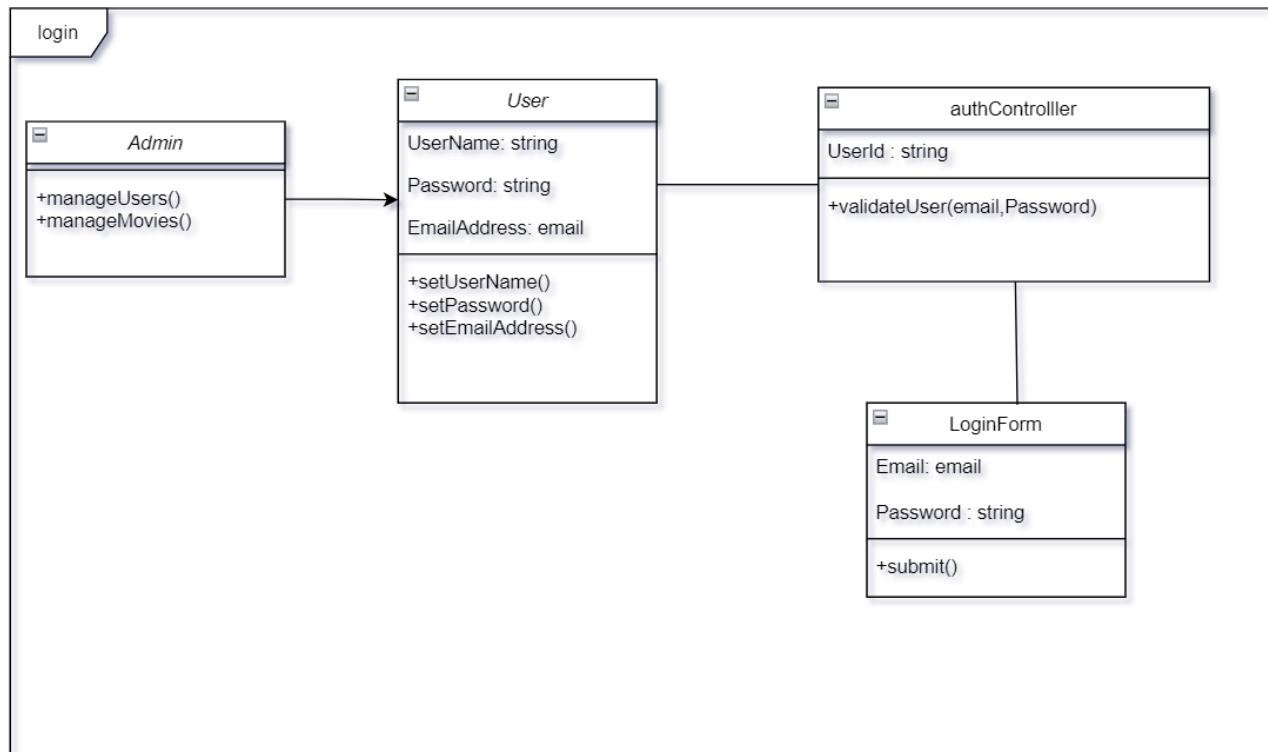
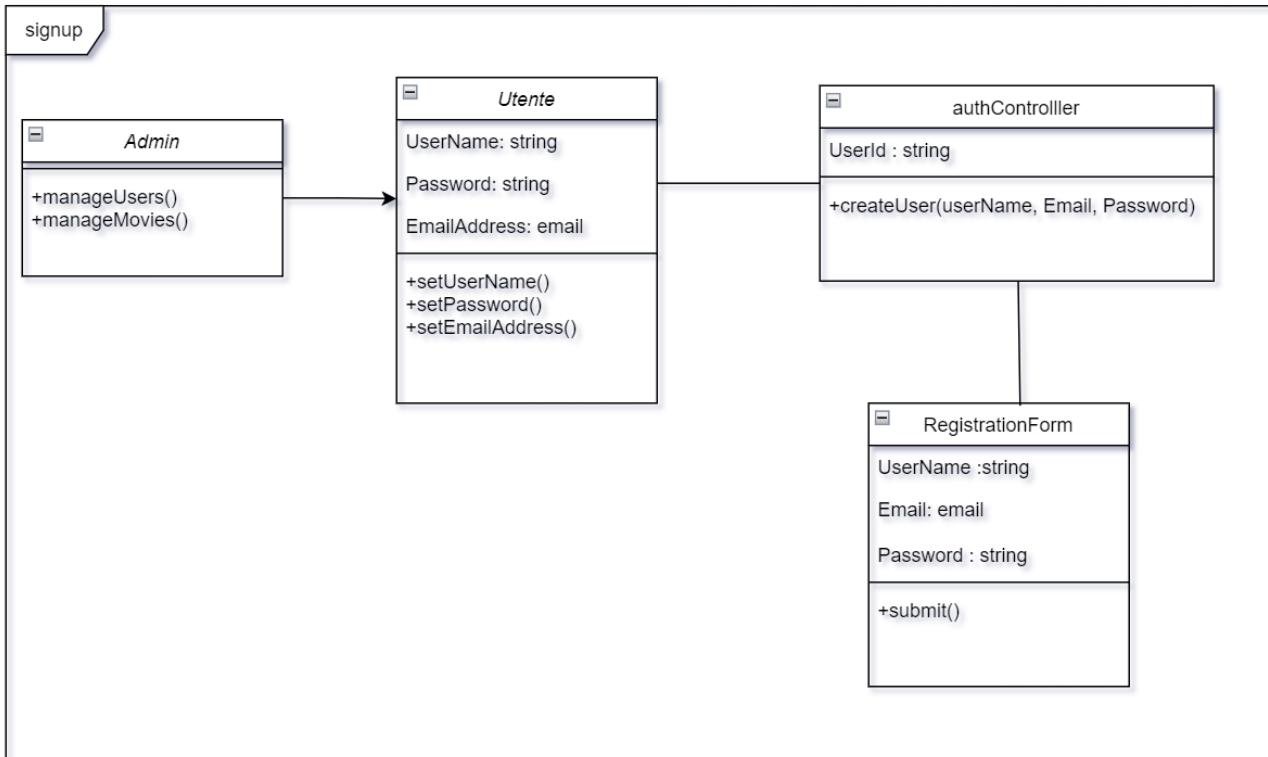


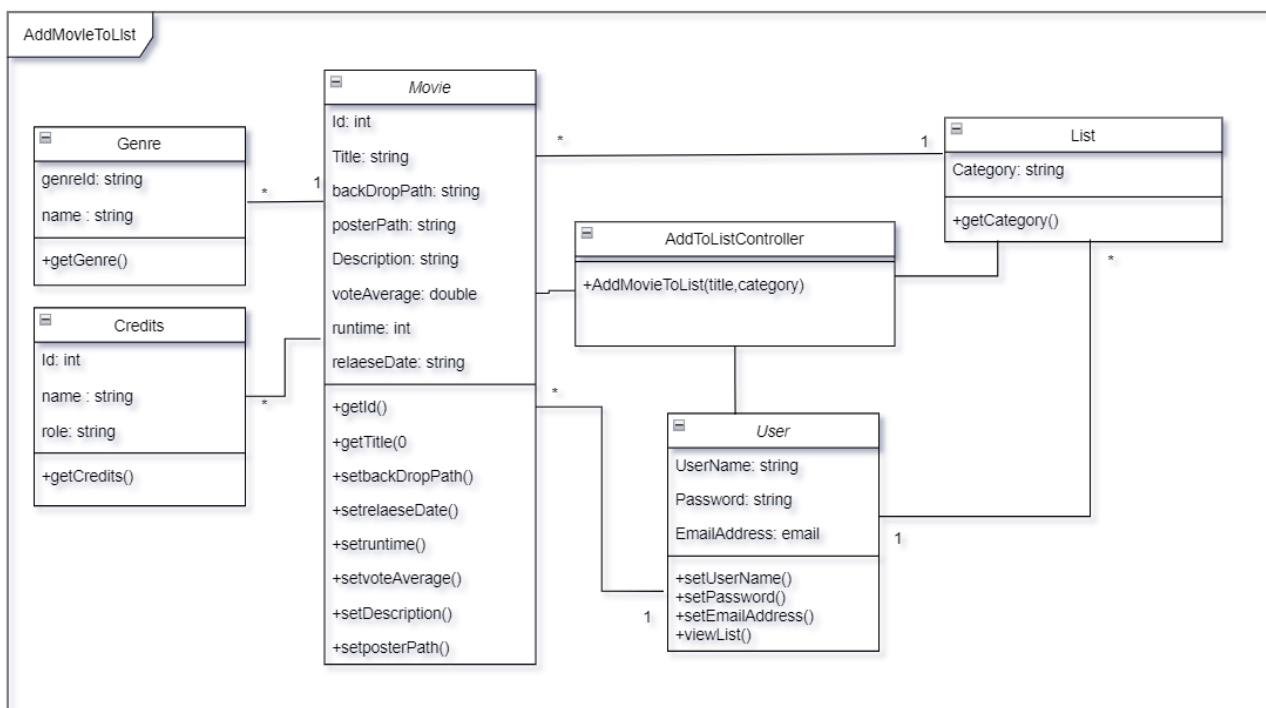
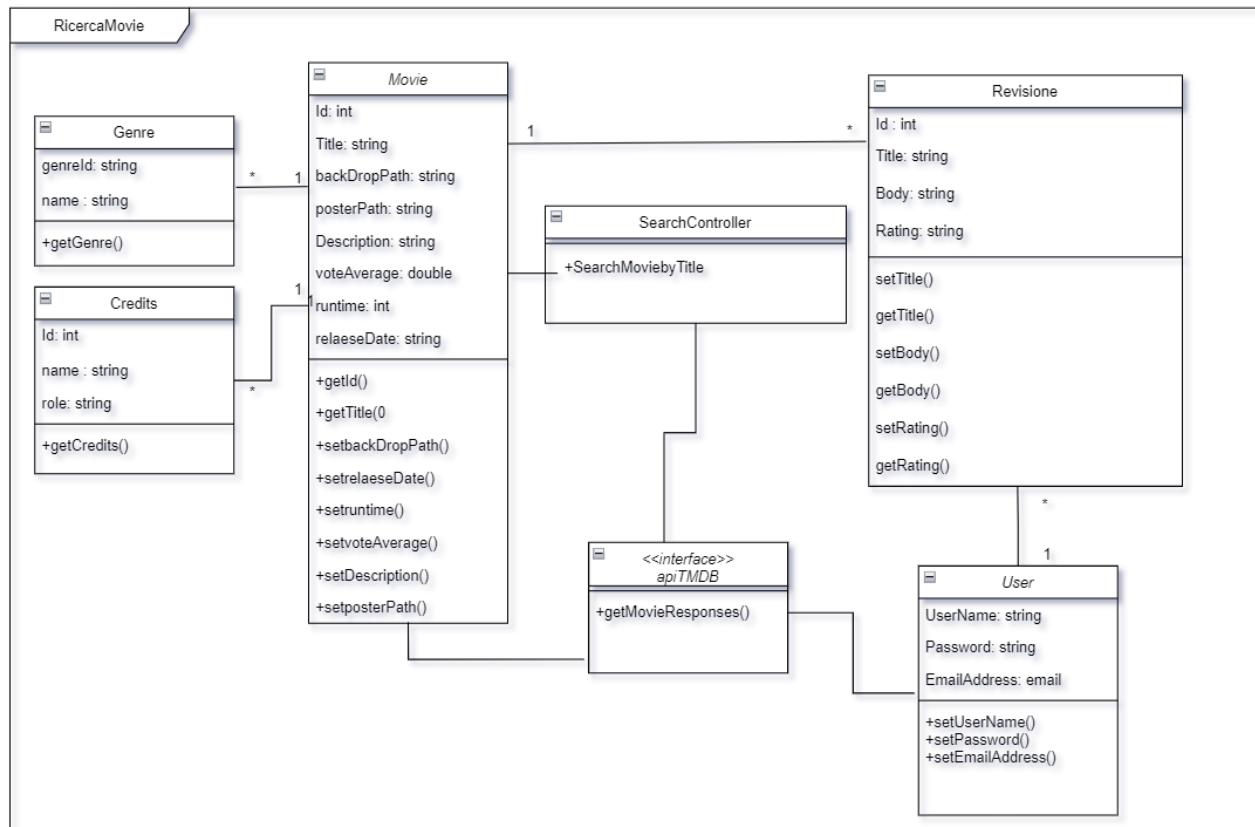
miro

