

Luca Maccarini

# Relazione progetto

2021/2022

**E-voting platform**

# Indice

<b>Descrizione del problema</b>	<b>2</b>
1.1. Analisi e specifica dei requisiti	2
1.1.2 Requisiti funzionali	6
1.1.3 Requisiti non funzionali	6
Requisiti di sviluppo	6
Requisiti di sicurezza	6
1.4. Glossario	7
<b>Progettazione del sistema</b>	<b>9</b>
2.1. Diagramma dei casi d'uso	10
2.1.1 Descrizione testuale dei casi d'uso (del 3° diagramma)	16
2.2 Descrizione degli scenari	18
2.3. Diagramma delle classi	30
Modello di progetto	31
Modello di programma	34
2.3.1 Discussione dei Design Pattern utilizzati	34
2.4. Diagrammi di sequenza	39
Modifica password	47
Visualizza vincitori	48
Visualizza sessioni di voto	56
Crea sessione di voto	57
Calcola vincitore	60
Visualizza log	61
2.5. Diagrammi di attività	65
Ricerca elettore nella sessione di voto	70
Inserisci voti in presenza	71
Calcolo del vincitore	72
Visualizza log	72
2.6. Macchine di stato	73
2.7. Diagramma dei componenti	75
<b>Implementazione del sistema</b>	<b>75</b>
3.1. Diagramma delle classi a livello di programma	75
3.2. Gestione dei dati persistenti	76
3.3. Descrizione dell'interfaccia grafica	77
3.5. Specifica e verifica dei vincoli	91
3.6. Descrizione del testing	96
3.7. Note per l'installazione e l'utilizzo	99
3.7.1 Progetto	99
3.7.2 Database	99

# Note Iniziali

Tutte le immagini, eccetto quelle riguardanti la grafica (punto 3.3), presentano un'etichetta sottostante contenente il nome “Figura” seguito dal numero di quest’ultima. Questo perché **alcune immagini sono molto grosse e metterle nella relazione le ha rese poco leggibili**, quindi utilizzando la nomenclatura sopra indicata è possibile trovare la corrispondente immagine nella cartella “immagini”, inserita nella zip consegnata.

## Assignments

Ho riutilizzato il lavoro prodotto negli assignment e l’ho inserito all’interno della relazione. I diagrammi dei casi d’uso, il diagramma delle classi e le users stories sono stati poco rielaborati rispetto a quelli consegnati negli assignment.

## 1. Descrizione del problema

Il progetto richiede la realizzazione di un sistema di votazione elettronico. Il sistema che verrà realizzato dovrà essere utilizzabile sia dagli utenti che vogliono esprimere il proprio voto (**elettori**) sia dal personale (**gestori del sistema**) che si occuperà della gestione delle sessioni di voto e della creazione di account per nuovi elettori.

### 1.1. Analisi e specifica dei requisiti

Il sistema che andrò a realizzare sarà disponibile solo per gli elettori e i gestori del sistema, quindi un utente che vuole usufruire della piattaforma deve recarsi presso un seggio elettorale e richiedere, ad un gestore del sistema, la sua iscrizione alla piattaforma.

Il processo di iscrizione di un elettore prevede una prima fase di autenticazione mediante documento identificativo e codice fiscale. Dopo che il gestore del sistema si è accertato della identità dell’utente, procederà alla registrazione del nuovo account e quindi inserirà i dati personali, uno username e la e-mail dell’utente nel sistema. Quest’ultimo verificherà che l’utente non abbia già un account registrato, dopo di che creerà l’account impostando una password casuale generata dal sistema, la quale sarà inviata via e-mail all’utente.

Un elettore può votare o lasciare la scheda bianca (astenuto) ad una sessione di voto se ha il diritto per farlo. Questo è deciso dal gestore del sistema che ha creato la sessione di voto ed, all'atto della creazione, ha specificato le gli elettori aventi diritto al voto.

Il voto può essere espresso in 2 modalità: in presenza o a distanza.

Il voto in presenza prevede che un' elettore si presenti ad un seggio elettorale e si identifichi mostrando il proprio documento identificativo e codice fiscale ad un gestore del sistema. Quest'ultimo dovrà verificare l'autenticità dei documenti, verificare l'identità della persona mediante la foto sul documento, controllare se la persona ha già un account da elettore registrato nel sistema, verificare se ha diritto al voto per la sessione di voto in questione ed, infine, controllare se l'elettore non ha già espresso un voto per la sessione di voto in questione.

Il voto a distanza o voto elettronico prevede che l'utente, mediante un dispositivo collegato ad internet, si autentichi nel sistema e-voting usando il proprio username e password. Successivamente il sistema permetterà di selezionare solo le sessioni di voto dove l'elettore ha il diritto di votare; dopo averne selezionata una, l'elettore potrà esprimere il proprio voto.

Un gestore del sistema è un amministratore che può creare e gestire le sessioni di voto, visualizzare i log del sistema, controllare l'identità degli utenti ai seggi elettorali ed inserire nel sistema sw i voti espressi dagli elettori nei seggi elettorali.

Per creare una sessione di voto il gestore del sistema deve specificare:

- il nome della sessione di voto;
- una descrizione;
- la data di termine della sessione, dopo la quale la votazione verrà bloccata;
- la lista dei candidati;
- la tipologia di votazione e il modo in cui verrà calcolato il vincitore;
- gli elettori che hanno diritto al voto.

Le tipologie di voto e di estrazione del vincitore sono riportate nelle seguenti tabelle:

Tipologia di voto	Descrizione
voto ordinale	all'elettore è richiesto di ordinare i candidati (o gruppi/partiti) presenti nella scheda in base alle proprie preferenze
voto categorico	l'elettore inserisce una preferenza per un candidato
voto categorico con preferenze	l'elettore inserisce una preferenza per un gruppo/partito e ha la possibilità di indicare una o più preferenze tra i candidati del gruppo/partito selezionato (niente voto disgiunto)

referendum	Consiste in una domanda fatta all'elettorato con la quale si chiede se si sia favorevoli o contrari a un determinato quesito
------------	--

Tipologia di calcolo del vincitore	Descrizione
maggioranza	il vincitore è il candidato che ha ottenuto il maggior numero di voti
maggioranza assoluta	il vincitore è il candidato che ha ottenuto la maggioranza assoluta dei voti, cioè il 50% + 1 dei voti espressi
referendum senza quorum	si procede al conteggio dei voti indipendentemente se abbia partecipato o meno alla consultazione la maggioranza degli aventi diritto al voto
referendum con quorum	si procede al conteggio dei voti espressi solo nel caso in cui abbia partecipato alla consultazione la maggioranza degli aventi diritto al voto

La fase di scrutinio e calcolo del vincitore può essere avviata solo dopo che la data di termine della sessione di voto è passata. Calcolato il vincitore, tutti gli elettori e i gestori del sistema possono vedere chi è il vincitore della sessione di voto.

Infine il sistema deve prevedere un sistema di auditing per verificare il corretto funzionamento dell'intero sistema, così da poter tracciare e ricostruire ciò che accade internamente, rilevando possibili problemi.

## Requisiti utente

(User stories estratte dalle varie interviste fatte agli stakeholders)

- Gestore del sistema: “per la registrazione di un account, secondo le nostre politiche, è necessario che l’identità di una persone sia verificata in presenza e con l’utilizzo di un documento identificativo”;
- Utente: “Per autenticarmi a distanza vorrei una procedura semplice come l’utilizzo della mia e-mail o di uno username e di una password, inoltre in caso perdessi la password sarebbe comodo un sistema di recupero o reset di quest’ultima”;
- Elettore: “vorrei che il sistema mi facesse vedere tutte le votazioni a cui posso votare e di conseguenza esprimere il mio voto”;

- Gestore del sistema: “*tutti i gestori del sistema potranno creare delle sessioni di voto e caricare in una sessione di voto i voti fatti in presenza ai seggi elettorali*”;
- Gestore del sistema: “*nessuno può chiudere una sessione di voto perché la chiusura di una sessione di voto avverrà solo quanto il suo termine temporale scadrà*”;
- Elettore: “*è importante avere la possibilità di votare in presenza o a distanza per ogni sessione di voto, soprattutto in questo periodo in cui bisogna mantenere il distanziamento sociale*”;
- Gestore del sistema: “*se una persona vuole votare in presenza mi serve un modo per controllare se ha il diritto di voto alla sessione di voto e anche sapere se ha già votato nella sessione di voto*”;
- Gestore del sistema: “*è importante che il calcolo del vincitore avvenga solo dopo che la chiusura della sessione di voto*”;
- Gestore del sistema: “*abbiamo concordato che servirà implementare un sistema che registri tutte le azioni che avvengono sul servizio per poter analizzare che cosa accade nel sistema soprattutto in caso di problemi nelle security policy*”;
- Elettore: “*penso che sia un mio diritto quello di mantenere il mio voto segreto, sia che io lo esprima in presenza o a distanza*”;
- Gestore del sistema: “*in caso di attacco al sistema, il voto degli elettori deve rimanere assolutamente segreto insieme alle loro password*”;
- Gestore del sistema: “*l'esperienza di utilizzo della piattaforma dovrà essere facile e intuitiva*”;
- Gestore del sistema: “*il sistema software dovrà essere sempre disponibile durante tutto l'anno, sono ammesse solo 48 o 72 ore di disservizio perché date le scadenze delle sessioni di voto vogliamo essere sempre operativi per permettere agli elettori di votare*”;
- Gestore del sistema: “*saranno i gestori del sistema ad inserire i partiti e le persone che ne fanno parte*”.

## Requisiti di sistema

## 1.1.2 Requisiti funzionali

### Elettore

- autenticazione sulla piattaforma;
- modifica della password;
- reset della password;
- visualizzazione dei vincitori;
- visualizzazione delle sessioni di voto alla quale l'elettore può votare;
- esprimere il voto elettronico/cartaceo in una sessione di voto;
- registrazione in presenza con un gestore del sistema.

### Gestore del sistema

- creazione e delle sessioni di voto;
- inserimento dei voti cartacei nel sistema elettronico;
- controllare se un utente ha già espresso un voto in una sessione di voto;
- controllare l'identità delle persone che vogliono votare al seggio elettorale;
- visualizzare i log del sistema;
- registrare un nuovo elettore;
- inserire e rimuovere i candidati e i partiti nel sistema;
- registrare che un elettore ha votato in presenza.

## 1.1.3 Requisiti non funzionali

### Requisiti di sviluppo

- l'applicazione e-voting deve essere sviluppata in java + javafx per l'interfaccia utente;
- l'applicazione si deve interfacciare con un database relazionale mysql mediante driver jdbc

### Requisiti di sicurezza

- verifica degli input inseriti dall'utente mediante l'interfaccia, per assicurarsi di non elaborare input malevoli o non significativi;
- il sistema deve utilizzare i prepared statement, dove occorrono, per inviare query al database così da evitare attacchi SQL injection;
- verifica di robustezza della password per evitare attacchi di password cracking;
- il voto degli elettori deve rimanere sempre anonimo perché è diritto dell'elettore mantenere segreto il proprio voto;

- il sistema deve avere un sistema di auditing perché è importante ricostruire gli eventi di causa/effetto avvenuti nel sistema per scoprire problemi nelle security policies come, ad esempio, un utente che riesce a votare a due volte;
- le password non devono essere salvate in chiaro nel database ma verrà salvato il loro hash, perché così chiunque abbia accesso alle password degli utenti non le potrà vederle in chiaro;
- eseguire backup periodici del database per permettere il ripristino in caso di problemi gravi e limitare le perdite di dati.

## **Qualità del software da raggiungere**

- **Facilità di modifica**

Rendere il codice strutturato per identificare con facilità gli errori, realizzare una struttura che permetta di aggiungere nuove funzionalità per sviluppi futuri e che permetta di adattare il sistema software ad un'altro dominio applicativo.

- **Portabilità**

Il sistema deve funzionare su diverse piattaforme, così da permettere agli utenti con dispositivi eterogenei di utilizzare il servizio. Questa qualità è raggiunta soddisfacendo i vincoli di sviluppo (utilizzo di java).

- **Correttezza e affidabilità**

Il sistema software dovrà implementare tutte le specifiche e comportarsi come previsto da queste ultime. Dovranno essere eseguiti e documentati dei test per valutare la correttezza del sistema.

- **Riusabilità**

Rendere il codice modulare e dove possibile generico per poter riutilizzare parti del sistema per creare nuovi sistemi.

## 1.4. Glossario

<b>Termino</b>	<b>Definizione</b>
<b>Voto ordinale</b>	All'elettore è richiesto di ordinare i candidati (o gruppi/partiti) presenti nella scheda in base alle proprie preferenze.
<b>Voto categorico</b>	L'elettore inserisce una preferenza per un candidato (o gruppo/partito).
<b>Voto categorico con preferenze</b>	L'elettore inserisce una preferenza per un

	gruppo/partito e ha la possibilità di indicare una o più preferenze tra le persone del gruppo/partito selezionato (niente voto disgiunto).
<b>Referendum</b>	Consiste in una domanda fatta all'elettorato con la quale si chiede se si sia favorevoli o contrari a un determinato quesito.
<b>Maggioranza</b>	Il vincitore è il candidato che ha ottenuto il maggior numero di voti.
<b>Maggioranza assoluta</b>	Il vincitore è il candidato che ha ottenuto la maggioranza assoluta dei voti, cioè il 50% + 1 dei voti espressi.
<b>Referendum senza quorum</b>	Si procede al conteggio dei voti indipendentemente se ha partecipato o meno alla consultazione la maggioranza degli aventi diritto al voto.
<b>Referendum con quorum</b>	Si procede al conteggio dei voti espressi solo nel caso in cui abbia partecipato alla consultazione la maggioranza degli aventi diritto al voto.
<b>Utente</b>	persona non autenticata e con un account di qualsiasi tipo registrato nel sistema sw
<b>Utente A.</b>	Utente autenticato nella piattaforma sw
<b>Elettore</b>	Utente autenticato nella piattaforma sw con un account di tipo elettore
<b>Gestore del sistema</b>	Utente autenticato nella piattaforma sw con un account di tipo gestore del sistema
<b>Persona elettore</b>	Persona con un account di tipo elettore registrato nel sistema
<b>Persona elettore A.</b>	Una persona elettore che ha superato la procedura di autenticazione, in un seggio, da parte dei gestori del sistema
<b>Password</b>	Una sequenza di caratteri alfanumerici e caratteri speciali
<b>Password sicura</b>	una password contenente: - almeno una lettera maiuscola; - almeno un numero; - almeno un carattere speciale;

	e che abbia lunghezza compresa tra 8 e 16
<b>Politica</b>	esprime i diritti di un utente del sistema su una risorsa del sistema: <u>chi</u> può fare <u>cosa</u> su che <u>risorsa</u> e <u>come</u>
<b>Security policy</b>	stesso significato di Politica
<b>Hash</b>	il risultato di una funzione di hash
<b>Funzione di hash</b>	È una funzione non invertibile che mappa una stringa di lunghezza arbitraria in una stringa di lunghezza predefinita. Esistono numerosi algoritmi che realizzano funzioni hash con particolari proprietà.
<b>pre-condizioni</b>	È una condizione o predicato che deve essere sempre vero prima dell'esecuzione di una sezione di codice o prima di un'operazione.
<b>post-condizioni</b>	È una condizione o un predicato che deve essere sempre vero immediatamente dopo l'esecuzione di una sezione di codice o dopo un'operazione
<b>trigger</b>	È un evento
<b>user story</b>	Descrizione informale e in linguaggio naturale delle funzionalità o caratteristiche di un sistema software.
<b>O.T.P.</b>	One time password un codice che viene inviato all'utente mediante un canale come s.m.s. o e-mail e viene richiesto al momento di una autenticazione o registrazione.
<b>Documento identificativo</b>	Carta d'identità o patente
<b>SW</b>	software

## 2. Progettazione del sistema

A questo punto dello sviluppo progettuale si passa, in seguito a diverse letture dei requisiti, dalla definizione del diagramma dei casi d'uso e di tutti gli scenari associati ad ogni caso d'uso al diagramma delle classi a livello di progetto. Quest'ultimo contiene gli elementi principali che sono stati rilevati durante l'analisi degli scenari, dei requisiti e dei casi d'uso. Tramite raffinamento successivo, si arriva alla definizione del diagramma delle classi a livello di programma, il quale fornisce una visione esaustiva del sistema ed è direttamente mappato nel codice.

Successivamente, ho eseguito la stesura dei diagrammi comportamentali delle attività e delle macchine a stato (solo per gli oggetti stateful che hanno uno stato mutabile). Questo per poi arrivare alla stesura del diagramma delle componenti con le varie interfacce fornite/richieste.

## 2.1. Diagramma dei casi d'uso

I diagrammi dei casi d'uso vanno sviluppati per iterazioni successive, rileggendo più volte i requisiti ed interfacciandosi con il committente per verificare che i requisiti siano stati catturati correttamente. Si inizia, quindi, con un diagramma più astratto arrivando a dei diagrammi sempre più precisi e dettagliati.

Il primo diagramma prodotto dopo una sola lettura dei requisiti è il seguente:

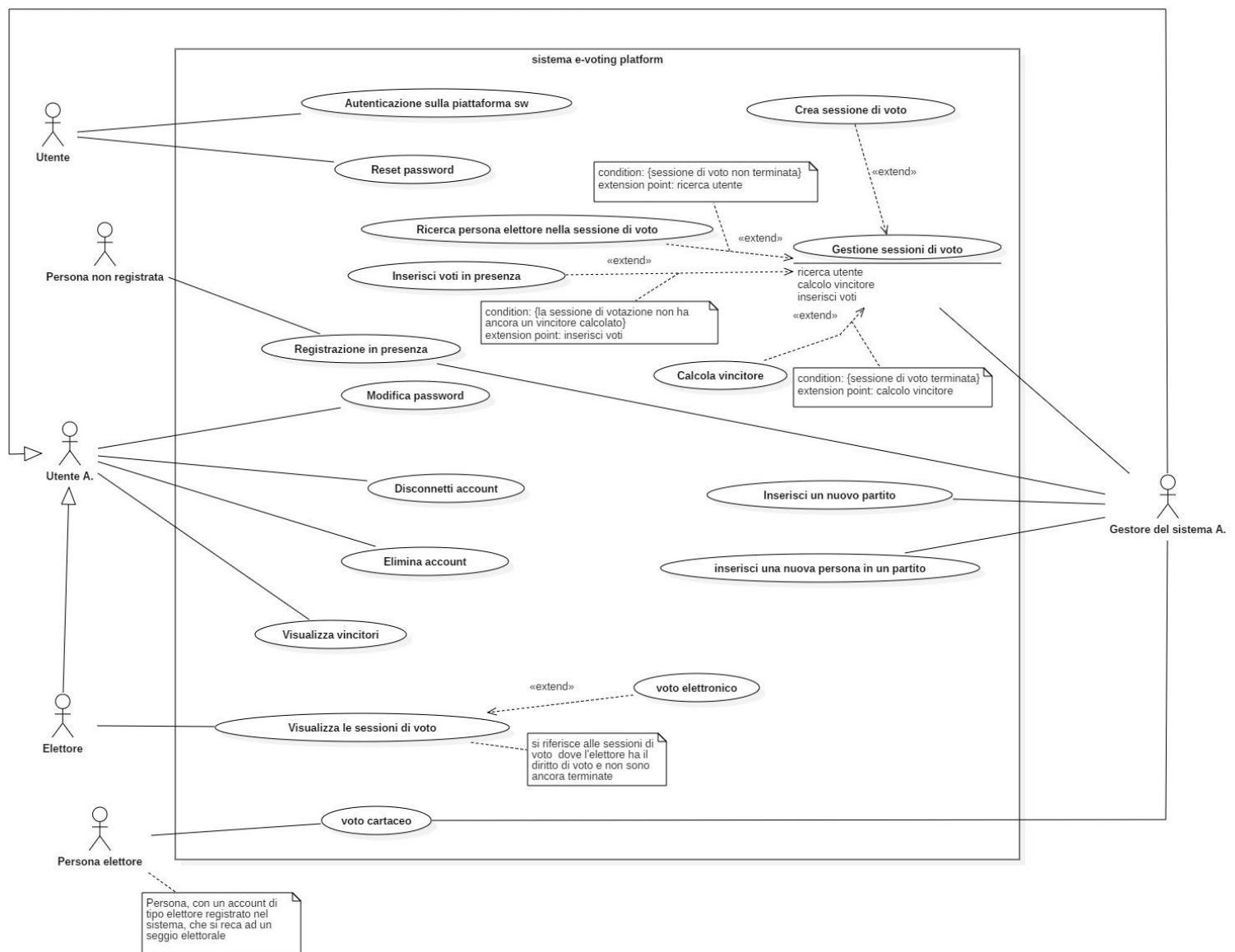


Figura 1: primo diagramma dei casi d'uso

Il diagramma è molto generico e impreciso.

Cattura i seguenti requisiti:

- Un “Utente” può autenticarsi o resettare la propria password. Una volta autenticato, l’utente diventa un utente autenticato “utente A.” che può modificare la password, eliminare il proprio account, visualizzare i vincitori o disconnettersi ritornando ad essere un utente non autenticato. Un “Utente A.” di fatto non è un vero attore in quanto, per il sistema, esistono solo gli elettori e i gestori del sistema, ma è anche vero che queste due tipologie di utenti condividono i casi d’uso di “Utente A.” Quindi, nel prossimo raffinamento del diagramma, l’attore “Utente A” diventerà un attore astratto;
- Un “Elettore”, oltre ai casi d’uso di “Utente A.”, può visualizzare le sessioni di voto dove può votare ed eventualmente esprimere il proprio voto;
- Un “Gestore del sistema”, oltre ai casi d’uso di “Utente A.”, può registrare una persona che richiede un account di elettore presso un seggio elettorale, aggiungere dei partiti nel sistema, aggiungere delle nuove persone nei vari partiti, creare una sessione di voto e gestire le sessioni di voto.  
In queste ultime opzionalmente può:
  - inserire i voti cartacei fatti al seggio, solo se abilitato dal creatore della sessione di voto;
  - ricercare se una persona elettore ha diritto di voto, se ha già votato in una sessione di voto non ancora terminata;
  - calcolare il vincitore solo se la sessione di voto è terminata;
- una “Persona non registrata” può registrarsi alla piattaforma mediante un “Gestore del sistema”;
- una “Persona elettore” può esprimere il voto cartaceo presentandosi ad un seggio elettorale, ma dal diagramma non è molto chiaro il ruolo del gestore del sistema in questo caso d’uso.

Dopo una rilettura dei requisiti, delle users stories e aver analizzato nuovamente il diagramma prodotto, ho riscontrato i seguenti problemi:

- il caso d’uso “voto cartaceo” è molto astratto e non chiaro, infatti dal diagramma non si capisce che l’elettore deve:
  - aver verificato la propria identità,
  - avere il diritto di votare
  - non aver già espresso un voto per la sessione di voto in questione.

Infine il “Gestore del sistema” deve registrare nel sistema sw che la “persona elettore” ha espresso un voto cartaceo nella sessione di voto;

- “modifica password”, “elimina account”, “disconnetti account” potrebbero essere visti come figli di un caso d’uso astratto “gestione account”, così da accomunarli sotto un unico caso d’uso generico;
- Per la piattaforma software gli utenti sono divisi in 2 categorie: elettori e gestori del sistema. Quindi ha senso che l’attore “Utente A”. sia un attore astratto;
- manca il caso d’uso “visualizza log” per quanto riguarda l’attore “gestore del sistema A.”;
- il reset della password può essere visto come un’estensione dell’autenticazione, in quanto se nella procedura di autenticazione l’utente non ricorda la password può scegliere di resettarla.

Il secondo diagramma dei casi d'uso prodotto dal raffinamento del primo:

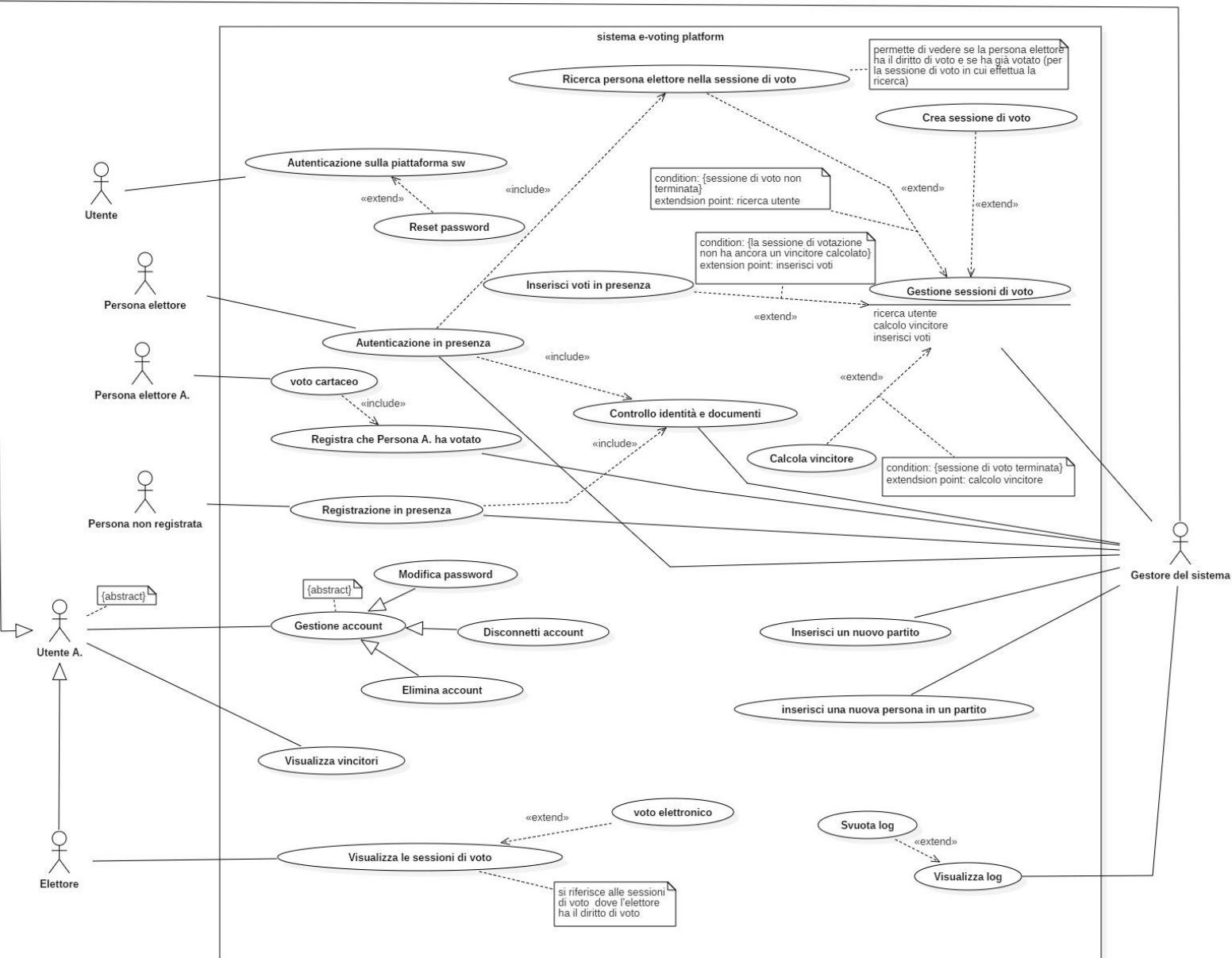


Figura 2: secondo diagramma dei casi d'uso, ottenuto dal raffinamento del primo

Il diagramma è già più preciso del precedente, in particolare:

- c'è un utilizzo più intelligente delle clausole <<include>> ed <<<extends>> . Ad esempio il caso d'uso "controllo identità e documenti", che è incluso in 2 altri casi d'uso;
- "Utente A." è un attore astratto in quanto nel sistema sw possono solo operare "Elettori" o "Gestori del sistema", nonostante ciò hanno dei casi d'uso comuni;

- “modifica account”, “disconnetti account”, “elimina Account” sono tre casi d’uso generalizzati dal caso d’uso “gestione account”. Quest’ultimo è un caso d’uso astratto perchè di fatto non è prevista una vera e propria gestione account. Questo perchè le informazioni dell’account, inserite dal gestore del sistema alla registrazione, rimangono invariate per tutta la vita dell’account;
- ho aggiunto il caso d’uso “visualizza log” per i gestori del sistema, collegato ad il caso d’uso opzionale “svuota log”;
- il caso d’uso “voto cartaceo” è associato ad un nuovo attore, ovvero “Persona elettore A.”. Questo è un elettore che vuole votare in presenza ed ha passato con successo la fase di identificazione da parte dei gestori del sistema. Invece l’attore “Persona elettore” è collegato al caso d’uso “autenticazione in presenza” che gli permette di diventare una “Persona elettore A.” se superata con successo. Inoltre, questo caso d’uso, adesso include il caso d’uso “Registra Che persona A. ha votato”, collegato all’attore “Gestore del sistema”. Questo perché bisogna registrare nella piattaforma sw che la “Persona elettore A.” ha votato, bloccando così la possibilità di votare nuovamente alla stessa sessione di voto sia in forma cartacea che elettronica;

Dopo un’ultima analisi del precedente diagramma e una rilettura dei requisiti, ho ritenuto opportuno dividere meglio il diagramma per definire i casi d’uso eseguiti sulla piattaforma sw e quelli eseguiti manualmente al seggio elettorale.

## Il terzo e ultimo diagramma dei casi d'uso:

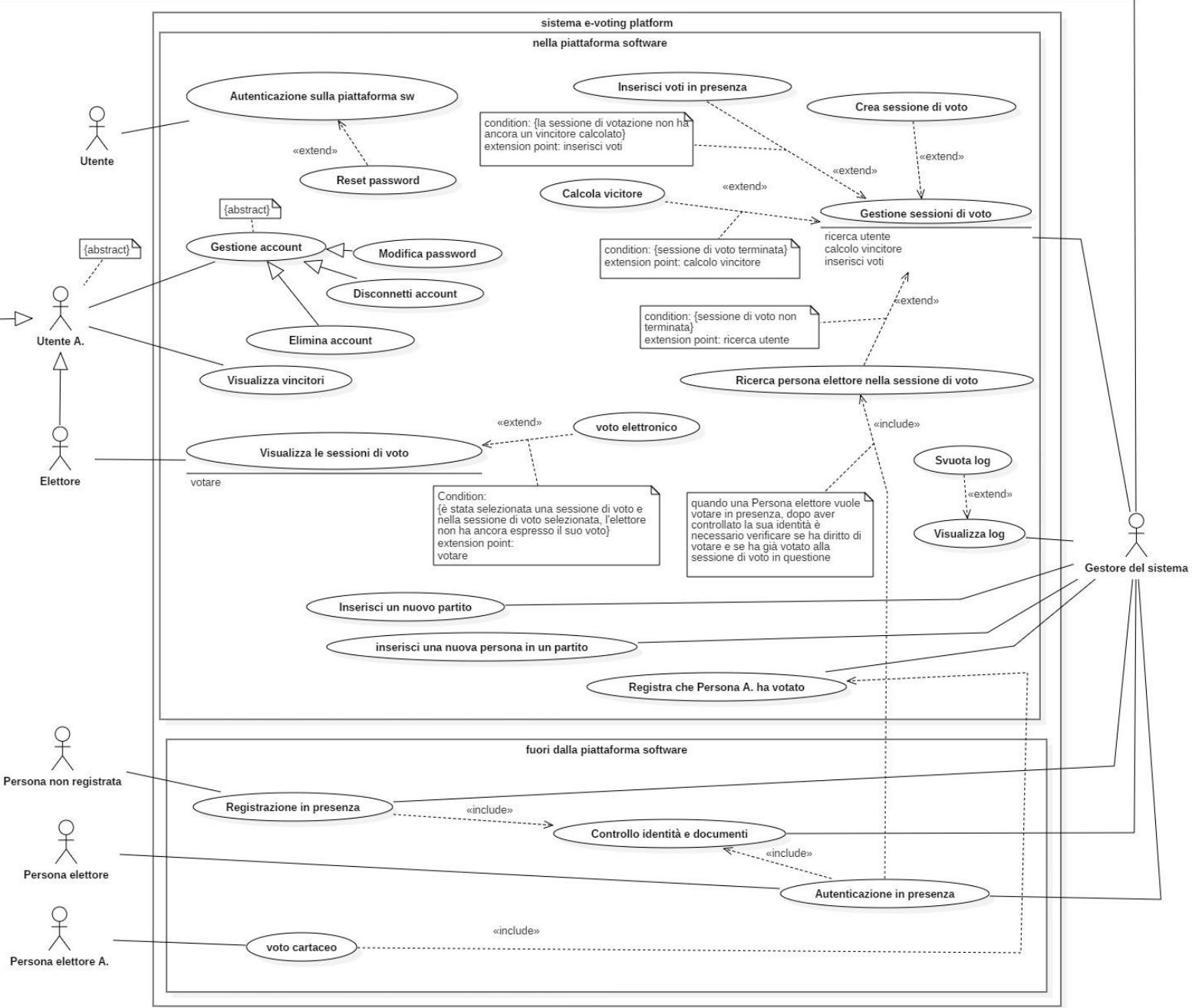


Figura 3: terzo diagramma dei casi d'uso, ottenuto dal raffinamento del secondo

Nel diagramma i casi d'uso sono divisi in due use case subjects, così ho potuto dividere i casi d'uso "eseguiti" mediante la piattaforma software e quelli che vedono una interazione manuale in presenza nei seggi elettorali. Di conseguenza sono presenti due frecce <<include>> che attraversano i due case subjects. Queste collegano un caso d'uso "eseguito" fuori dalla piattaforma ed uno effettuato sulla piattaforma sw.

