

Università degli Studi di Padova

DIPARTIMENTO DI MATEMATICA "TULLIO LEVI-CIVITA"

CORSO DI LAUREA IN INFORMATICA



Confronto tra Angular e React nella realizzazione di una piattaforma di e-Voting

Tesi di laurea triennale

Relatore

Prof. Gilberto Filè

Laureando

Gabriel Bizzo

ANNO ACCADEMICO 2020-2021

Sommario

Il presente documento descrive il lavoro svolto durante il periodo di stage, della durata di circa trecentoventi ore, dal laureando Gabriel Bizzo presso l'azienda Sync Lab S.r.l. Il lavoro di stage si inserisce in un progetto, denominato Voting-Online, che consiste nello sviluppo di un'applicazione web nell'ambito e-Voting, ossia un sistema per permettere di eseguire delle votazioni online e il relativo monitoraggio da parte di un amministratore.

Gli obiettivi da raggiungere erano molteplici.

In primo luogo era richiesto lo studio del linguaggio di programmazione Java e del [framework](#) Spring, da effettuare come semplice approfondimento ma in seguito utilizzato per lo sviluppo del [Backend \(BE\)](#) del progetto Voting-Online.

In secondo luogo era richiesto lo studio delle tecnologie necessarie allo sviluppo del [Frontend \(FE\)](#) del suddetto progetto. In particolare i linguaggi di programmazione Javascript e Typescript, il [framework](#) Angular e la libreria React.

Infine era richiesta la progettazione, e successiva implementazione, delle maschere necessarie al funzionamento dell'applicazione web, con lo sviluppo di un relativo documento tecnico.

Ringraziamenti

Innanzitutto desidero ringraziare con affetto i miei genitori e tutta la mia famiglia per il sostegno, il grande aiuto e per essermi stati vicini in ogni momento durante questi difficili anni di studio. Ci tengo in particolare a dedicare questo traguardo a mia nonna Teresa, con la speranza che possa essere fiera del mio percorso.

Vorrei esprimere inoltre la mia gratitudine al Prof. Gilberto Filè, relatore della mia tesi, per avermi accettato come tirocinante e per l'aiuto fornitomi durante la stesura del lavoro.

Ho desiderio di ringraziare infine tutti i miei amici per tutti i bellissimi anni passati insieme e le mille avventure vissute.

Padova, Settembre 2021

Gabriel Bizzo

Indice

1	Introduzione	1
1.1	L'azienda	1
1.2	Scelta dell'azienda	1
1.3	Il progetto Voting-Online	2
1.4	Strumenti di sviluppo e di supporto	2
1.5	Strumenti organizzativi	3
1.6	Organizzazione del testo	4
1.6.1	Struttura del documento	4
1.6.2	Convenzioni tipografiche	4
2	Descrizione dello stage	5
2.1	Obiettivi	5
2.2	Pianificazione del lavoro	6
2.3	Analisi preventiva dei rischi	7
3	Analisi dei requisiti	9
3.1	Casi d'uso	9
3.1.1	Attori principali	9
3.1.2	Elenco dei casi d'uso	9
3.2	Tracciamento dei requisiti	29
4	Progettazione e codifica	35
4.1	Backend	35
4.1.1	Tecnologie	35
4.1.2	Descrizione	37
4.1.3	Architettura	37
4.1.4	Design pattern utilizzati	38
4.2	Tecnologie per il Frontend	38
4.3	Frontend con Angular	40
4.3.1	Descrizione	40
4.3.2	Architettura di Angular	40
4.3.3	Architettura dell'applicazione sviluppata	40
4.3.4	Design pattern utilizzati	41
4.4	Frontend con React	42
4.4.1	Descrizione	42
4.4.2	Architettura dell'applicazione sviluppata	42
4.4.3	Il pre-rendering di Next.js	43
4.4.4	Design pattern utilizzati	44

4.5	Maschere prodotte	44
4.5.1	Homepage	45
4.5.2	Autenticazione	45
4.5.3	Registrazione	46
4.5.4	Admin Dashboard	46
4.5.5	User Dashboard	48
4.5.6	Maschera di voto di un'elezione	49
5	Confronto tra Angular e React	51
5.1	Introduzione	51
5.1.1	Angular	51
5.1.2	React	52
5.1.3	I componenti	52
5.2	Dettagli tecnici	53
5.2.1	Linguaggio di programmazione	53
5.2.2	Gestione dei componenti	53
5.2.3	Gestione dello stato	54
5.2.4	Gestione dello stile	54
5.3	Performance	54
5.3.1	Memoria	54
5.3.2	Site Rendering	55
5.3.3	Rendering della UI	56
5.4	Conclusioni	57
6	Conclusioni	59
6.1	Raggiungimento degli obiettivi	59
6.2	Analisi del lavoro svolto	59
6.3	Valutazione personale	59
	Glossary	61
	Acronyms	63
	Bibliografia	65