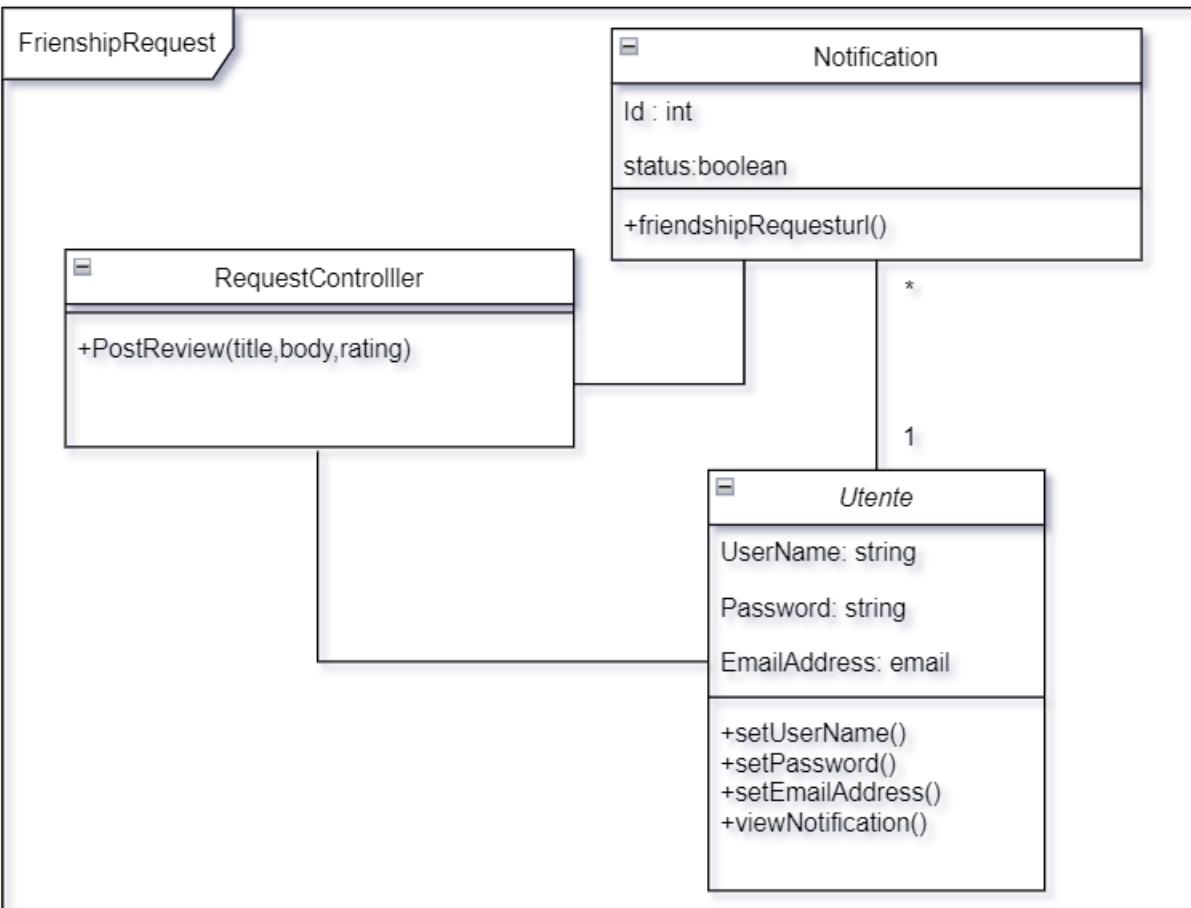
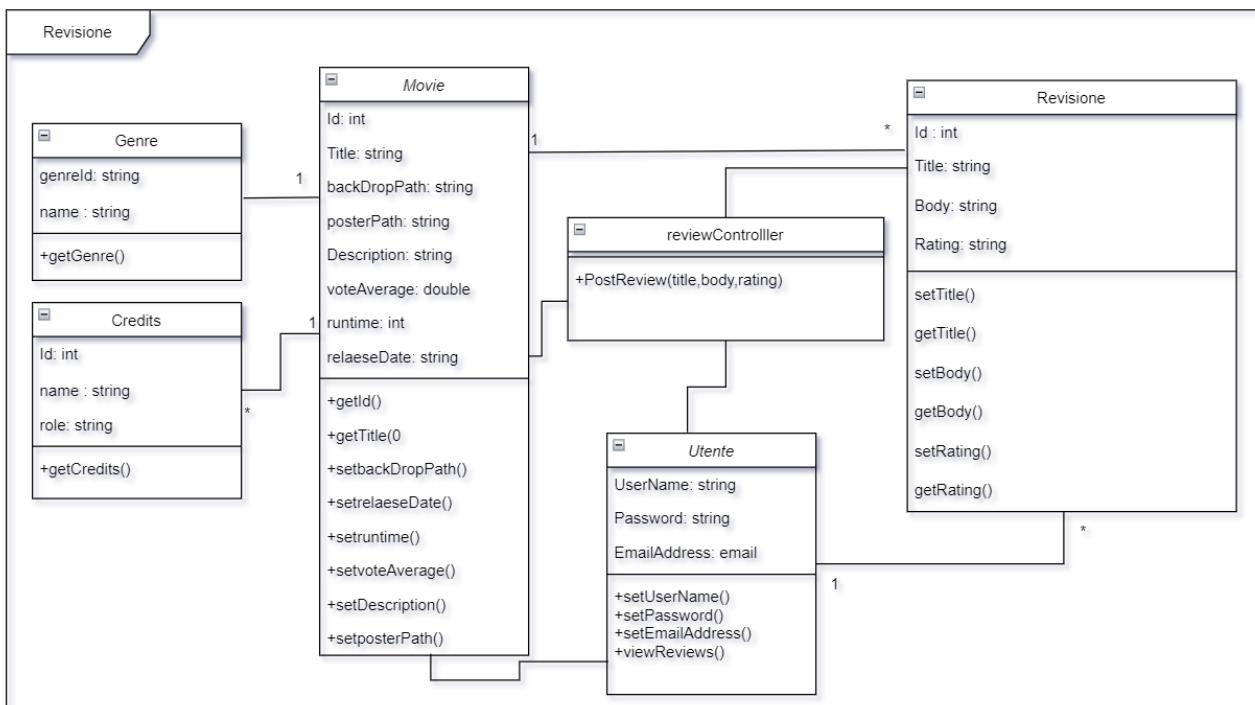
Dettagli: <https://miro.com/app/board/uXjVOqCV2fY=?moveToWidget=3458764532610921884&cot=14>