Ritengo che il terzo diagramma dei casi d'uso sia il più adatto e completo per rappresentare i requisiti che ci sono stati forniti.

## 2.1.1 Descrizione testuale dei casi d'uso (*del 3° diagramma*)

- **Autenticazione sulla piattaforma sw:** permette ad un utente di autenticarsi sulla piattaforma software mediante username e password, diventando così un utente A.
- **Reset password (estensione di “Autenticazione sulla piattaforma sw”):** permette ad un utente di resettare la password in fase di autenticazione, nel caso in cui l'utente non la ricordasse. Per resettare la password è richiesto l'username associato all'account e un codice O.T.P. inviato via e-mail.
- **Gestione account (caso d'uso astratto):** accomuna i 3 casi d'uso che permettono ad un utente A. di effettuare azioni sul proprio account: “Modifica password”, “Disconnetti account”, “Elimina account”.
- **Modifica password:** permette ad un utente A. di modificare la propria password, inserendo la password attuale e la nuova password.
- **Disconnetti account:** permette ad un utente A. di disconnettersi dal proprio account tornando nella interfaccia di login. A questo punto l'utente A. ritornerà ad essere solo un utente.
- **Elimina account:** permette ad un utente autenticato di eliminare definitivamente il proprio account.
- **Visualizza vincitori:** permette ad un utente A. di visualizzare i vincitori delle sessioni di voto dove è stata fatta la procedura di calcolo del vincitore.
- **Visualizza le sessioni di voto:** permette ad un elettore di visualizzare tutte le sessioni di voto dove ha diritto di voto.
- **Voto elettronico (estensione di “Visualizza le sessioni di voto”):** permette ad un elettore di esprimere il proprio voto o di astenersi ad una delle sessioni di voto mostrate dal caso che estende, ovvero “Visualizza le sessioni di voto”.
- **Gestione sessioni di voto:** permette ad un gestore del sistema di gestire le sessioni di voto, visualizzando tutte le sessioni di voto e permettendogli di applicare delle azioni opzionali specificate nei successivi casi d'uso, i quali estendono questo caso d'uso.
- **Crea sessione di voto (estensione di “Gestione sessioni di voto”):** permette ad un gestore del sistema di creare una nuova sessione di voto, inserendo il nome, la descrizione, la data di termine, la modalità di voto, la modalità di calcolo

del vincitore, indicare chi ha il diritto di voto e, nel caso non sia un referendum, indicare i candidati.

- **Inserisci voti in presenza (estensione “Gestione sessioni di voto”):** permette ad un gestore del sistema di inserire i voti cartacei fatti in presenza.
- **Calcola vincitore (estensione di “Gestione sessioni di voto”):** permette di avviare il calcolo del vincitore se la sessione di voto in questione è terminata.
- **Ricerca persona elettore nella sessione di voto (estensione di “Gestione sessioni di voto” e incluso in “Autenticazione in presenza”):** permette ad un gestore del sistema di controllare se un utente ha il diritto di voto e se ha già votato in una determinata sessione di voto, se la sessione di voto in questione non è ancora terminata.
- **Visualizza log:** permette ai gestori del sistema di accedere allo storico dei log.
- **Svuota log:** permette ai gestori del sistema di cancellare lo storico dei log.
- **Inserisci un nuovo partito:** permette ad un gestore del sistema di inserire un nuovo partito nel sistema dei partiti, così che possano essere aggiunti come candidati nelle nuove sessioni di voto.
- **Inserisci una nuova persona in un partito:** permette ad un gestore del sistema di aggiungere una nuova persona all'interno di un partito.
- **Registra che persona A. ha votato (incluso in “voto cartaceo”):** serve a permettere ad un gestore del sistema di registrare nella piattaforma sw che una “Persona elettore A.” ha votato, così la prossima volta che verrà votare alla medesima sessione di voto non potrà farlo né in modo cartaceo né elettronico.
- **Registrazione in presenza:** permette ad un gestore del sistema di registrare una persona non registrata nel sistema software, quindi di aggiungere un nuovo elettore.
- **Voto cartaceo:** permette ad una “Persona elettore A.” di esprimere il proprio voto cartaceo nei seggi elettorali.
- **Autenticazione in presenza:** permette ad una “Persona elettore” di autenticarsi in presenza e diventare una “Persona elettore A.” così che successivamente possa esprimere un voto cartaceo.

- **Controllo identità e documenti (incluso in “Registrazione in presenza” e in “Autenticazione in presenza”):** controllo dell'identità di una persona e di regolarità dei suoi documenti da parte di un gestore del sistema.

## 2.2 Descrizione degli scenari

<b>Nome</b>	Autenticazione sulla piattaforma sw.
<b>Scopo</b>	Permettere all'utente di autenticarsi al sistema software, quindi diventare un Utente A.
<b>Attore/i</b>	Utente.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “Autenticati” nell'interfaccia di autenticazione.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. Mediante il form presente nella interfaccia di login l'utente dovrà compilare i campi “username” e “password”;</li> <li>2. una volta aver attivato il trigger nella interfaccia di login il sistema controlla username e password;</li> <li>3. se l'utente è presente nel database e la password è corretta il software mostrerà la home dell'utente.</li> </ol>
<b>Alternativa/e</b>	3a. Se non c'è corrispondenza dell'username nel database verrà visualizzato un messaggio di errore 3b. se non c'è corrispondenza della password nel database verrà visualizzato un messaggio di errore
<b>Post-condizioni</b>	L'utente è autenticato e quindi è un utente A.

<b>Nome</b>	Reset password.
<b>Scopo</b>	Permettere ad un utente di resettare la password del proprio account.
<b>Attore/i</b>	Utente.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “Password dimenticata” nell'interfaccia di autenticazione.

<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. Dopo aver attivato il trigger Il sistema chiede l'username dell'utente, se questo non è già stato inserito nella interfaccia di login;</li> <li>2. il sistema invia una mail (alla mail associata all'username inserito al punto 1) contenente un codice O.T.P;</li> <li>3. l'utente dovrà inserire il codice O.T.P. in un apposito form, mostrato dal sistema;</li> <li>4. se il codice O.T.P. è corretto il sistema genera ed imposta una nuova password sicura per l'account in questione ed invia la nuova password via e-mail all'utente;</li> </ol>
<b>Alternativa/e</b>	<ol style="list-style-type: none"> <li>2a. Se l'username non appartiene a nessun account, nessuna e-mail verrà inviata e verrà visualizzato un messaggio di errore;</li> <li>4a. se il codice O.T.P. è errato il sistema mostra un messaggio di errore "codice O.T.P. errato" e non procede con il reset della password.</li> </ol>
<b>Post-condizioni</b>	L'utente ha resettato la propria password sicura.

<b>Nome</b>	Registrazione in presenza.
<b>Scopo</b>	Permettere la creazione di un account di tipo elettore per una persona non registrata nel sistema.
<b>Attore/i</b>	Persona non registrata, Gestore del sistema A.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Presentarsi ad un seggio elettorale e chiedere di autenticarsi ad un gestore del sistema.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. Il gestore del sistema A. procede al controllo dell'identità e dei documenti;</li> <li>2. se la fase 1 è superata con successo, Il gestore del sistema registrerà la Persona non registrata inserendo in un apposito form nella interfaccia di registrazione (disponibile solo ai gestori del sistema) i seguenti dati: <ul style="list-style-type: none"> <li>- username che l'elettore vuole per l'account</li> <li>- nome;</li> <li>- cognome;</li> </ul> </li> </ol>

	<ul style="list-style-type: none"> <li>- data di nascita;</li> <li>- codice fiscale;</li> <li>- comune di nascita</li> <li>- sesso</li> <li>- identificatore della provincia di nascita</li> <li>- e-mail.</li> </ul> <p>ed infine clicca il bottone “registra elettore”</p> <ol style="list-style-type: none"> <li>3. se il codice fiscale, l'username e la e-mail inseriti al punto 2 non sono associati a nessun altro account Il sistema invierà un codice O.T.P. alla e-mail inserita al punto 1</li> <li>4. la persona non registrata comunicherà il codice O.T.P. al gestore del sistema che la sta registrando, il quale lo inserirà nell'apposito form mostrato dal software</li> <li>5. se il codice O.T.P. è corretto il sistema creerà l'account fornendo una password generata casualmente la quale verrà inviata mediante e-mail alla e-mail inserita al punto 1</li> </ol>
<b>Alternativa/e</b>	<ol style="list-style-type: none"> <li>1a. se il controllo dell'identità e dei documenti non va a buon fine, perché il documento non è autentico o la foto sul documento non corrisponde alla Persona non registrata: non si procederà alla registrazione;</li> <li>3a. dopo aver cliccato il bottone “Registra elettore” se i dati inseriti non coincidono con il codice fiscale inserito verrà mostrato un messaggio di errore;</li> <li>3b. dopo aver cliccato il bottone “Registra elettore” se l'username, la mail o il codice fiscale sono già associati ad altri account verrà mostrato un messaggio di errore.</li> <li>5a. se il codice O.T.P. non è corretto l'account non verrà creato</li> </ol>
<b>Post-condizioni</b>	Il nuovo account è stato registrato

<b>Nome</b>	Modifica della password.
<b>Scopo</b>	permettere ad un utente A. di modificare la password del proprio account.
<b>Attore/i</b>	Utente A.

<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “Modifica password” nell’interfaccia di gestione dell’account.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. L’utente A. dovrà inserire in un apposito form la password sicura attuale, la password sicura nuova e confermare la nuova password sicura;</li> <li>2. dopo aver attivato il trigger, se la password sicura attuale inserita al punto 1 è corretta, la nuova password è una password sicura ed, infine, la conferma della nuova password coincide, il sistema procederà a modificare la vecchia password sicura in quella nuova. Inoltre, il sistema notificherà via e-mail che la password è stata modificata, senza ovviamente mostrare la nuova password.</li> </ol>
<b>Alternativa/e</b>	<ol style="list-style-type: none"> <li>2a. Se l’utente A. inserisce una password attuale errata il sistema mostrerà un messaggio di errore;</li> <li>2b. se la password attuale è corretta ma la password nuova inserita non è una password sicura, viene mostrato un messaggio di errore;</li> <li>2c. se la password attuale è corretta e la password inserita è una password sicura, ma la nuova password inserita non coincide con la conferma della nuova password verrà mostrato un messaggio di errore.</li> </ol>
<b>Post-condizioni</b>	l’utente A. ha modificato la propria password.

<b>Nome</b>	Disconnessione account.
<b>Scopo</b>	permettere ad un utente A. di disconnettersi dal proprio account.
<b>Attore/i</b>	Utente A.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “Disconnessione” nell’interfaccia di gestione dell’account.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. Una volta attivato il trigger il sistema disconnetterà l’utente;</li> <li>2. il sistema mostrerà l’interfaccia di autenticazione.</li> </ol>

<b>Alternativa/e</b>	
<b>Post-condizioni</b>	l'utente A. si è disconnesso dal sistema.

<b>Nome</b>	Elimina account.
<b>Scopo</b>	permettere ad un Elettore di eliminare il proprio account.
<b>Attore/i</b>	Elettore
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “Elimina account” nell’interfaccia di gestione dell’account.
<b>Descrizione (sequenza eventi)</b>	1. il sistema elimina l’account e mostrerà l’interfaccia di autenticazione.
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	L’elettore ha eliminato il proprio account.

<b>Nome</b>	Visualizza vincitori.
<b>Scopo</b>	Visualizzare i vincitori delle sessioni di voto dove è stato calcolato il vincitore.
<b>Attore/i</b>	Utente A.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “Visualizza vincitori” nella Home.
<b>Descrizione (sequenza eventi)</b>	1. Dopo aver attivato il trigger verranno mostrati all’utente A. tutti i vincitori delle sessioni di voto delle quali è stato calcolato il vincitore, mostrando per primi i vincitori delle sessioni di voto terminate con la data più recente.
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	l’utente A. sta visualizzando i vincitori.

<b>Nome</b>	Voto elettronico ( <i>a distanza</i> ).
<b>Scopo</b>	Votare ad una sessione di voto mediante il sistema software.

<b>Attore/i</b>	Elettore.
<b>Pre-condizioni</b>	L'elettore deve aver selezionato una sessione di voto per la quale non ha già espresso un voto ed ha il diritto di voto.
<b>Trigger</b>	Click del bottone “Vota” nell'interfaccia della sessione di voto.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. L'elettore deve inserire le proprie preferenze di voto seguendo la modalità di votazione della sessione di voto;</li> <li>2. l'elettore attiva il trigger e se ha compilato correttamente le preferenze di voto, quest'ultimo verrà registrato.</li> </ol>
<b>Alternativa/e</b>	2a. Se l'elettore non ha selezionato correttamente le preferenze di voto, tra cui la facoltà di astenersi, verrà visualizzato un messaggio di errore ed il voto non sarà registrato.
<b>Post-condizioni</b>	L' elettore ha votato.

<b>Nome</b>	Visualizza le sessioni di voto.
<b>Scopo</b>	Visualizzare tutte le sessioni di voto non ancora terminate dove l'elettore ha diritto di votare.
<b>Attore/i</b>	Elettore.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “Visualizza sessioni di voto” nella Home.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. Il sistema mostra all'elettore tutte le sessioni di voto non ancora terminate per le quali ha il diritto di voto, indicando per ogni sessione di voto se l'elettore ha già espresso il proprio voto.</li> </ol>
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	L'elettore sta visualizzando le sessioni di voto per le quali ha diritto di voto.

<b>Nome</b>	Autenticazione in presenza.
<b>Scopo</b>	Permettere ad una Persona elettore di autenticarsi in

	presenza ad un seggio elettorale diventando così una Persona elettore A.
<b>Attore/i</b>	Persona elettore, Gestore del sistema
<b>Pre-condizioni</b>	presentarsi ad un seggio elettorale e avere un documento identificativo
<b>Trigger</b>	Presentarsi ad una sessione di voto presso un seggio elettorale.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. La persona elettore deve consegnare al gestore del sistema predisposto per l'autenticazione un documento identificativo e la tessera sanitaria o il codice fiscale;</li> <li>2. il gestore del sistema A. procede al controllo dell'identità e del documento;</li> <li>3. se il punto 2 è superato con successo, allora il gestore del sistema ricerca l'utente nella sessione di voto per controllare se la persona elettore ha il diritto di votare e se ha già votato a quella sessione di voto;</li> <li>4. se il punto 3 è superato con successo, la Persona elettore si è autenticata e gli viene consegnata la scheda elettorale.</li> </ol>
<b>Alternativa/e</b>	<p>2a. Se il controllo dell'identità e del documento ha esito negativo la persona elettore non verrà autenticata;</p> <p>3a. la persona elettore non ha diritto di votare non verrà autenticata per la sessione di voto in questione;</p> <p>3b. la persona elettore ha già votato e quindi non verrà autenticata per la sessione di voto in questione.</p>
<b>Post-condizioni</b>	La persona elettore ora è una Persona elettore A.

<b>Nome</b>	Voto cartaceo ( <i>in presenza</i> ).
<b>Scopo</b>	Votare ad una sessione di voto presso un seggio elettorale e senza l'utilizzo del sistema software.
<b>Attore/i</b>	Persona elettore A., Gestore del sistema
<b>Pre-condizioni</b>	

<b>Trigger</b>	Consegna della scheda di voto.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. il gestore del sistema notifica sulla piattaforma sw che nella sessione di voto in questione la persona elettore A. sta procedendo al voto. Così se la persona elettore A. proverà nuovamente a votare alla stessa sessione di voto, verrà rilevato che il voto è già stato espresso e non potrà procedere;</li> <li>2. la persona elettore A. compila la scheda di voto o lascia la scheda bianca;</li> <li>3. la persona elettore A. imbuca la scheda di voto nell'apposito contenitore e lascia il seggio elettorale.</li> </ol>
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	La persona elettore A. ha votato.

<b>Nome</b>	Controllo identità e controllo documenti.
<b>Scopo</b>	Accertarsi dell'identità della persona controllata.
<b>Attore/i</b>	Gestore del sistema, Persona non registrata oppure Persona elettore (sarà il soggetto controllato).
<b>Pre-condizioni</b>	
<b>Trigger</b>	Quando viene richiesto il controllo dai casi d'uso "Registrazione in presenza" o "Autenticazione in presenza".
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. Il gestore del sistema chiede al soggetto controllato un documento identificativo e la tessera sanitaria o codice fiscale;</li> <li>2. il gestore controlla la legittimità dei documenti e mediante la foto sul documento si accerta che appartengano realmente al soggetto controllato;</li> <li>3. se il punto 2 è superato con successo il controllo ha esito positivo.</li> </ol>
<b>Alternativa/e</b>	2a. se i documenti sono falsi, scaduti o non appartengono alla persona che li ha mostrati il controllo ha esito negativo ed il soggetto controllato non verrà identificato
<b>Post-condizioni</b>	Il soggetto controllato è stato identificato, quindi:

	<ul style="list-style-type: none"> <li>- una persona non registrata può procedere alla registrazione del proprio account;</li> <li>- una Persona elettore diventa una Persona elettore A.</li> </ul>
--	--

<b>Nome</b>	Crea sessione di voto.
<b>Scopo</b>	Creare una nuova votazione o referendum.
<b>Attore/i</b>	Gestore del sistema.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “Crea sessione di voto” nell’interfaccia di gestione delle sessioni di voto.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. il gestore del sistema deve compilare un form inserendo le seguenti informazioni: <ul style="list-style-type: none"> <li>- nome della votazione;</li> <li>- descrizione della votazione / quesito del referendum;</li> <li>- modalità di votazione;</li> <li>- modalità di calcolo del vincitore;</li> <li>- selezione dei partiti candidati (solo se la modalità di voto non è referendum);</li> <li>- selezione degli elettori che hanno diritto di voto;</li> <li>- data e ora in cui si chiuderà la sessione di voto;</li> </ul> </li> <li>2. se tutti i dati inseriti sono corretti verrà creata la sessione di voto.</li> </ol>
<b>Alternativa/e</b>	2a. Se i dati non sono corretti verrà visualizzato un messaggio di errore e la sessione di voto non verrà creata.
<b>Post-condizioni</b>	Il gestore del sistema ha creato una sessione di voto.

<b>Nome</b>	Ricerca persona elettore nella sessione di voto
<b>Scopo</b>	permette ad un gestore del sistema di controllare se una persona elettore ha il diritto di votare e se ha già votato nella sessione di voto selezionata.
<b>Attore/i</b>	Gestore del sistema.
<b>Pre-condizioni</b>	Aver selezionato la sessione di voto, la quale non deve essere terminata.

<b>Trigger</b>	Click del bottone "controlla" nell'interfaccia di gestione della sessione di voto.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. il gestore del sistema inserisce l'username della persona elettore che si vuole cercare in un apposito form nell'interfaccia di "gestione della sessione di voto" ed attiva il trigger;</li> <li>2. il sistema cercherà l'elettore nella sessione di voto e mostrerà se l'elettore ha il diritto di voto e se ha già espresso il proprio voto.</li> </ol>
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	il gestore del sistema sa se la Persona elettore cercata ha diritto di voto e se ha già votato nella sessione di voto in questione.

<b>Nome</b>	Inserisci voti in presenza.
<b>Scopo</b>	Inserire i voti fatti in presenza ai seggi elettorali nel sistema software.
<b>Attore/i</b>	Gestore del sistema.
<b>Pre-condizioni</b>	Aver selezionato la sessione di voto, la quale non deve avere il vincitore già calcolato.
<b>Trigger</b>	Click del bottone "Carica i voti in presenza" nell'interfaccia di gestione della sessione di voto.
<b>Descrizione (sequenza eventi)</b>	<ol style="list-style-type: none"> <li>1. Dopo aver attivato il trigger verrà mostrata la medesima interfaccia di voto degli elettori;</li> <li>2. il gestore del sistema potrà registrare tutti i voti effettuati al seggio, uno per volta.</li> </ol>
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	I voti espressi in presenza sono stati caricati nella sessione di voto.

<b>Nome</b>	Calcola vincitore.
<b>Scopo</b>	calcolare il vincitore della sessione di voto
<b>Attore/i</b>	Gestore del sistema.

<b>Pre-condizioni</b>	Aver selezionato la sessione di voto, la quale deve essere terminata.
<b>Trigger</b>	Click del bottone “calcola vincitore” nell’interfaccia di gestione della sessione di voto.
<b>Descrizione (sequenza eventi)</b>	1. il sistema calcola il vincitore
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	Il vincitore è stato calcolato.

<b>Nome</b>	Gestione sessioni di voto
<b>Scopo</b>	visualizzare e selezionare le sessioni di voto, per poi utilizzare i comportamenti opzionali (<<extends>>)
<b>Attore/i</b>	Gestore del sistema.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone "aggiorna" nella interfaccia di gestione delle sessioni di voto
<b>Descrizione (sequenza eventi)</b>	1. il sistema mostra tutte le sessioni dove non è stato calcolato il vincitore; 2. il gestore del sistema seleziona una delle sessioni di voto.
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	Il gestore del sistema ha selezionato una sessione di voto dove non è ancora stato calcolato il gestore del sistema.

<b>Nome</b>	Visualizza log
<b>Scopo</b>	visualizzare i log prodotti dal sistema
<b>Attore/i</b>	Gestore del sistema.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “aggiorna” nell’interfaccia dei log
<b>Descrizione (sequenza eventi)</b>	1. il sistema mostra tutti i log prodotti dal sistema.

<b>Alternativa/e</b>	
<b>Post-condizioni</b>	Il gestore del sistema sta visualizzando i log prodotti dal sistema.

<b>Nome</b>	Svuota log
<b>Scopo</b>	Eliminare i log registrati sul sistema
<b>Attore/i</b>	Gestore del sistema.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone "elimina log" nell'interfaccia dei log
<b>Descrizione (sequenza eventi)</b>	1. il sistema elimina tutti i log.
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	Il gestore del sistema ha eliminato i log del sistema.

<b>Nome</b>	Inserisci un nuovo partito
<b>Scopo</b>	Inserire un nuovo partito nel sistema
<b>Attore/i</b>	Gestore del sistema.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone "aggiungi partito" nell'interfaccia di gestione dei partiti.
<b>Descrizione (sequenza eventi)</b>	1. indicare nel form apposito il nome, la descrizione del partito ed il nome e cognome della prima persona che fa parte del partito
<b>Alternativa/e</b>	1a. se il nome del partito coincide con quello di un altro partito già registrato nel sistema, il partito non sarà aggiunto e verrà mostrato un messaggio di errore.  1b. se i dati inseriti non sono corretti verrà mostrato un messaggio di errore.
<b>Post-condizioni</b>	il gestore del sistema ha aggiunto un partito nuovo ed una nuova persona che ne fa parte.

<b>Nome</b>	Inserisci una nuova persona in un partito
<b>Scopo</b>	inserire una nuova persona in un partito esistente
<b>Attore/i</b>	Gestore del sistema.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “inserisci persona” nell’interfaccia di gestione dei partiti
<b>Descrizione (sequenza eventi)</b>	1. indicare nel form apposito nome, cognome e partito di appartenenza della nuova persona da aggiungere ed attivare il trigger
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	il gestore del sistema ha aggiunto una nuova persona ad un partito esistente

<b>Nome</b>	Registra che persona elettore A. ha votato
<b>Scopo</b>	registrare nel sistema sw che una persona elettore A. ha votato
<b>Attore/i</b>	Gestore del sistema, persona elettore A.
<b>Pre-condizioni</b>	
<b>Trigger</b>	Click del bottone “registra” dell’interfaccia della sessione di voto
<b>Descrizione (sequenza eventi)</b>	1. indicare nel form apposito l’username dell’ account della persone elettore A. ed attivare il trigger
<b>Alternativa/e</b>	
<b>Post-condizioni</b>	il gestore del sistema ha registrato nel sistema sw che la persona elettore A. ha votato

## 2.3. Diagramma delle classi

Questa fase prevede la realizzazione di vari diagrammi delle classi partendo da diagrammi astratti a livello di progetto fino ad arrivare alla produzione di un diagramma delle classi a livello di programma, perciò molto preciso e direttamente mappato dal codice che ho successivamente prodotto.

Anche in questa fase ho proceduto per step e per raffinamento successivo così da garantire una migliore precisione e tracciabilità del sistema.

## Modello di progetto

Per disegnare il primo diagramma delle classi a livello di progetto ho eseguito l'analisi nome-verbo del testo fornito. Questo mi ha permesso di identificare le principali classi (nomi) e di assegnargli le prime responsabilità (verbi).

Riporto di seguito il diagramma prodotto.

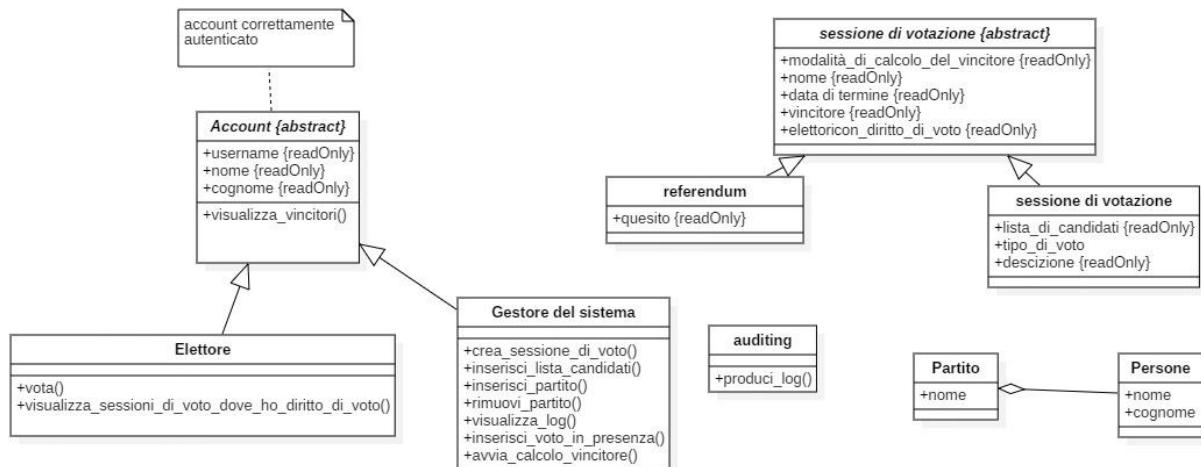


Figura 4: primo diagramma delle classi

Il diagramma è molto scarno poiché non sono presenti i tipi degli attributi, i parametri dei metodi e non sono riportate le associazioni, tranne l'aggregazione tra partito e persone. Questo diagramma mi è servito solo per iniziare a pensare alle principali classi del mio progetto e alle loro principali responsabilità.

Successivamente, ho iniziato a pensare a come queste classi avrebbero collaborato e così ho rielaborato il precedente diagramma rendendolo molto più completo e preciso. Il diagramma che è riportato in figura 5 è stato prodotto come la consegna di un Assignment (leggermente rielaborato) e, di preciso, è stato prodotto precedentemente alla spiegazione dei design pattern. Ho deciso di inserirlo nella relazione perché è stato il diagramma che ha fatto da base per la realizzazione del diagramma delle classi a livello di programma.

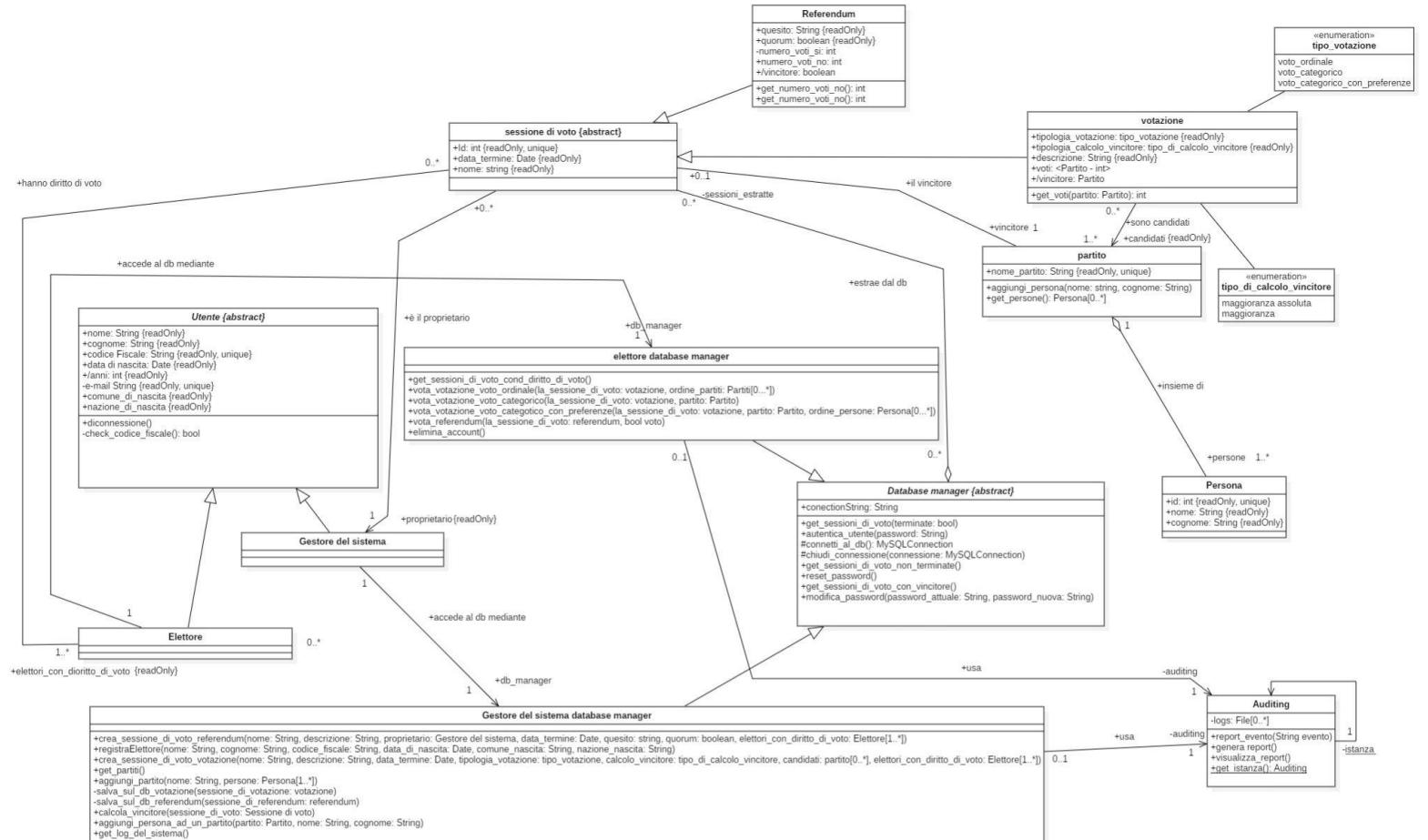


Figura 5: secondo diagramma delle classi

**per una migliore visione ricordo che ho allegato l'immagine nella cartella "immagini", è il file Figura 5**

Questo diagramma mostra già maggiormente l'architettura ad alto livello del sistema. Pertanto realizzarlo ha richiesto molto impegno poiché non è stato facile assegnare le responsabilità in modo tale da mantenere un'alta coesione e un basso accoppiamento tra le classi, senza conoscere i vari design pattern. Infatti, dopo le lezioni su questo argomento li ho studiati e quindi introdotti.

In realtà, involontariamente, ho utilizzato il design pattern GRASP Pure fabrication, nel quale le classi database manager sono dei pure fabrication.