Elenco delle figure

1.1	Logo dell'azienda Sync Lab	1
3.1	Scenario principale	10
3.2	Use Case - UC1: Registrazione	11
3.3	Use Case - UC3: Autenticazione	13
3.4	Use Case - UC5: Visualizzazione storico dei voti dei voti dell'utente	14
3.5	Use Case - UC6: Visualizzazione lista delle elezioni disponibili	15
3.6	Use Case - UC7: Visualizzazione maschera di voto di una elezione	16
3.7	Use Case - UC7.1: Visualizzazione lista dei partiti partecipanti	17
3.8	Use Case - UC10: Espressione di un voto in un'elezione	18
3.9	Use Case - UC11: Inserimento di una nuova elezione	20
3.10	Use Case - UC13: Inserimento di un nuovo partito	22
3.11	Use Case - UC15: Modifica di una elezione	24
3.12	Use Case - UC21: Visualizzazione lista di tutte le elezioni	28
3.13	Use Case - UC22: Visualizzazione lista di tutti i partiti	29
4.1	Logo di Apache Maven	36
4.2	Logo del framework Spring	36
4.3	Modulo Backend	38
4.4	Logo di Javascript	39
4.5	Logo di Typescript	39
4.6	Modulo Frontend realizzato con Angular	41
4.7	Modulo Frontend realizzato con Next.js e React	43
4.8	Schema generale del Frontend in Next.js e React	44
4.9	Homepage	45
4.10	Pagina di autenticazione	46
4.11	Pagina di Registrazione	46
4.12	Pagina della Admin Dashboard	47
4.13	Finestra contenente i dettagli di un'elezione	47
4.14	Form di creazione di una nuova elezione	48
4.15	Pagina della User Dashboard	48
4.16	Maschera di voto di un'elezione	49
4.17	Pop-up di conferma della preferenza dell'elettore	49
5.1	Angular Logo	51
5.2	React Logo	52
5.3	Next.js Logo	56
5.4	Angular Universal Logo	56

Elenco delle tabelle

3.1	Tabella del tracciamento dei requisiti funzionali	30
3.2	Tabella del tracciamento dei requisiti qualitativi	33
3.3	Tabella del tracciamento dei requisiti di vincolo	33
6.1	Tabella di riepilogo dello stato degli obiettivi	59

Capitolo 1

Introduzione

1.1 L'azienda

Sync Lab nasce come Software house tramutatasi rapidamente in System Integrator attraverso un processo di maturazione delle competenze tecnologiche, metodologiche ed applicative nel dominio del software.

L'azienda, propone sul mercato interessanti quanto innovativi prodotti software, nati nel loro laboratorio di ricerca e sviluppo. Attraverso questi prodotti Sync Lab ha gradualmente conquistato significativamente fette di mercato nei seguenti settori: mobile, videosorveglianza e sicurezza delle infrastrutture informatiche aziendali.

Attualmente, Sync Lab ha più di 150 clienti diretti e finali, con un organico aziendale di 200 dipendenti distribuiti tra le 5 sedi dislocate in tutta Italia.



Figura 1.1: Logo dell'azienda Sync Lab

1.2 Scelta dell'azienda

Il primo contatto avuto con l'azienda Sync Lab è avvenuto allo StageIT 2021, effettuato in modalità telematica. Il presentatore dell'azienda è stato l'ingegnere Fabio Pallaro, il quale ha spiegato in modo chiaro ed esaustivo le caratteristiche dell'azienda. La motivazione principale che mi ha spinto a scegliere Sync Lab è stata la conferma da parte del sig. Pallaro di accontentare gli stagisti rispetto alle loro preferenze nell'ambito informatico, permettendomi di effettuare un approfondimento a tutto tondo nell'ambito dello sviluppo di applicazioni web.

1.3 Il progetto Voting-Online

La votazione elettronica, anche detta e-Voting (dall'inglese «electronic voting»), consiste in un insieme di metodologie che permettono ai cittadini l'espressione del proprio voto e la gestione delle preferenze attraverso tecnologie elettroniche e informatiche.

Gli applicativi software che realizzano questa tipologia di sistema permettono all'utente di accedere ad una maschera per effettuare una votazione, passando precedentemente per una procedura di autenticazione in modo da collegare il voto ad una persona fisica in modo sicuro. Questi sistemi, oltre ad acquisire le preferenze degli elettori, offrono ad un insieme di amministratori della piattaforma degli strumenti per creare e configurare diverse tipologie di elezioni, oltre che per gestire l'inserimento e/o la rimozione dei partiti partecipanti.

Il progetto di stage *Voting-Online*, proposto da Sync Lab, consiste nell'analisi, progettazione e realizzazione di un'applicazione web riguardante il suddetto ambito. Tale progetto nasce per mostrare, principalmente ad aziende private, quella che potrebbe essere un'applicazione web di e-Voting attraverso un prototipo.