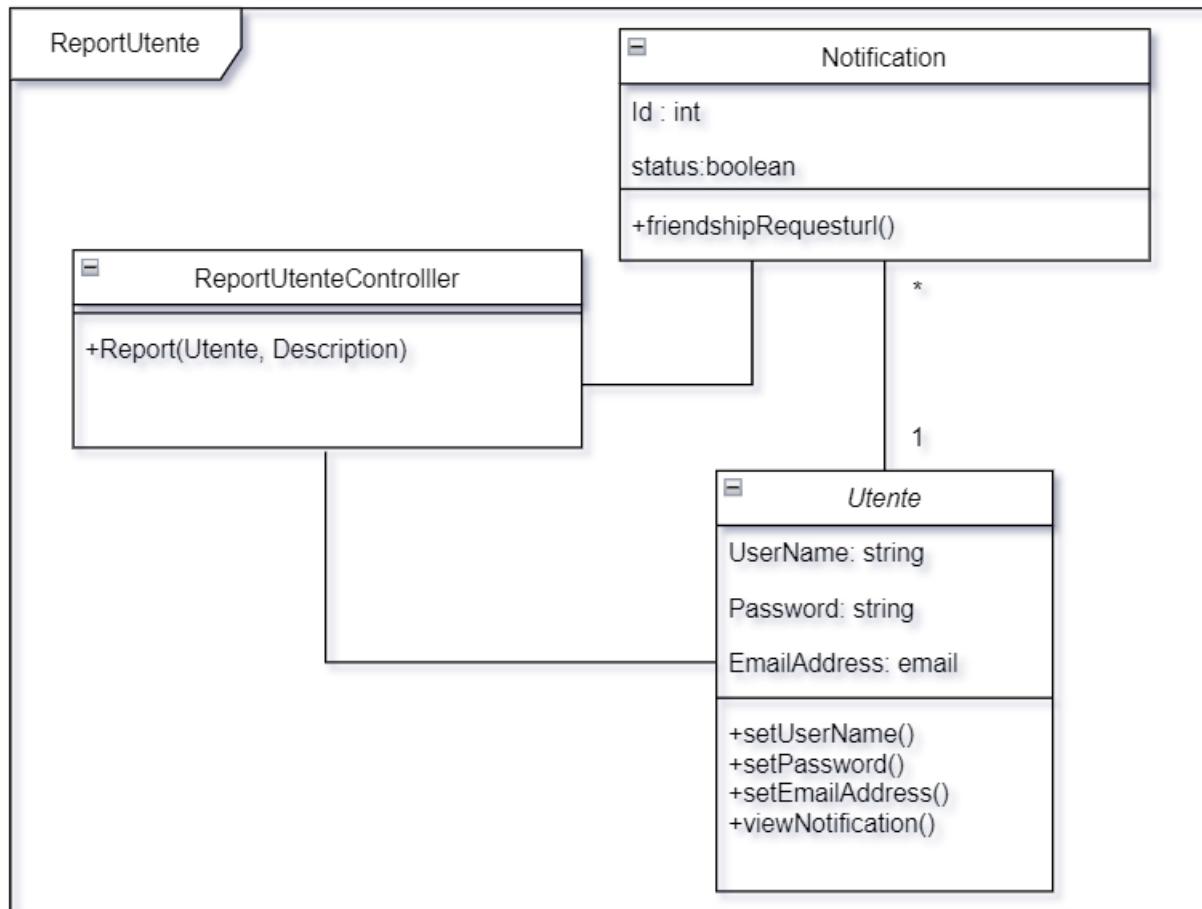
2.2 Modelli di Dominio

2.2.1 Class Diagram



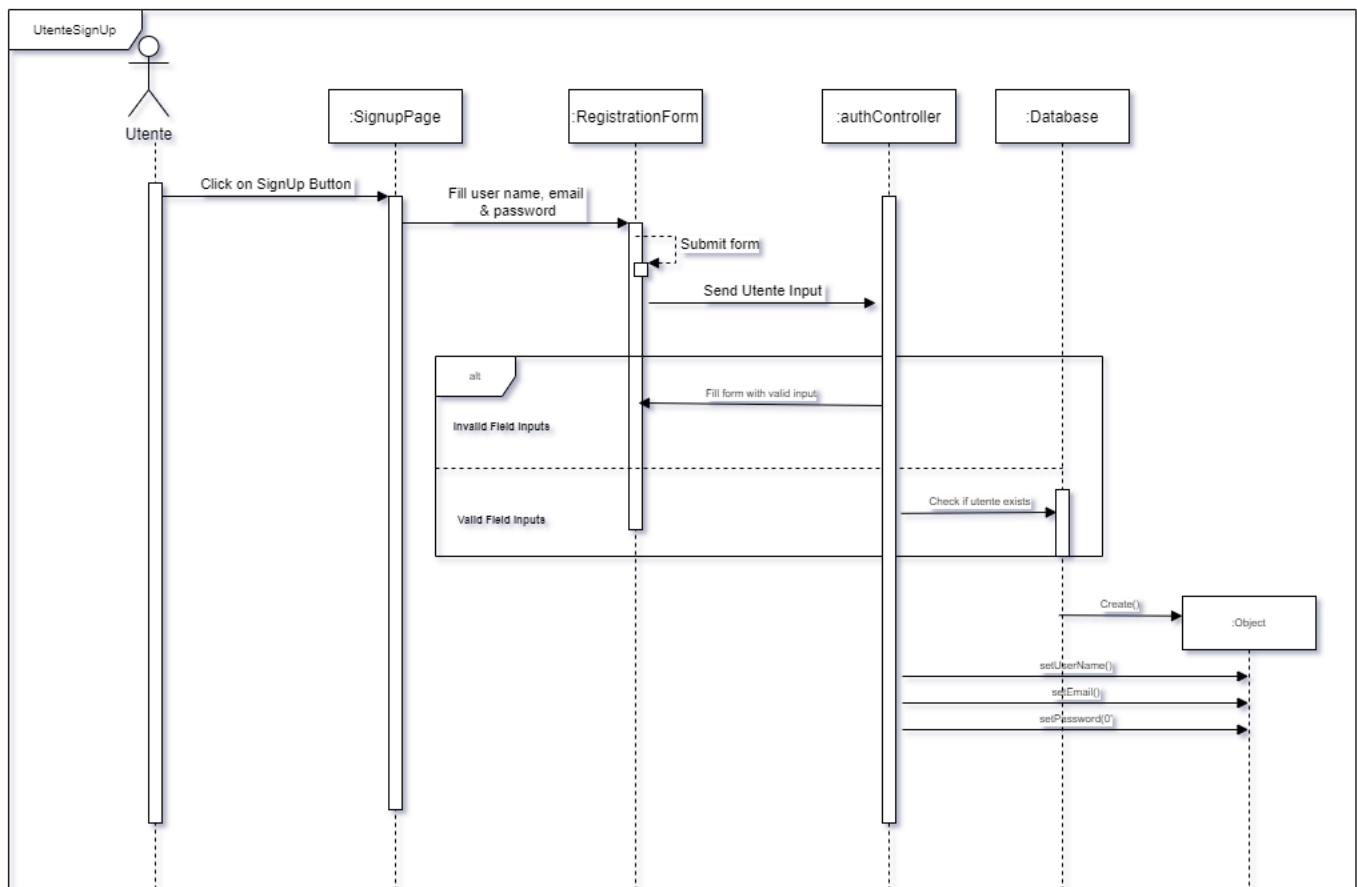


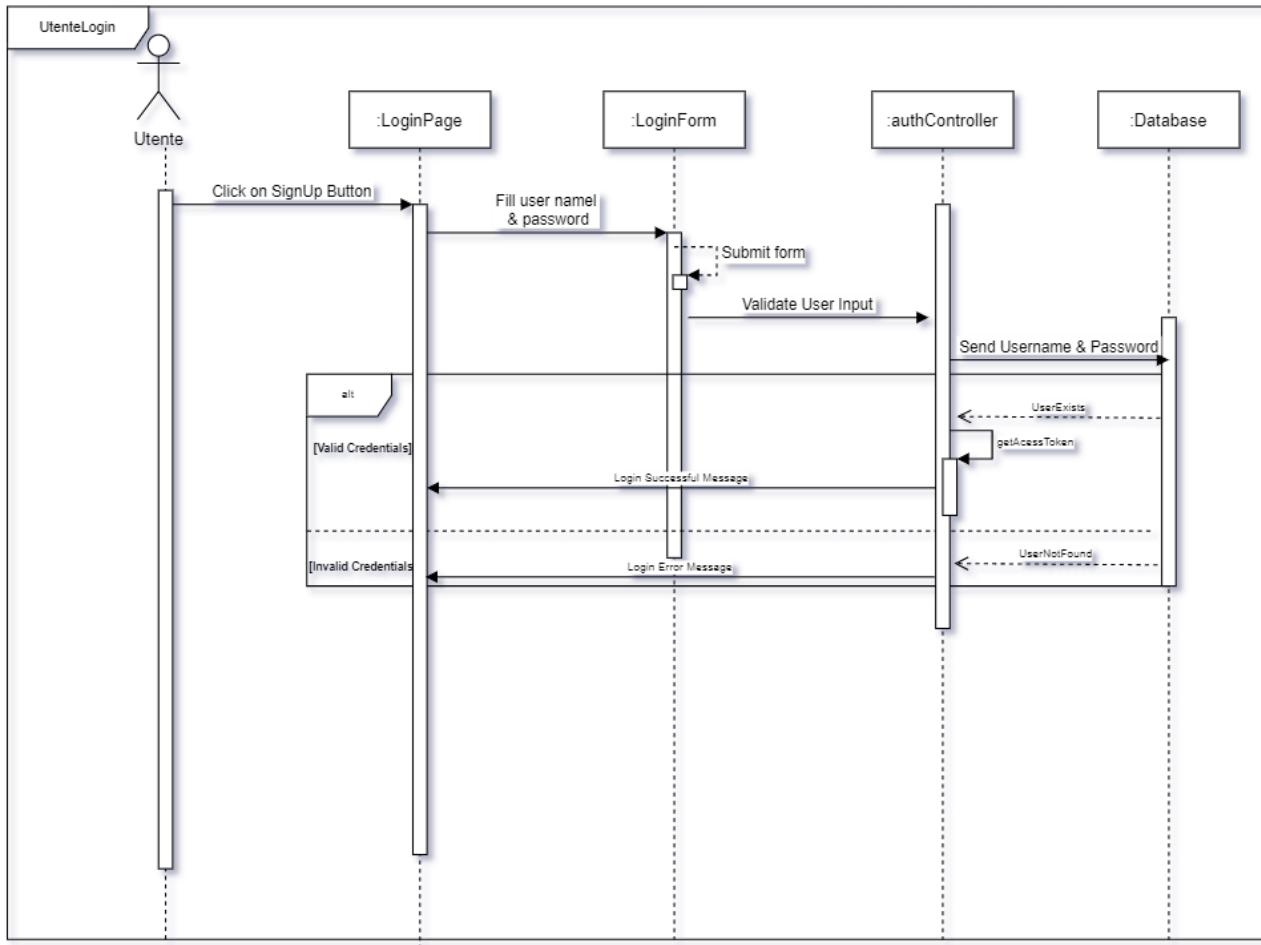


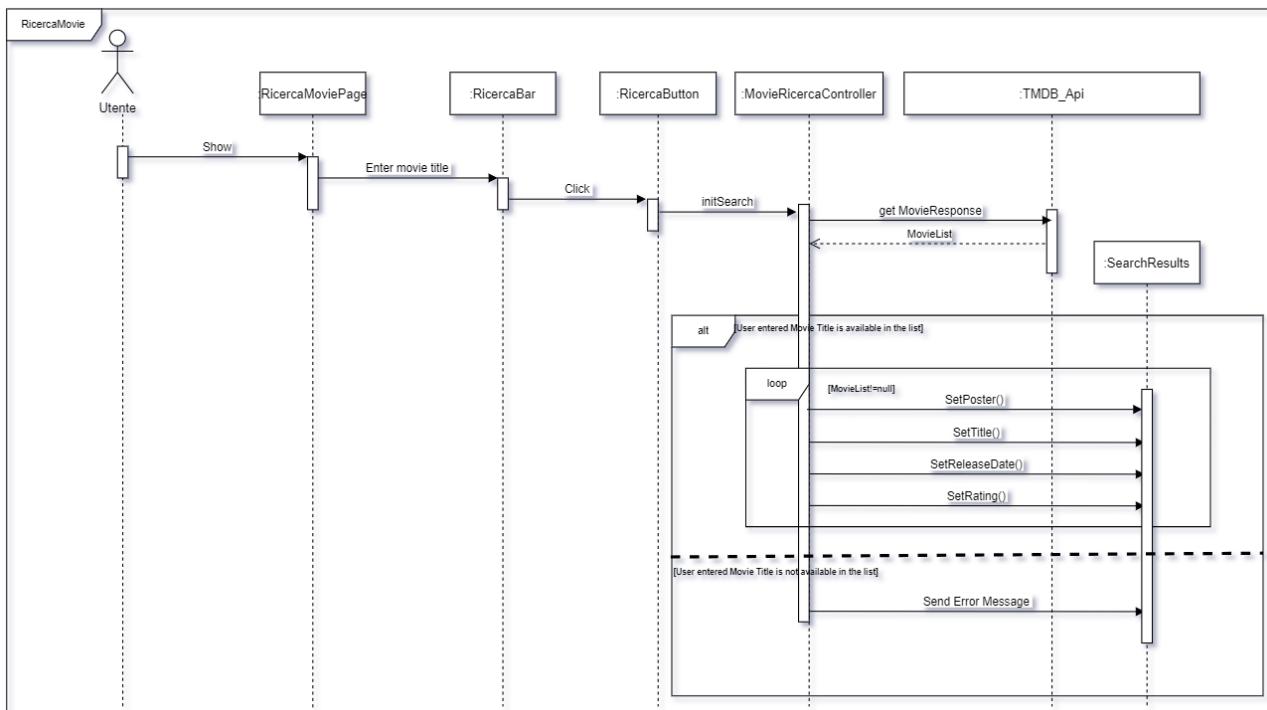
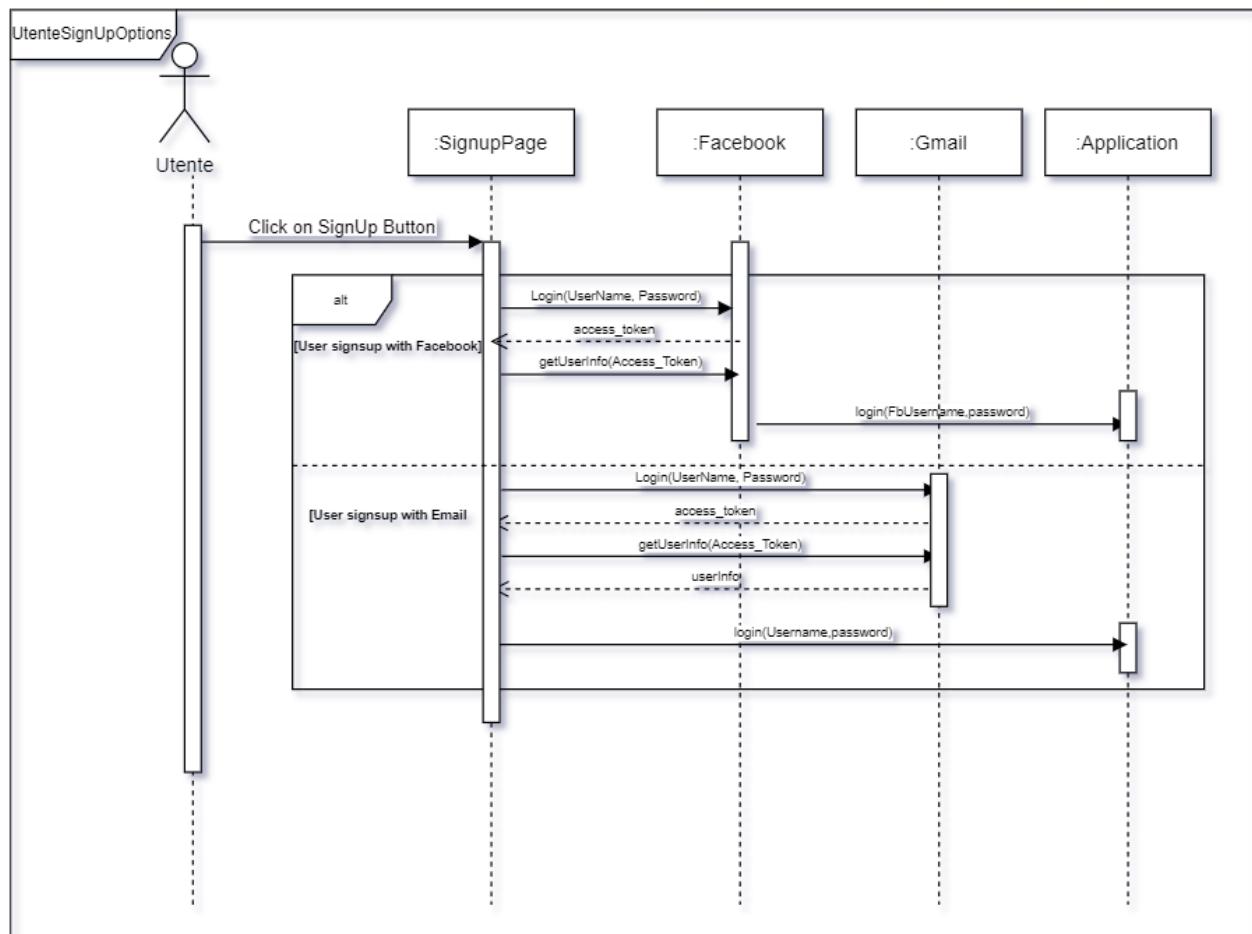


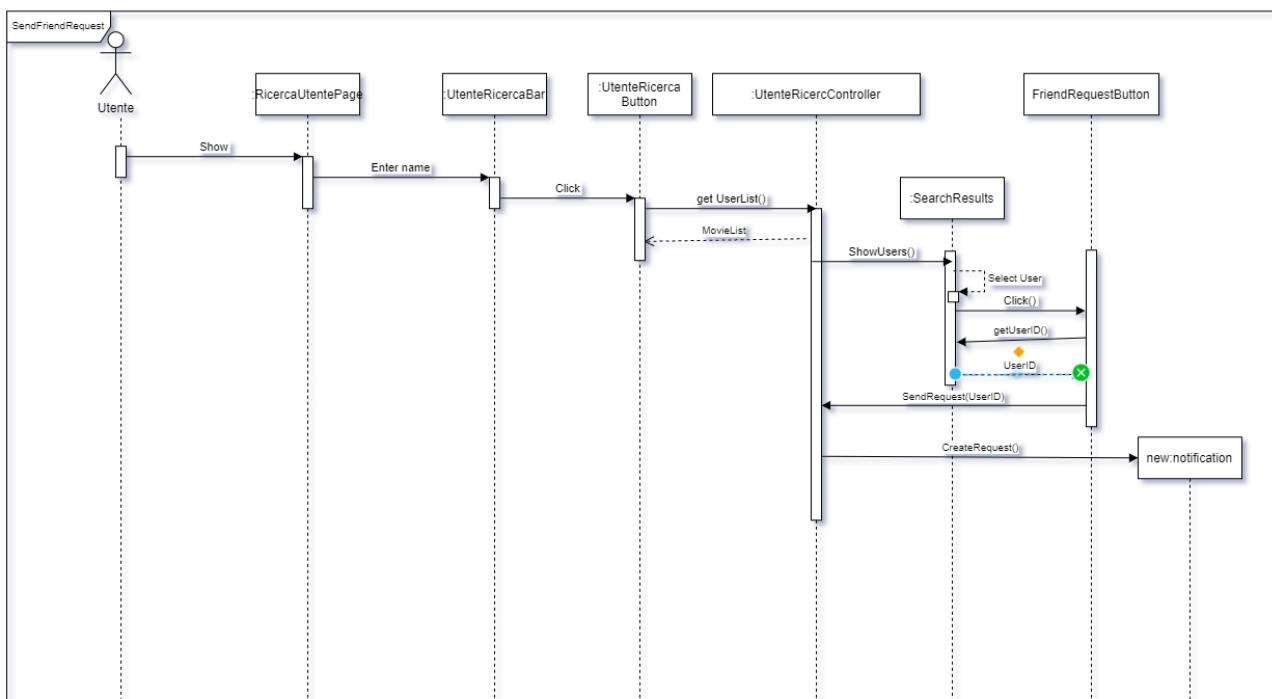
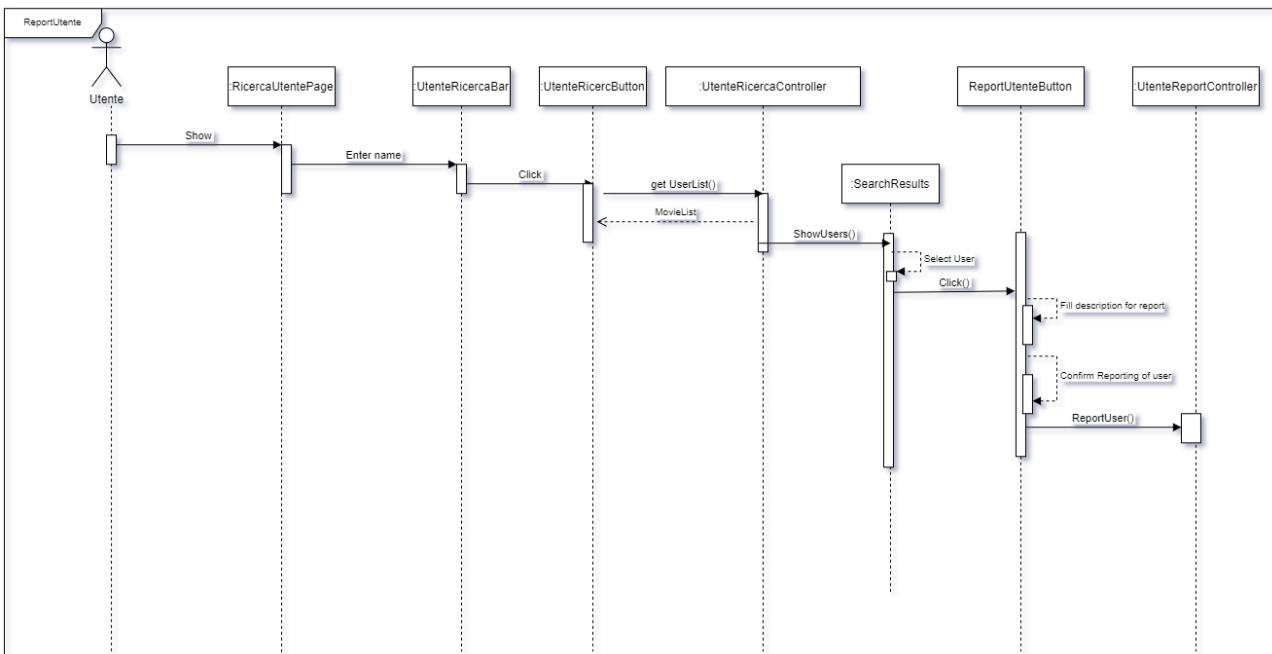
2.2.2 Sequence Diagram