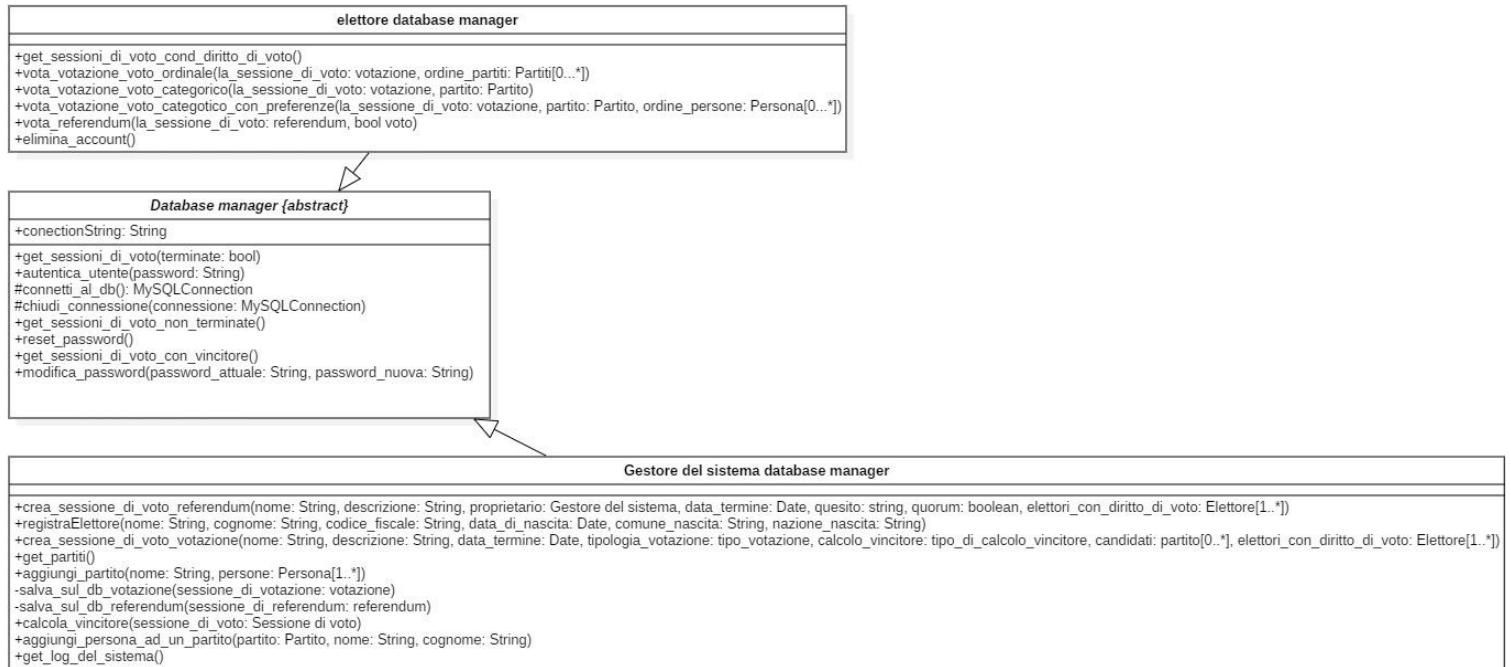


Figura 6: pure fabrication

Quest'ultime classi sono dei pure fabrication poiché sono state create per contenere azioni ed informazioni strettamente correlate alla gestione del Database. Ciò mi ha permesso di mantenere una maggiore coesione tra le classi che ho creato, perché non sarebbe stato corretto assegnare la responsabilità di connettersi al database ed eseguire le query direttamente alla classe `elettore` o alla classe `gestore del sistema`. Quindi tutte le azioni che il `gestore del sistema` deve eseguire ed interessano il database sono racchiuse in “`gestore del sistema database manager`” e la classe “`gestore del sistema`” ha un riferimento ad una istanza di “`gestore del sistema database manager`”. Nello stesso modo, la classe “`elettore`” accede al database con la classe “`elettore database manager`”.

In seguito queste classi verranno modificate per introdurre il design pattern DAO.

Il diagramma, tuttavia, è ancora astratto ed alcune classi andrebbero rielaborate per essere specializzate maggiormente. Quindi bisogna aggiungere i tipi ai parametri e ai valori restituiti dai metodi dove mancano ed anche sistemare le associazioni tra le varie classi.

A questo punto ho deciso di inserire il diagramma delle classi a livello di programma, così da esplicitare come saranno composte le varie classi del programma finale. Inoltre, questo mi ha permesso di procedere alla realizzazione dei diagrammi di sequenza mantenendo un’alta tracciabilità con il diagramma delle classi a livello di programma ed anche con il codice prodotto. Tuttavia nella fase di implementazione del sistema e stesura del codice, in caso di un’eventuale modifica di un aspetto

progettuale, ho dovuto modificare nuovamente anche il diagramma delle classi a livello di programma ed i relativi diagrammi di sequenza.

## Modello di programma

Dopo ulteriori rielaborazioni del precedente diagramma delle classi ed aver applicato i vari design pattern studiati, sono arrivato alla produzione del diagramma delle classi a livello di programma, riportato di seguito nella figura 7.

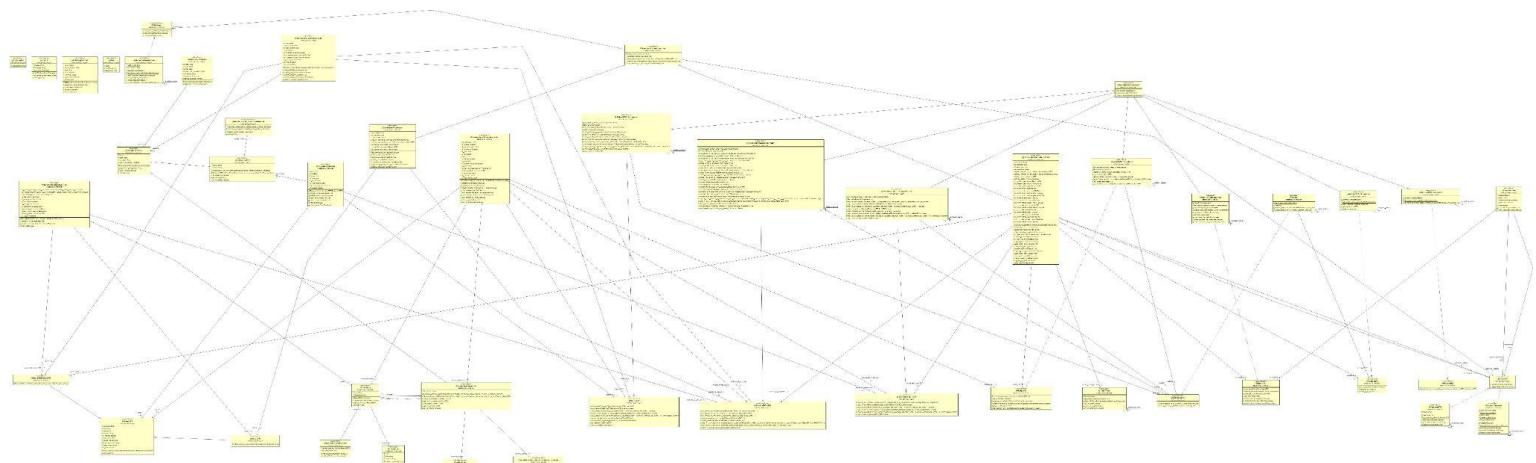


Figura 7: diagramma delle classi definitivo, a livello di programma

**per una migliore visione ricordo che ho allegato l'immagine nella cartella "immagini", è il file Figura 7**

Per garantire una migliore leggibilità, mi sono avvalso dell'utilizzo di un programma diverso da starUML. Questo poiché si creavano troppe sovrapposizioni tra le numerose classi ed associazioni.

### 2.3.1 Discussione dei Design Pattern utilizzati

Nel mio progetto sono utilizzati i seguenti design pattern:  
MVC, Singleton, DAO, Strategy, Observer.

#### DAO

Mi sono avvalso dell'utilizzo del design pattern DAO per gestire tutte le operazioni che prevedono un'interazione con il Database. Questo perché realizzando delle interfacce che si interpongono tra il Database e la logica della applicazione, il codice può essere basato sui metodi messi a disposizione da queste interfacce e non essere quindi direttamente dipendente dal database, il quale può subire modifiche strutturali o addirittura essere sostituito nel tempo.

Il vero lavoro di interazione con il Database è effettuato dalle classi che implementano le varie interfacce DAO e si preoccupano di costruire le varie query

per inserire dati nel database, per estrarli e trasformarli in oggetti DTO, poi utilizzati dalle altre classi presenti nel progetto. Le classi che implementano le interfacce DAO sono figlie della classe “DAOImplementationBasics” ed ereditano i metodi per connettersi e chiudere la connessione con il Database. Quindi questa classe contiene la stringa di connessione con il Database.

All'interno del mio progetto le interfacce DAO sono utilizzate **principalmente** per:

- il salvataggio dei voti (ElettoreDAO);
- la registrazione delle sessioni di voto e dei voti espressi al seggio elettorale (GestoreDelSistemaDAO);
- il calcolo del vincitore e scaricare le sessioni di voto / partiti / persone dei partiti (SessioneDiVotoDAO, PartitoDAO, PersonaDAO);
- registrare i log sul database (AuditingDAO).

Inoltre, per una migliore chiarezza dei ruoli assunti dalle classi all'interno ho adottato la seguente nomenclatura:

- <nome>DAO: interfaccia DAO;
- <nome>DAOImplementation: la classe che implementa l'interfaccia DAO;
- <nome>DTO: le classi DTO.

Riporto di seguito una figura contenente due interfacce DAO:

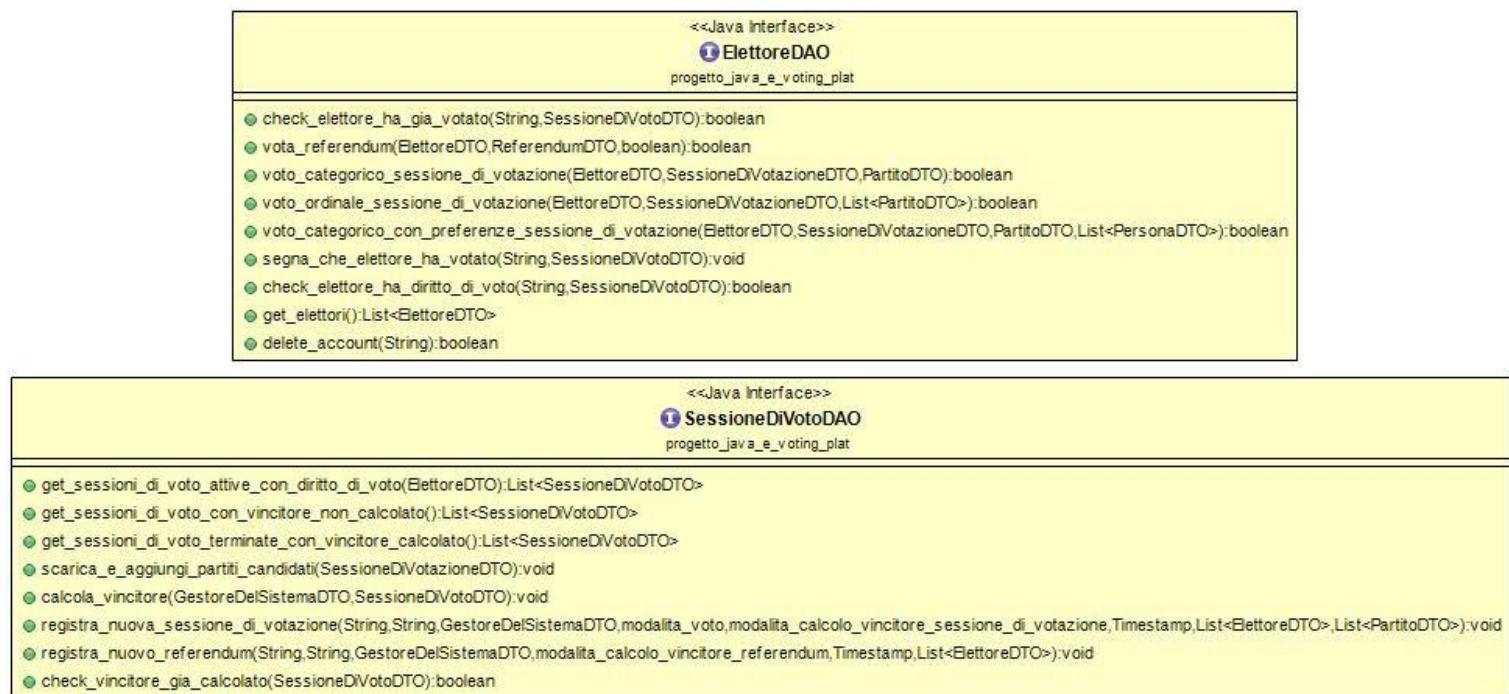


Figura 8: esempio di interfacce DAO

## Singleton

Ho usato il design pattern Singleton ogni qual volta che ritenevo inutile la creazione di più istanze di una classe, vincolando una sola istanza condivisa a runtime, quindi l'ho utilizzato nelle classi che contengono uno stato minimale, se non nullo, e che

vengono utilizzate solo per i metodi che mettono a disposizione. In ognuna di queste classi ho messo un attributo statico privato chiamato “singleton\_instance()” contenente la singola istanza della classe e il metodo pubblico “get\_singleton\_instance()”, il quale restituisce l’istanza solamente se questa esiste, altrimenti la crea (accadrà la prima volta che il metodo sarà chiamato).

Esempio di classi a cui ho applicato il design pattern singleton:

- la classe “mailSenderImplementation”, i quali metodi sono utilizzati per inviare e-mail agli utenti;
- la classe “AccountPassword”, i quali metodi sono usati per generare una password alla registrazione e per controllare se una password è sicura;
- in tutte le classi che implementano un’interfaccia DAO poiché queste sono provviste di uno stato minimale ed immutabile che non modifica il comportamento dei metodi, ma solo di metodi che agiscono sul database.

Inoltre la singola applicazione non gestisce più account contemporaneamente, ma un elettore o un gestore loggato nel sistema per volta. Quindi creare più istanze delle classi DAO avrebbe poco senso.

In seguito è riportata una figura contenente una classe “ElettoreDAOImplementation”, la quale implementa l’interfaccia Elettore DAO mostrata in figura 8 e utilizza il design pattern singleton.

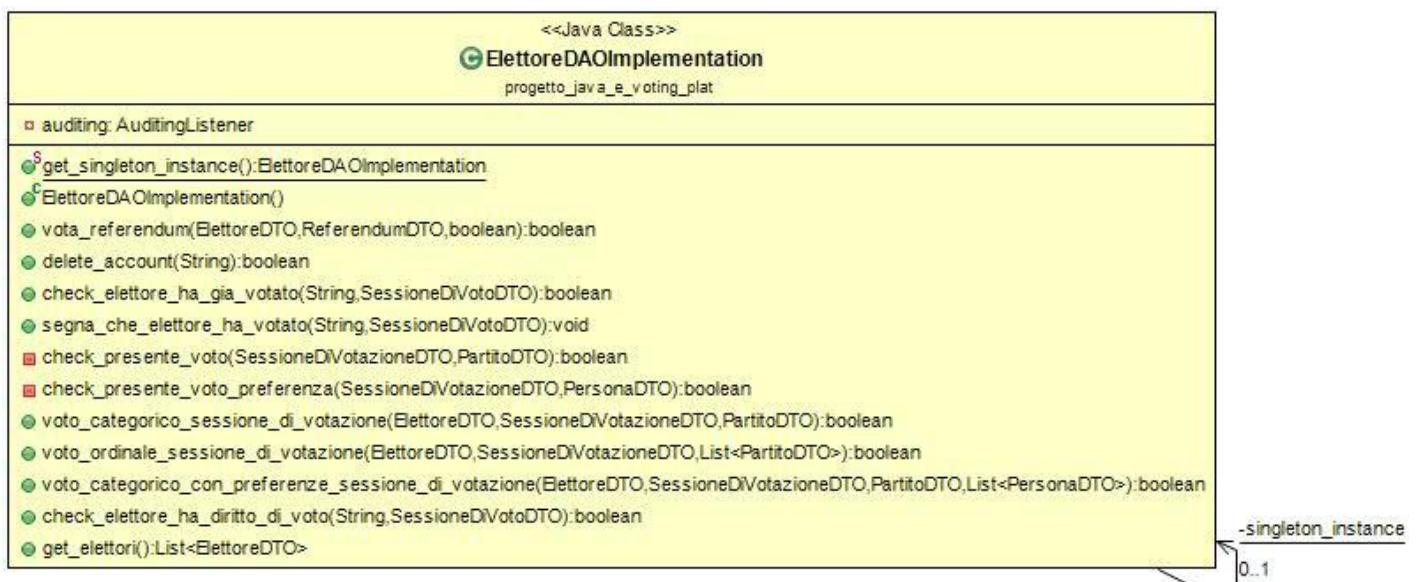


Figura 9: esempio di singleton

## MVC

Il design pattern MVC è utile per gestire l’interazione della parte grafica e le classi del programma. Sono presenti 3 componenti: la view, che è la GUI ovvero l’interfaccia utente creata mediante scene builder, il quale produce i file fxml; il Model, ovvero le classi che contengono la logica e i dati dell’applicazione; infine, il Controller che contiene negli attributi i vari elementi della GUI dalla quale può estrarre gli input dell’utente e i vari metodi richiamati dagli eventi generati dall’utente mediante la GUI.

Un esempio di uso di questo pattern: quando un utente vota ad una votazione clicca sul pulsante “vota”; a questo punto, la classe “VotoSessioneDiVotazioneController” reagisce a questo evento e registra il voto nel database utilizzando le classi del modello, infine, blocca la possibilità di esprimere un altro voto disabilita il pulsante “vota” nell’interfaccia grafica.

Per identificare meglio le classi controller nel mio progetto, queste sono chiamate con la seguente nomenclatura: <nome>Controller.

Riporto di seguito la classe “VotoSessioneDiVotazioneController”.



Figura 10: esempio di controller

## Strategy

Mi sono avvalso dell'uso di questo design pattern per gestire diverse possibili politiche di generazione e controllo di password, in quanto in alcuni momenti vi era la necessità di controllare o generare semplici codici O.T.P. ed, in altri casi, una password sicura per l'account. In questo modo, anche seguendo il design pattern GRASP protected variations, ho inserito un'interfaccia stabile "Password" e diverse implementazioni per varie strategie di controllo e generazione di password. Questo comportamento è pensato anche per possibili sviluppi futuri dove, se cambierà ancora la strategia di creazione della password o dei codici O.T.P., potrà essere aggiunta estendendo l'interfaccia comune "Password" a quelle già esistenti ed implementate nel progetto.

Riporto di seguito l'interfaccia Password e le due strategie di generazione e controllo della password.

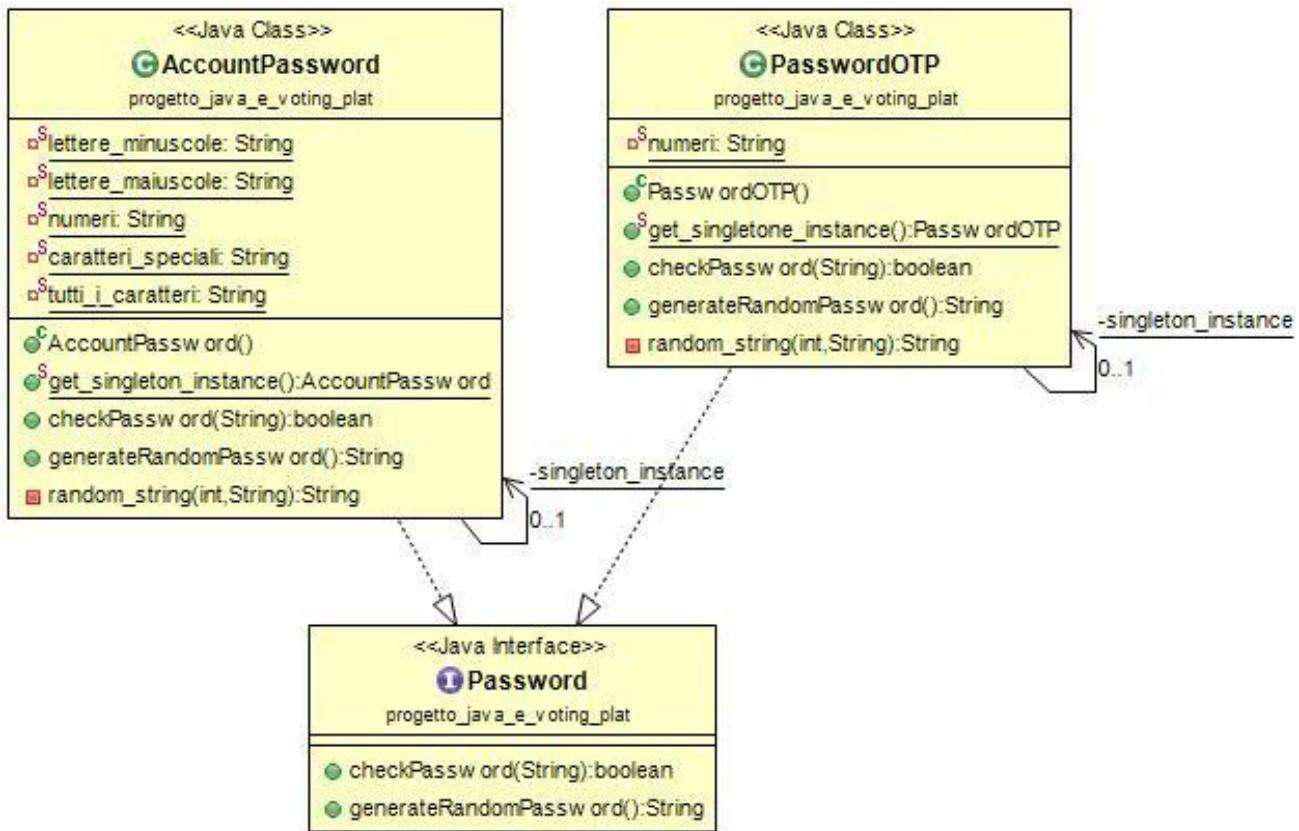


Figura 11: esempio di strategy

## Observer (Publish - Subscribe)

Ho utilizzato questo design pattern per realizzare il sistema di auditing, rendendo dei publisher tutte quelle classi che nei loro metodi eseguivano delle operazioni che dovevano essere riportate nei log. Queste sono alcune classi chiamate "<nome>DAOImplementation", le quali pubblicano un evento ogni qual volta vengono eseguite azioni importanti che interessano il database come, per esempio, la registrazione di un nuovo elettore, la creazione di una sessione di voto, ecc.

La classe subscribe è la classe "Auditing" che gestisce l'auditing e che implementa l'interfaccia subscriber "AuditingListener". Questa contiene i metodi utilizzati dai publisher per pubblicare un evento al subscriber. Nel progetto le classi publisher non registrano dinamicamente più subscriber perché, di fatto, contengono staticamente un solo subscriber al quale inviare tutti gli eventi da registrare nei log.

Riporto di seguito l'interfaccia del subscriber "AuditingListener", la classe Auditing che implementa l'interfaccia e qualche classe publisher.

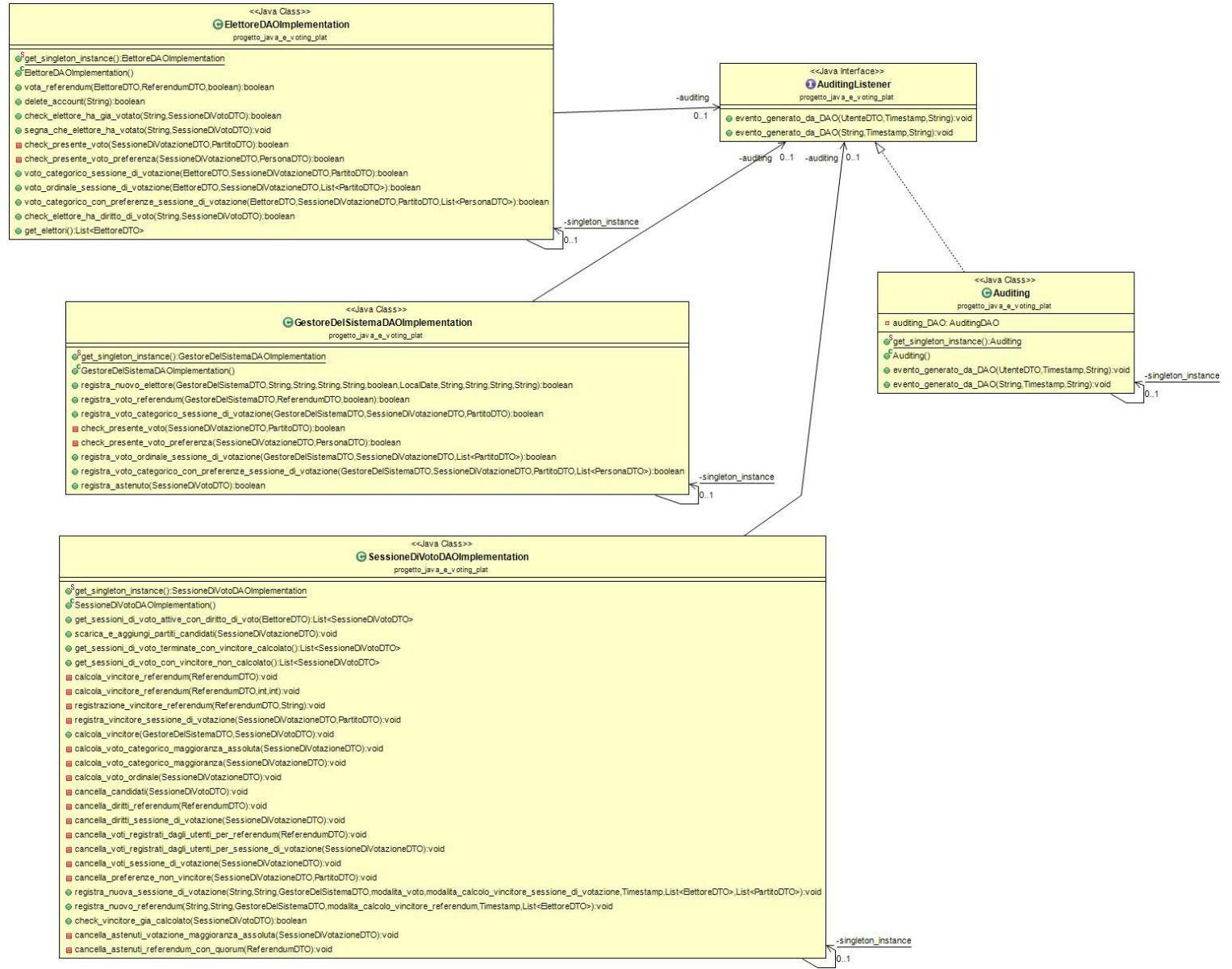


Figura 12: subscriber e qualche publisher

## 2.4. Diagrammi di sequenza

I diagrammi di sequenza sono una rappresentazione grafica degli scenari dei casi d'uso e permettono una modellazione degli scenari più agevole e chiara.

Prendendo come riferimento gli scenari che ho riportato sopra, ho tralasciato la rappresentazione di alcuni di questi per evitare di soffermarmi troppo su scenari secondari, riducendo la focalizzazione su quelli più importanti. Sono consapevole del fatto che, durante la stesura di un eventuale lavoro, questo dovrà essere valutato successivamente da un potenziale cliente, il quale deve ricevere un documento completo di diagrammi di sequenza rappresentanti tutti gli scenari.

Gli scenari che non ho riportato come diagrammi di sequenza sono:

- “Disconnessione account”, “Elimina account”: perché di minor importanza;
- “Controllo identità e controllo documenti”: perché è un controllo fatto manualmente da un gestore del sistema e quindi “eseguito” al di fuori della piattaforma sw;
- “Autenticazione in presenza”: perché è “eseguito” in parte al di fuori della piattaforma sw e la parte “eseguita” sulla piattaforma sw è coperta dallo scenario “Ricerca persona elettore nella sessione di voto” del quale ho prodotto il diagramma di sequenza;
- “Voto cartaceo (in presenza)”: perché è “eseguito” in parte al di fuori della piattaforma sw e la parte “eseguita” sulla piattaforma sw è coperta dallo scenario “registra che persona elettore A. ha votato” e "Inserisci voti in presenza" dei quali ho prodotto il diagramma di sequenza;

Per garantire una migliore leggibilità mi sono avvalso dell'utilizzo di un programma diverso da starUml.

Riporto e commento i diagrammi di sequenza.

## Autenticazione sulla piattaforma sw

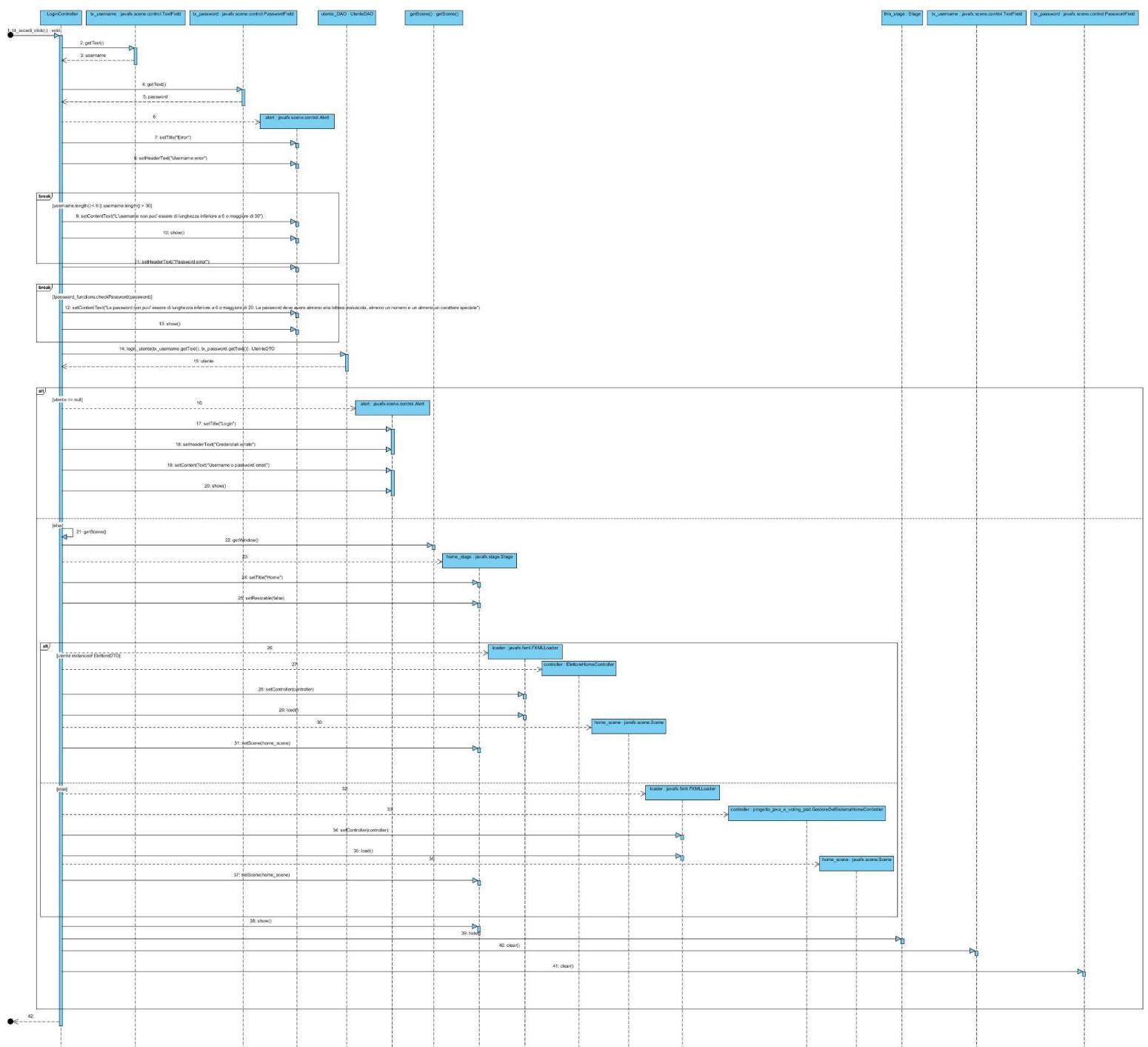


Figura 13: diagramma di sequenza scenario “Autenticazione sulla piattaforma sw”

La richiesta di autenticazione è gestita da un’istanza della classe “LoginController”, la quale estrae l’username e la password dell’utente rispettivamente dagli oggetti “tx\_username” e “tx\_password”, chiamando il metodo “getText()”. Successivamente i due input estratti vengono controllati: l’username deve avere una lunghezza compresa tra 6 e 30 caratteri, mentre la password viene controllata mediante il metodo “checkPassword()” dell’oggetto “password\_functions”, il quale è di tipo

apparente “Password” (interfaccia strategy) e di tipo effettivo “AccountPassword” (strategia utilizzata). In seguito avviene il login vero e proprio. Infatti, viene chiamato il metodo “login\_utente()” dell’oggetto “utente.DAO”, il quale controllerà nel database se è presente un utente con lo stesso username e hash della password. Se quest’ultimi vengono trovati dal metodo “login\_utente()”, ritorna un oggetto con tipo apparente “UtenteDTO” chiamato “utente”, altrimenti ritorna “null”. Infine, viene controllato il valore dell’oggetto utente:

- se è null viene mostrato un messaggio di errore;
- se non è null ed il suo tipo effettivo (instanceof) è “ElettoreDTO”, viene nascosta l’interfaccia di login e viene mostrata la home dell’utente “elettore” che si è appena autenticato;
- se non è null ed il suo tipo effettivo (instanceof) è “GestoreDelSistemaDTO”, viene nascosta l’interfaccia di login e viene mostrata la home dell’utente “gestore del sistema” che si è appena autenticato.