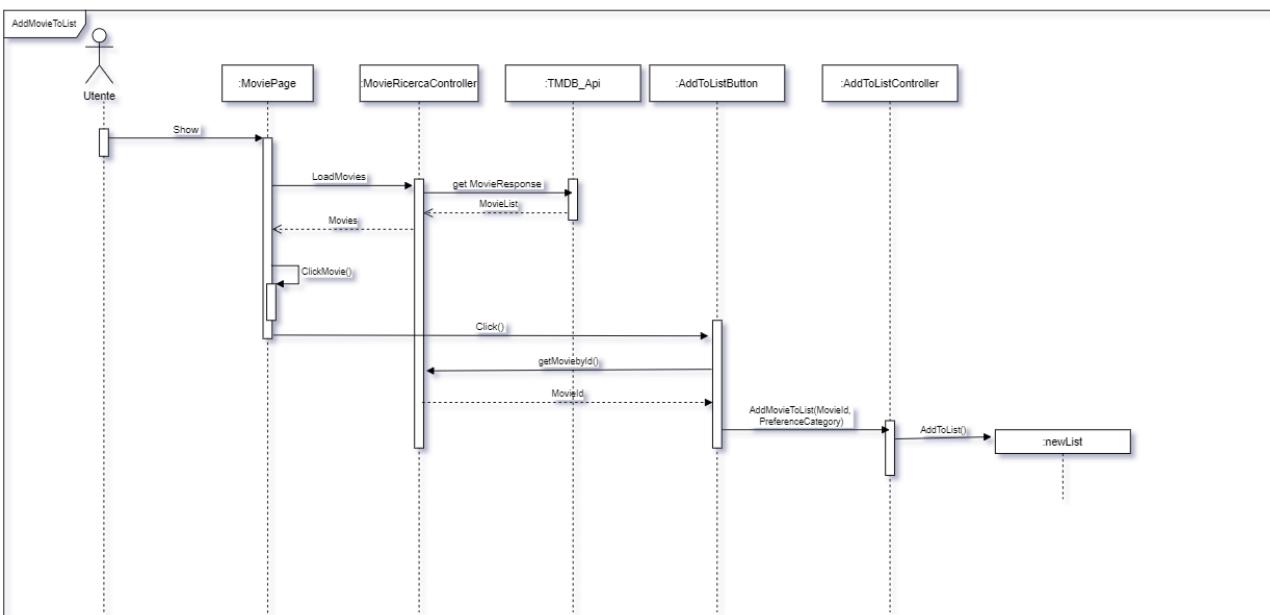
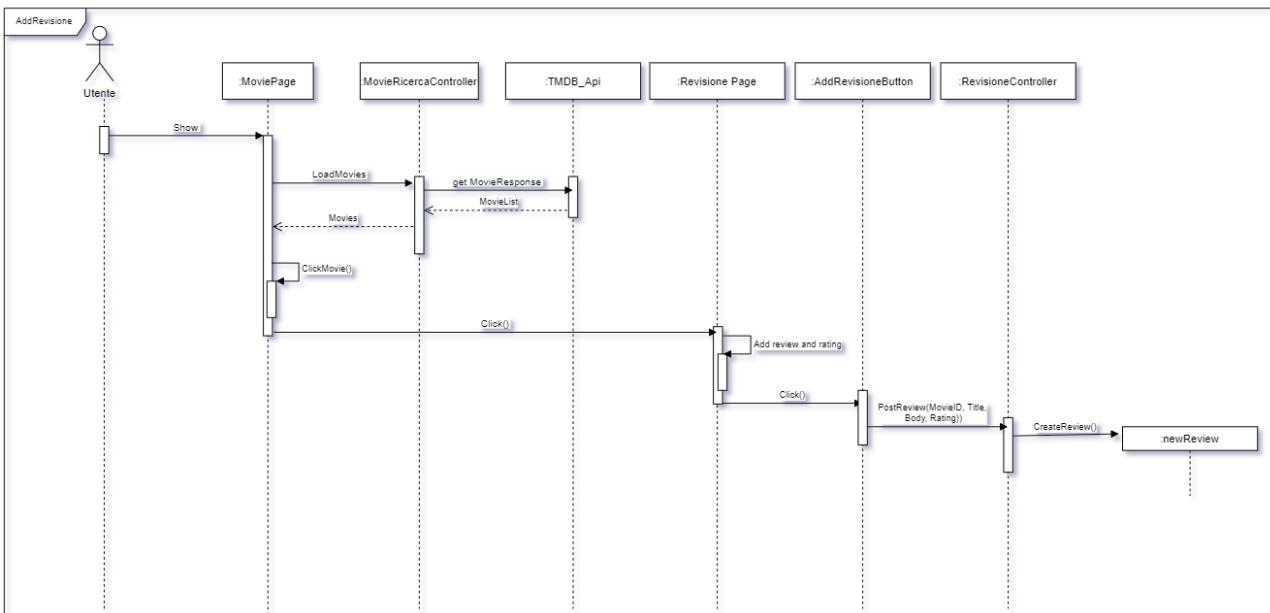
Di seguito verranno mostrati i vari sequence diagram descriventi i vari scenari di interazioni tra oggetti in linea temporale che definiscono le varie funzionalità dando un'idea delle scelte implementative intraprese.

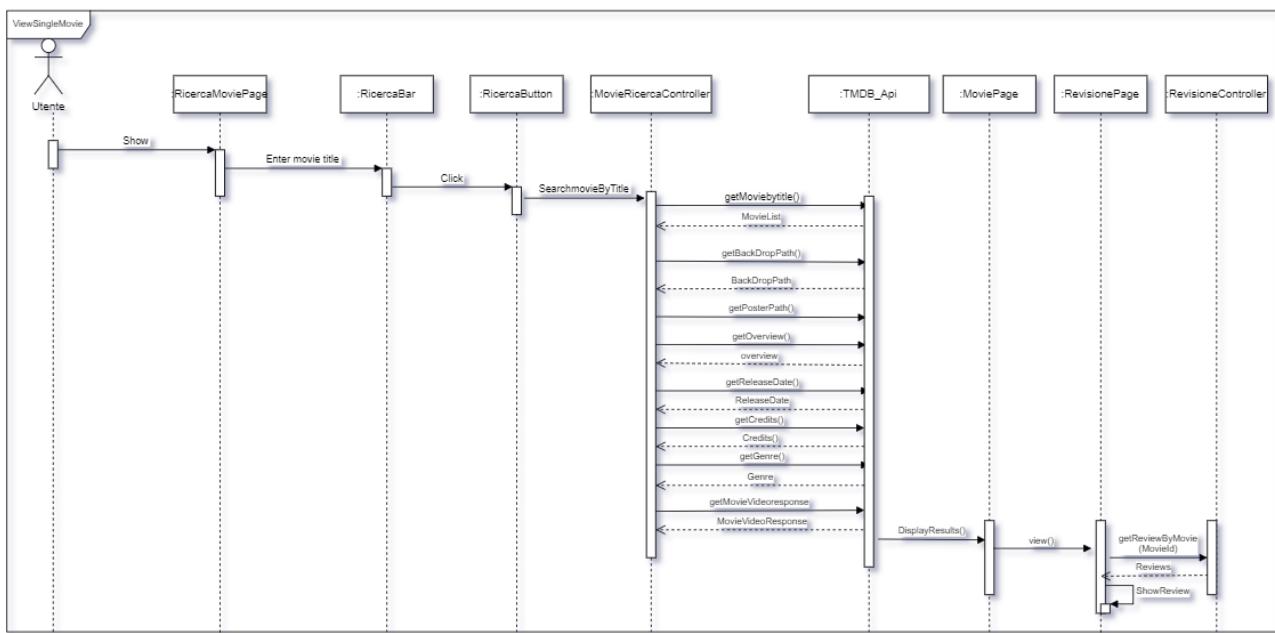




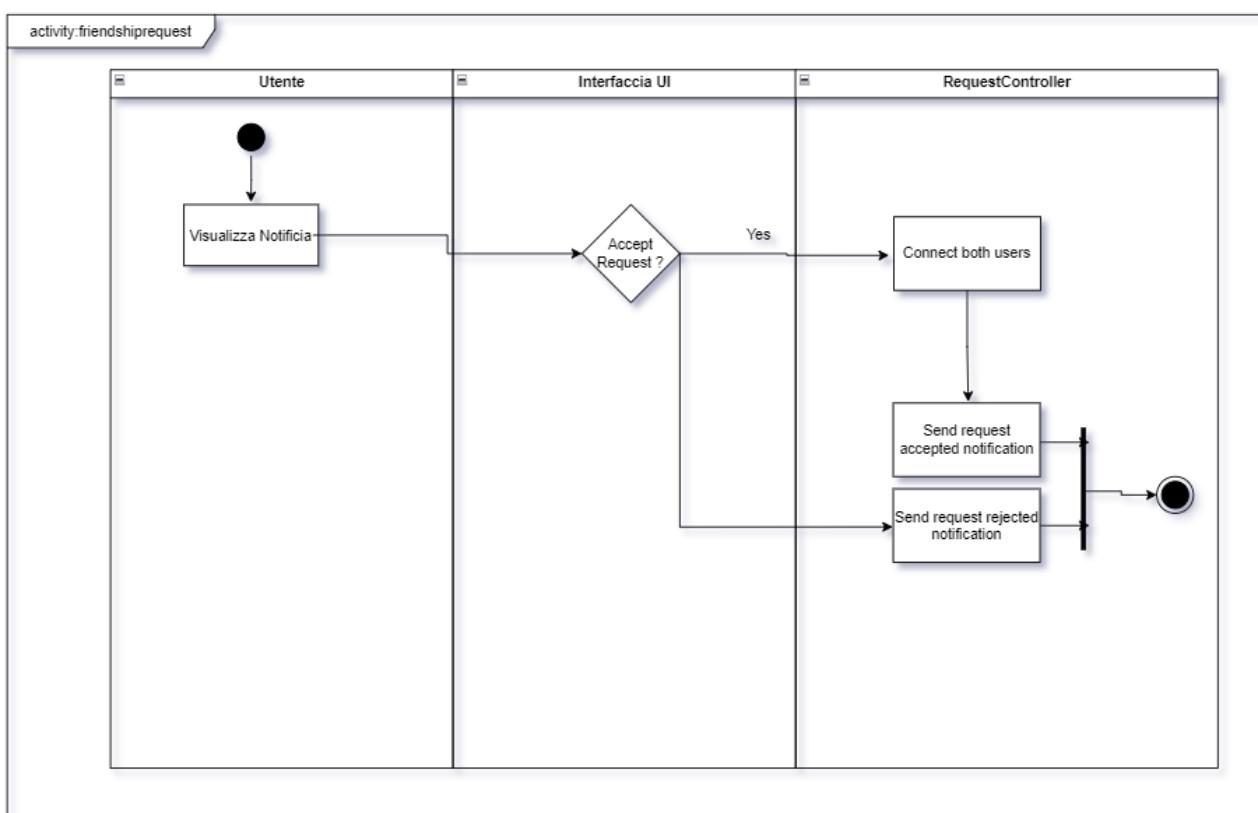
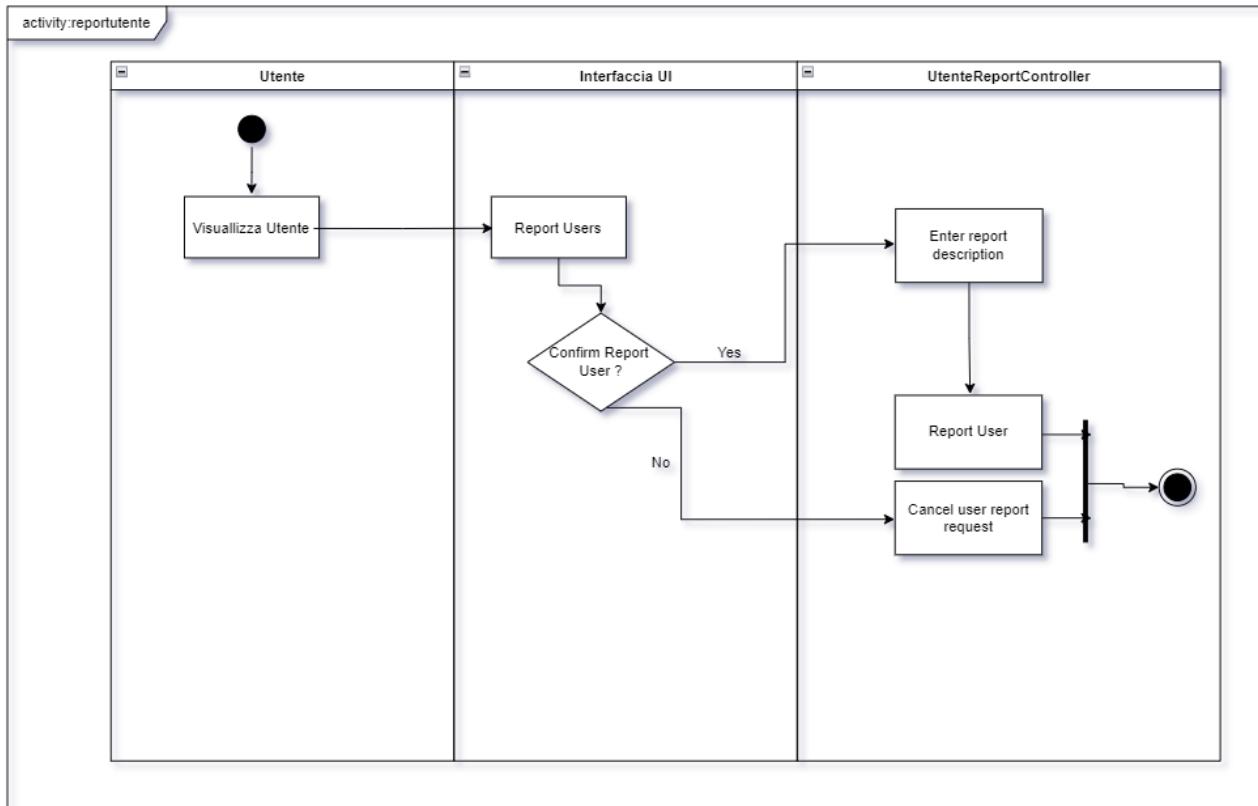


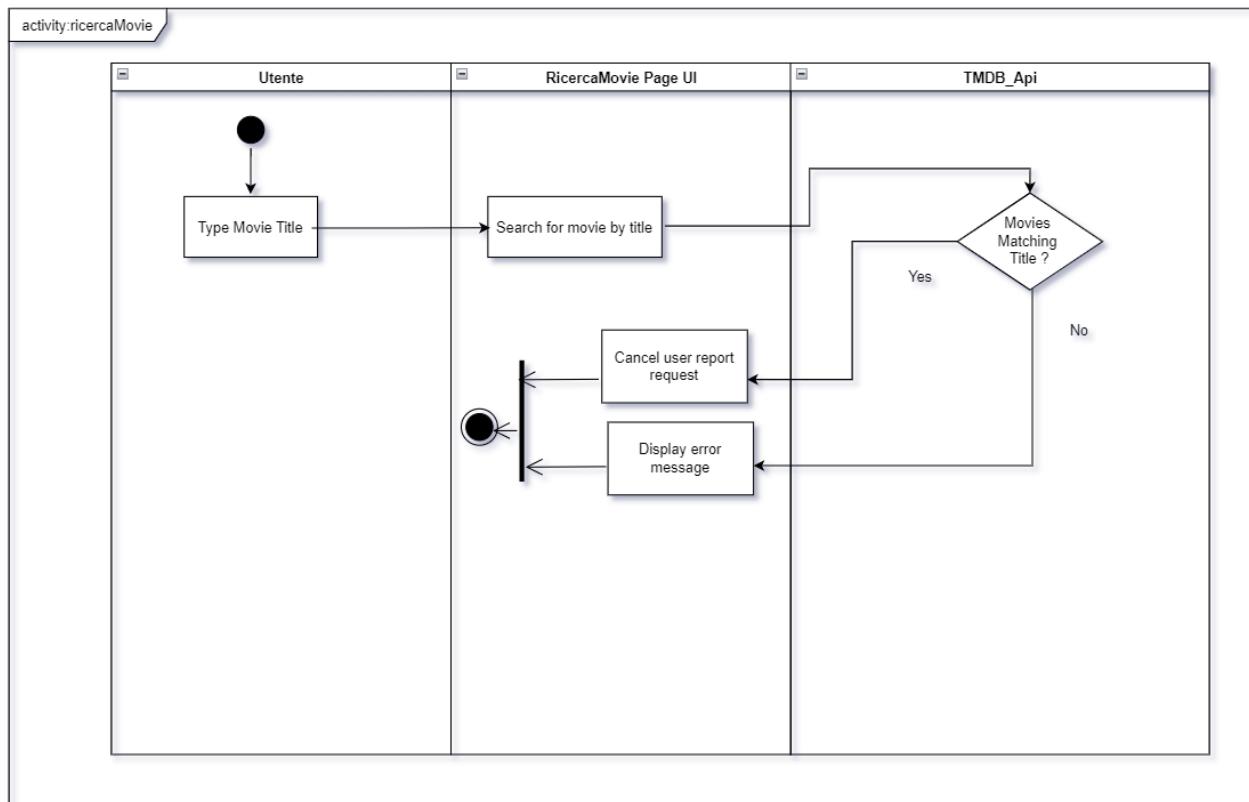




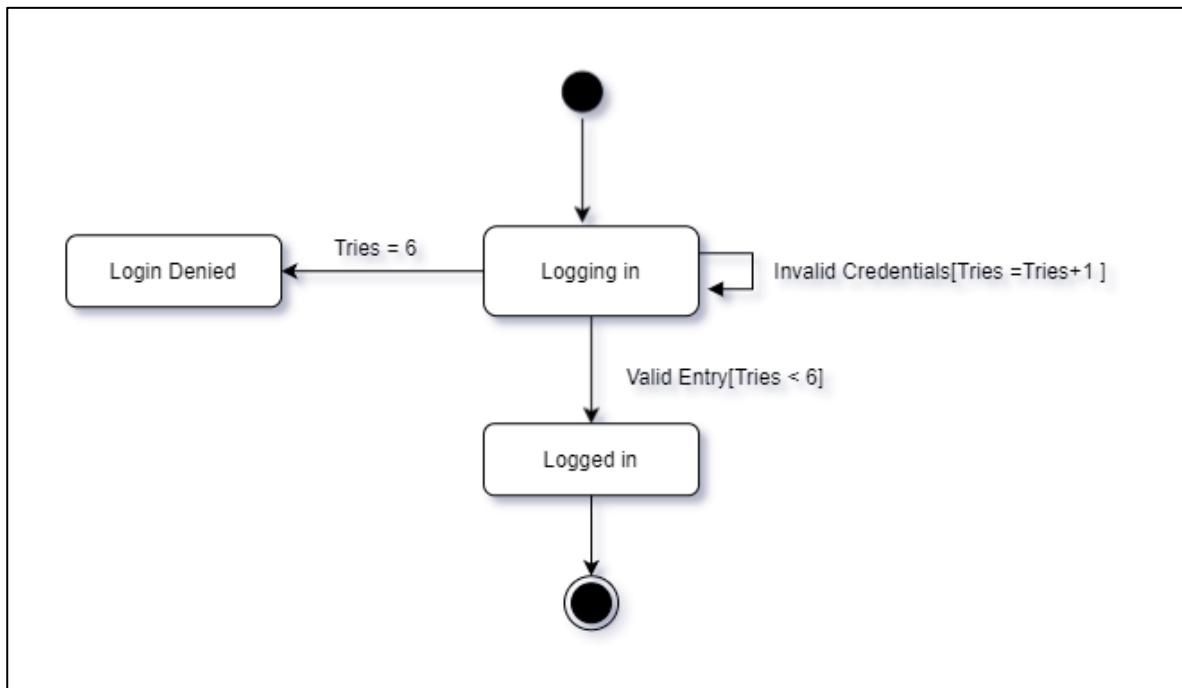
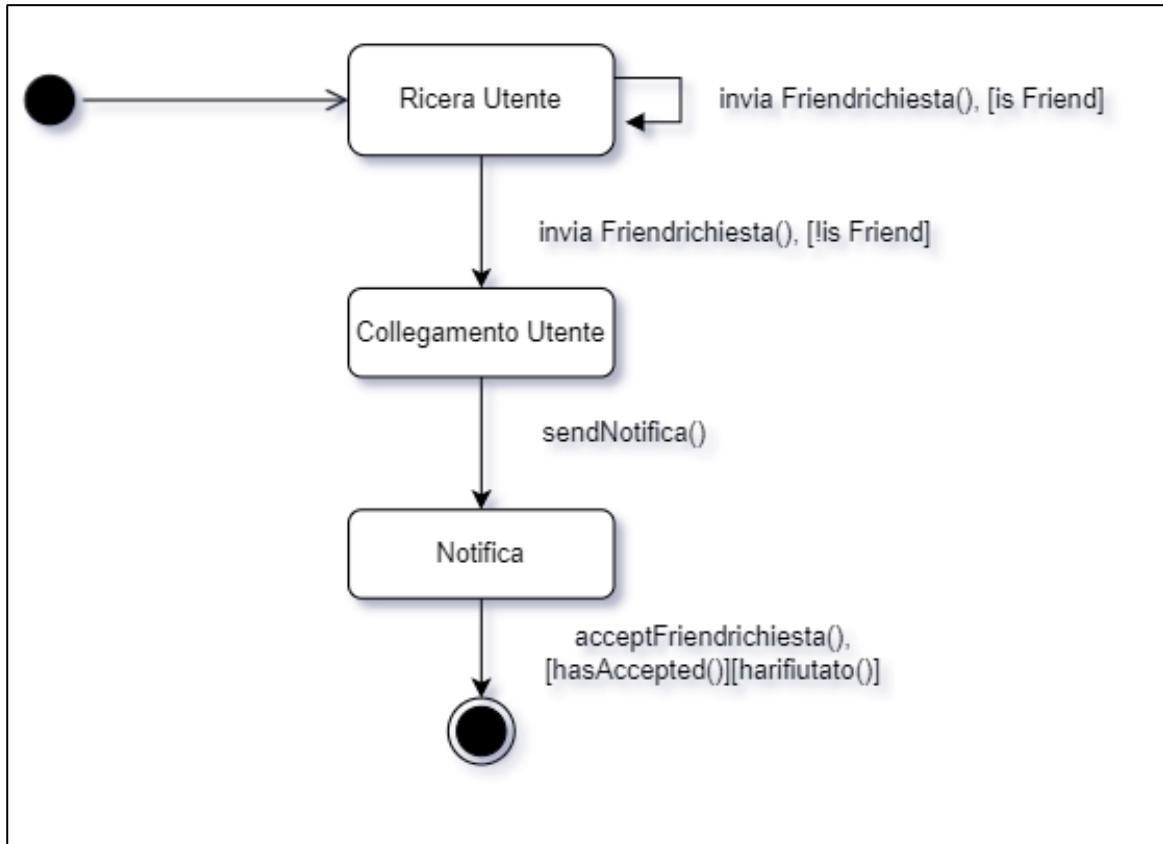


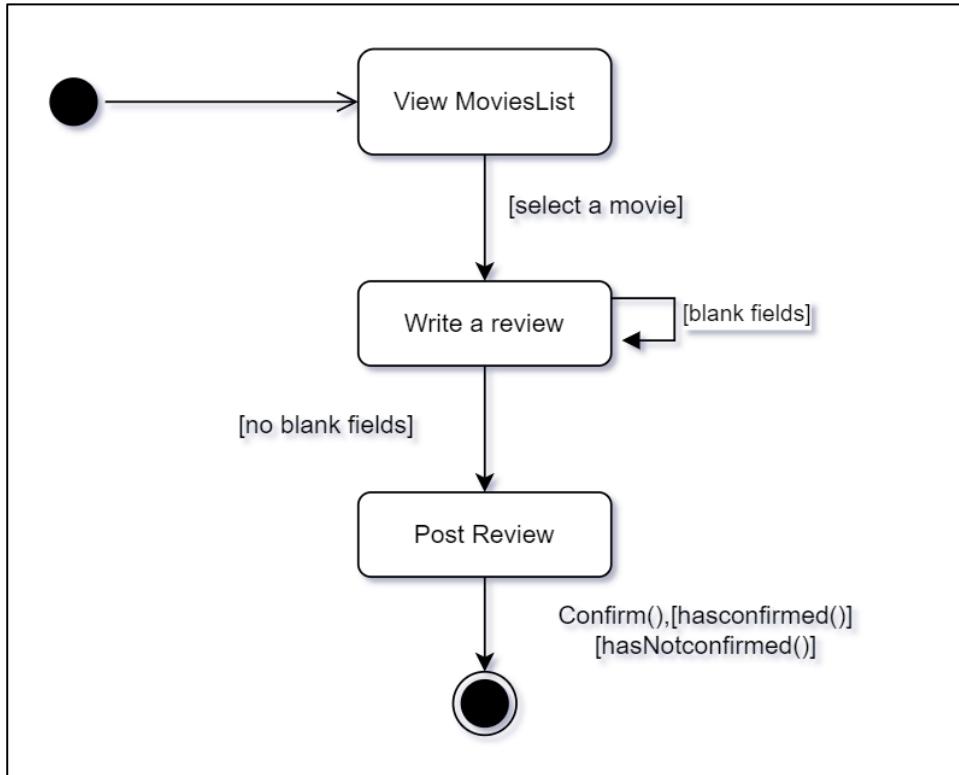
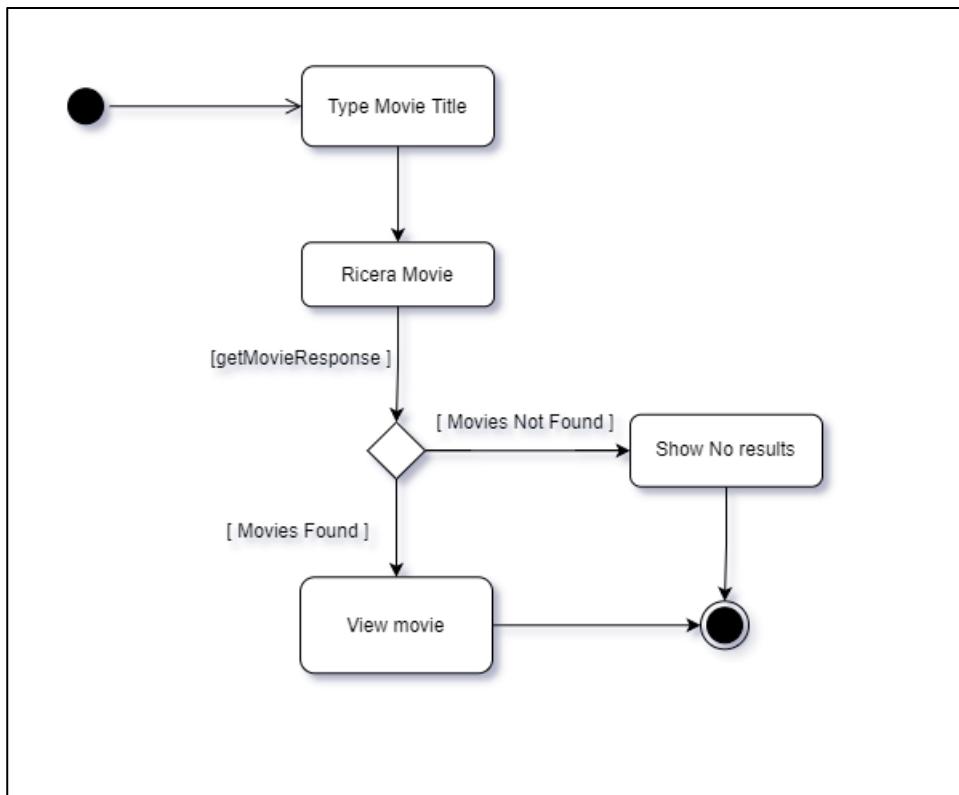
2.2.3 Activity Diagram