## Reset password

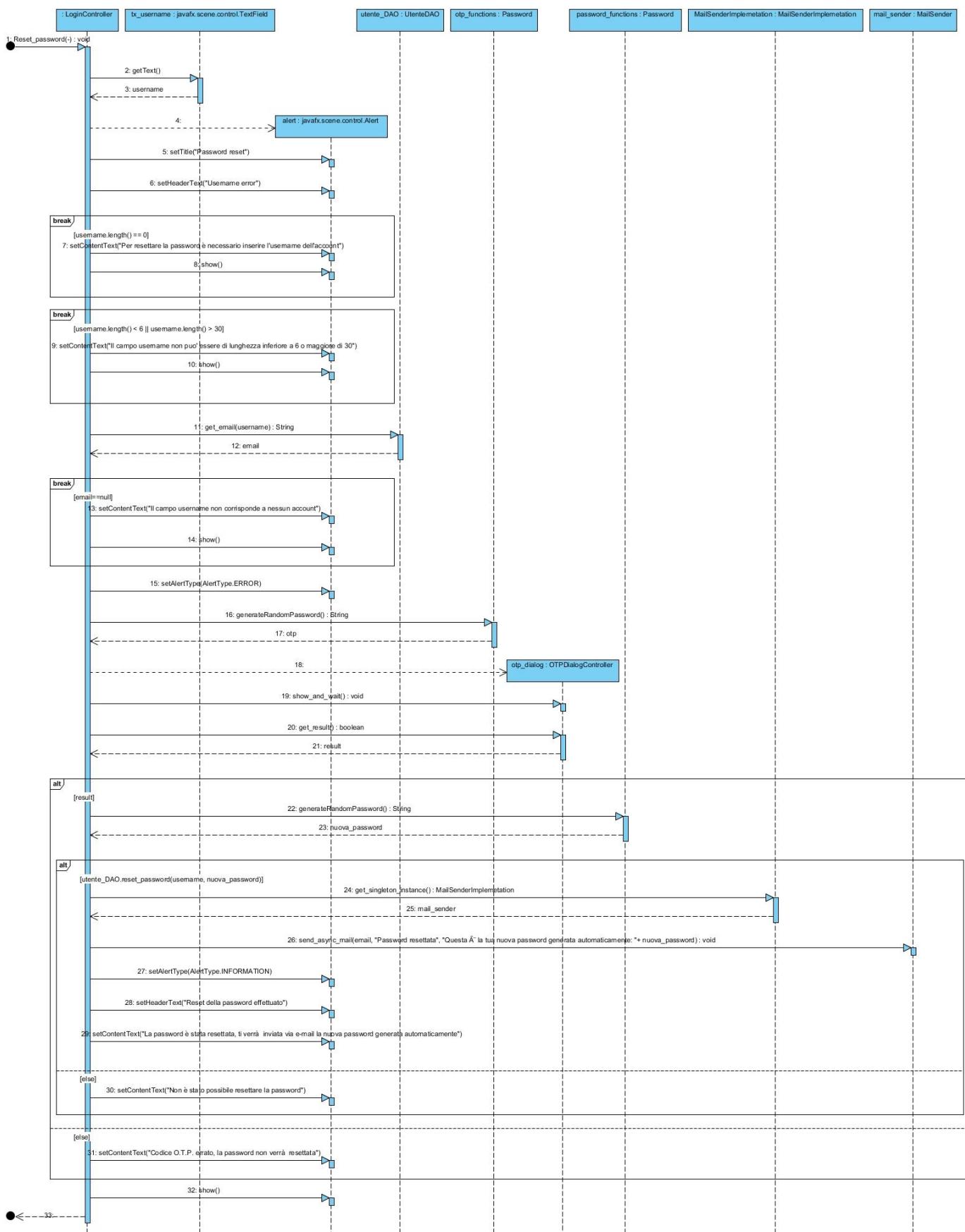


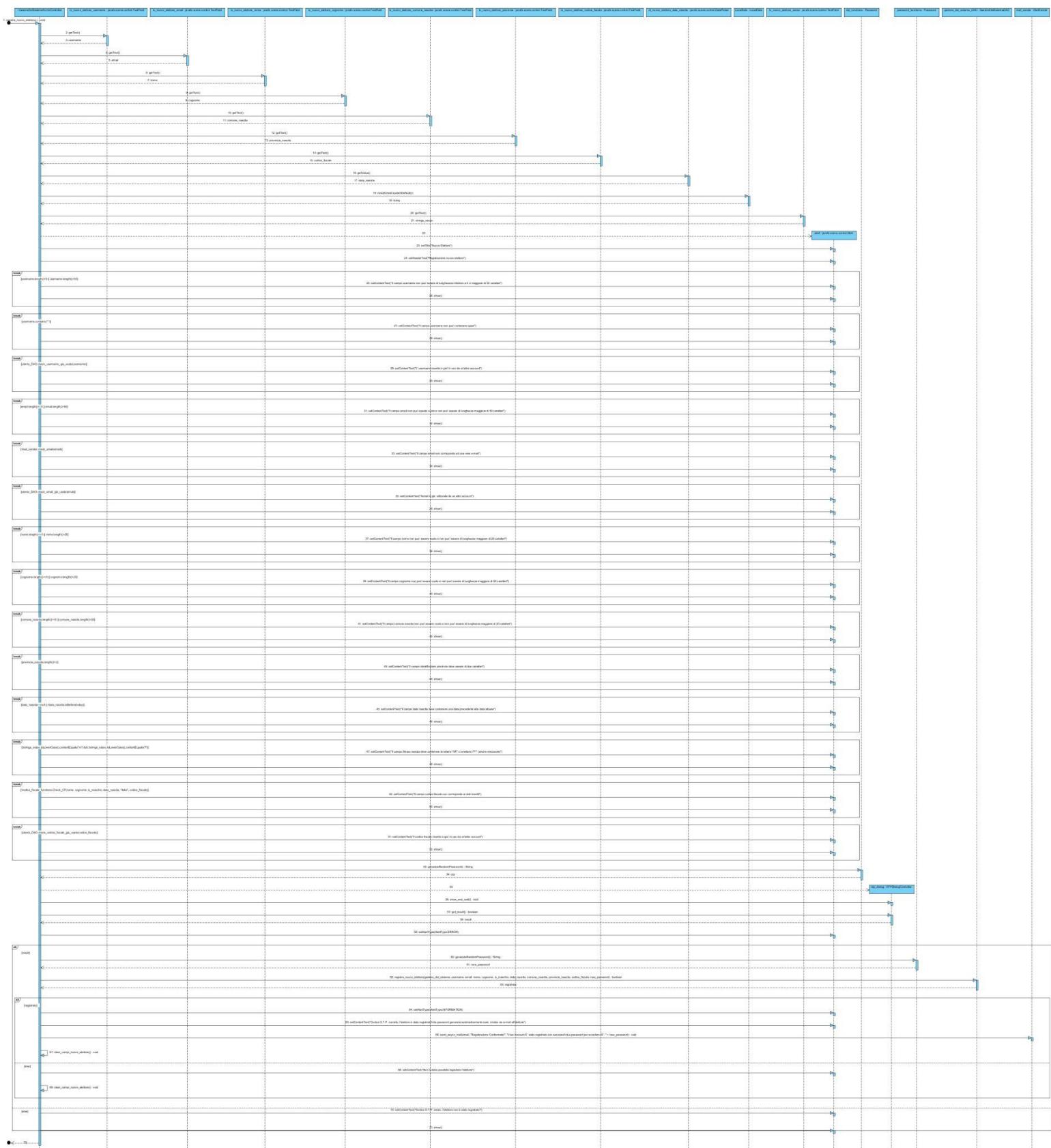
Figura 14: diagramma di sequenza scenario “Reset password”

La richiesta di reset della password è gestita da un’istanza della classe “LoginController”, la quale, come nel precedente diagramma di sequenza, estrae l’username e lo controlla. Se quest’ultimo è corretto, viene estratta dal database l’e-mail associata all’account tramite il metodo “get\_email()” dell’oggetto “utente.DAO”, mentre se l’username non è presente nel database l’email avrà il valore null e quindi verrà mostrato un messaggio di errore. Se la email estratta non è null, viene generato un codice O.T.P. mediante l’oggetto “otp\_functions”, il quale è di tipo apparente “Password” (interfaccia strategy) e di tipo effettivo “PasswordOTP” (strategia utilizzata).

Successivamente viene istanziato l’oggetto “otp\_dialog”, al quale viene passato il codice O.T.P. e l’e-mail precedentemente estratta. Quest’ultimo si occuperà di inviare il codice O.T.P. all’e-mail, questo permetterà all’utente di copiarlo dalla propria posta elettronica ed inserirlo nell’apposito form.

L’oggetto “otp\_dialog” controllerà che coincida con quello inviato ed in seguito, l’oggetto “LoginController”, mediante il metodo “get\_result()” dell’oggetto “otp\_dialog”, potrà capire se il processo di verifica del codice O.T.P. è andato a buon fine oppure no. Nel secondo caso verrà mostrato un messaggio di errore e non si procederà, mentre nel primo caso si procederà al vero e proprio reset della password. Quindi, viene generata una password nuova per l’account tramite il metodo “generateRandomPassword()” dell’oggetto “password\_functions” di tipo apparente “Password” e tipo effettivo “AccountPassword” (strategia utilizzata). Infine viene resettata la password mediante il metodo “reset\_password()” dell’oggetto “utente.DAO”. Pertanto, se il reset della password sarà andato a buon fine, verrà mostrato all’utente un messaggio di avvenuto reset della password e la nuova password sarà inviata mediante e-mail all’utente con il metodo “send\_async\_mail()” dell’oggetto “mail\_sender”, altrimenti verrà mostrato un messaggio di errore.

## **Registrazione in presenza**



*Figura 15: diagramma di sequenza scenario “Registrazione in presenza”*

La richiesta di registrazione di un nuovo elettore è gestita da un'istanza della classe “GestoreDelSistemaHomeController”. Come nei precedenti scenari, vengono estratti e controllati i dati di input, i quali sono: username, email, nome, cognome, comune\_nascita, provincia\_nascita, codice\_fiscale, data\_nascita, stringa\_sesso.

**Nota:** il controllo del codice fiscale è eseguito dal medesimo codice **consegnato negli assignments**, nel diagramma il controllo è eseguito mediante il metodo “check\_CF()” dell’oggetto “codice\_fiscale functions”.

Dopo la fase di controllo, viene verificato che la email appartenga veramente alla persona che vuole registrarsi mediante la medesima procedura di codice O.T.P. mostrata nel precedente diagramma. Se la procedura del codice O.T.P. non avviene con successo, verrà mostrato un messaggio di errore e non si procederà alla registrazione. Nel caso invece che essa venga superata, il nuovo elettore viene registrato mediante il metodo “registra\_nuovo\_elettore()” dell’ oggetto “gestore\_del\_sistema.DAO”. Se quest’ultimo metodo restituisce “true” viene mostrato un messaggio di avvenuta registrazione e la nuova password viene inviata via email all’elettore, altrimenti viene mostrato un messaggio di errore.

## Modifica password

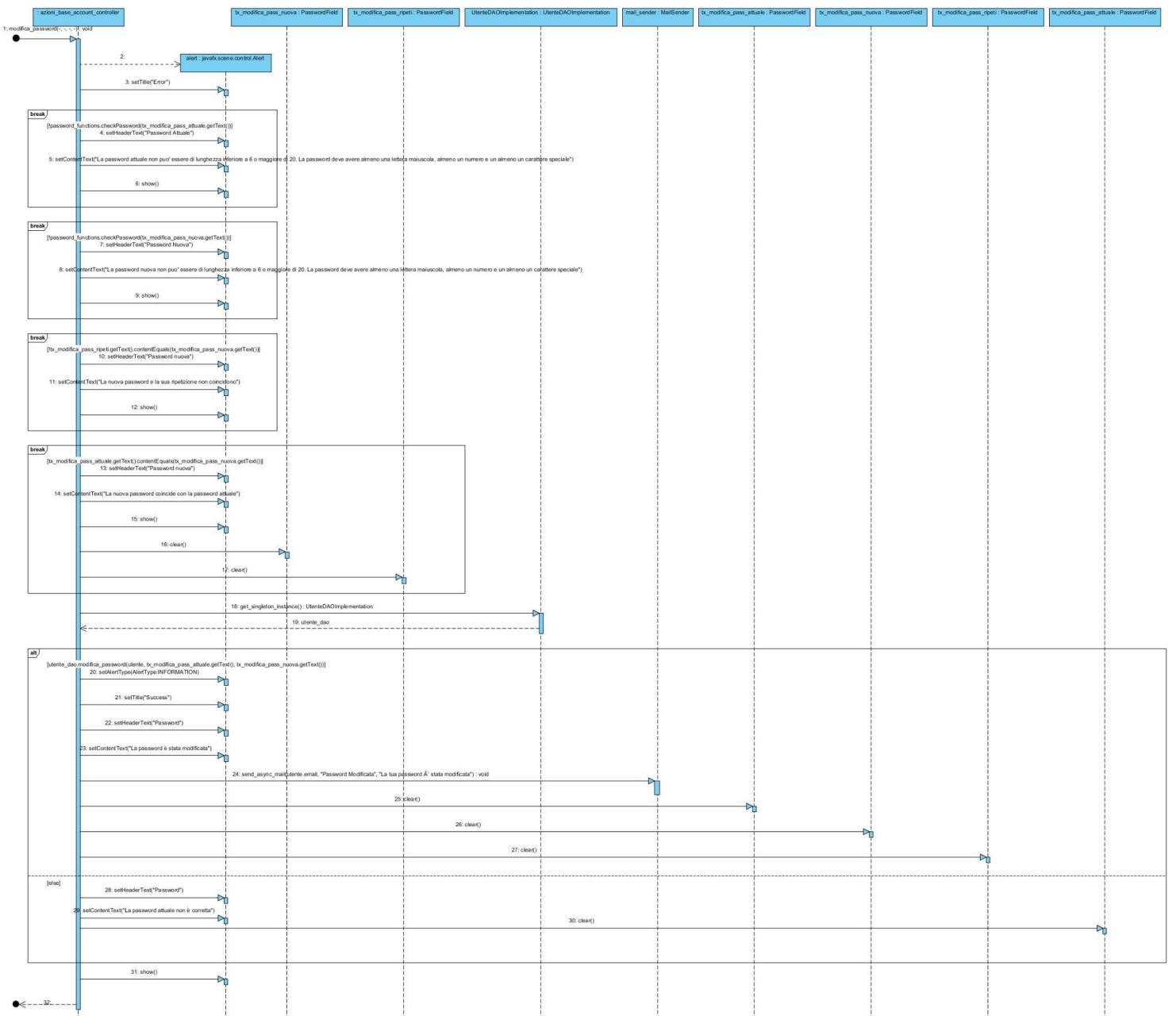


Figura 16: diagramma di sequenza scenario "modifica password"

La richiesta di modifica della password è gestita da un'istanza della classe “`azioni_base_account_controller`” e, come nei precedenti scenari, vengono estratti e controllati i dati di input, i quali sono: password attuale, password nuova e conferma della password nuova.

Inizialmente si controlla che il formato delle password sia corretto tramite l'oggetto “`password_functions`” di tipo apparente “`Password`” e tipo effettivo “`AccountPassword`” (strategia), il cui metodo “`checkPassword`” controlla che il formato sia quello di una password sicura. Dopo il superamento del controllo del formato, viene controllato che la nuova password e la conferma della nuova password coincidano.

A questo punto, mediante il metodo “get\_singleton\_instance()” della classe “UtenteDAOImplementation”, viene restituito l’oggetto “utente\_dao”. Di quest’ultimo oggetto viene chiamato il metodo “modifica\_password()”, il quale restituisce true o false.

Se il risultato è true, viene inviata una e-mail all’elettore dove gli viene comunicato che la sua password è stata modificata e viene mostrato un messaggio di successo. Se il risultato è false, invece, viene mostrato un messaggio di errore e la password non viene modificata.

## Visualizza vincitori

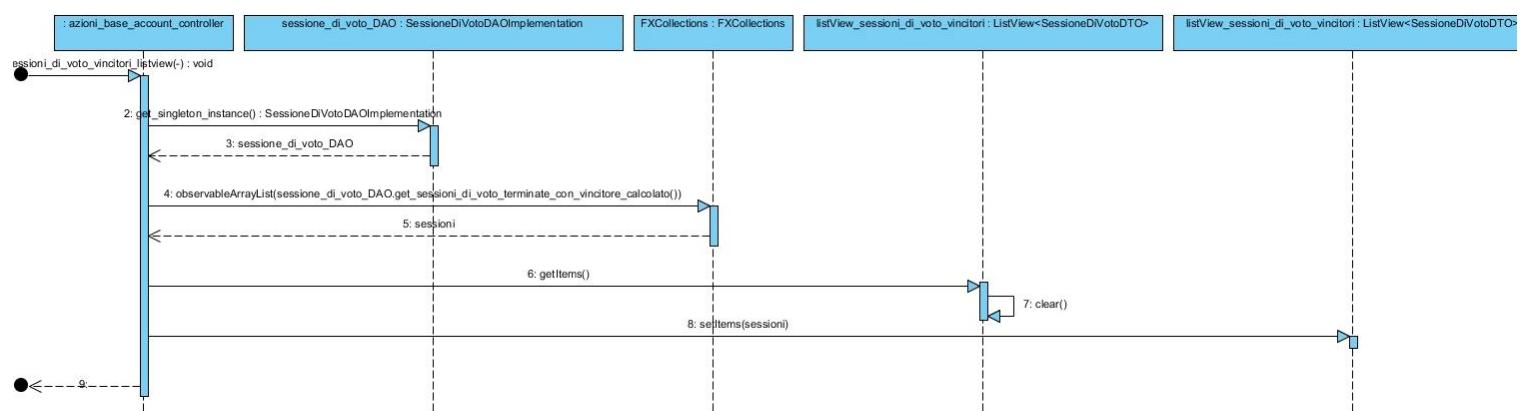


Figura 17: diagramma di sequenza scenario “visualizza vincitori”

La richiesta di visualizzare i vincitori è gestita da un’istanza della classe “azioni\_base\_account\_controller”, che estrae la lista delle sessioni di voto con vincitore mediante il metodo “get\_sessions\_di\_voto\_terminate\_con\_vincitore()” dell’oggetto “sessione\_di\_voto.DAO”. Questa lista viene visualizzata attraverso il metodo “setItems()” della listView “listView\_sessions\_di\_voto\_vincitori”.

## **Voto elettronico (a distanza)**

Questo scenario è composto da più diagrammi di sequenza, in quanto il controller che riceve i voti dei referendum è diverso dal controller che riceve i voti delle votazioni.

Inoltre nei diagrammi è rappresentato non solo lo scenario di voto elettronico dell'elettore, ma anche lo scenario di registrazione di un voto cartaceo da parte di un gestore del sistema. Infatti, quando un gestore del sistema vuole inserire i voti cartacei del seggio elettorale, gli verrà mostrata la medesima interfaccia di un elettore che intende votare, con la differenza che il gestore di sistema potrà registrare più di un voto (uno per volta).

Riporto di seguito i vari diagrammi

## voto “sì” ad un referendum

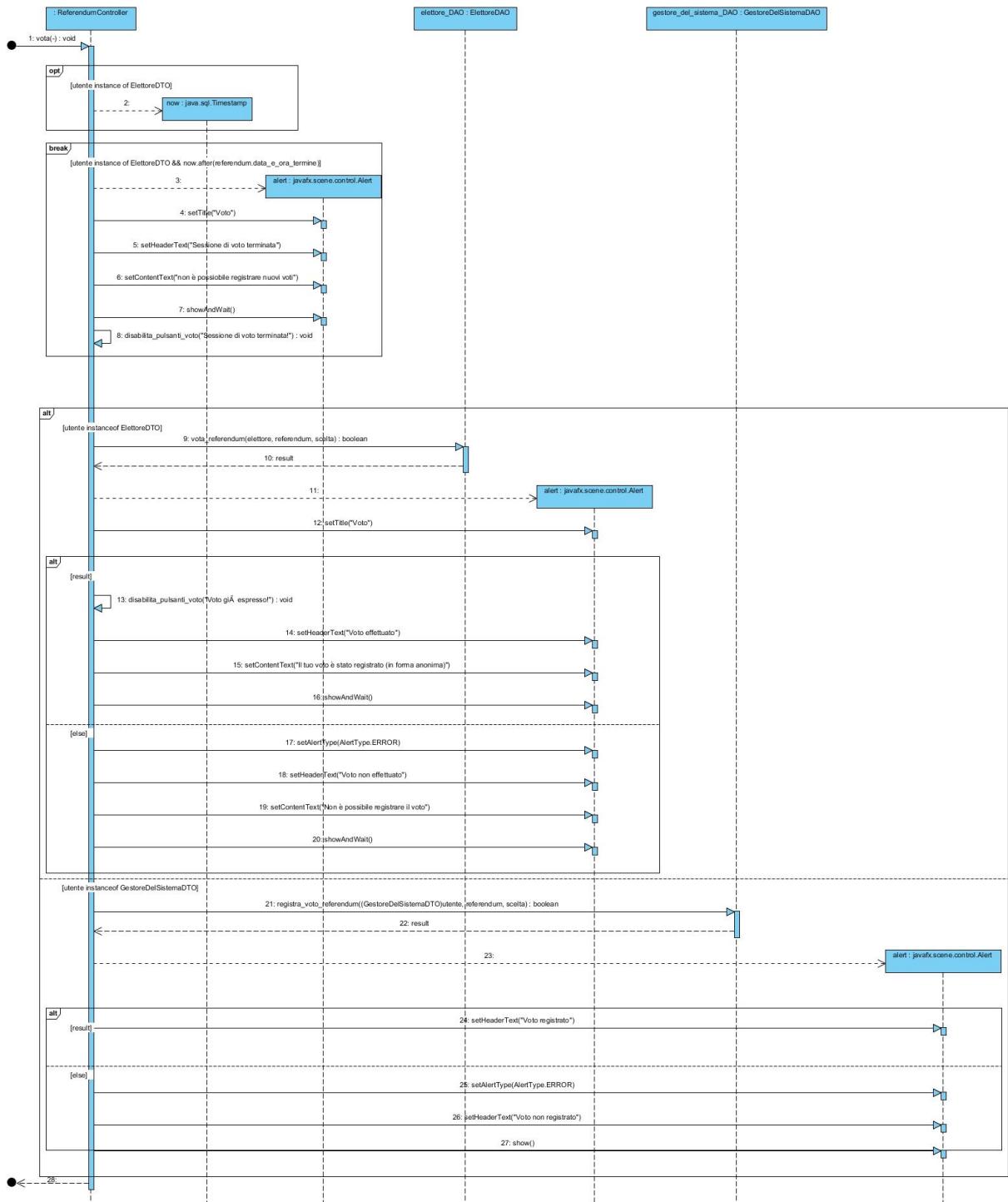


Figura 18: diagramma di sequenza scenario “voto elettronico (a distanza), voto si ad un referendum”

La richiesta di votare “sì” ad un referendum è gestita da un’istanza della classe “ReferendumController”, la quale controlla se l’utente è un elettore o un gestore del sistema.

Nel primo caso viene registrato il voto mediante il metodo “`voto_referendum(elettore, referendum, scelta)`” dell’oggetto “`elettore_DAO`” con `scelta=true`. Quindi, se la registrazione è andata a buon fine, verrà mostrato un messaggio di successo e verrà

bloccata la possibilità di votare nuovamente, altrimenti verrà mostrato un messaggio di errore.

Nel secondo caso viene registrato il voto tramite il metodo “registra\_voto\_referendum(..., scelta)” con scelta=true e non verrà bloccata la possibilità di votare ulteriormente. Quindi, se la registrazione è andata a buon fine, verrà mostrato un messaggio di successo, altrimenti di errore.

Il voto “no” ad un referendum è analogo al diagramma appena mostrato, solo che le due chiamate al metodo “vota\_referendum(elettore, referendum, scelta)” hanno il parametro scelta=false.

astenersi ad un referendum

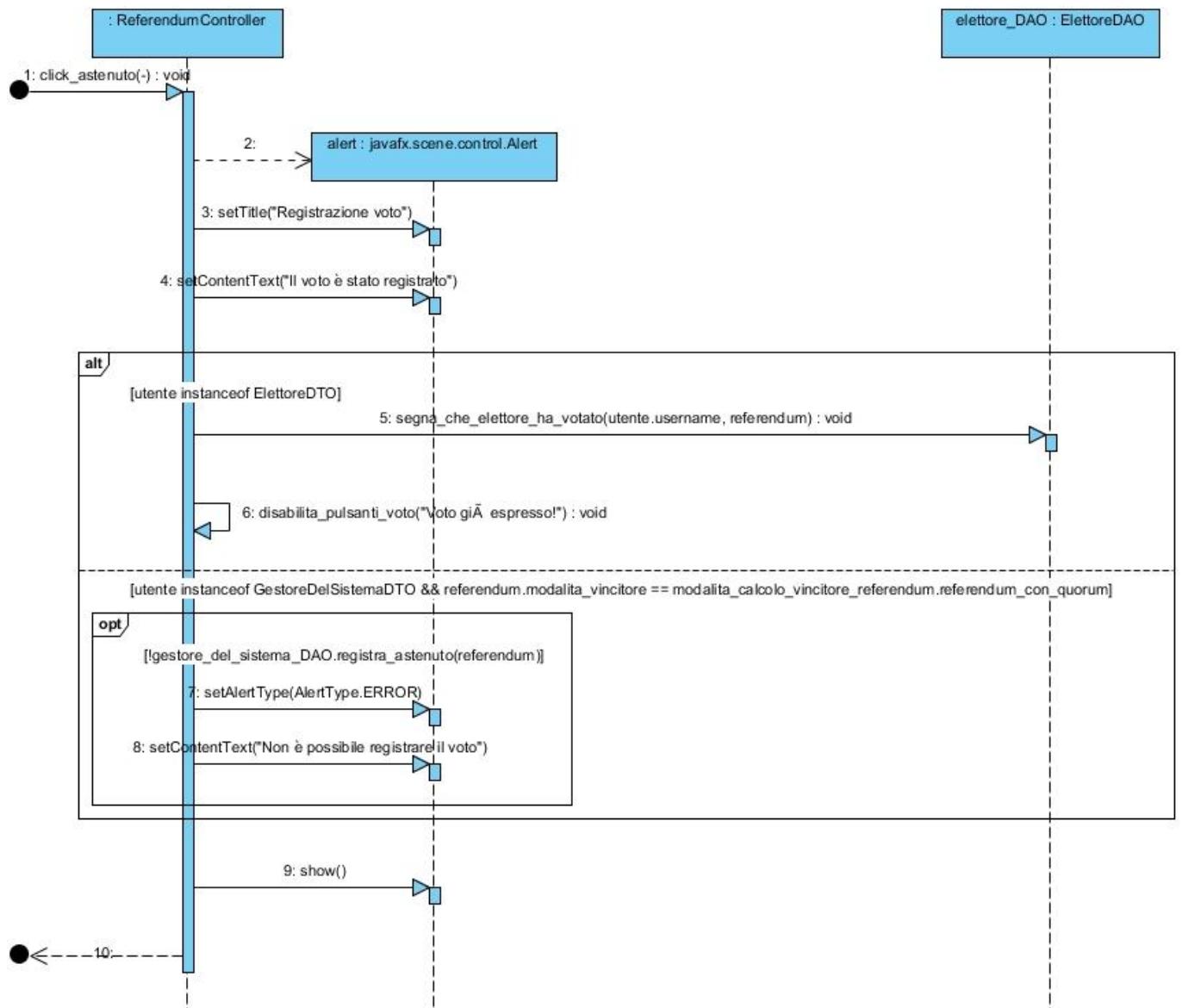


Figura 19: diagramma di sequenza scenario “voto elettronico (a distanza), astenersi ad un referendum”

La richiesta di astenersi ad un referendum è gestita da un'istanza della classe "ReferendumController", la quale controlla se l'utente è un elettore o un gestore del sistema.

Nel primo caso viene registrato che l'elettore si è astenuto mediante il metodo "segna\_che\_elettore\_ha\_votato()", il quale restituisce true o false. Se quest'ultimo restituisce true, viene mostrato un messaggio di successo e viene bloccata la possibilità di votare ulteriormente, altrimenti se esso restituisce false viene mostrato un messaggio di errore.

Nel secondo caso viene registrata la scelta di astenersi solo se la modalità di calcolo del vincitore è con quorum, in quanto se fosse senza quorum non sarebbe utile. Analogamente all'elettore viene mostrato un messaggio di successo o errore a seconda della riuscita della registrazione.

## voto ad una votazione

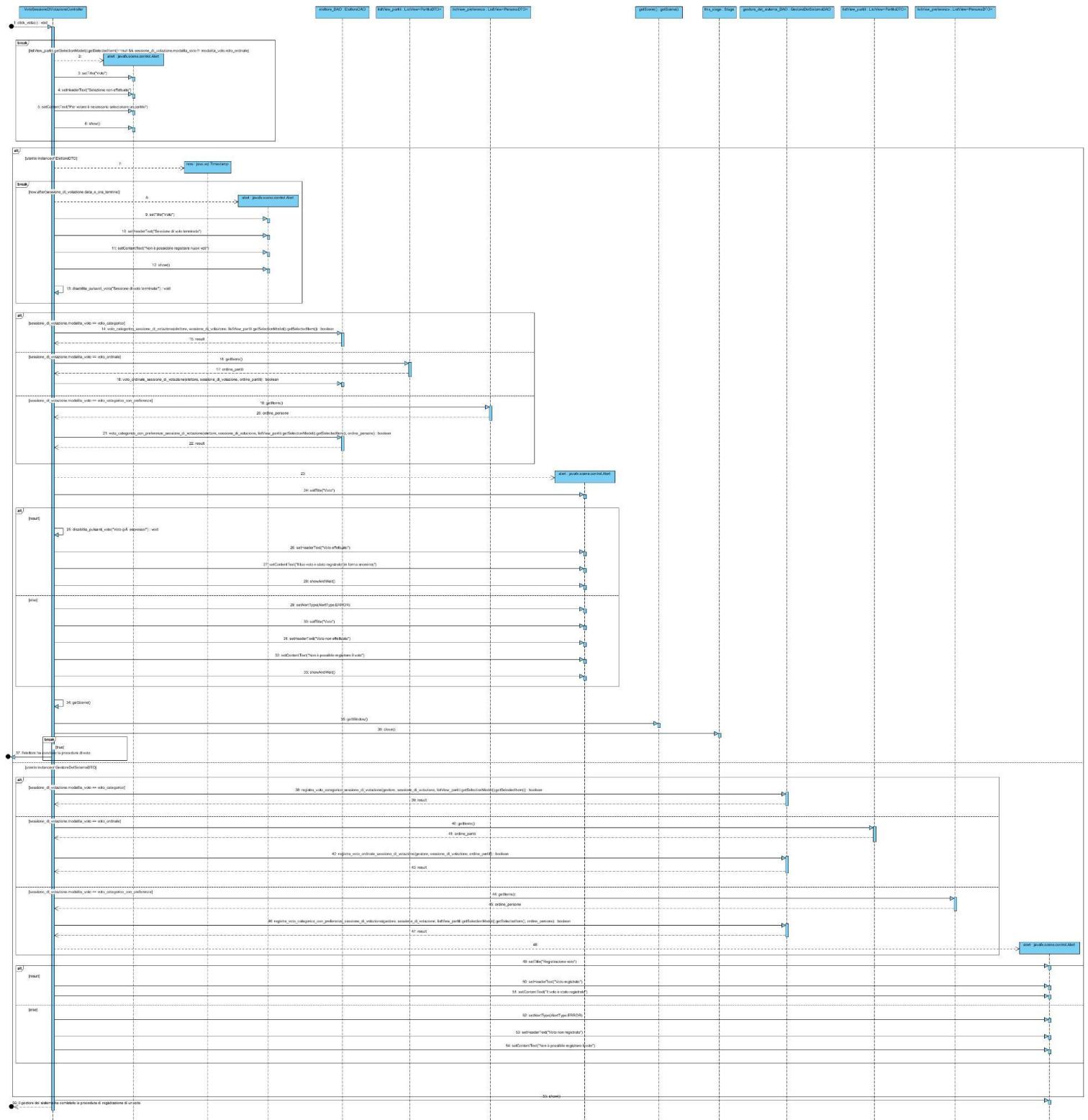


Figura 20: diagramma di sequenza scenario “Voto ad una votazione”

La richiesta “votare ad una votazione” è gestita da un’istanza della classe “`VotoSessioneDiVotazioneController`”, la quale, dopo una fase di controllo iniziale, verifica se l’utente è un gestore del sistema o un elettore.

Nel caso sia un elettore, viene controllato che la sessione di voto non sia terminata e, successivamente, a seconda del tipo di voto viene chiamato un metodo differente dell'oggetto “elettore.DAO” .

Rispettivamente:

- per il voto categorico viene chiamato  
“voto\_categorico\_sessione\_di\_votazione()”;
- per il voto ordinale viene chiamato “voto\_ordinale\_sessione\_di\_votazione()”;
- per il voto categorico con preferenze viene chiamato  
“voto\_categorico\_con\_preferenze\_sessione\_di\_votazione()”.

Infine, se il voto è stato registrato, viene visualizzato un messaggio di successo, altrimenti di errore, la schermata viene chiusa e non potrà essere espresso un ulteriore voto.

Nel caso del gestore del sistema, la registrazione del voto cartaceo, a seconda del tipo di votazione, viene chiamato un metodo differente dell'oggetto “gestore\_del\_sistema.DAO” .

Rispettivamente:

- per il voto categorico viene chiamato  
“registra\_voto\_categorico\_sessione\_di\_votazione()”;
- per il voto ordinale viene chiamato  
“registra\_voto\_ordinale\_sessione\_di\_votazione()”;
- per il voto categorico con preferenze viene chiamato  
“registra\_voto\_categorico\_con\_preferenze\_sessione\_di\_votazione()”.