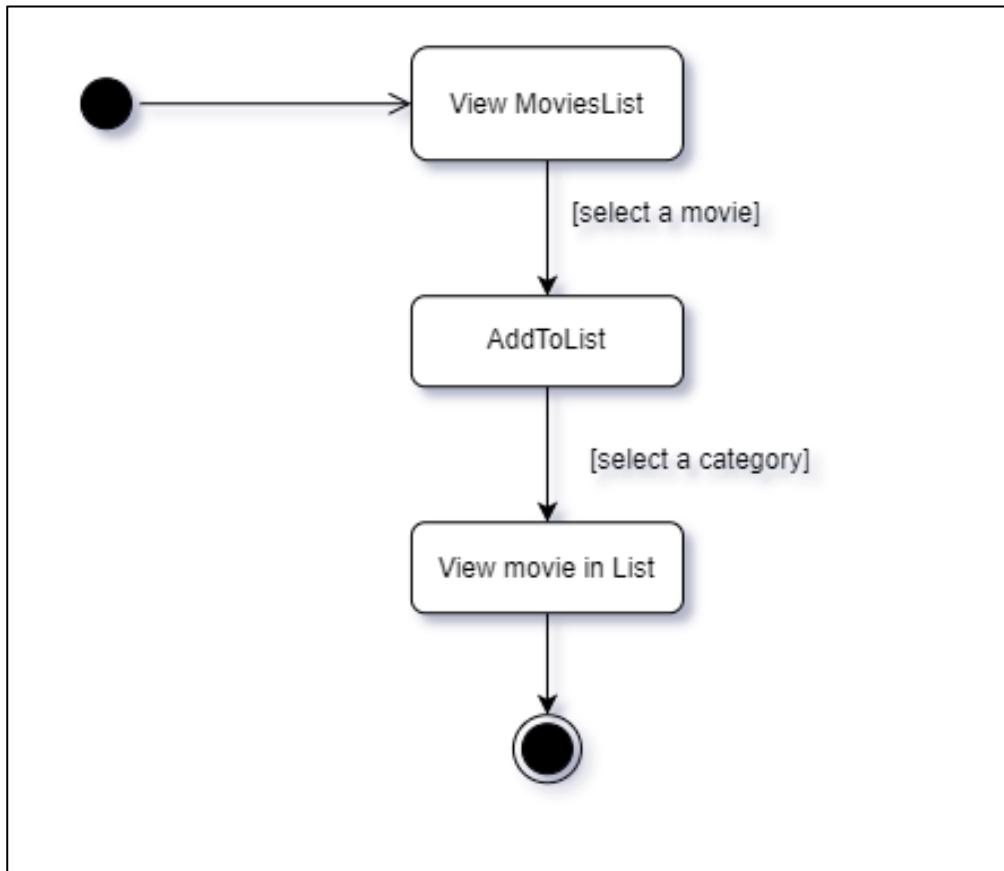
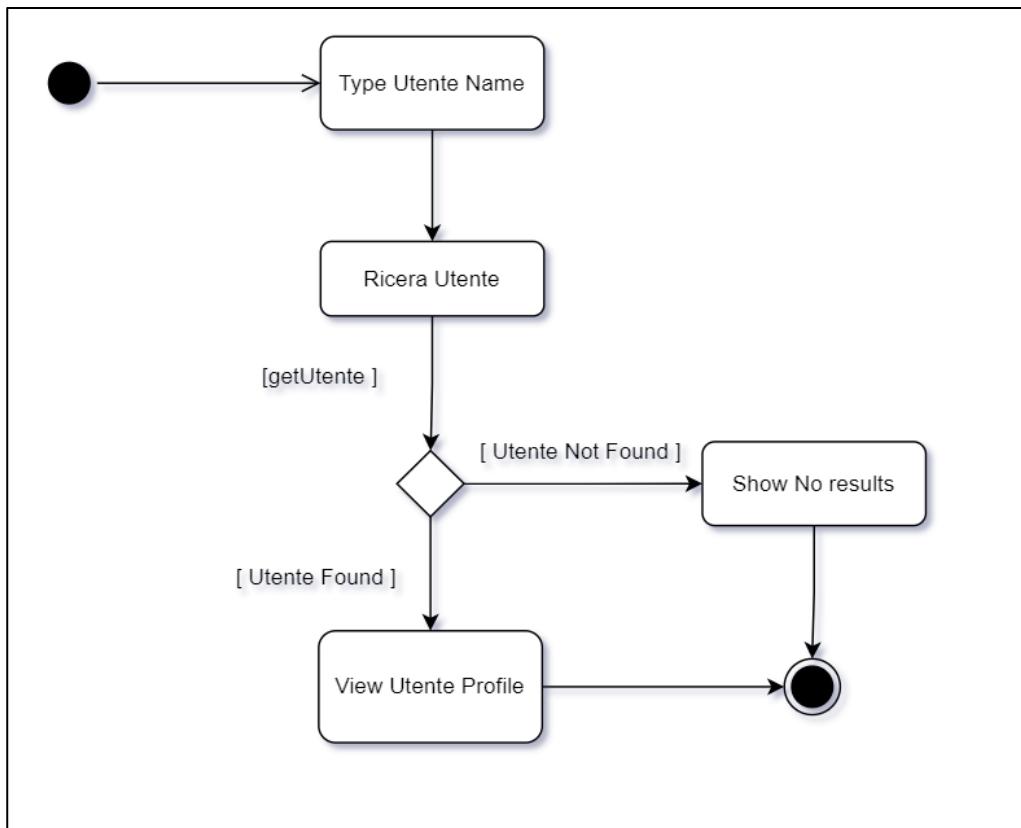




2.2.4 State Diagrams







3. Design del Sistema

3.1 Analisi dell'architettura:

CineMates è un'app mobile sviluppata per iOS.

È stato creato utilizzando Swift e SwiftUI insieme a UIKit.

L'utilizzo di SwiftUI migliora costantemente la scalabilità delle applicazioni su più piattaforme Apple. Questo framework consente l'anteprima dal vivo durante lo sviluppo, accelerando così notevolmente lo sviluppo di applicazioni mobili reattive e facili da usare.

Abbiamo adottato un modello architetturale MVVM che consiste nella separazione degli aspetti della nostra applicazione in tre componenti:

- **Model**
- **View**
- **ViewModel**

permettendoci di evitare di mescolare il codice che fornisce la logica a quello che gestisce la UI. Il **Model** rappresenta il punto di accesso ai dati. Trattasi di una o più classi che leggono dati dal DB, oppure da un servizio Web di qualsivoglia natura.

La **View** rappresenta la vista dell'applicazione, l'interfaccia grafica che mostrerà i dati.

Il **ViewModel** è il punto di incontro tra la View e il Model: i dati ricevuti da quest'ultimo sono elaborati per essere presentati e passati alla View.

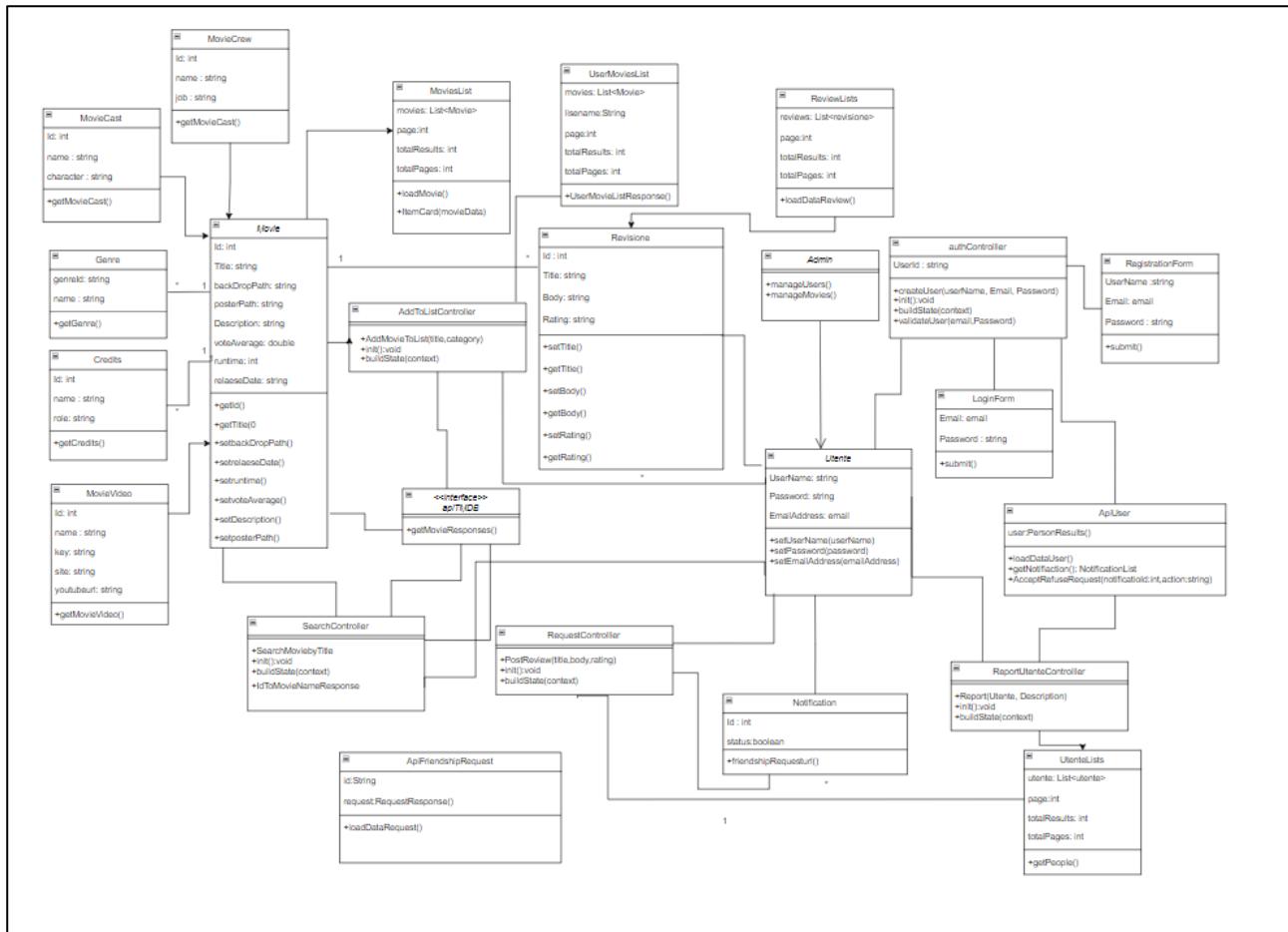
Il fulcro del funzionamento di questo pattern è la creazione di un componente, il ViewModel appunto, che rappresenta tutte le informazioni e i comportamenti della corrispondente View. La View si limita infatti, a visualizzare graficamente quanto esposto dal ViewModel, a riflettere in esso i suoi cambi di stato oppure ad attivarne dei comportamenti. Questi tre componenti sono interconnessi: il Model viene mostrato tramite la View all'utente, il quale produce gli input con cui il ViewModel aggiorna il Model. Mantenerli logicamente separati però ha grandi vantaggi nella gestione del codice, infatti questo pattern favorisce lo sviluppo, il test e la manutenzione di ciascuna parte indipendentemente dall'altra.

L'app è stata sviluppata con particolare attenzione all'usabilità, all'efficienza, alla portabilità e alle prestazioni dell'applicazione.

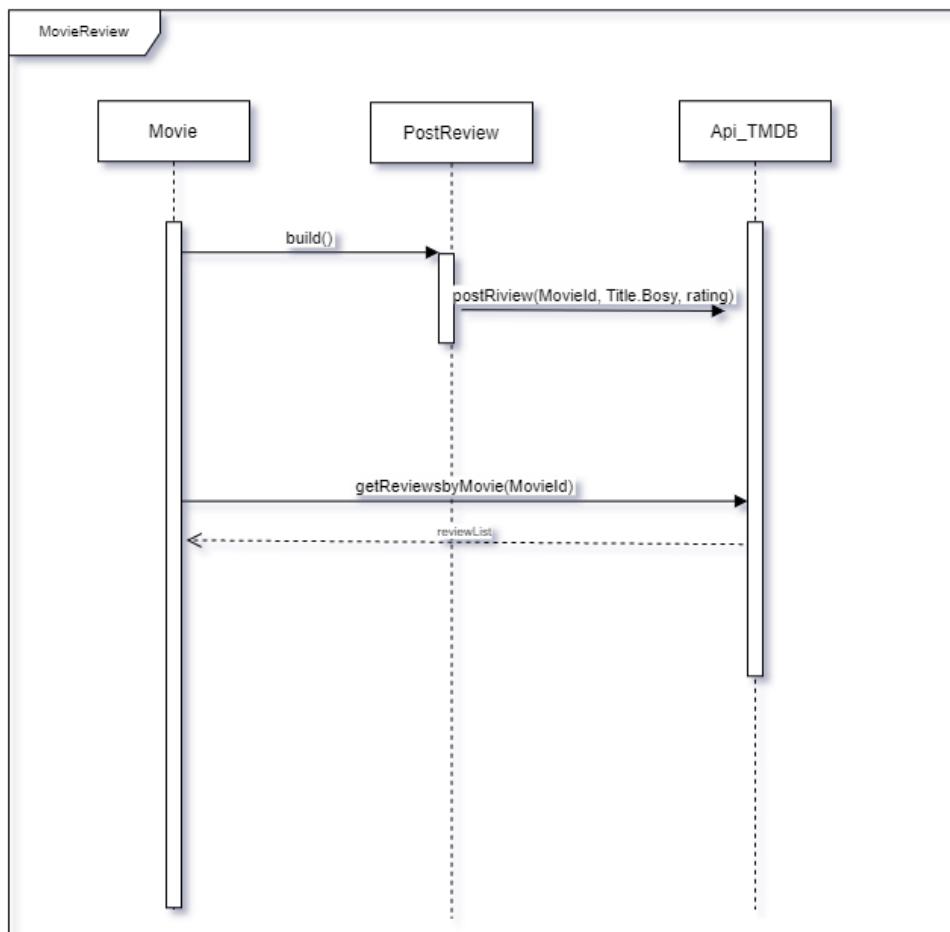
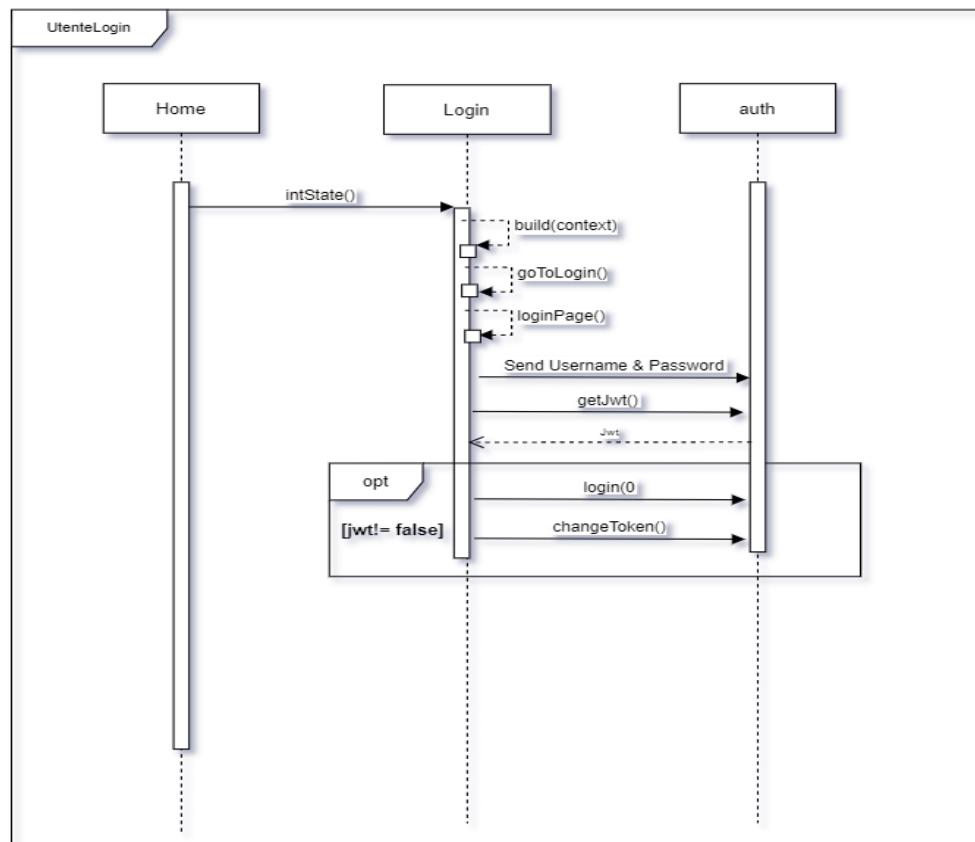
Abbiamo utilizzato l'API TMDB di terze parti come origine dati primaria richiesta per l'applicazione. Questa API invia tutti i dati richiesti per le informazioni sul film.

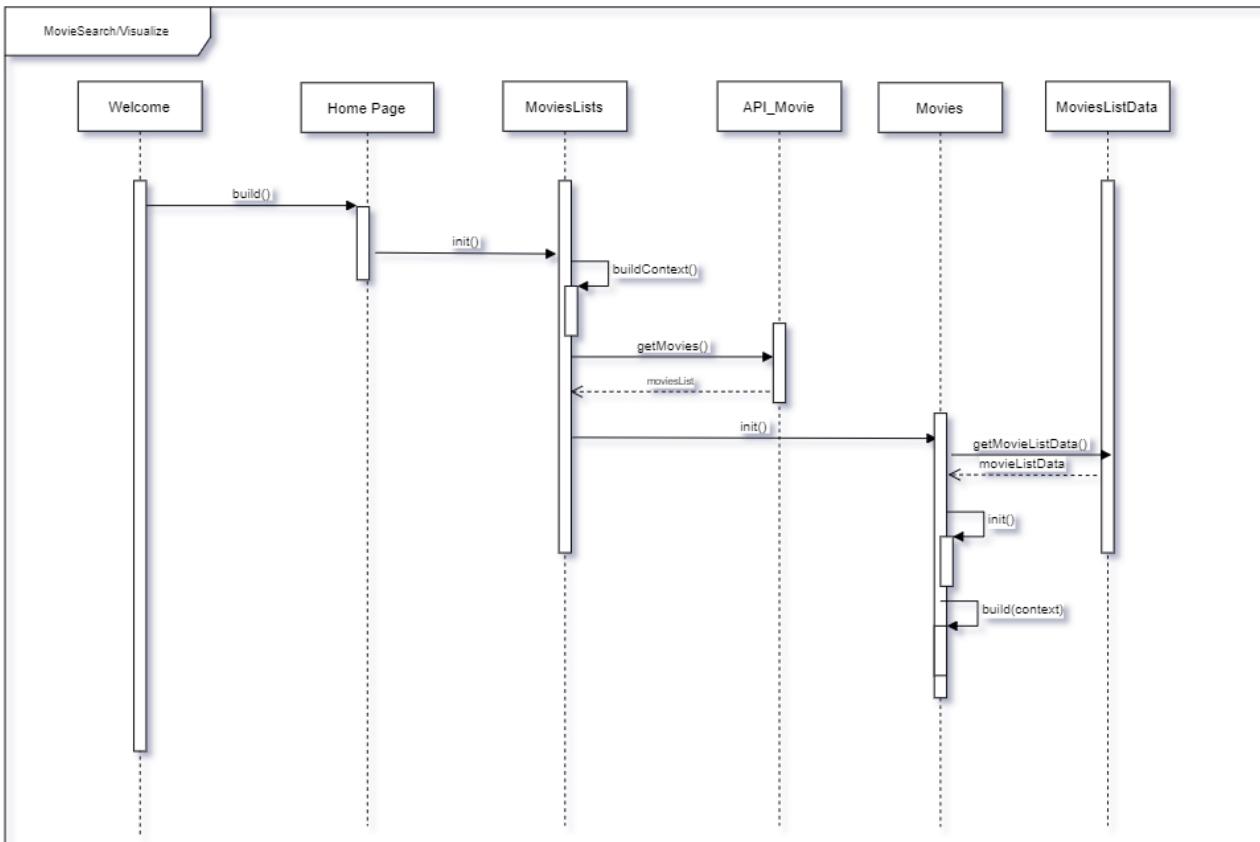
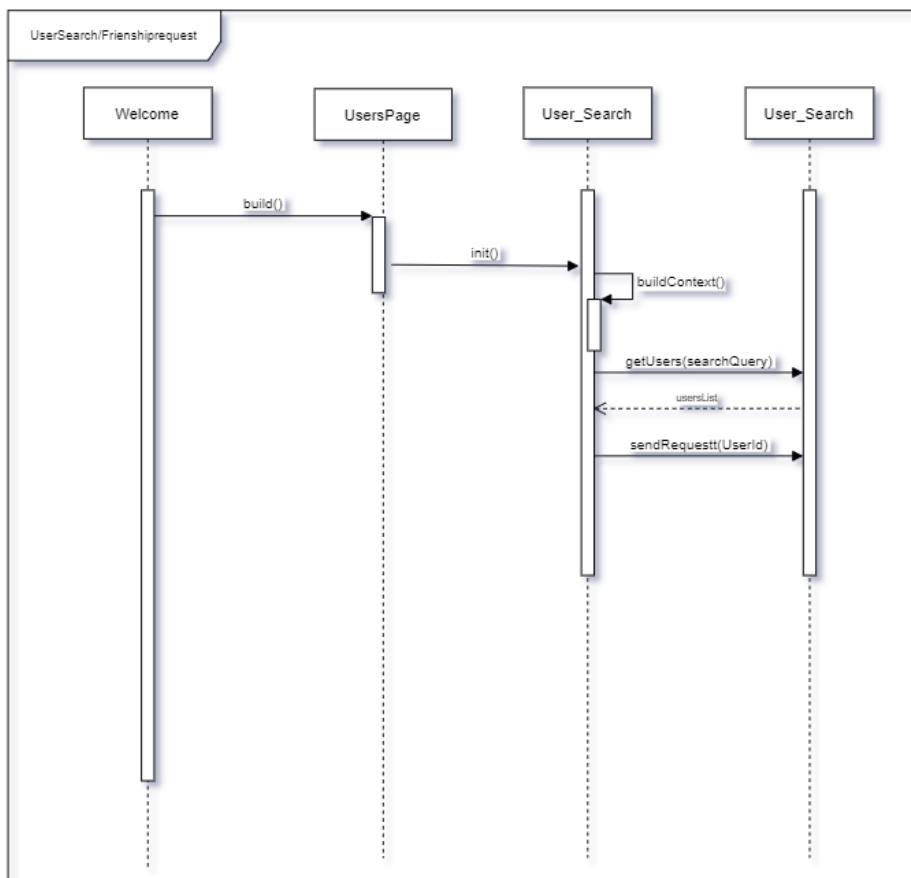
L'app è ricca di funzionalità di base necessarie per creare un ambiente social per gli appassionati di cinema, come da richiesta.

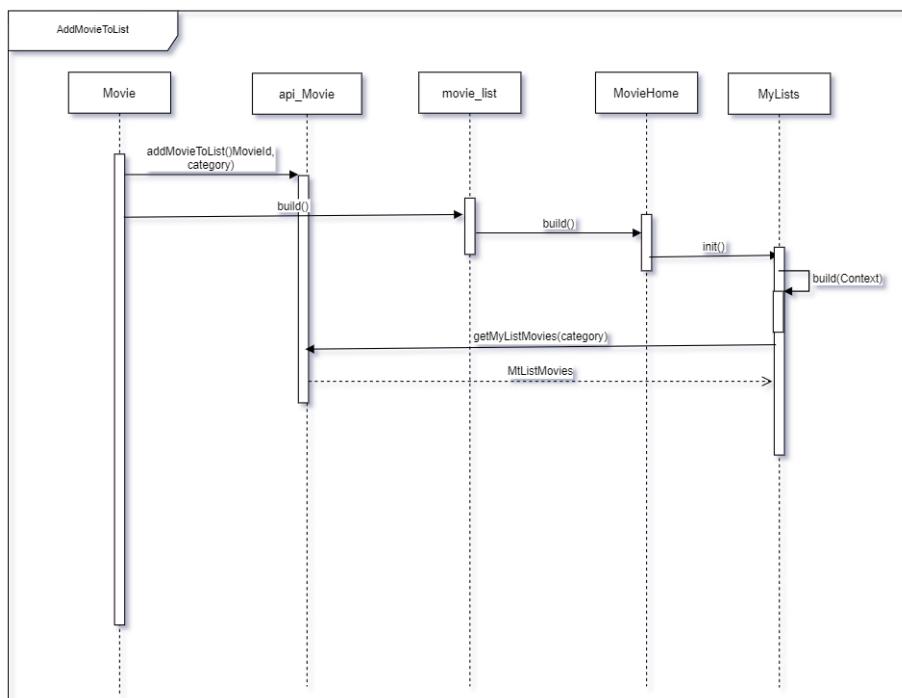
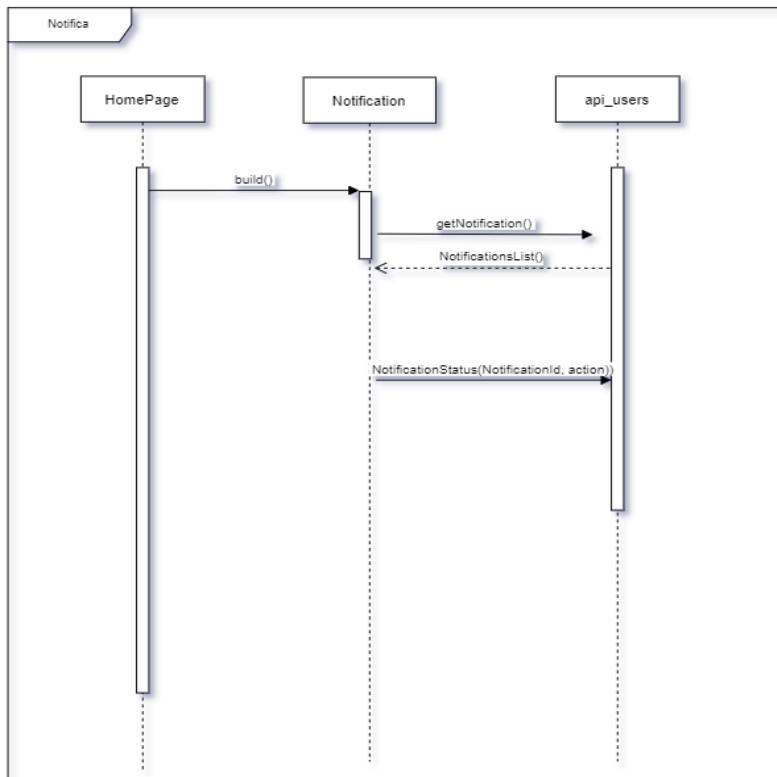
3.2 Design Class Diagram