Se il voto è stato registrato, viene visualizzato un messaggio di successo, altrimenti uno di errore. La schermata rimarrà aperta e sarà pronta per la registrazione di un nuovo voto.

## astenersi ad una votazione

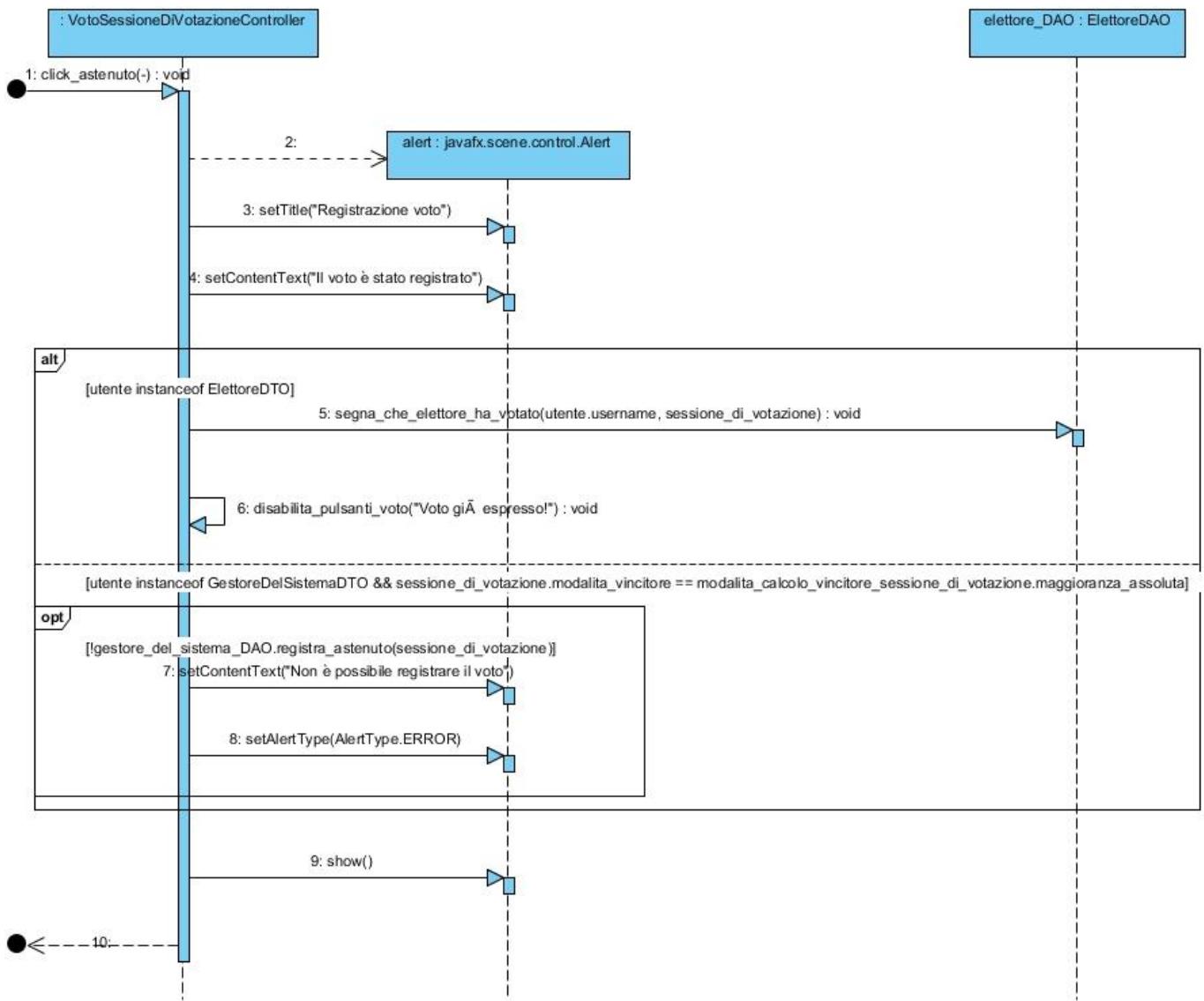


Figura 21: diagramma di sequenza scenario “voto elettronico (a distanza), astenersi ad una votazione”

La richiesta di astenersi ad una votazione è gestita da un’istanza della classe “VotoSessioneDiVotazioneController”, la quale controlla se l’utente è un elettore o un gestore del sistema.

Nel primo caso, essa regista che l’elettore si è astenuto attraverso il metodo “segna\_che\_elettore\_ha\_votato()”, che segna che l’elettore ha espresso un proprio voto. Nonostante ciò, nessun voto verrà effettivamente registrato perché l’elettore ha scelto di astenersi. Dopodiché, se l’azione si è svolta con successo, viene mostrato un messaggio di successo e viene bloccata la possibilità di votare ulteriormente, altrimenti viene visualizzato un messaggio di errore.

Nel secondo caso viene registrata la scelta di astenersi solo se la modalità di calcolo del vincitore della votazione è maggioranza assoluta, poiché se la modalità è

maggioranza non serve tenere conto degli astenuti. Come per l'elettore, viene mostrato un messaggio di successo o errore a seconda della riuscita della registrazione.

## Visualizza sessioni di voto

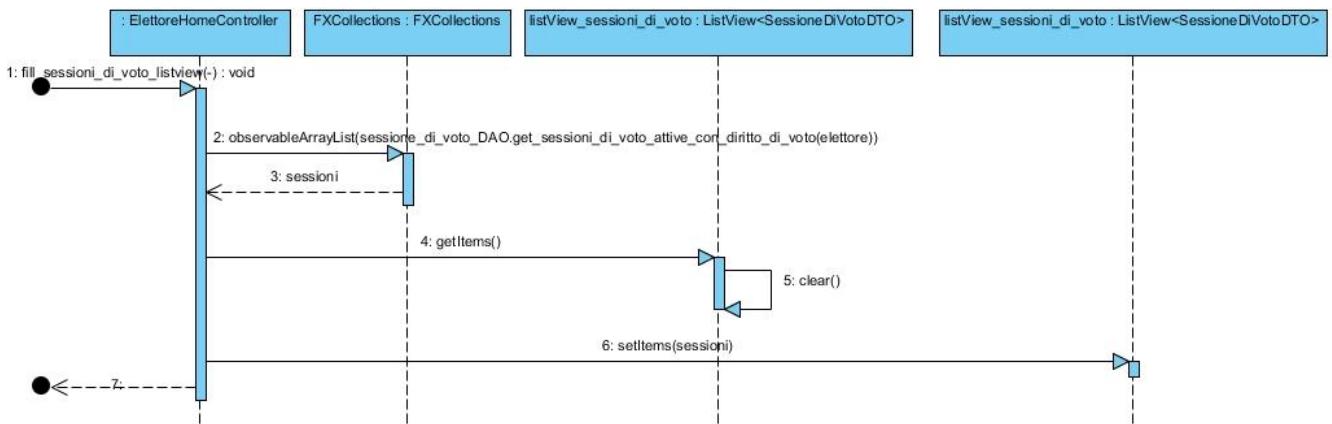


Figura 22: diagramma di sequenza scenario “Visualizza sessione di voto”

La richiesta di visualizzare le sessioni di voto è gestita da un'istanza della classe “ElettoreHomeController”, la quale estrae la lista delle sessioni dove l'elettore ha il diritto di voto mediante il metodo “`get_sessions_di_voto_attive_con_diritto_di_voto()`” dell'oggetto “`sessione_di_voto.DAO`”. Successivamente questa lista viene visualizzata tramite il metodo “`setItems()`” della listView “`listView_sessions_di_voto`”.

## Crea sessione di voto

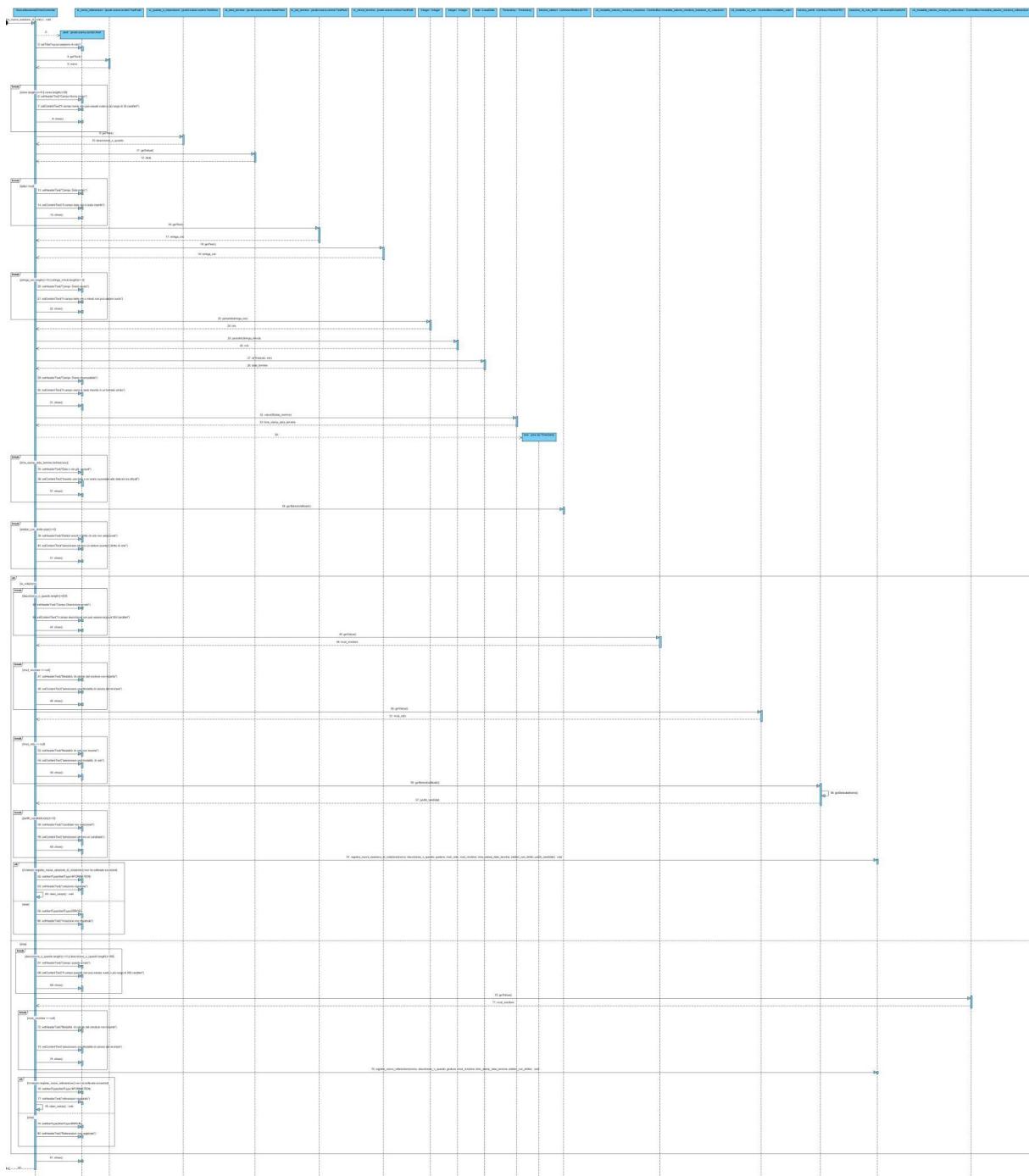


Figura 23: diagramma di sequenza scenario “Crea sessione di voto”

La richiesta di creare una nuova sessione di voto è gestita da un’istanza della classe “NuovaSessioneDiVotoController”, la quale inizialmente controlla i vari input inseriti dall’utente mostrando un messaggio di errore se questi non sono corretti. Una volta controllati gli input, viene controllato se la sessione di voto che si vuole creare è una votazione o un referendum.

Nel primo caso, tra i vari controlli, viene controllato anche che sia stato selezionato almeno un partito candidato alla votazione e questa viene quindi registrata mediante

il metodo “`registra_nuova_votazione()`”. Se il metodo non solleva eccezioni viene visualizzato un messaggio di successo, altrimenti un messaggio di errore.

## Ricerca elettore nella sessione di voto

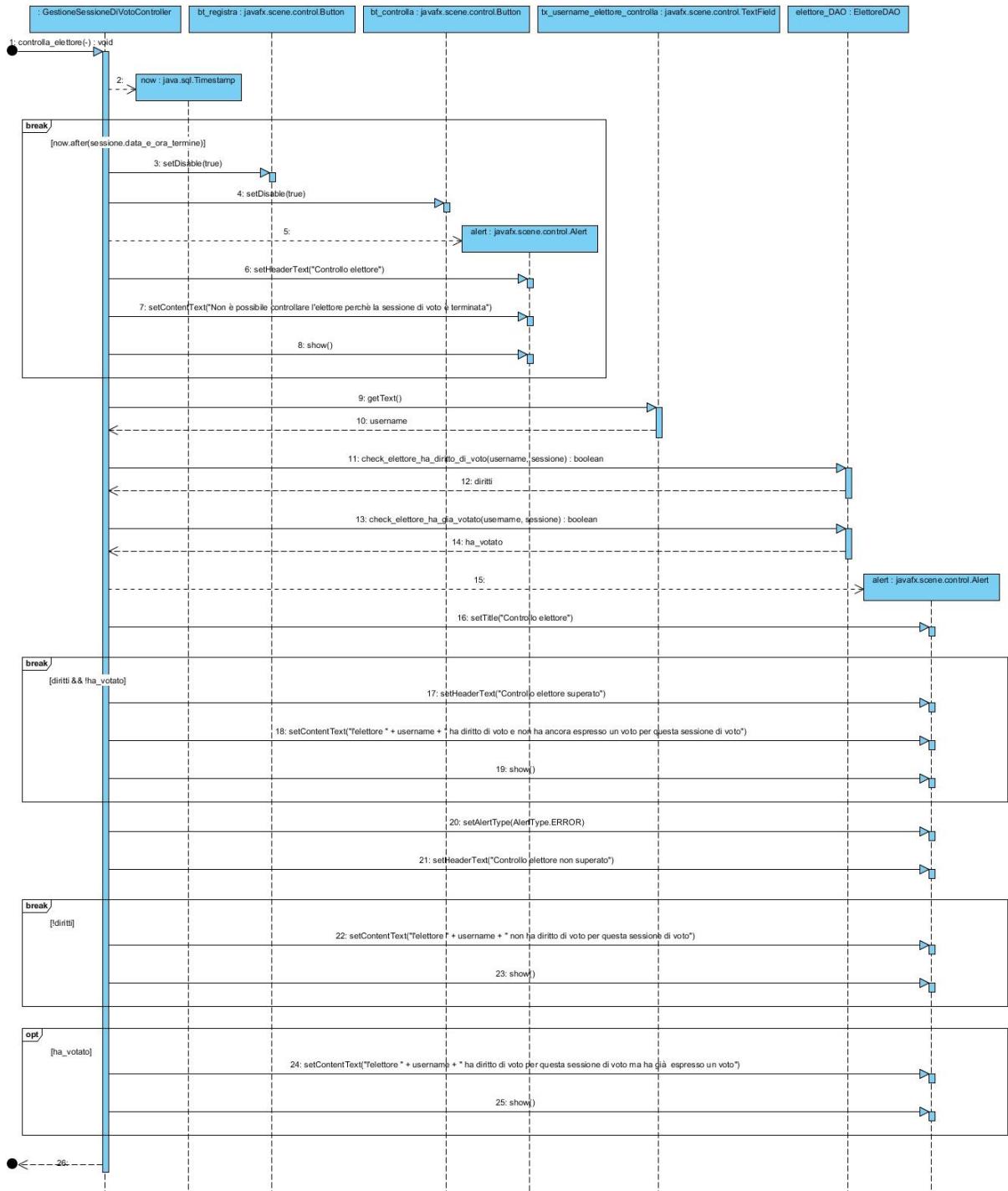


Figura 24: diagramma di sequenza scenario “Ricerca persona elettore nella sessione di voto”

La richiesta di ricercare un elettore nella sessione di voto è gestita da un'istanza della classe “GestioneSessioneDiVotoController”, la quale controlla che la sessione di voto non sia terminata, poiché se lo fosse verrebbe mostrato un messaggio di errore. Successivamente viene controllato se l'elettore ha il diritto di voto mediante il

metodo “check\_elettore\_ha\_diritto\_di\_voto()” dell’oggetto “elettore.DAO”. Se l’elettore non ha il diritto di voto viene visualizzato un messaggio di errore, altrimenti si procede con il controllo del voto mediante il metodo “check\_elettore\_ha\_gia\_votato()” sempre dell’oggetto “elettore.DAO”. Infine viene visualizzato un messaggio che indica se l’elettore ha già votato nella sessione di voto oppure no.

## Inserisci voti in presenza

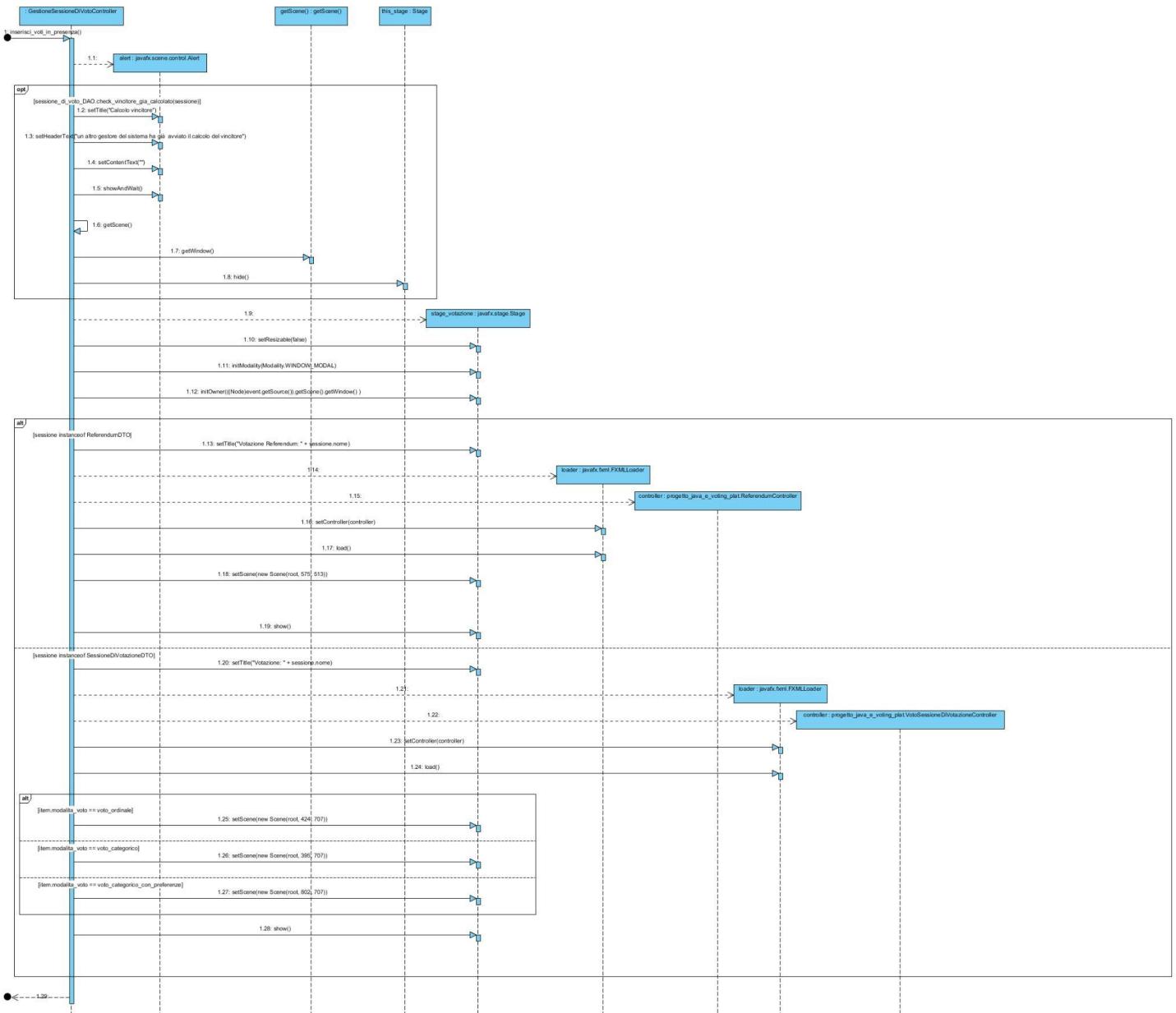


Figura 25: diagramma di sequenza scenario “Inserisci voti in presenza”

La richiesta di inserire i voti in presenza in una sessione di voto è gestita da un’istanza della classe “GestioneSessioneDiVotoController”, la quale controlla se è già stato calcolato il vincitore della sessione di voto mediante il metodo “check\_vincitore\_gia\_calcolato()” dell’oggetto “sessione\_di\_voto\_DAO”. Se il vincitore è già stato calcolato non si potrà procedere, altrimenti si procederà

controllando se la sessione di voto è un referendum o una votazione.

Se è un referendum viene mostrata la stessa schermata mostrata agli elettori per votare ad un referendum, altrimenti viene mostrata la stessa schermata che viene mostrata agli elettori per votare ad una votazione.

## Calcola vincitore

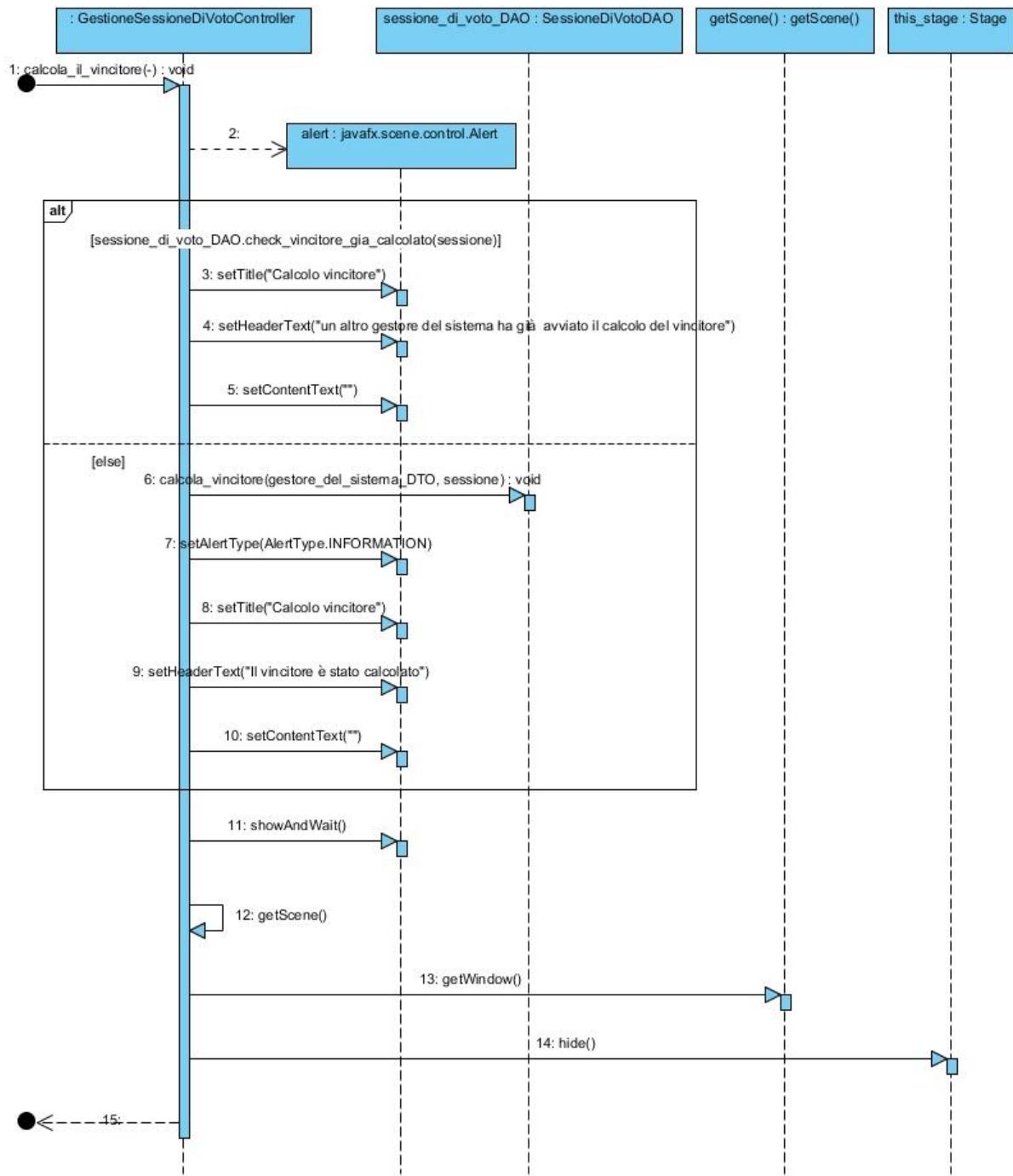


Figura 26: diagramma di sequenza scenario "calcola vincitore"

La richiesta di calcolare il vincitore è gestita da un'istanza della classe "GestionSessioneDiVotoController", la quale, come nel precedente scenario, controlla che il vincitore non sia già stato calcolato. Successivamente calcola il

vincitore mediante il metodo “calcola\_vincitore()” dell’oggetto “sessione\_di\_voto.DAO” ed infine viene mostrato un messaggio di successo.

## Gestione sessioni di voto

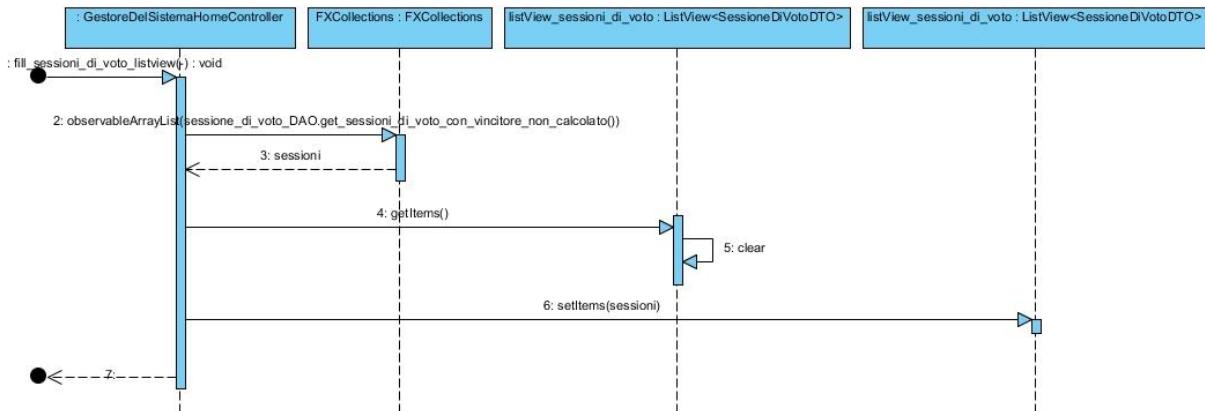


Figura 27: diagramma di sequenza scenario “Gestione sessioni di voto”

La richiesta di gestire le sessioni di voto è gestita da un’istanza della classe “GestoreDelSistemaHomeController”, la quale ottiene tutte le sessioni di voto con il vincitore non calcolato mediante il metodo “get\_sessions\_di\_voto\_con\_vincitore\_non\_calcolato()” dell’oggetto “sessione\_di\_voto.DAO” e, successivamente, le mostra all’utente mediante la listview “listView\_sessions\_di\_voto”.

## Visualizza log

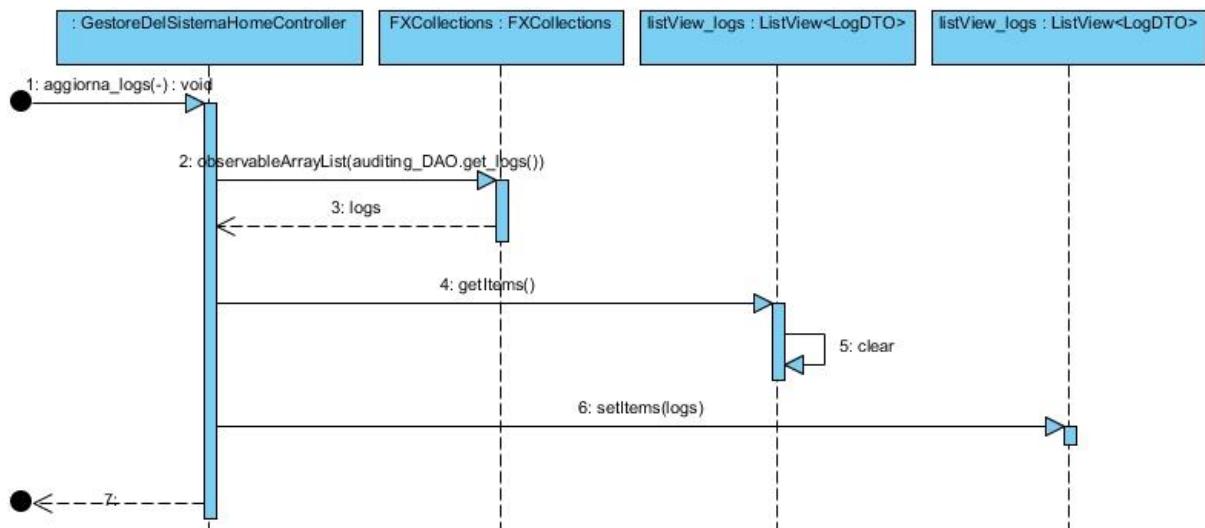


Figura 28: diagramma di sequenza scenario “Visualizza log”

La richiesta di visualizzare i log è gestita da un’istanza della classe “GestoreDelSistemaHomeController”, la quale ottiene tutti i log mediante il metodo “get\_logs()” dell’oggetto “auditing.DAO” e successivamente li mostra all’utente mediante la listview “listView\_logs”.

## Inserisci un nuovo partito

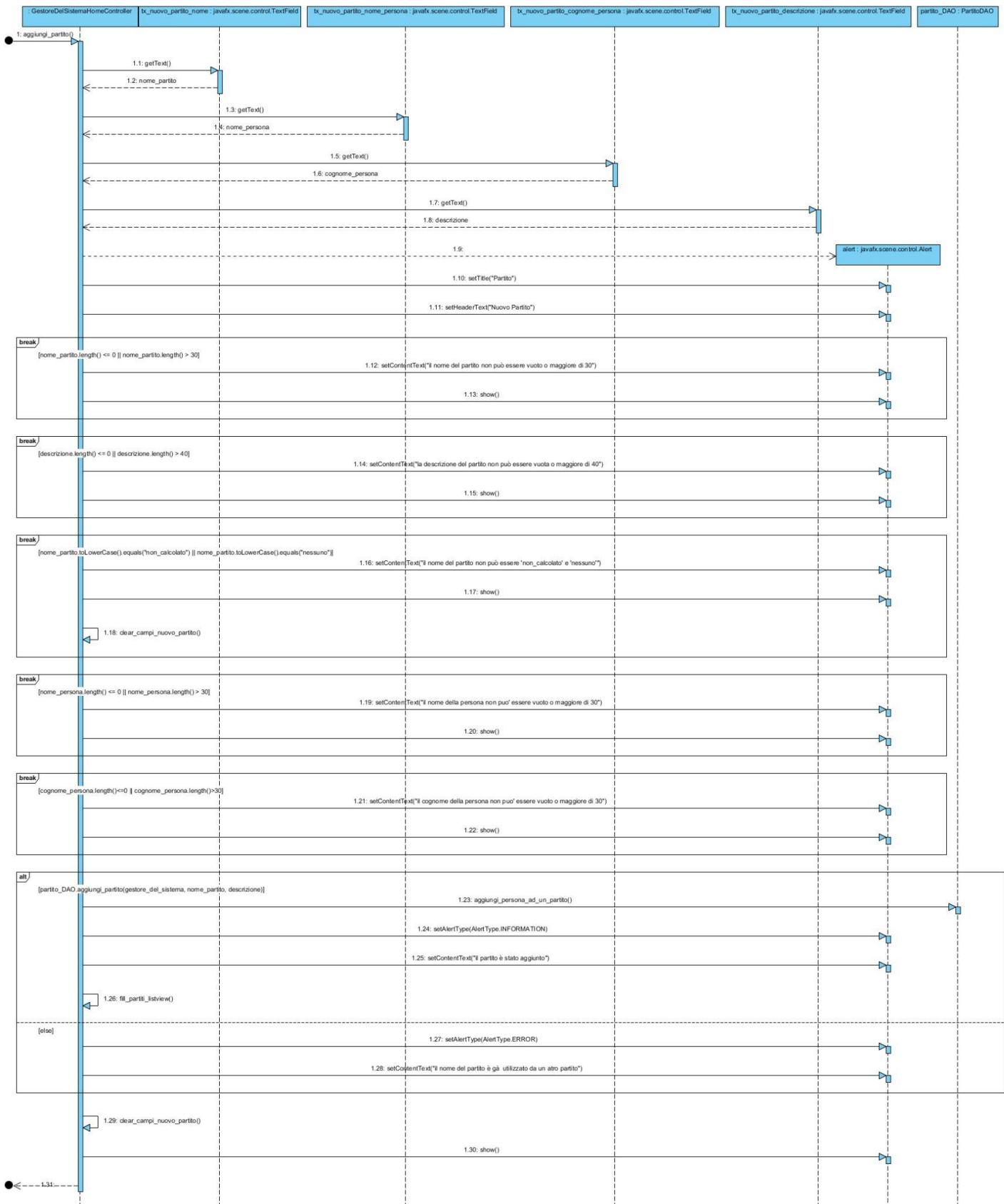


Figura 29: diagramma di sequenza scenario "Inserisci un nuovo partito"

La richiesta di registrare nel sistema un nuovo partito è gestita da un'istanza della classe “GestoreDelSistemaHomeController”, la quale controlla gli input dell'utente: nome partito, descrizione partito, nome e cognome della prima persona del partito. Se il controllo è superato con successo procede a registrare il nuovo partito mediante il metodo “aggiungi\_partito” dell'oggetto “partito.DAO” e successivamente viene aggiunta la prima persona del partito mediante il metodo “aggiungi\_persona\_ad\_un\_partito” dello stesso oggetto. Se l'operazione è andata a buon fine viene visualizzato un messaggio di successo, altrimenti di errore.