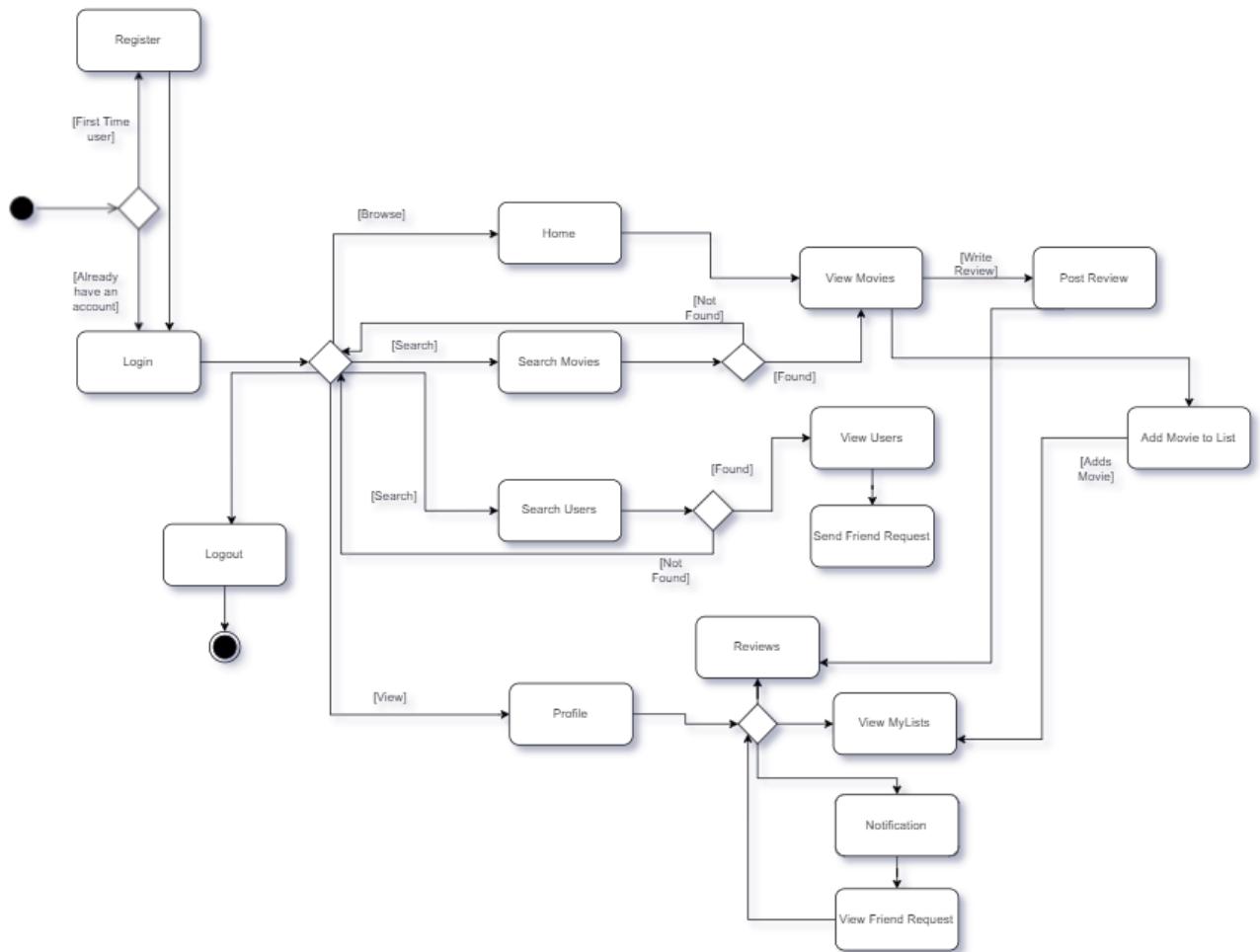
3.3 Design Sequence Diagrams







3.4 Design State Chart Diagram



3.5CRC Cards

Di seguito verranno riportate le CRC cards (Class Responsability Collaborators) usate per analizzare la collaborazione tra le classi e le loro responsabilità.

Class Name	Utente
Superclasses	
Subclasses	
Responsibilities	Collaborators
Signup into App	SignUp
Login to App	Login
Post Revisione	Reviews, Movie
Send Friend Request	FriendRequests ,Utente
View Notifications	Notification, Utente
Report Utente	Utente
View Movies	Movies
Ricerca Movie	Movies, TMDB_ Api

Class Name	Movies
Superclasses	
Subclasses	
Responsibilities	Collaborators
Browse Movie	Movies
View Movie categories in Home Page	MoviesPage
Ricerca Movie	Movies, TMDB_ Api
Write Revisione	Movies, Utente
Read Revisione	Movies, Utente

Class Name	Revisione
Superclasses	
Subclasses	
Responsibilities	Collaborators
View Revisione	Revisione, Utente

Class Name	AddToList
Superclasses	
Subclasses	
Responsibilities	Collaborators
Add Favourite movies to a preferred category	Movies, Utente, AddToList

Class Name	RicercaMovie
Superclasses	
Subclasses	
Responsibilities	Collaborators
Search Movies by Title	Movies, TMDB_ Api

Class Name	ReportController
Superclasses	
Subclasses	
Responsibilities	Collaborators

Report a Utente	Utente
-----------------	--------

Class Name	authController
Superclasses	
Subclasses	
Responsibilities	Collaborators
Validate User	Utente, Database
Create User	Utente
LoginUser	Utente, Database

Class Name	requestController
Superclasses	
Subclasses	
Responsibilities	Collaborators
Notifica utente	Utente

3.6 Testing del sistema

3.6.1 Test Planning

Il seguente capitolo tratterà la fase di testing del progetto dove ci si occuperà della fase di validazione e verifica dimostrando che tutti i requisti implementati siano correttamente funzionanti.

Test ID	1	
Descrizione	Registrazione	
Input	Goal	Result
Viene premuto il pulsante “Registrati via email”	Viene mostrato un form contenente gli input per poter procedere alla registrazione dell’utente (email,username,password)	Positivo
L’utente si registra con i dettagli richiesti corretti nel modulo.	L’utente riceve il messaggio di successo.	Positivo
Viene premuto il pulsante “Registrati” con alcuni campi del form non compilati o errati	L’utente non riesce a registrarsi correttamente	Positivo
Un utente che ha già un account utilizza la stessa email per registrarsi nuovamente	L’utente non riesce a registrarsi e riceve un messaggio di errore	Positivo

Test ID	2	
Descrizione	Login e Logout	
Input	Goal	Result
Viene premuto il pulsante “Login”	Viene mostrato un form contenente gli input per poter procedere all’accesso sulla app	Positivo
L’utente fa clic sul pulsante “Accedi” dopo aver inserito credenziali valide	L’utente accede correttamente all’applicazione reindirizzati alla pagina di benvenuto dell’app mobile	Positivo
L’utente accede con credenziali non corrispondenti o errate	All’utente viene negato l’accesso e riceve un messaggio di errore	Positivo
L’utente fa clic sul pulsante di disconnessione	L’utente è disconnesso dall’applicazione	Positivo

Test ID	3	
Descrizione	Ricerca e visualizzazione di film	
Input	Goal	Result
L'utente accede all'app	L'utente viene indirizzato alla home page.	Positivo
L'utente inserisce il nome di un film esistente nella barra di ricerca e viene premuto il pulsante di ricerca.	Vengono visualizzati tutti i film che includono il nome inserito nella barra di ricerca, nel titolo del film.	Positivo
L'utente inserisce il nome di un film inesistente nella barra di ricerca e viene premuto il pulsante di ricerca.	Non vengono visualizzati film.	Positivo
L'utente seleziona un film dai film visualizzati nella pagina	L'utente viene indirizzato a una pagina che mostra tutte le informazioni sul film selezionato.	Positivo

Test ID	4	
Descrizione	Scrittura recensione	
Input	Goal	Result
L'utente fa clic sul pulsante di revisione	L'utente è indirizzato alla pagina di revisione.	Positivo
L'utente compila tutti i campi nella pagina di revisione e scorre il cursore per valutare il film e fa clic sul pulsante di conferma.	La recensione è pubblicata.	Positivo
L'utente non riesce a compilare il campo richiesto nella pagina di revisione.	La recensione non viene pubblicata e all'utente verrà chiesto di compilare tutti i campi se desidera pubblicare la recensione.	Positivo

Test ID	5	
Descrizione	Elenco film	
Input	Goal	Result
L'utente fa clic su "Le mie liste" nella sua pagina del profilo	L'utente viene indirizzato a una pagina con diverse liste.	Positivo
L'utente fa clic su una particolare film nell'elenco.	L'utente viene indirizzato alla pagina del film.	Positivo
L'utente seleziona un film dai risultati della ricerca. Quindi fare clic sul pulsante per aggiungere il film a una lista.	Il film viene aggiunto alla lista.	Positivo

Test ID	6	
Descrizione	Cerca utenti	
Input	Goal	Result
Digitare il nome di un utente esistente nella barra di ricerca e il pulsante di ricerca viene premuto.	Uno o più utenti che hanno il nome menzionato nella barra di ricerca vengono visualizzati nei risultati della ricerca.	Positivo
Digitare il nome di un utente non esistente nella barra di ricerca e il pulsante di ricerca viene premuto.	Nessun utente è elencato.	Positivo

Test ID	7	
Descrizione	Richiesta di connessione	
Input	Goal	Result
L'utente fa clic sul pulsante di richiesta di collegamento sulla scheda dell'utente selezionato.	L'utente riceve un messaggio di richiesta di conferma per inviare la richiesta di connessione all'utente selezionato.	Positivo
L'utente conferma la richiesta di connessione da inviare all'utente selezionato.	Notifica della richiesta di connessione inviata all'utente selezionato.	Positivo
L'utente fa clic sul pulsante "annulla" quando viene richiesto di confermare la richiesta di connessione da inviare all'utente.	La richiesta di connessione non verrà inviata.	Positivo

Test ID	8	
Descrizione	Notifica	
Input	Goal	Result
L'utente fa clic sul pulsante di notifica.	L'utente può visualizzare le richieste di connessione che ha ricevuto.	Positivo
L'utente preme il pulsante "accetta" relativo alla richiesta di connessione inviata da un altro utente.	I due utenti sono collegati.	Positivo
L'utente preme il pulsante "rifiuta" relativo alla richiesta di connessione inviata da un altro utente.	La connessione viene rifiutata.	Positivo

Test ID	9	
Descrizione	Segnala utente	
Input	Goal	Result
Visualizza un utente e fai clic sul pulsante di segnalazione.	L'utente viene indirizzato a una pagina in cui viene chiesto di inserire il motivo della segnalazione dell'utente.	Positivo
L'utente fa clic sul pulsante di conferma per segnalare l'utente.	L'utente selezionato verrà segnalato all'amministratore.	Positivo

3.6.2 Unit test

```
func testShouldGetFavouritesMovies() throws {
    // MARK: - Arranging test case dependencies
    let movieListsRepo = ApiList()
    var response : [MovieList]? = nil
    var errorResponse : Error? = nil
    let expectation = self.expectation(description: "Waiting for a non empty movie list")
    // MARK: - Acting to get the expected result
    movieListsRepo.loadData(num: "1") { movieLists, error in
        // Fullfil the expectation to let the test runner
        // know that it's OK to proceed
        response = movieLists.movieLists
        errorResponse = error
        expectation.fulfill()
    }
    waitForExpectations(timeout: 1, handler: nil)
    // MARK: - Asserting that expected results are the got ones.
    XCTAssertNil(errorResponse)
    response?.forEach({ list in
        if (list.listName == "preferiti") {
            XCTAssertEqual(list.movies.count, 0)
        }
    })
}

func testShouldGetCorrectUserById() throws {
    // MARK: - Arranging test case dependencies
    let userRepo = ApiUser()
    var userResponse : Person? = nil
    let expectation = self.expectation(description: "Waiting for the exiting user")
    // MARK: - Acting to get the expected result
    userRepo.loadUserDataUser(num: "1") { user in
        // Fullfil the expectation to let the test runner
        // know that it's OK to proceed
        userResponse = user.results.first
        expectation.fulfill()
    }
    waitForExpectations(timeout: 1, handler: nil)
    // MARK: - Asserting that expected results are the got ones.
    XCTAssertEqual(userResponse?.username, "accr0")
    XCTAssertEqual(userResponse?.email, "accro@gmail.com")
}

func testRatingTextShouldBeCorrectAccordingToAverage() throws {
    // MARK: - Arranging test case stubs
    let threeStarMovie = Movie(id: 0, voteAverage: 3.49)
    let fourStarMovie = Movie(id: 0, voteAverage: 3.51)
    // MARK: - Acting to get the expected result
    let threeStarRatingText = threeStarMovie.ratingText
    let fourStarRatingText = fourStarMovie.ratingText
    // MARK: - Asserting that expected results are the got ones.
    XCTAssertEqual(threeStarRatingText, "★★★")
    XCTAssertEqual(fourStarRatingText, "★★★★")
}
```

```
func testYearTextShouldBeCorrectAccordingToRelease() throws {
    // MARK: - Arranging test case stubs
    let newMilleniumMovie = Movie(id: 0, releaseDate: "2001-01-01")
    let malformedReleaseDateMovie = Movie(id: 0, releaseDate: "notARealDate")
    // MARK: - Acting to get the expected result
    let yearText = newMilleniumMovie.yearText
    let uncorrectYearText = malformedReleaseDateMovie.yearText
    // MARK: - Asserting that expected results are the got ones.
    XCTAssertEqual(yearText, "2001")
    XCTAssertEqual(uncorrectYearText, "n/a")
}
```