## Inserisci una nuova persona in un partito

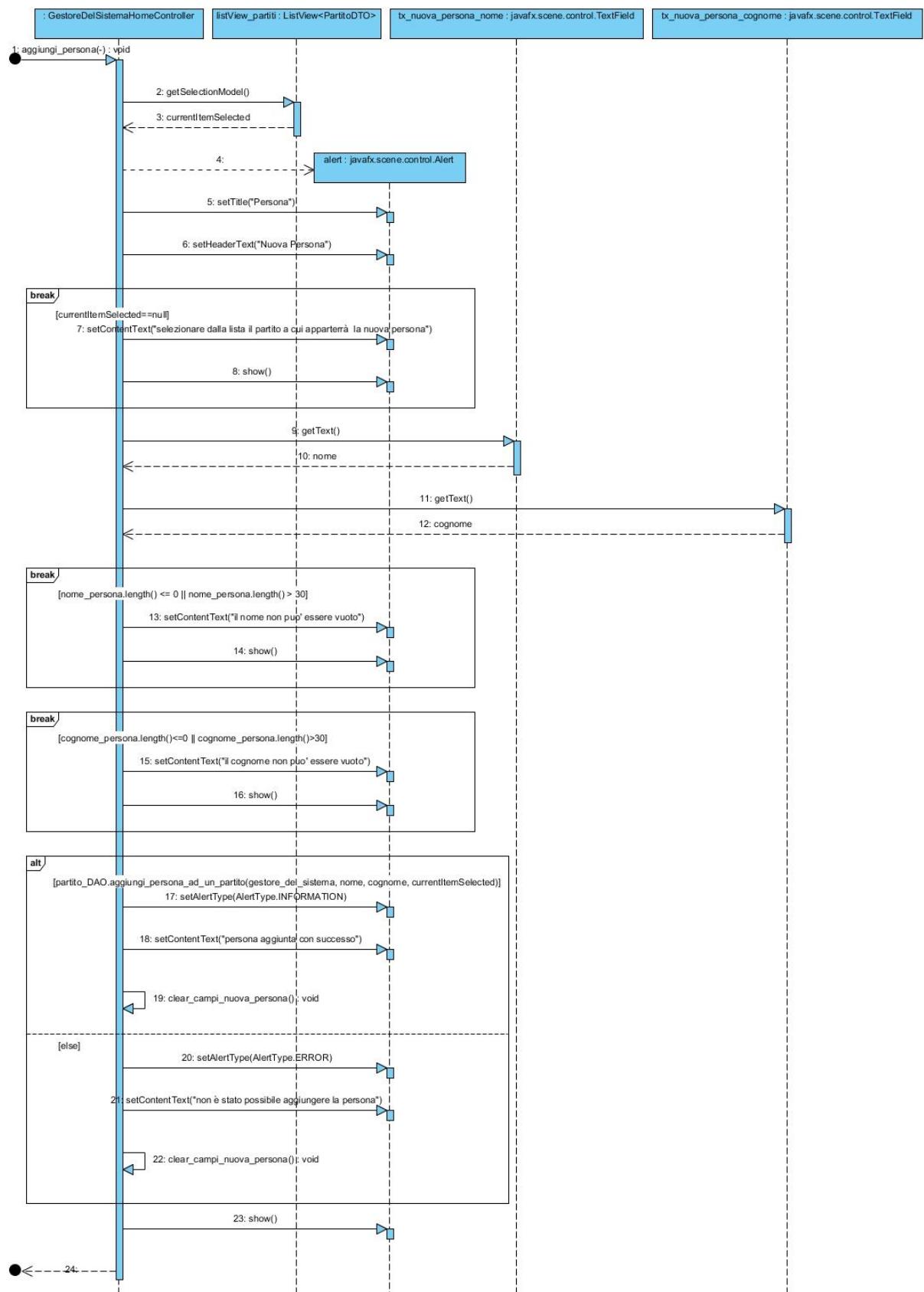


Figura 30: diagramma di sequenza scenario “Inserisci una nuova persona in un partito”

La richiesta di aggiungere una nuova persona in un partito è gestita da un'istanza della classe “GestoreDelSistemaHomeController”, la quale controlla nome, cognome e se è stato selezionato il partito alla quale sarà aggiunta mediante il metodo “getSelectionModel” dell'oggetto “listView\_partiti”. Se il controllo è superato con successo si procede all'aggiunta della persona al partito mediante il metodo “aggiungi\_persona\_ad\_un\_partito” dell'oggetto “partito.DAO”. Se l'operazione è andata a buon fine viene visualizzato un messaggio di successo, altrimenti di errore.

### Registra che persona elettore A. ha votato

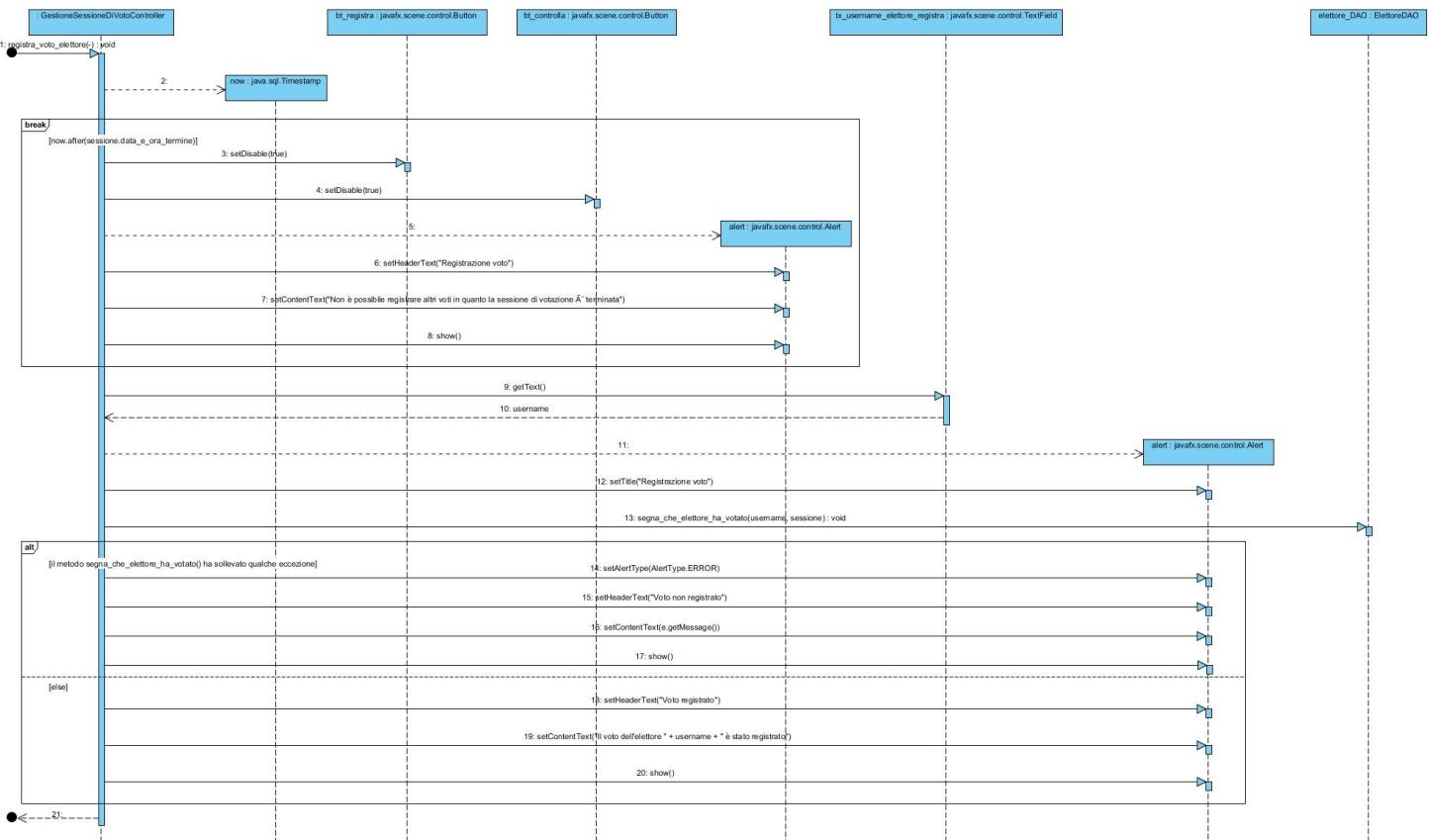


Figura 31: diagramma di sequenza scenario “registra che persona elettore A. ha votato”

La richiesta di registrare che una persona elettore A. ha votato in presenza è gestita da un'istanza della classe “GestioneSessioneDiVotoController”, la quale controlla se la sessione di voto è terminata. In caso che quest'ultima lo sia, mostra un messaggio di errore e non procede, mentre, se la sessione di voto non è ancora terminata, viene registrato che la persona elettore A. sta esprimendo il proprio voto cartaceo mediante il metodo “segna\_che\_elettore\_ha\_votato()” dell'oggetto “elettore.DAO”. Infine se il metodo solleva qualche eccezione viene mostrato un messaggio di errore, altrimenti uno di successo.

## 2.5. Diagrammi di attività

In questa parte sono riportati e discussi i diagrammi delle attività da me prodotti, ossia solo quelli per i diagrammi di sequenza più significativi e corposi, in quanto ho

prodotto molti diagrammi di sequenza e non vorrei soffermarmi troppo su questa fase della relazione.

I diagrammi di attività prodotti sono stati disegnati parallelamente a quelli di sequenza poiché questi ultimi sono molto a “basso livello” e sono direttamente mappati nel codice prodotto. Al contrario, i diagrammi di attività realizzati sono ad un livello astrattivo più alto. A causa di questo salto di astrazione, il rischio sarebbe stato quello di diminuire la loro tracciabilità. Tuttavia, crearli parallelamente mi ha permesso di astrarre con criterio, garantendo così la tracciabilità dei diagrammi.

## Autenticazione sulla piattaforma sw

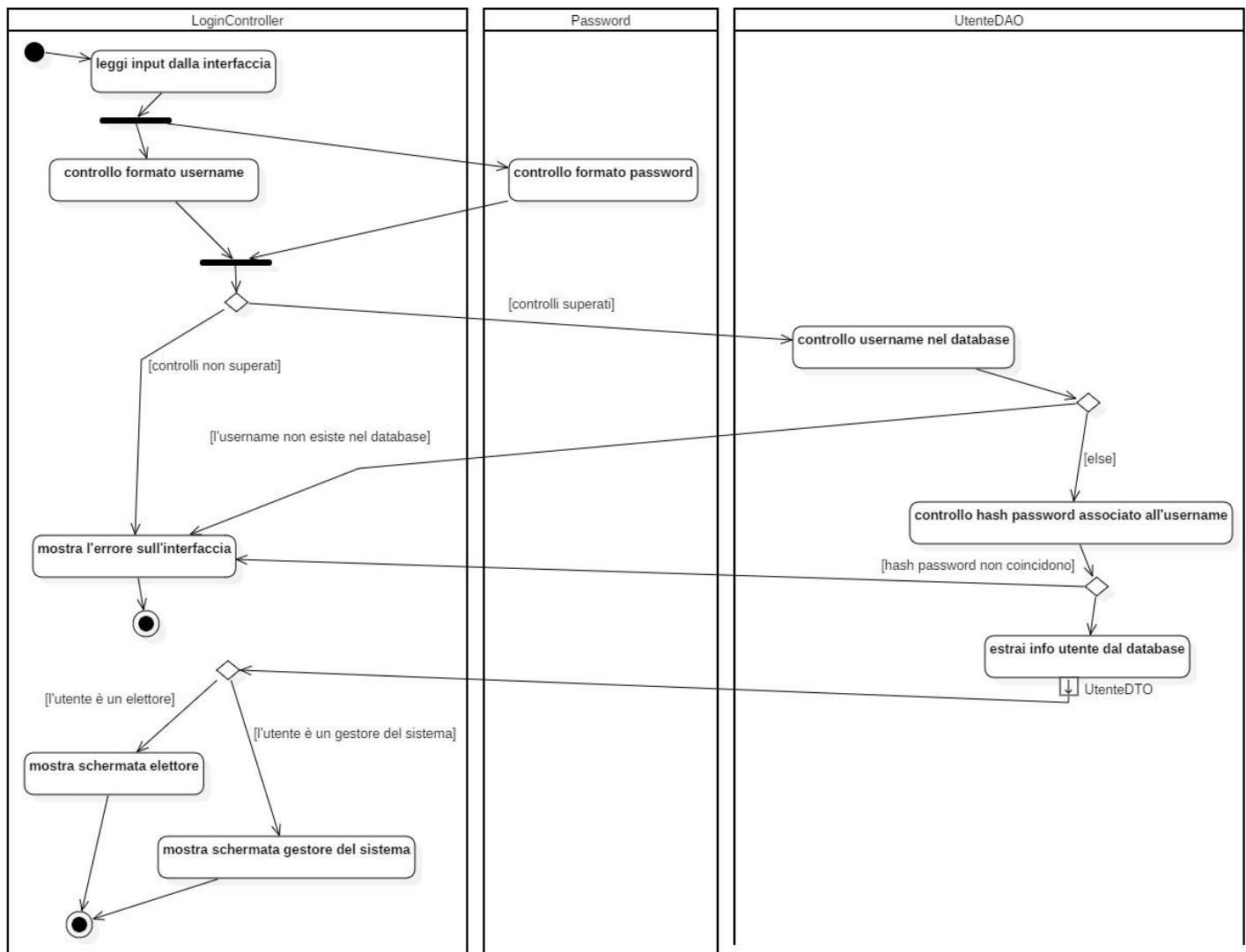


Figura 32: diagramma delle attività “autenticazione sulla piattaforma sw”

Questo diagramma delle attività rappresenta l’ordine di esecuzione delle azioni che permettono all’utente di autenticarsi sulla piattaforma sw quando, quest’ultimo, ha inserito gli input richiesti ed ha attivato il trigger dello scenario.

## Registrazione in presenza

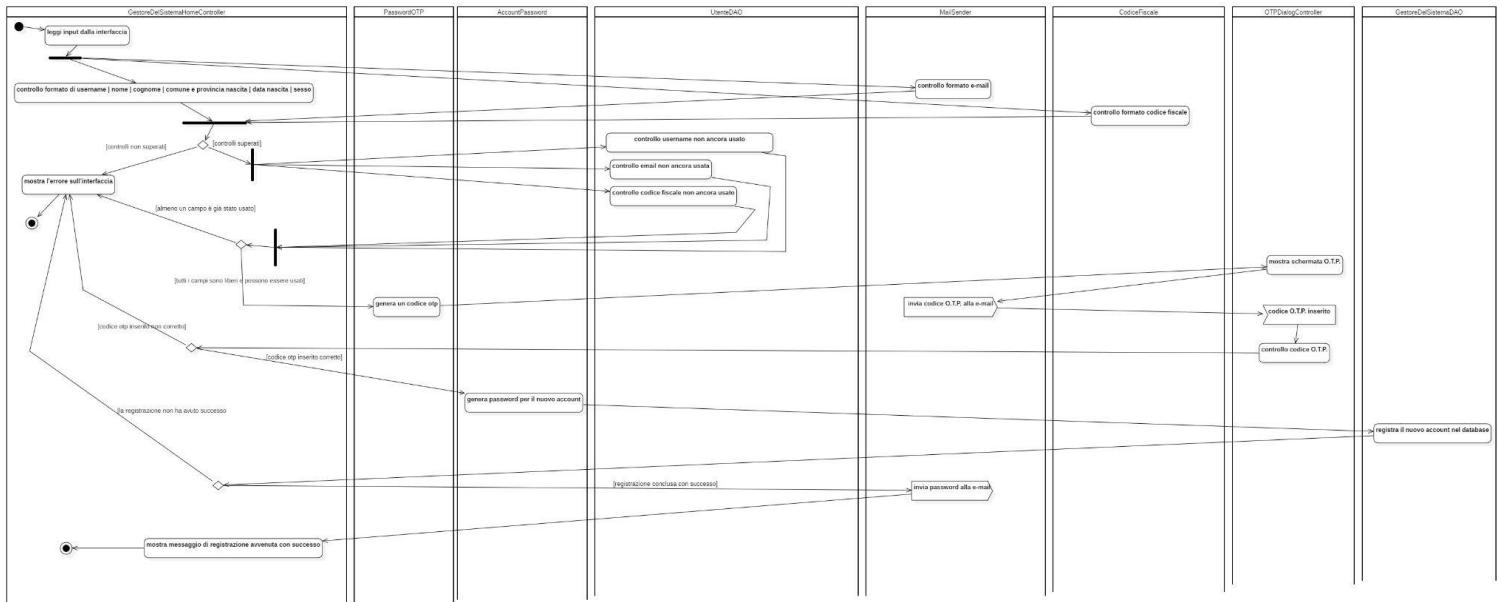


Figura 33: diagramma delle attività "Registrazione in presenza"

Questo diagramma delle attività rappresenta l'ordine di esecuzione delle azioni che permettono a un gestore del sistema di registrare un nuovo elettorato sulla piattaforma sw quando, il gestore del sistema, ha inserito gli input richiesti ed ha attivato il trigger dello scenario.

## Voto elettronico (a distanza)

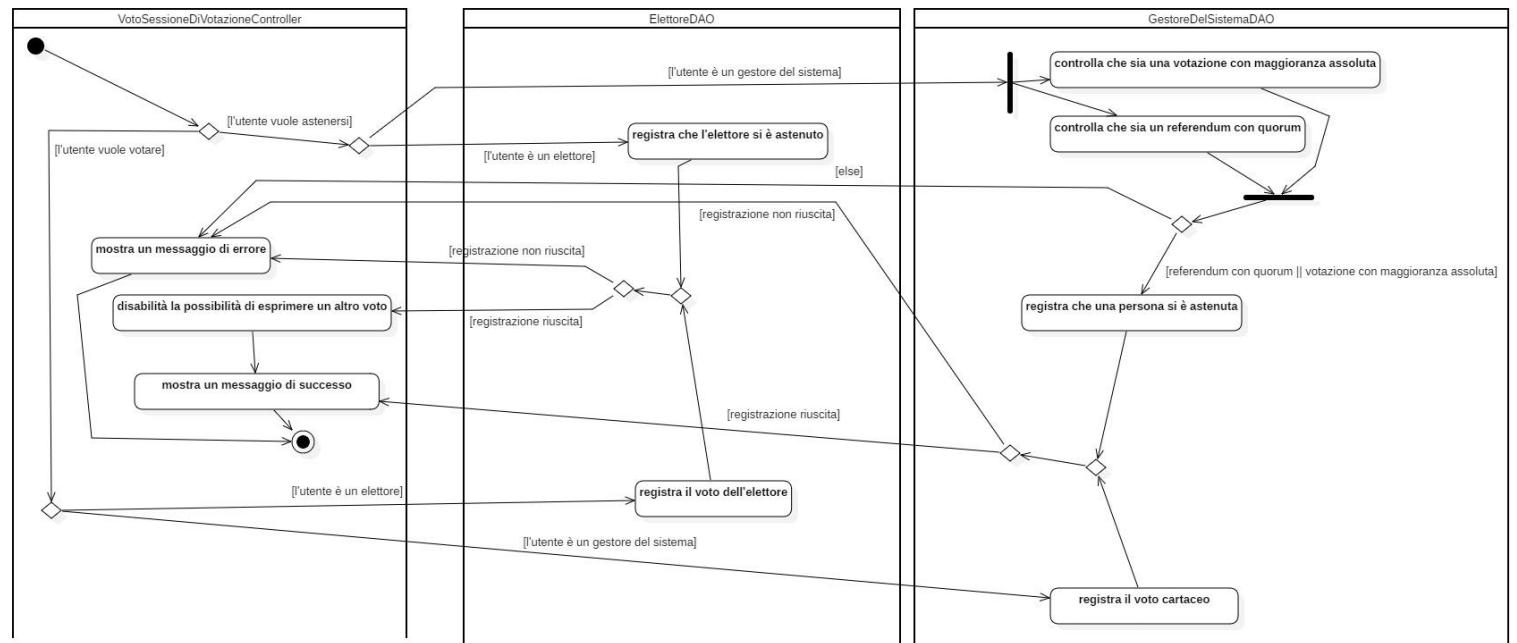


Figura 34: diagramma delle attività "voto elettronico (a distanza)"

Questo diagramma delle attività rappresenta l'ordine di esecuzione delle azioni che

permettono a un elettore di votare ed ad un gestore del sistema di registrare un voto fatto in presenza.

Utilizzando un' astrazione maggiore non ho dovuto dividere in più diagrammi, come nei diagrammi di sequenza, i vari casi (votazione, referendum, voto ad una votazione, astenuto ad una votazione, astenuto ad un referendum, voto ad un referendum).

## Crea sessione di voto

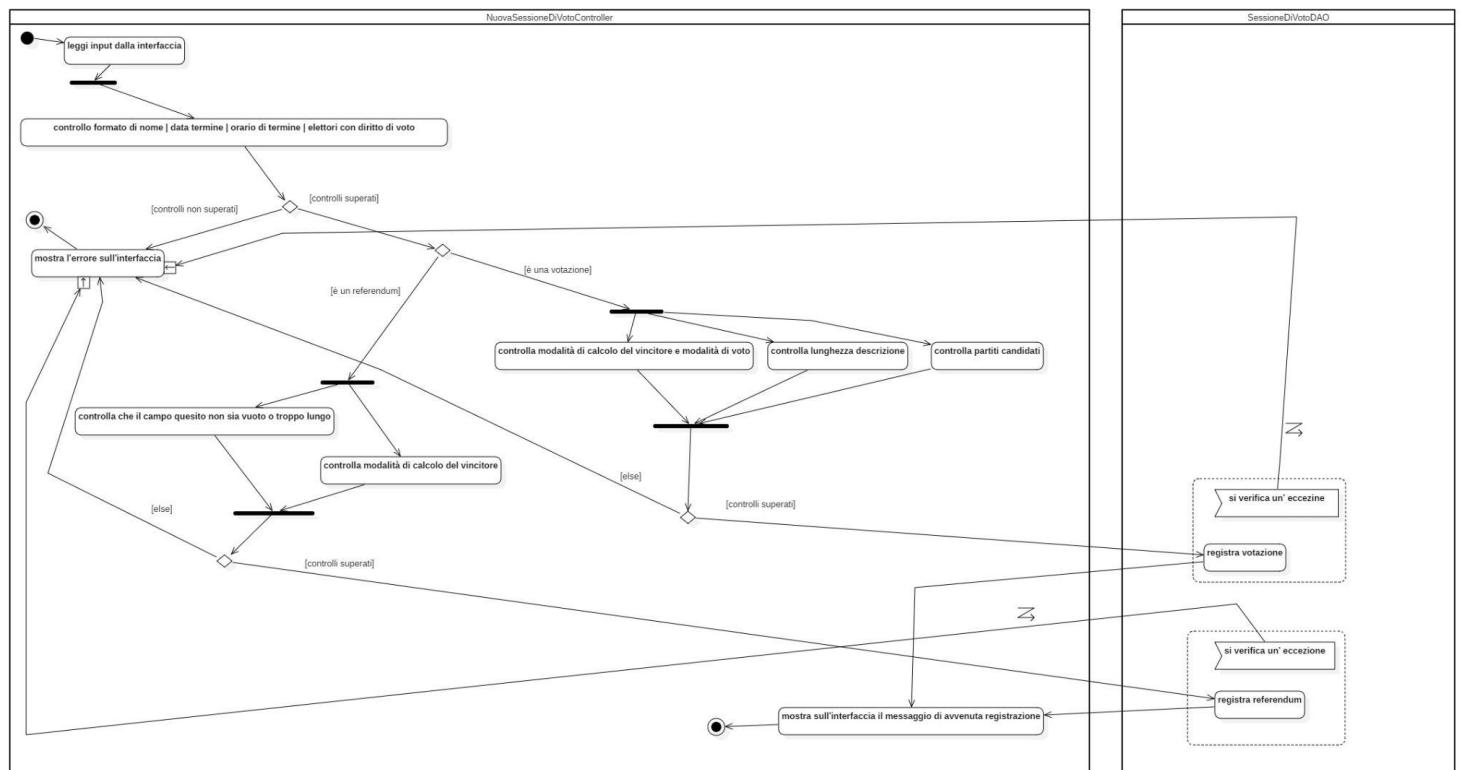


Figura 35: diagramma delle attività "crea sessione di voto"

Questo diagramma delle attività rappresenta l'ordine di esecuzione delle azioni che permettono a un gestore del sistema di creare una sessione di voto e registrarla nel sistema sw quando, quest'ultimo, ha inserito gli input richiesti ed ha attivato il trigger dello scenario.

## Ricerca elettore nella sessione di voto

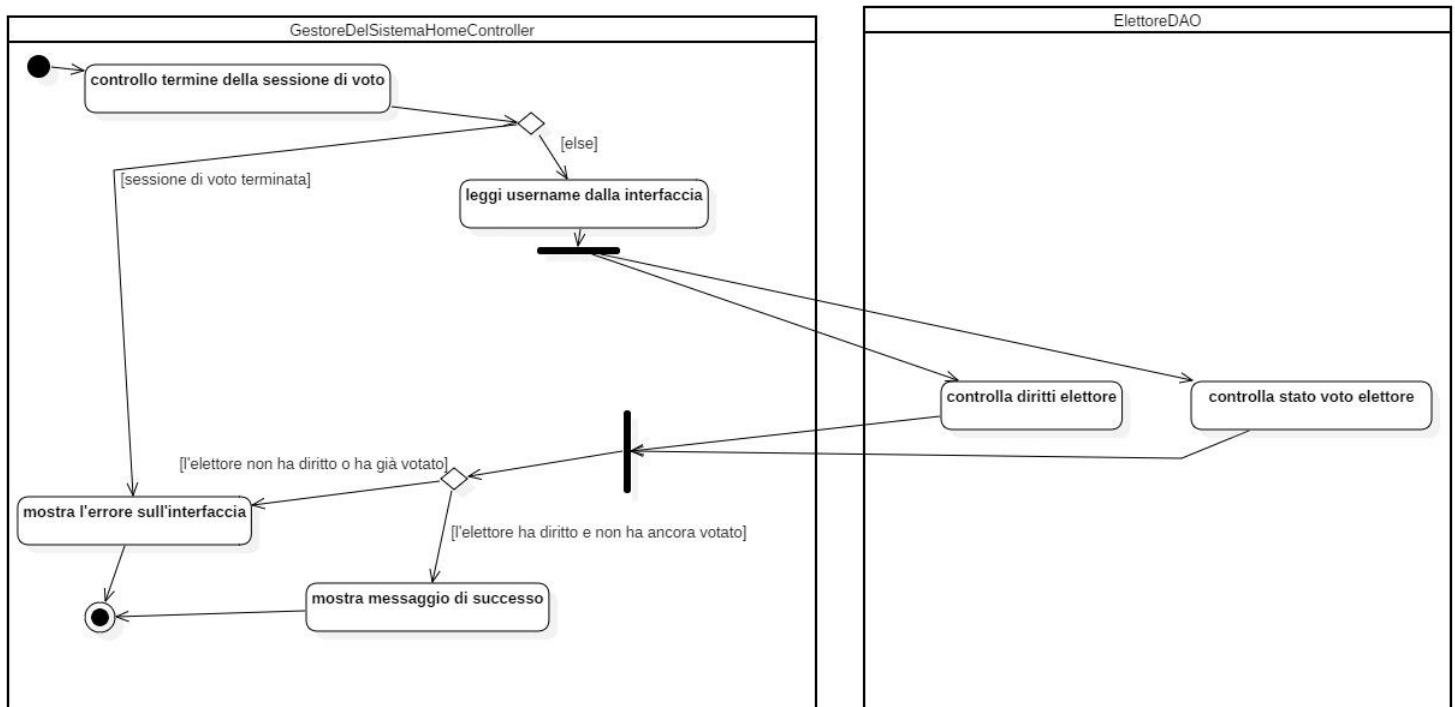


Figura 36: diagramma delle attività “ricerca elettore nella sessione di voto”

Questo diagramma delle attività rappresenta l'ordine di esecuzione delle azioni che permettono a un gestore del sistema di cercare un elettore all'interno di una sessione di voto per scoprire se ha il diritto di votare e se ha già espresso un voto in una determinata sessione di voto.

## Inserisci voti in presenza

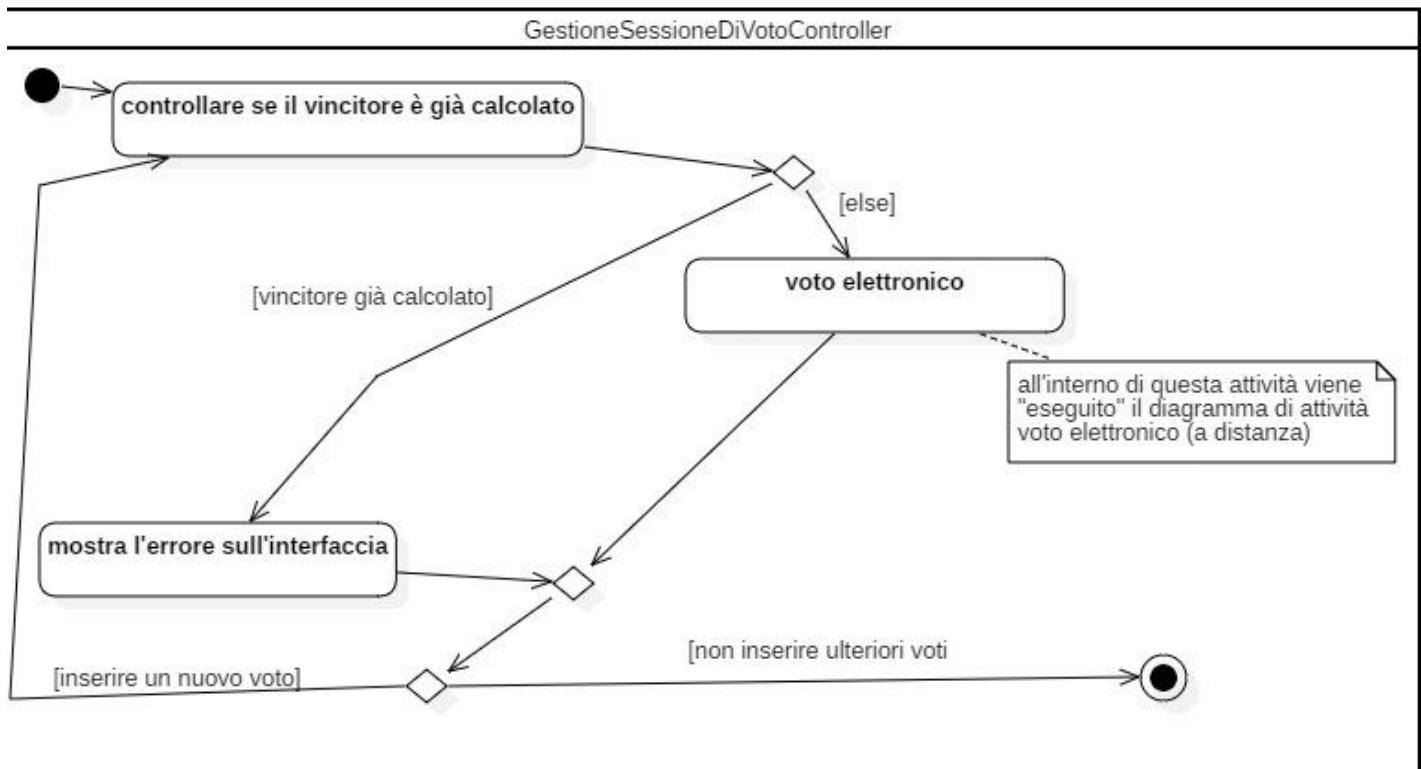
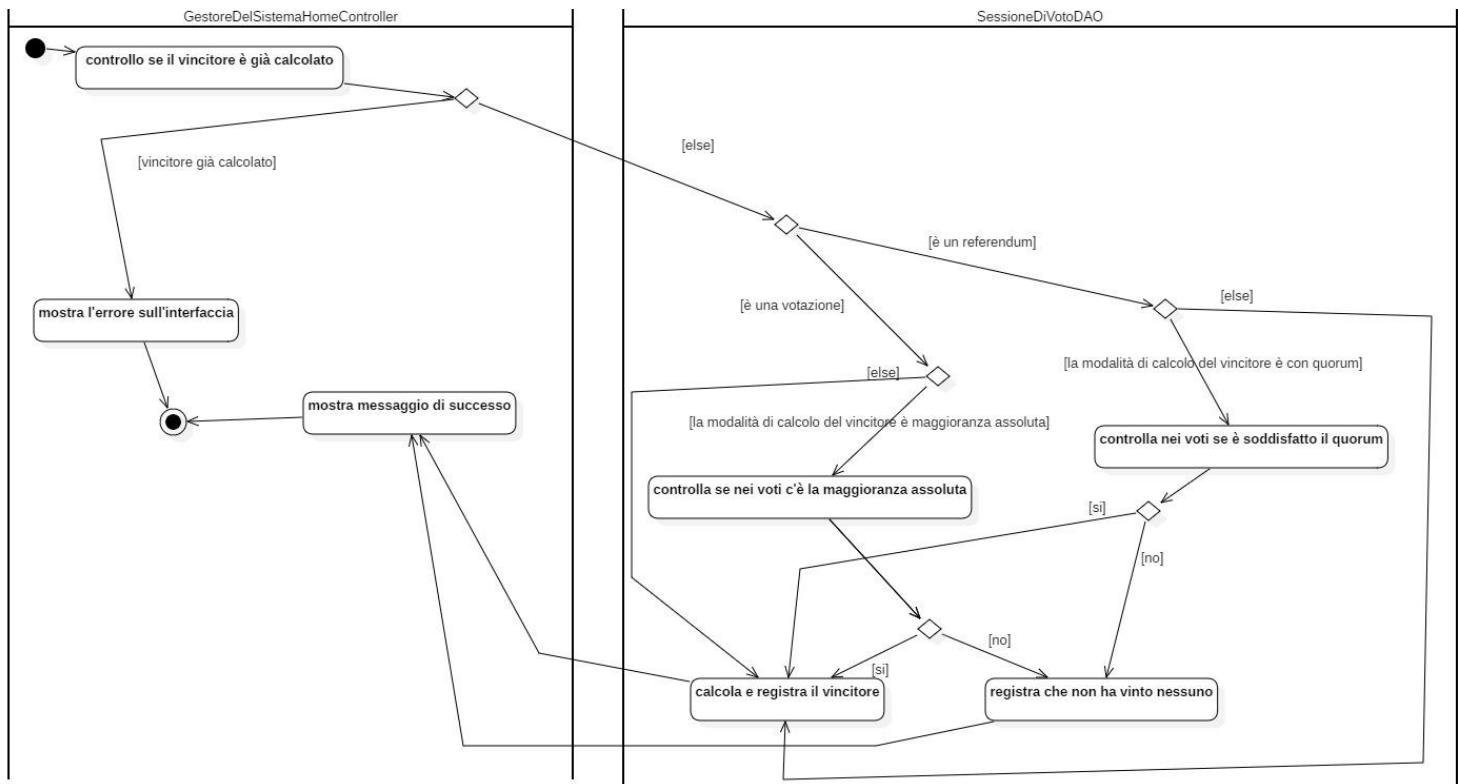


Figura 37: diagramma delle attività “inserisci voti in presenza”

Questo diagramma delle attività rappresenta l’ordine di esecuzione delle azioni che permettono a un gestore del sistema di inserire i voti cartacei espressi al seggio. Sottolineo che con l’attività “voto elettronico” si intende che quell’attività rimanda al diagramma di attività “voto elettronico (a distanza)”, dove il gestore del sistema potrà registrare uno alla volta i voti cartacei.

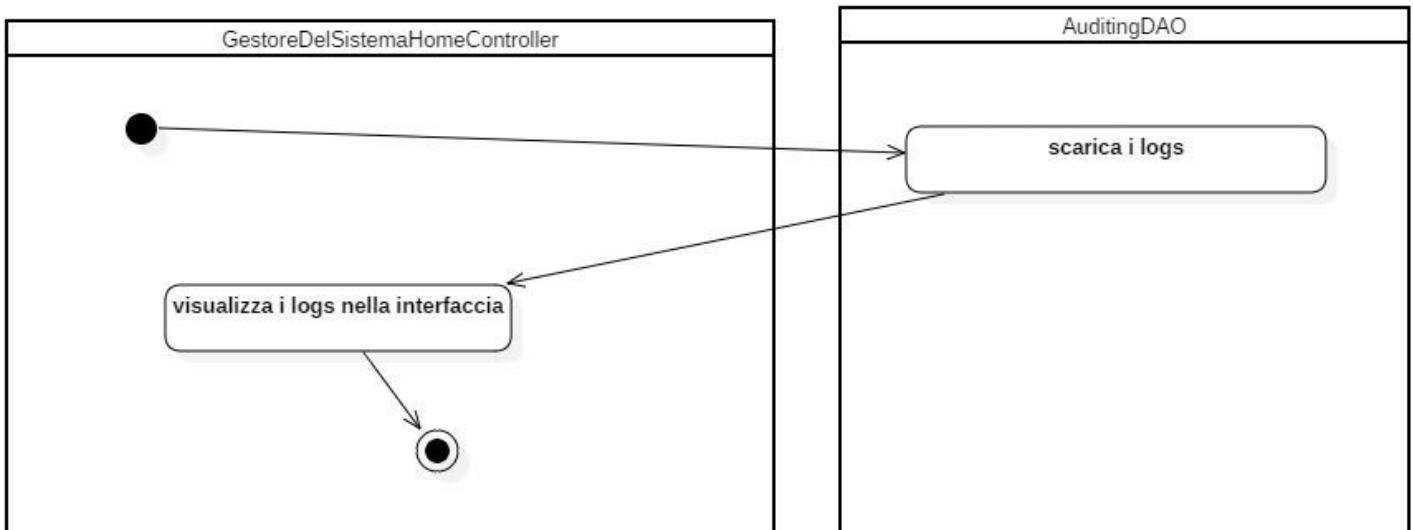
## Calcolo del vincitore



*Figura 38: diagramma delle attività “calcolo del vincitore”*

Questo diagramma delle attività rappresenta l’ordine di esecuzione delle azioni che permettono al sistema di calcolare il vincitore di una sessione di voto, quando un gestore del sistema attiva il trigger per calcolare un vincitore.

## Visualizza log



*Figura 39: diagramma delle attività “visualizza log”*

Questo diagramma delle attività rappresenta l’ordine di esecuzione delle azioni che permettono a un gestore del sistema di visualizzare i log del sistema doo aver attivato il trigger dello scenario.

### Inserisci un nuovo partito

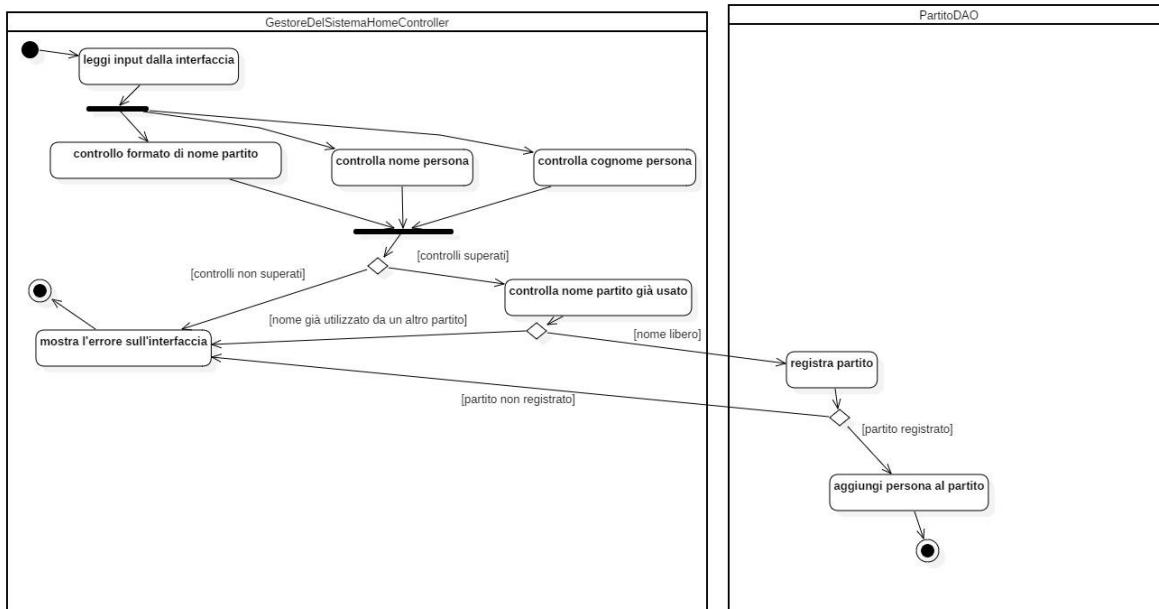


Figura 40: diagramma delle attività “inserisci un nuovo partito”

Questo diagramma delle attività rappresenta l’ordine di esecuzione delle azioni che permettono a un gestore del sistema di aggiungere un nuovo partito nel sistema sw dopo aver inserito gli input richiesti ed aver attivato il trigger dello scenario.

## 2.6. Macchine di stato

Le macchine di stato da me prodotte sono due e si riferiscono una alla sessione di voto (classe “SessioneDiVotoDTO”), mentre la seconda agli utenti (classe UtenteDTO). Non ho realizzato altre macchine di stato perché le altre classi DTO hanno uno stato quasi immutabile oppure con mutamenti non particolarmente significativi ed inoltre le altre classi non DTO non sono statefull oppure hanno uno o due attributi immutabili.

## SessioneDiVotoDTO

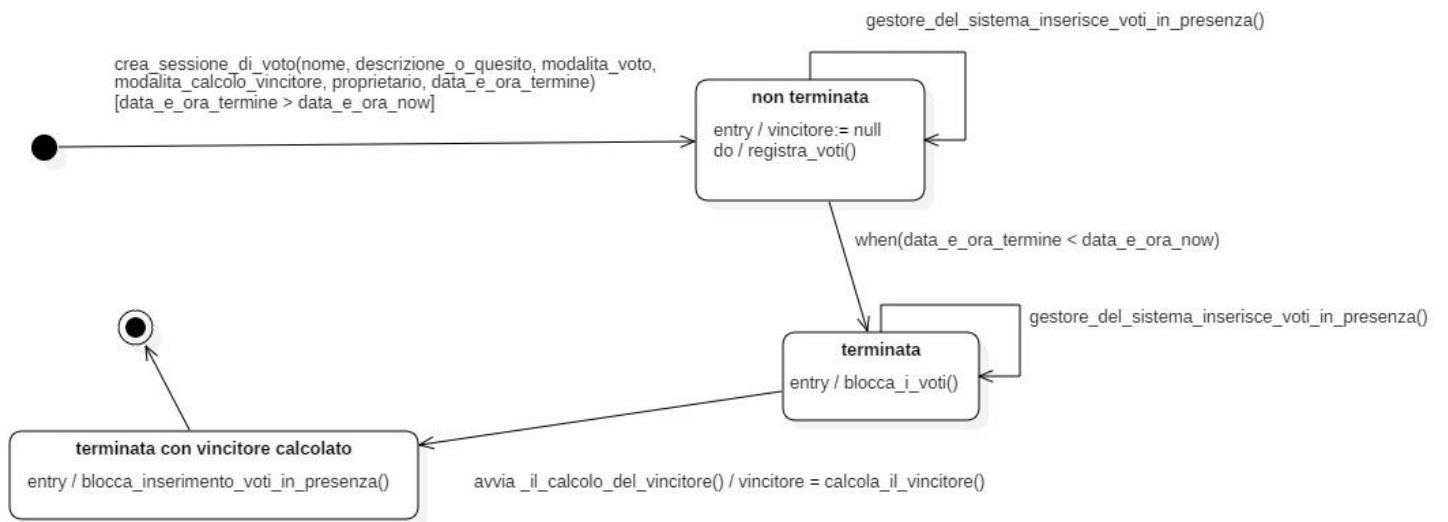


Figura 41: Macchina di stato di una sessione di voto

## UtenteDTO

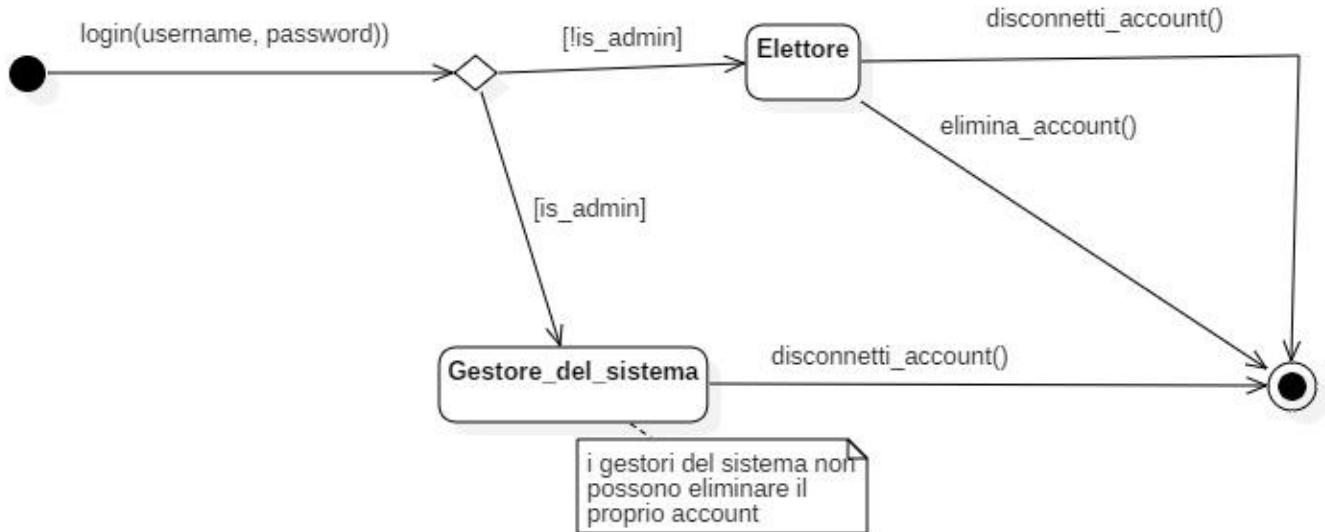


Figura 42: Macchina di stato di un utente

## 2.7. Diagramma dei componenti

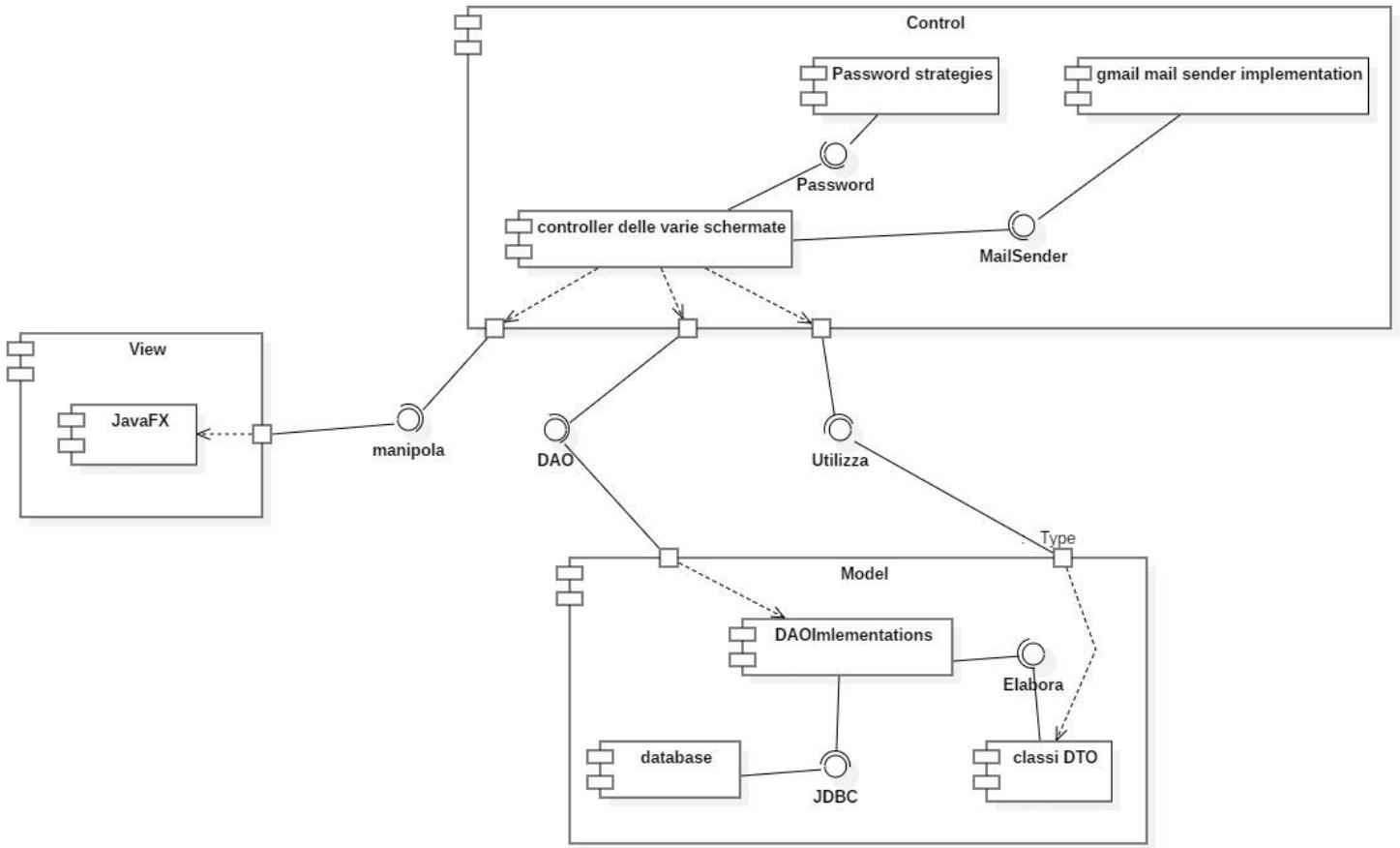


Figura 43: diagramma dei componenti

**Nota sul diagramma i componenti:** “password strategies” e “gmail mail sender implementation” sono dei componenti molto piccoli, composti il primo da 2 classi ed il secondo da una sola classe. Ho deciso comunque di esplicitarli nel diagramma come componenti perché sono completamente indipendenti e possono essere riutilizzate in altri sistemi mediante le loro interfacce. Un’altra considerazione è che in questo progetto sono dei componenti molto piccoli ma magari in altri contesti potrebbe essere utile avere molte più strategie di controllo e generazione delle password oppure avere diversi server di posta al quale collegarsi.

## 3. Implementazione del sistema

### 3.1. Diagramma delle classi a livello di programma

Ricordo che il diagramma delle classi a livello di programma l’ho inserito nella sezione di progettazione insieme alla discussione dei design pattern utilizzati.

## 3.2. Gestione dei dati persistenti

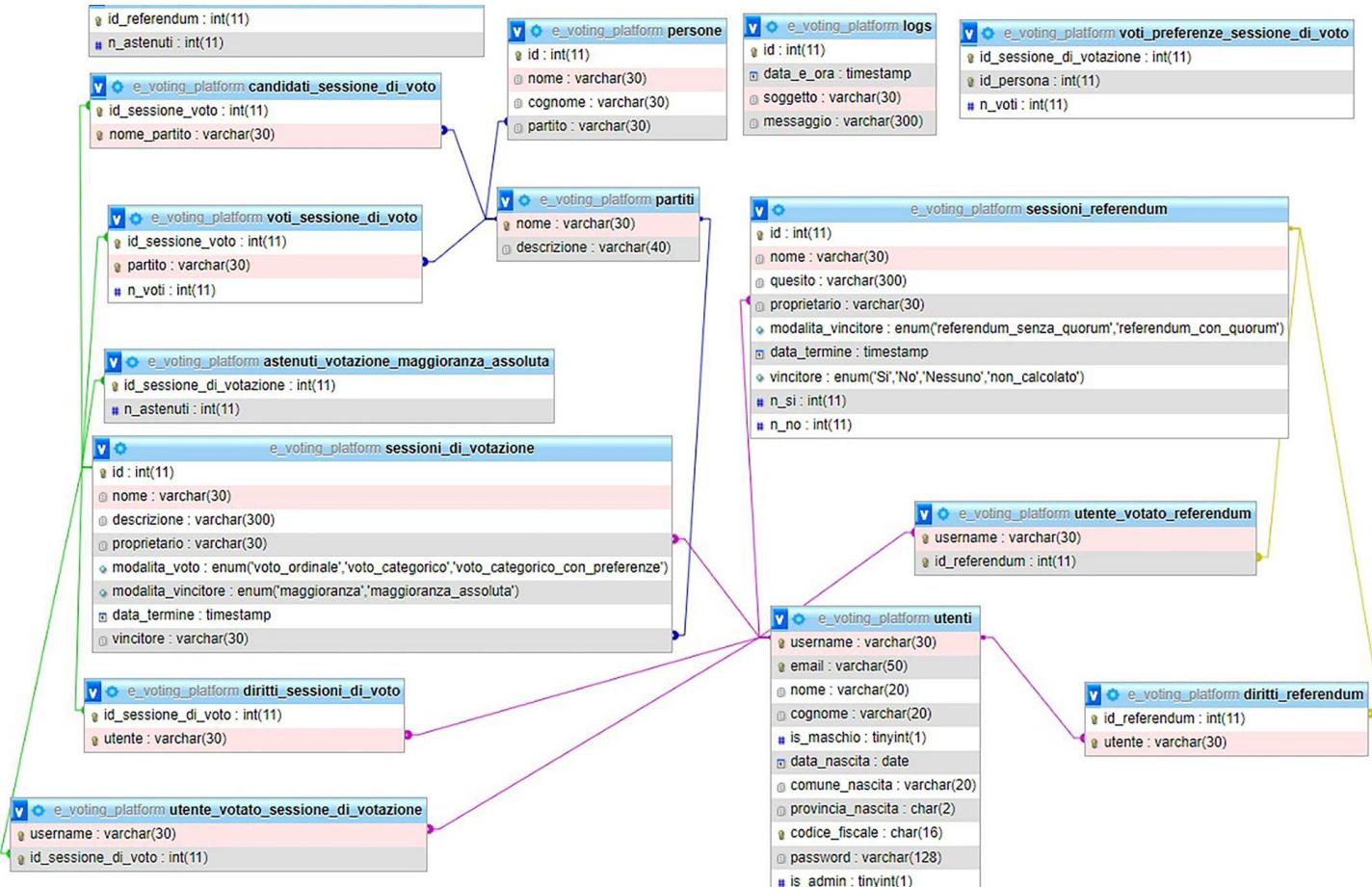


Figura 44: Schema del database MySQL

### Breve descrizione delle tabelle:

- **Utenti**: tabella utilizzata per salvare le informazioni degli elettori e dei gestori del sistema insieme agli hash delle password;
- **Diritti\_referendum e diritti\_sessions\_di\_voto**: servono per associare un utente ad una sessione di voto se ha il diritto di voto;
- **sessioni\_di\_votazione**: contiene le informazioni delle votazioni;
- **sessioni\_referendum**: contiene le informazioni dei referendum;
- **candidati\_sessione\_di\_voto**: serve per associare un partito candidato ad una sessione di voto;
- **utente\_votato\_referendum e utente\_votato\_sessione\_di\_votazione**: serve per registrare che un utente ha espresso un voto ad una sessione di voto o ad un referendum. Non viene registrata la preferenza espressa mediante il voto ma solo che l'elettore ha espresso il proprio voto. Questo per garantire la segretezza del voto, infatti nel database non è presente in nessuna tabella l'associazione

<utente - voto - sessione di voto>, ma anche per sapere quando un elettore ha già espresso il proprio voto in una sessione di voto;

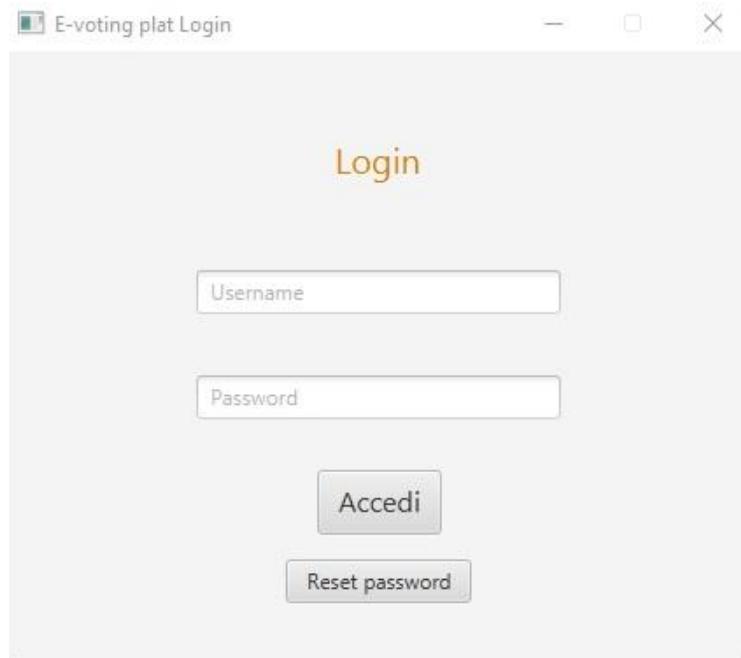
- **voti\_sessione\_di\_voto:** serve per salvare i voti ricevuti da ogni candidato in una sessione di voto. I voti vengono salvati come numero cumulativo appunto per non poter ricondurre un voto ad un elettore;
- **voti\_referenze\_sessione\_di\_voto:** voti riferiti alle persone di un partito nelle votazioni con modalità di voto “voto categorico con preferenze”;
- **partiti:** contiene le informazione dei partiti;
- **persone:** contiene le informazioni delle persone ed il partito al quale appartengono;
- **logs:** contengono i log del sistema;
- **astenuti\_referendum\_con\_quorum e astenuti\_votazione\_maggioranza\_assoluta:** serve per registrare il numero di astenuti ad una sessione di voto. Gli astenuti sono salvati come un numero cumulativo e non è possibile ricondurre il numero di astenuti agli elettori.

Ovviamente le informazioni che diventano “inutili” dopo aver calcolato il vincitore di una sessione di voto, vengono eliminate per mantenere, per quanto possibile, un dimensione ridotta delle tabelle.

### 3.3. Descrizione dell’interfaccia grafica

In questa sezione riporto gli screenshots delle principali schermate e le corrispondenti valutazioni fatte dall’interfaccia sul formato degli input, ad esempio controllando la loro lunghezza, se contendono determinati caratteri, ecc. Non sono riportate le validazioni effettuate mediante il database come, ad esempio, il controllo di unicità di un input. Queste tipologie di controlli sono visibili dai diagrammi di sequenza.

## Login



### ***validazione degli input:***

- username deve avere una lunghezza compresa tra 6 e 30 (compresi);
- password deve avere una lunghezza compresa tra 6 e 20 (compresi) e deve contenere almeno una lettera maiuscola ed un carattere speciale.

## Gestione account (elettore e gestore del sistema) / registrazione nuovo elettore (gestore del sistema)

Utente: luca\_maccarini  
Gestore del sistema

▼ Gestione Account / Registrazione nuovo elettore

Modifica password

Registra un nuovo elettore

Username

E-mail

Nome

cognome

Data di nascita

Comune nascita

Codice fiscale:

Sesso ("M" o "F")

Identificatore della provincia di nascita (es: MI)

Registra elettore

Gestisci le sessioni di voto

Visualizza vincitori

Gestione Partiti

Visualizza log

### **validazione degli input per modificare la password:**

- password attuale, nuova password e conferma della nuova password devono avere una lunghezza compresa tra 6 e 20 (compresi) e deve contenere almeno una lettera maiuscola ed un carattere speciale. Inoltre la nuova password e la sua ripetizione devono coincidere.

### **validazione degli input per registrare un nuovo elettore:**

- username deve avere una lunghezza compresa tra 6 e 30 (compresi);

- l'e-mail deve avere il formato corretto di una e-mail ed una lunghezza compresa tra 1 e 50 (compresi);
- il nome deve avere una lunghezza compresa tra 1 e 20 caratteri (compresi);
- il cognome deve avere una lunghezza compresa tra 1 e 20 caratteri (compresi);
- la data di nascita non può essere vuota e deve essere una data precedente a quella attuale;
- il comune di nascita deve avere una lunghezza compresa tra 1 e 20 caratteri (compresi);
- il formato del codice fiscale viene controllato mediante il medesimo codice consegnato negli assignments;
- il campo del sesso deve contenere una sola lettera tra "M" o "F" (anche minuscole);
- l'identificatore della provincia deve contenere solo due lettere.

## Creazione di una votazione (gestore del sistema)

The screenshot shows a window titled "Nuova votazione". On the left, there are fields for "Nome" (Name) and "Descrizione" (Description). Below these are dropdown menus for "Modalità calcolo del vincitore" (Calculation mode) and "Modalità di voto" (Voting mode), along with a date/time input field showing "Alle ore" (All hours) and "hh : mn" (hours : minutes). A "Crea" (Create) button is at the bottom left. In the center and right, two lists are displayed under the heading "Seleziona gli elettori che avranno diritto al voto tieni premuto CTRL + click su chi vuoi selezionare" (Select voters who have the right to vote, hold down CTRL + click on whom you want to select) and "Seleziona i partiti candidati CTRL + click su chi vuoi selezionare" (Select candidate parties, hold down CTRL + click on whom you want to select). The voter list contains entries like "alessia\_bertoli | alessia, bertoli", "alessio\_verga | alessio, verga", "Ester\_Monaldo | Ester, Monaldo", and "test | test, test". The candidate party list contains "test partito 1 | descrizione 1", "test partito 2 | descrizione 2", and "test partito 3 | descrizione 3".

### **validazione degli input:**

- il nome deve avere una lunghezza compresa tra 1 e 20 (compresi);
- la descrizione deve avere una lunghezza minore o uguale di 300 caratteri;
- la modalità di voto e di calcolo del vincitore devono essere selezionate;
- la data e l'ora di termine devono indicare una data e ora successivi alla data e ora attuali;
- deve essere selezionato almeno un elettore avente diritto al voto dalla lista degli elettori;
- deve essere selezionato almeno un candidato dalla lista dei candidati.

## Creazione di una referendum (gestore del sistema)

Nuovo referendum

Nome

Quesito

max 300 caratteri

Modalità calcolo del vincitore

Alle ore hh : mn

Crea

Seleziona gli elettori che avranno diritto al voto  
tieni premuto CTRL + click su chi vuoi selezionare

alessia_bertoli   alessia, bertoli
alessio_verga   alessio, verga
Ester_Monaldo   Ester, Monaldo
test   test, test

### **validazione degli input:**

- il nome deve avere una lunghezza compresa tra 1 e 20 (compresi);
- il quesito deve avere una lunghezza compresa tra 1 e 300 caratteri (compresi);
- la modalità di calcolo del vincitore deve essere selezionata;
- la data e l'ora di termine devono indicare una data e ora successivi alla data e ora attuali;
- deve essere selezionato almeno un elettore avente diritto al voto dalla lista degli elettori.

## Gestione dei partiti e delle persone nei partiti (gestore del sistema)

Utente: luca\_maccarini  
Gestore del sistema

▶ Gestione Account / Registrazione nuovo elettore  
▶ Gestisci le sessioni di voto  
▶ Visualizza vincitori  
▼ Gestione Partiti

aggiorna

test partito 1 | descrizione 1  
test partito 2 | descrizione 2  
test partito 3 | descrizione 3

Crea un nuovo partito

Nome partito  
Descrizione del partito  
Nome persona  
Cognome persona

Aggiungi partito

Aggiungi una persona al partito selezionato

Nome  
Cognome  
Aggiungi persona

▶ Visualizza log

### ***inserimento nuovo partito validazione degli input:***

- il nome del partito deve avere una lunghezza compresa tra 1 e 30 (compresi);
- la descrizione del partito deve avere una lunghezza compresa tra 1 e 40 (compresi).

### ***inserimento nuova persona in un partito validazione degli input:***

- il nome della persona deve avere una lunghezza compresa tra 1 e 30 (compresi);
- il cognome della persona deve avere una lunghezza compresa tra 1 e 30 (compresi);

- deve essere selezionato dalla lista dei partiti il partito al quale apparterrà la nuova persona.

## voto categorico e registrazione voto categorico (elettore e gestore del sistema)

### **validazione degli input:**

- prima di votare bisogna selezionare il partito.

**voto categorico con preferenze e registrazione voto categorico con preferenze  
(elettore e gestore del sistema)**

### **validazione degli input:**

- prima di votare bisogna selezionare il partito ed ordinare le preferenze.

**voto ordinale e registrazione voto ordinale (elettore e gestore del sistema)**

Votazione: test votazione ordinale

**Tipo di votazione:** Voto ordinale  
**Proprietario:** luca\_maccarini  
**Data di termine:** 18/3/2032 alle ore 11:11  
**Modalità di calcolo del vincitore:** maggioranza  
**Descrizione:**  
descrizione voto ordinale

**Istruzioni:**  
Riordina i seguenti partiti e clicca il pulsante vota

test partito 1 | descrizione 1

test partito 2 | descrizione 2

test partito 3 | descrizione 3

^  
▼

E possibile votare

**Vota**   **Scheda bianca (astenuto)**

### **validazione degli input:**

- prima di votare bisogna ordinare i partiti.

## Controllare il diritto di voto e lo stato del voto di un elettore in una sessione di voto (gestore del sistema)

Gestione Sessione di voto: test votazione ordinale

**Tipo di votazione:** Voto ordinale  
**Proprietario:** luca\_maccarini  
**Data di termine:** 18/3/2032 alle ore 11:11  
**Modalità di calcolo del vincitore:** Maggioranza  
**Descrizione:** descrizione voto ordinale

▼ Controlla diritto di voto e stato di voto di un elettore

Controllare in presenza se un elettore:  
- ha il diritto di voto per a questa sessione di voto  
- ha già espresso il proprio voto a questa sessione di voto

Username elettore  Controlla

▶ Segna che un elettore ha votato in presenza  
▶ inserimento voti in presenza e calcolo del vincitore

### validazione degli input:

- l'username deve avere una lunghezza compresa tra 1 e 30 (compresi).

## Registrare che una persona elettore A. ha espresso un voto cartaceo in una sessione di voto (gestore del sistema)

Gestione Sessione di voto: test votazione ordinale X

**Tipo di votazione:** Voto ordinale  
**Proprietario:** luca\_maccarini  
**Data di termine:** 18/3/2032 alle ore 11:11  
**Modalità di calcolo del vincitore:** Maggioranza  
**Descrizione:** descrizione voto ordinale

► Controlla diritto di voto e stato di voto di un elettore  
▼ Segna che un elettore ha votato in presenza

Registrare nel sistema che l'elettore ha votato in presenza a questa sessione di voto, così che non possa esprimere un altro voto mediante la piattaforma sw o ritornando al seggio in un altro momento

Username elettore  Registra

► inserimento voti in presenza e calcolo del vincitore

### **validazione degli input:**

- l'username deve avere una lunghezza compresa tra 1 e 30 (compresi).

### **Riporto altre schermate dell'applicazione che non hanno input:**

## **visualizza log (gestore del sistema)**

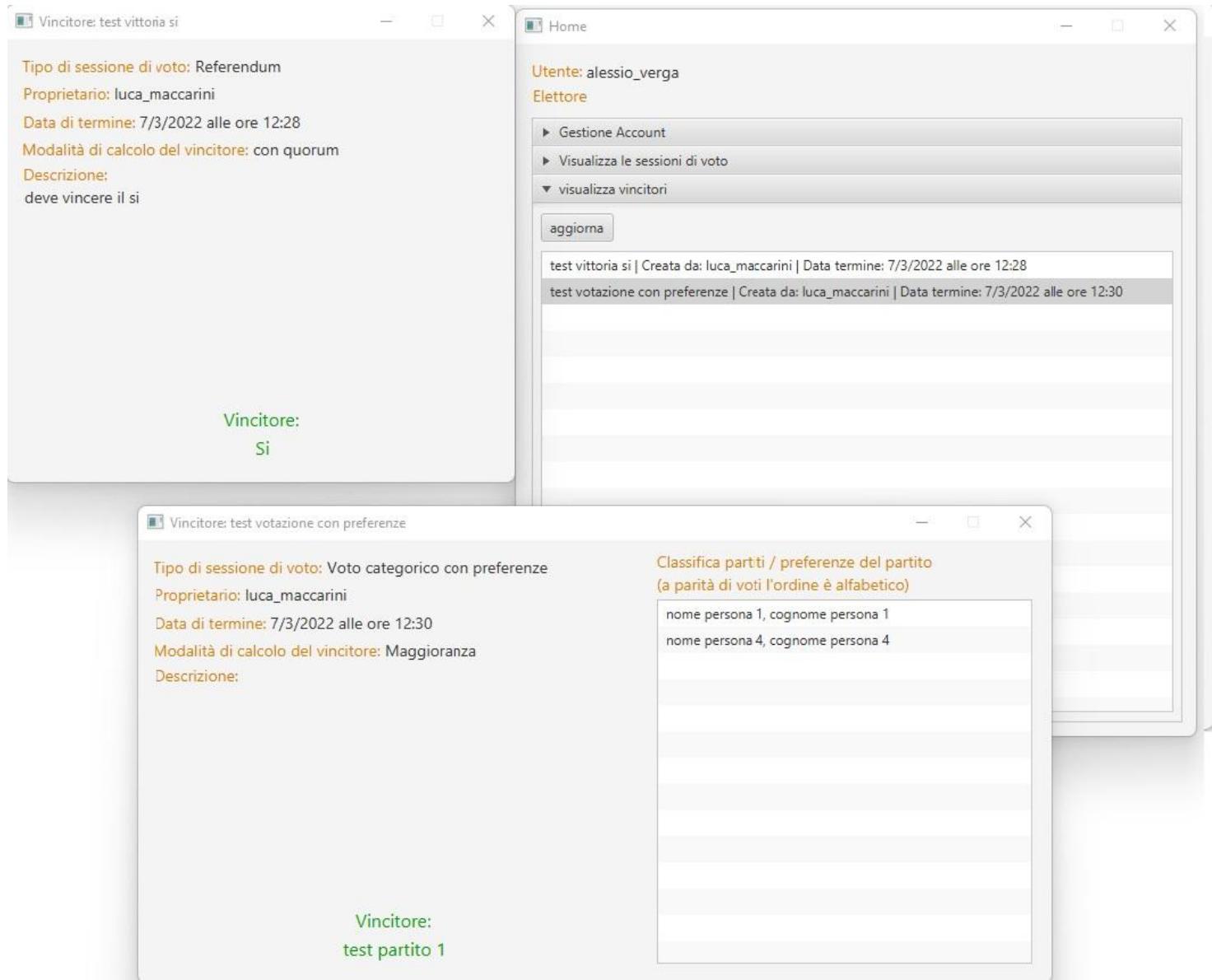
The screenshot shows a Windows application window titled "Home". Inside, there's a sidebar with the user "luca\_maccarini" and the title "Gestore del sistema". A list of options includes "Gestione Account / Registrazione nuovo elettore", "Gestisci le sessioni di voto", "Visualizza vincitori", "Gestione Partiti", and "Visualizza log". Below this is a section with two buttons: "aggiorna" and "elimina tutti i log". The main area displays a scrollable list of log entries:

- REGISTRATO DA: luca\_maccarini - DATA REGISTRAZIONE: 7/3/2022 alle ore 14:29 - MESSAGGIO DEL L...
- REGISTRATO DA: luca\_maccarini - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:30 - MESSAGGIO DEL L...
- REGISTRATO DA: luca\_maccarini - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:30 - MESSAGGIO DEL L...
- REGISTRATO DA: luca\_maccarini - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:29 - MESSAGGIO DEL L...
- REGISTRATO DA: luca\_maccarini - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:29 - MESSAGGIO DEL L...
- REGISTRATO DA: luca\_maccarini - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:26 - MESSAGGIO DEL L...
- REGISTRATO DA: luca\_maccarini - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:26 - MESSAGGIO DEL L...
- REGISTRATO DA: luca\_maccarini - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:26 - MESSAGGIO DEL L...
- REGISTRATO DA: alessia\_bertoli - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:20 - MESSAGGIO DEL L...
- REGISTRATO DA: alessio\_verga - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:20 - MESSAGGIO DEL L...
- REGISTRATO DA: alessio\_verga - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:20 - MESSAGGIO DEL L...
- REGISTRATO DA: alessia\_bertoli - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:20 - MESSAGGIO DEL L...
- REGISTRATO DA: alessia\_bertoli - DATA REGISTRAZIONE: 7/3/2022 alle ore 12:20 - MESSAGGIO DEL L...

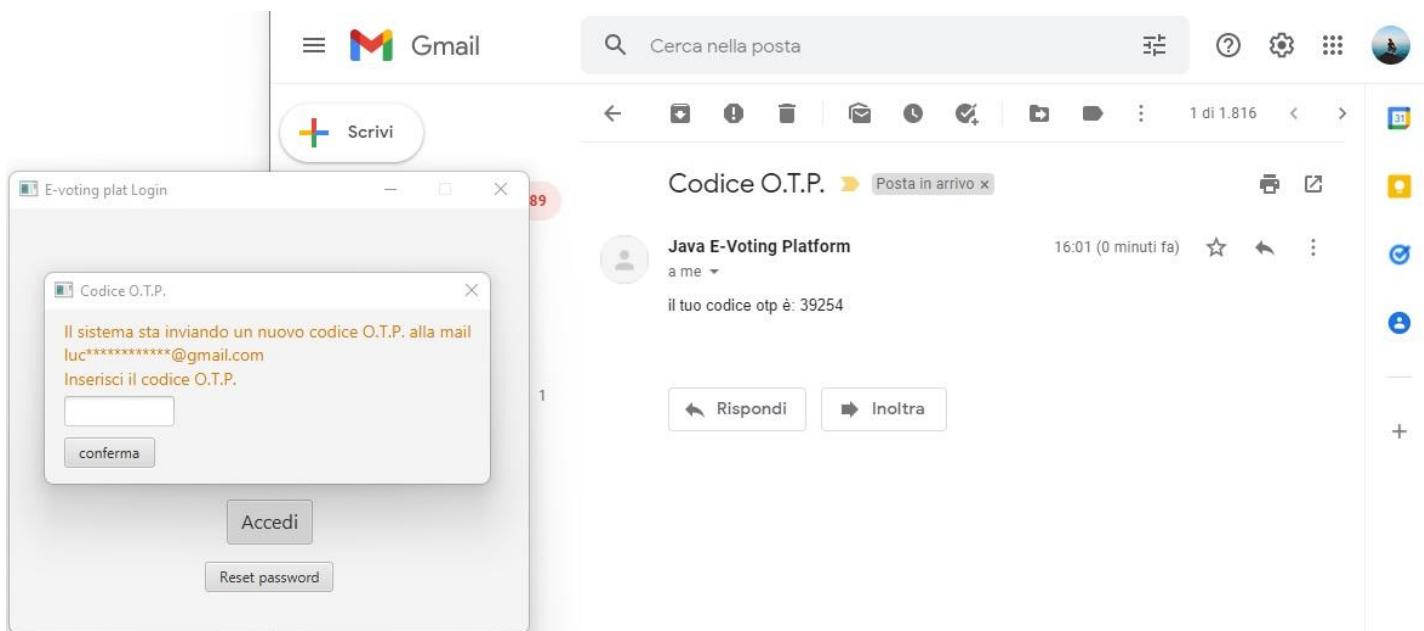
## **visualizza vincitore (elettore e gestore del sistema)**

riporto il vincitore di un referendum e di una votazione con modalità di voto

“categorico con preferenze” le altre modalità di voto presentano schermate simili



### Schermata per inserire il codice O.T.P. (elettore e gestore del sistema)



### 3.4. Diagramma di deployment

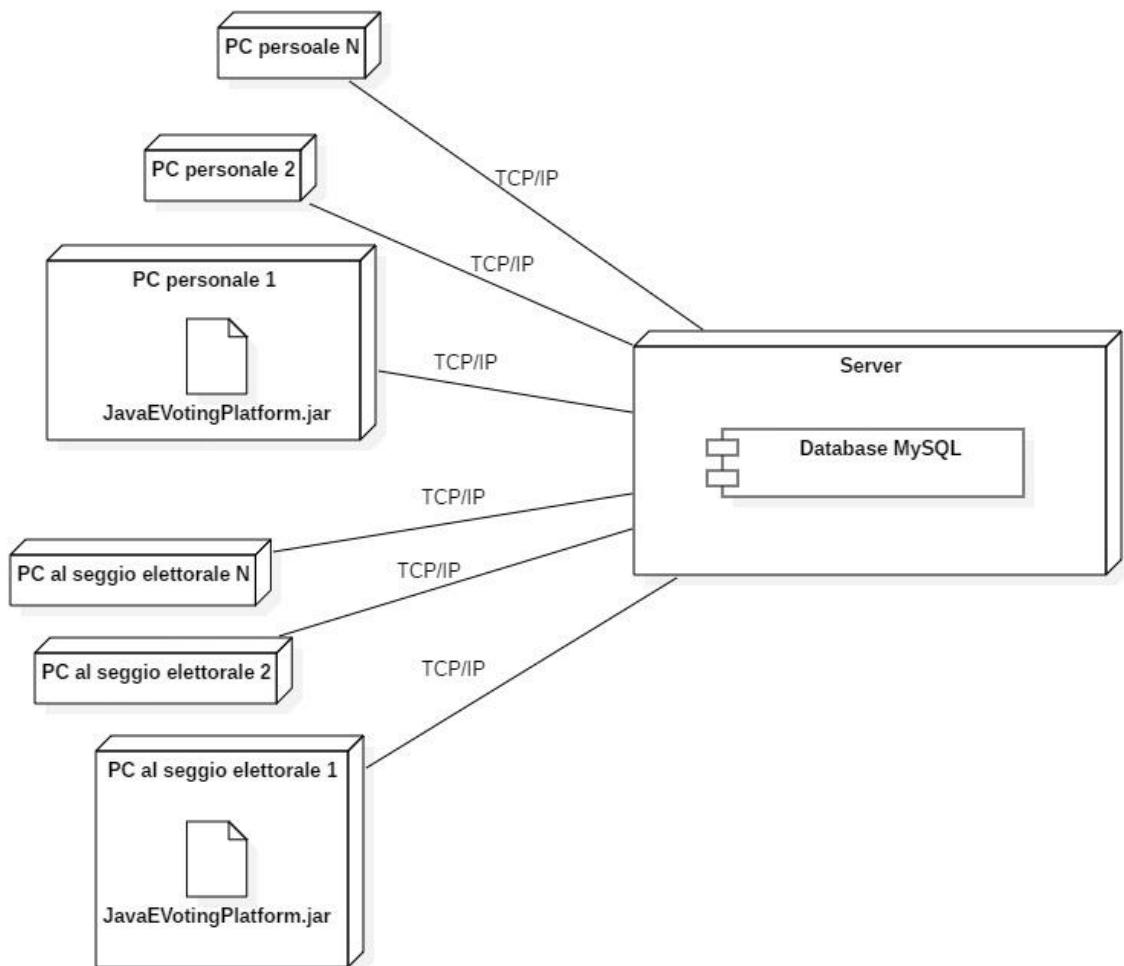


Figura 45: Diagramma di deployment

### 3.5. Specifica e verifica dei vincoli

Per non rendere troppo lunga la specifica di tutti i vincoli di tutte le classi ho deciso di riportare solo i vincoli delle principali classi DTO. Sono consapevole che se stessi creando un vero lavoro commissionato da un vero cliente dovrei riportare i vincoli di tutte le classi, anche le precondizioni e le post condizioni delle classi che implementano le interfacce DAO. Per una migliore chiarezza i vincoli mappati in jml sono di colore blu.

context **UtenteDTO** inv:

```
self.username.size() > 0 and self.username.size() <=30 and  
self.email.size() > 0 and self.email.size() <= 50 and  
mailSender.check_email(self.email) and  
self.nome.size() > 0 and self.nome.size() <= 20 and  
self.cognome.size() > 0 and self.cognome.size() <= 20 and  
self.data_nascita <> null and self.data_nascita.isBefore(LocalDateTime.now()) and  
self.comune_nascita.size() > 0 and self.comune_nascita.size() < 20 and  
self.provincia_nascita.size() = 2 and  
CodiceFiscale.check_CF(self.nome, self.cognome, self.is_maschio,  
self.data_nascita, "italia", self.codice_fiscale)
```

per la mappatura jml bisogna definire come “pure” i seguenti metodi:

```
public class MailSenderImplementation implements MailSender {  
    ...  
    public static /*@ pure */ MailSenderImplementation get_singleton_instance(){  
        ...  
    }  
    ...  
    public /*@ pure */ boolean check_email(String email){...}  
    ...  
}  
  
public class CodiceFiscale{  
    ...  
    public static /*@ pure */ CodiceFiscale get_singleton_instance() {...}  
    ...  
    public /*@ pure */ boolean Check_CF(.....) {...}  
    ...  
}  
  
public abstract class UtenteDTO {  
    ...  
    /*@
```

```

invariant
username.length() > 0 && username.length() <= 30 &&
email.length() > 0 && email.length() <= 50 &&
mailSenderImplementation.get_singleton_instance().check_email(email) &&
nome.length() > 0 && nome.length() <= 20 &&
cognome.length() > 0 && cognome.length() <= 20 &&
data_nascita != null && data_nascita.isBefore(LocalDateTime.now()) &&
comune_nascita.length() > 0 && comune_nascita.length() < 20 &&
provincia_nascita.length() = 2 &&
CodiceFiscale.get_singletone_istance().check_CF(self.nome, self.cognome,
self.is_maschio, self.data_nascita, "italia", self.codice_fiscale);
@*/
...
}

```

context **ElettoreDTO** inv:  
`self.is_admin = false`

```

public class ElettoreDTO extends UtenteDTO {
...
/*@
invariant is_admin == false;
@*/
...
}

```

context **GestoreDelSistemaDTO** inv:  
`self.is_admin = true`

```

public class GestoreDelSistemaDTO extends UtenteDTO {
...
/*@
invariant is_admin == true;
@*/
...
}

```

context **SessioneDiVotoDTO** inv:

```

self.nome.size() > 0 and self.nome.size()<=30 and
self.proprietario.size() > 0 and self.proprietario.size() < 30 and
self.data_e_ora_termine <> null

public abstract class SessioneDiVotoDTO {

...
/*@
invariant
nome.length() > 0 && nome.length() <= 30 &&
proprietario.length() > 0 && proprietario.length() < 30 &&
data_e_ora_termine != null;
@*/
...
}

```

context **SessioneDiVotazioneDTO** inv:

```

self.candidati->size()>0 and
self.modalita_di_voto <> null and
self.descrizione.size() <= 300 and
self.modalita_vincitore <> null and
self.vincitore <> null implies
Timestamp(System.currentTimeMillis()).after(self.data_e_ora_termine)

```

context **SessioneDiVotazioneDTO::set\_partiti\_candidati**(candidati: Sequence)

```

pre: candidati->size() > 0
post: self.candidati = candidati

```

context **SessioneDiVotazioneDTO::set\_vincitore**(vincitore: PartitoDTO)

```

pre: vincitore <> null and
Timestamp(System.currentTimeMillis()).after(self.data_e_ora_termine)
post: self.vincitore = vincitore

```

```

public class SessioneDiVotazioneDTO extends SessioneDiVotoDTO {

...
private /*@ spec public */ List<PartitoDTO> candidati;
private /*@ spec public */ PartitoDTO vincitore;
...
/*@
invariant

```

```

candidati.size()>0 &&
modalita_di_voto != null &&
descrizione.length() <= 300 &&
modalita_vincitore != null &&
vincitore != null =>
Timestamp(System.currentTimeMillis()).after(data_e_ora_termine);
@*/
...

/*@
requires candidati.size() > 0;

ensures this.candidati = candidati;
@*/
public void set_partiti_candidati(List<PartitoDTO> candidati){...}
...

/*@
requires vincitore != null &&
Timestamp(System.currentTimeMillis()).after(data_e_ora_termine);

ensures this.vincitore = vincitore;
@*/
public void set_vincitore(PartitoDTO vincitore){...}

...
}

```

context **ReferendumDTO** inv:

self.descrizione.size() > 0 and self.descrizione.size() <= 300 and  
 self.modalita\_vincitore<>null and  
 self.vincitore <> null implies  
 Timestamp(System.currentTimeMillis()).after(self.data\_e\_ora\_termine)

context **ReferendumDTO::set\_vincitore(vincitore: String)**  
 pre: (vincitore = "Si" or vincitore="No") and  
 Timestamp(System.currentTimeMillis()).after(self.data\_e\_ora\_termine)  
 post: self.vincitore = vincitore

public class ReferendumDTO extends SessioneDiVotoDTO {

```

...
/*@
invariant
descrizione.length() > 0 and descrizione.length() <= 300 &&
modalita_vincitore != null &&
self.vincitore != null =>
Timestamp(System.currentTimeMillis()).after(self.data_e_ora_termine);
@*/

```

```

/*@
requires
(vincitore.equals("Si") || vincitore.equals("No") ) &&
Timestamp(System.currentTimeMillis()).after(data_e_ora_termine);

ensures this.vincitore = vincitore;
@*/

```

```

public void set_vincitore(String vincitore){...}
...
```

context **PartitoDTO** inv:

self.nome.size() > 0 and self.nome.size() <= 30 and  
self.persone->size() > 0

context **PartitoDTO::set\_persone**(persone: Sequence(PersonaDTO))  
pre: persone->size > 0  
post: self.persone = persone

```

public class PartitoDTO {
...
    private /*@ spec public */ List<PersonaDTO> persone;
...
/*@
invariant
nome.length() > 0 && nome.length() <= 30 &&
persone.size() > 0;
@*/
...
/*@

```

```

    requires persone.size() > 0
    ensures this.persone = persone
    */
    public void set_persone(List<PersonaDTO> persone) {...}

    ...
}

```

## 3.6. Descrizione del testing

Il testing effettuato è incentrato su 3 classi: "UtenteDAOImplementation", "ElettoreDAOImplementation" e "GestoreDelSistemaDAOImplementation".

Di queste classi ho testato solo i metodi che ritenevo più critici ed il criterio utilizzato è il **criterio di copertura delle decisioni**. Riporto gli screenshots dei vari metodi testati mostrando la copertura del codice.

**Per vedere il codice delle classi di test, guardare nella directory “src/test/java”.**

L'esecuzione dei test modifica i dati sul database, perciò ho realizzato un esportazione sql del database con inseriti i dati per il testing. Il file contente l'esportazione è "e\_voting\_platform.sql", va importato e lo stato del database non deve essere modificato prima di eseguire i test. Inoltre, all'interno delle classi di test sono presenti dei metodi statici etichettati con @AfterAll, i quali permettono di eseguire più volte i test perché annullano le azioni effettuate dai test sul database.

### login\_utente (*UtenteDAOImplementation*)

```

@Override
public UtenteDTO login_utente(String username, String password) throws ClassNotFoundException, SQLException {
    Connection conn = connessione_db();

    String query = "SELECT * FROM `utenti` WHERE username = ? AND password LIKE SHA2(?, 512)";
    PreparedStatement statement= conn.prepareStatement(query);

    statement.setString(1, username);
    statement.setString(2, password);

    ResultSet resultSet = statement.executeQuery();

    //guarda se ci sono risultati
    if(resultSet.next()) {
        if(resultSet.getBoolean("is_admin")) {
            UtenteDTO output = new GestoreDelSistemaDTO(resultSet.getString("username"),resultSet.getString("email"), resultSet.getString("nome"),
                resultSet.getString("cognome"),resultSet.getBoolean("is_maschio"), resultSet.getDate("data_nascita"),
                resultSet.getString("comune_nascita"), resultSet.getString("provincia_nascita"), resultSet.getString("codice_fiscale"));
            chiudi_connessione(conn);
            return output;
        }
        else {
            UtenteDTO output = new ElettoreDTO(resultSet.getString("username"),resultSet.getString("email"), resultSet.getString("nome"),
                resultSet.getString("cognome"), resultSet.getBoolean("is_maschio"), resultSet.getDate("data_nascita"),
                resultSet.getString("comune_nascita"), resultSet.getString("provincia_nascita"), resultSet.getString("codice_fiscale"));
            chiudi_connessione(conn);
            return output;
        }
    }else {
        chiudi_connessione(conn);
        return null;
    }
}

```

Figura 46 copertura del metodo testato

## Vota\_referendum (*ElettoreDAOImplementation*)

```
    @Override
    //scelta == true ==> si altrimenti no
    public boolean vota_referendum(ElettoreDTO elettore, ReferendumDTO referendum, boolean scelta) throws ClassNotFoundException, SQLException, LogException {
        if(check_elettore_ha_gia_votato(elettore.username, referendum))
            return false;

        Connection conn = connessione_db();

        String query = "UPDATE `sessioni_referendum` SET ";
        if(scelta)
            query += " n_si = n_si + 1 WHERE sessioni_referendum.id = " + referendum.id;
        else
            query += " n_no = n_no + 1 WHERE sessioni_referendum.id = " + referendum.id;

        Statement statement = conn.createStatement();
        statement.executeUpdate(query);
        chiudi_connessione(conn);

        segna_che_elettore_ha_votato(elettore.username, referendum);
        Timestamp now = new Timestamp(System.currentTimeMillis());
        auditing.evento_generato_da.DAO(elettore, now, "ha votato al referendum: (id: " + referendum.id + ") " + referendum );
    }

    return true;
}
```

Figura 47 copertura del metodo testato

## voto\_ordinale\_sessione\_di\_votazione (*ElettoreDAOImplementation*)

```
public boolean voto_ordinale_sessione_di_votazione(ElettoreDTO elettore, SessioneDiVotazioneDTO sessione, List <PartitoDTO> partito) throws ClassNotFoundException, SQLException, LogException{
    if(check_elettore_ha_gia_votato(elettore.username, sessione))
        return false;

    Connection conn = connessione_db();
    String query = "";
    for(int i=0; i<partito.size(); i++) {
        if(check_presente_voto(sessione, partito.get(i)))
            query += "UPDATE `voti_sessione_di_voto` SET n_voti = n_voti+ " + i +" WHERE `id_sessione_voto` = " + sessione.id+ " AND `partito` ='" + partito.get(i).nome +"';\n";
        else
            query += "INSERT INTO `voti_sessione_di_voto`(`id_sessione_voto`, `partito`, `n_voti`) VALUES (" + sessione.id + ", '" + partito.get(i).nome + "', " + i + ");\n";
    }
    //System.out.println(query);
    Statement statement = conn.createStatement();
    statement.executeUpdate(query);
    chiudi_connessione(conn);

    segna_che_elettore_ha_votato(elettore.username, sessione);
    Timestamp now = new Timestamp(System.currentTimeMillis());
    auditing.evento_generato_da.DAO(elettore, now, "ha votato alla votazione: (id: " + sessione.id + ") " + sessione );
}

return true;
}
```

Figura 48 copertura del metodo testato

## voto\_categorico\_sessione\_di\_votazione (*ElettoreDAOImplementation*)

```

public boolean voto_categorico_sessione_di_votazione(ElettoreDTO elettore, SessioneDiVotazioneDTO sessione, PartitoDTO partito) throws ClassNotFoundException, SQLException, LogException{
    if(check_elettore_ha_gia_votato(elettore.username, sessione))
        return false;

    Connection conn = connessione_db();
    String query="";
    if(check_presente_voto(sessione, partito))
        query = "UPDATE `voti_sessione_di_voto` SET n_voti = n_voti+1 WHERE `id_sessione_di_voto` = "+sessione.id+" AND `voti_sessione_di_voto`.`partito` ='" + partito.nome +"'";
    else
        query = "INSERT INTO `voti_sessione_di_voto` (`id_sessione_voto`, `partito`, `n_voti`) VALUES ("+sessione.id+", '" +partito.nome+"', 1)";

    Statement statement = conn.createStatement();
    statement.executeUpdate(query);
    chiudi_connessione(conn);

    segna_che_elettore_ha_votato(elettore.username, sessione);
    Timestamp now = new Timestamp(System.currentTimeMillis());
    auditing.evento_generato_da.DAO(elettore, now, "Ha votato alla votazione: (id: " + sessione.id + ") " + sessione );
}

return true;
}

```

Figura 49 copertura del metodo testato

## voto\_categorico\_con\_preferenze\_sessione\_di\_votazione (ElettoreDAOImplementation)

```

public boolean voto_categorico_con_preferenze_sessione_di_votazione(ElettoreDTO elettore, SessioneDiVotazioneDTO sessione, PartitoDTO partito, List<PersonaDTO> ordine_persone) throws ClassNotFoundException, SQLException, LogException{
    if(voto_categorico_sessione_di_votazione(elettore, sessione, partito)) {
        Connection conn = connessione_db();
        String query = "";

        for(int i=0; i<ordine_persone.size(); i++) {
            if(check_presente_voto_preferenza(sessione, ordine_persone.get(i)))
                query += "UPDATE `voti_preference_sessione_di_voto` SET n_voti = n_voti+1 WHERE `id_sessione_di_votazione` = "+sessione.id+" AND `id_persona` = " + ordine_persone.get(i).id +";\n";
            else
                query += "INSERT INTO `voti_preference_sessione_di_voto` (`id_sessione_di_votazione`, `id_persona`, `n_voti`) VALUES ("+sessione.id+", "+ordine_persone.get(i).id+", "+i+");\n";
        }

        Statement statement = conn.createStatement();
        statement.executeUpdate(query);
        chiudi_connessione(conn);

        return true;
    }else {
        return false;
    }
}

```

Figura 50 copertura del metodo testato

## check\_elettore\_ha\_già\_votato (ElettoreDAOImplementation)

```

public boolean check_elettore_ha_votato(String username, SessioneDiVotoDTO sessione) throws SQLException, ClassNotFoundException {
    Connection conn = connessione_db();

    if(sessione instanceof SessioneDiVotazioneDTO) {
        String query = "SELECT username FROM `utente_votato_sessione_di_votazione` WHERE `id_sessione_di_voto` = "+ sessione.id +" AND `username` = '" + username + "'";
        Statement statement = conn.createStatement();
        ResultSet result = statement.executeQuery(query);
        boolean output = result.next();
        chiudi_connessione(conn);
        return output;
    }else {
        String query = "SELECT username FROM `utente_votato_referendum` WHERE `id_referendum` = "+ sessione.id +" AND `username` = '" + username + "'";
        Statement statement = conn.createStatement();
        ResultSet result = statement.executeQuery(query);
        boolean output = result.next();
        chiudi_connessione(conn);
        return output;
    }
}

```

Figura 51 copertura del metodo testato

## registra\_nuovo\_elettore (GestoreDelSistemaDAOImplementation)

```

@Override
public boolean registra_nuovo_elettore(GestoreDelSistemaDAO gestore, String username, String email, String nome, String cognome, boolean is_maschio, LocalDate data_nascita, String comune_nascita, String provincia_nascita, String codice_fiscale, String password) throws ClassNotFoundException, SQLException, LogException {
    //controlla se esiste già un utente con lo stesso username
    //oppure se esiste già un utente con lo stesso codice fiscale
    Connection conn = connessione_db();
    PreparedStatement statement = conn.prepareStatement(query);
    statement.setString(1, username);
    statement.setString(2, email);
    statement.setString(3, nome);
    statement.setString(4, cognome);
    statement.setBoolean(5, is_maschio);
    statement.setDate(6, Date.valueOf(data_nascita));
    statement.setString(7, comune_nascita);
    statement.setString(8, provincia_nascita);
    statement.setString(9, codice_fiscale);
    statement.setString(10, password);

    int r = statement.executeUpdate();
    chiudi_connessione(conn);

    Timestamp now = new Timestamp(System.currentTimeMillis());
    auditing.evento_generato_da.DAO(gestore, now, "Ha registrato un nuovo elettore con username " + username);
    return true;
}

```

## 3.7. Note per l'installazione e l'utilizzo

### 3.7.1 Progetto

Il progetto è un progetto maven compilato con Java11.

Per lo sviluppo ho utilizzato Eclipse 2019-09 R (4.13.0).

Come pacchetti esterni ho utilizzato:

- JavaFX11 per l'interfaccia;
- mysql-connector-5.0.8 come JDBC (allegato nella zip del progetto);
- hamcrest e junit-5 per il testing;
- javax.mail 5.0.8 per inviare le email

sono stati usati i seguenti vm arguments:

```
--module-path "$DIRECTORY_TO_JAVAFX/javafx-sdk-11\lib" --add-modules javafx.controls,javafx.fxml
```

dove per **DIRECTORY\_TO\_JAVAFX** è stato impostato il percorso dove risiede il pacchetto javafx-sdk-11

Per inviare le e-mail ho utilizzato un account gmail, le credenziali sono nella classe MailSenderImplementation.

### 3.7.2 Database

Per il database ho utilizzato un database mysql fornito dal pacchetto XAMPP, ho scelto il pacchetto xampp in quanto mi fornisce anche l'interfaccia phpmyadmin con la quale avevo già familiarità.

Sul database ho utilizzato le seguenti credenziali di accesso:

- username: root
- password -> per facilità d'accesso e scopi didattici non ne ho impostata una

#### **Import del database**

crea un nuovo database chiamato “e\_voting\_platform” con codifica dei caratteri “utf8mb4\_general\_ci”.

Naviga nella cartella “dump database” all'interno della zip del progetto dove sono presenti 2 esportazioni sql:

- “e\_voting\_platform con valori di test.sql”: contiene i dati per il testing junit
- “e\_voting\_platform installazione pulita.sql”: per avere una installazione nuova e pulita del sistema

Quindi per una nuova installazione pulita importa il file “e\_voting\_platform installazione pulita.sql”.

Questa impostazione fornisce già i seguenti utenti:

<b>Username</b>	<b>Password</b>	<b>è gestore del sistema</b>
alessia_bertoli	LaMiaPassword1#	no
alessio_verga	LaMiaPassword1#	no
ester_monaldo	LaMiaPassword1#	no
ale_zolla	LaMiaPassword1#	si
luca_maccarini	LaMiaPassword1#	si