Il progetto in questione si focalizza sullo sviluppo del [FE](#), e quindi sull'approfondimento delle tecnologie ad esso connesse come Angular e React, anche se una parte del lavoro prevede lo studio di Java e del [framework](#) Spring in modo da acquisire una conoscenza di base anche sul lato [BE](#). L'argomento principale di tale progetto risulta quindi lo sviluppo delle interfacce grafiche corrispondenti alle maschere per fornire le funzionalità necessarie agli utenti e agli amministratori, utilizzando i linguaggi Javascript e Typescript.

Lo sviluppo del [FE](#), utilizzando le due tecnologie precedentemente citate, ha permesso di effettuare in conclusione un'attenta analisi comparativa tra di esse, approfondita nell'[apposita sezione](#).

1.4 Strumenti di sviluppo e di supporto

Strumenti software utilizzati per lo sviluppo, il versionamento e la documentazione dell'applicativo durante le diverse fasi del suo ciclo di vita.

Draw.io

Draw.io è una piattaforma online gratuita per la creazione di varie tipologie di diagrammi, esportabili come file in diversi formati (tra i quali PDF o JPEG). Tra le diverse possibilità offerte dal software sono presenti i diagrammi di flusso, di processo, [UML](#), [Entita'-Relazione](#) e di rete. Questa piattaforma è stata utilizzata per creare i diagrammi dei casi d'uso, utilizzati per l'analisi dei requisiti illustrata nel terzo capitolo.

Visual Studio Code

Visual Studio Code è un «Integrated development environment» (IDE), ovvero una piattaforma che consente di migliorare l'esperienza di sviluppo software e, tra i vari strumenti che rende disponibile, contiene un editor di codice sorgente. Grazie alle numerose estensioni che è possibile installare, si può utilizzare una vasta gamma di linguaggi di programmazione e funzionalità di supporto alla scrittura del codice. Uno strumento disponibile grazie a questo IDE che risulta molto utile è l'intelliSense, ovvero una forma di completamento automatico del codice e di visualizzazione grafica di informazioni durante lo sviluppo.

Overleaf

Overleaf è un editor LaTeX collaborativo basato su cloud e viene utilizzato per scrivere, modificare e pubblicare varie tipologie di documenti. Questo software è stato utilizzato per la scrittura del documento tecnico, richiesto dall'azienda insieme al codice sorgente.

Git

Git è uno strumento per il controllo di versione distribuito utilizzabile da interfaccia a riga di comando. E' possibile utilizzare il software in questione per collaborare con più membri di un team e per controllare la versione del codice prodotto così da poter ritornare ad una versione stabile in caso di problemi.

1.5 Strumenti organizzativi

Strumenti software utilizzati per il coordinamento e la pianificazione in accordo con l'azienda.

Discord

Discord è una piattaforma di VoIP, messaggistica istantanea e distribuzione digitale progettata per la comunicazione tra comunità di videogiocatori, ma utilizzabile per diversi scopi. Gli utenti possono comunicare con chiamate vocali, video-chiamate, messaggi di testo, media e file in chat private o come membri di un server. Questo strumento permette lo scambio di messaggi e materiale digitale tra gli stagisti e i dipendenti dell'azienda in modo facile e veloce.

Google Sheets

Google Sheets è un programma per fogli di calcolo incluso come parte della suite di editor di documenti basata sul Web gratuita offerta da Google. La piattaforma consente agli utenti di creare e modificare file collaborando con altri utenti in tempo reale. Attraverso questo software è stato possibile compilare un diario digitale che ha permesso al tutor aziendale di controllare lo stato di avanzamento del lavoro dello stage in modo accurato e giornaliero.

Notion

Notion è una piattaforma che fornisce componenti come note, database, calendari, promemoria. Gli utenti possono collegare questi componenti per creare i propri sistemi per la gestione della conoscenza, prendere appunti, gestire dati e progetti. L'azienda Sync Lab, sfruttando questo applicativo, gestisce in modo semplice ed efficace la problematica del flusso di persone in ufficio in tempo di pandemia da Covid-19, permettendo di segnalare la propria presenza in sede fino ad un numero massimo di persone raggiungibile ogni giorno.

1.6 Organizzazione del testo

1.6.1 Struttura del documento

Il documento, suddiviso in sei capitoli, è strutturato nella seguente modalità:

Il primo capitolo effettua una breve introduzione all'azienda e al lavoro effettuato durante lo stage curricolare presso Sync Lab.

Il secondo capitolo descrive lo stage elencando gli obiettivi da raggiungere, la pianificazione del lavoro e l'analisi preventiva dei rischi.

Il terzo capitolo approfondisce l'analisi dei requisiti effettuata per il progetto Voting-Online.

Il quarto capitolo approfondisce le tecnologie utilizzate nel progetto Voting-Online, la progettazione del **BE** e del **FE** e una descrizione dettagliata dei relativi software ottenuti dalla loro codifica.

Il quinto capitolo approfondisce le tecnologie utilizzate per lo sviluppo del **FE**, ovvero Angular e React, effettuando un confronto tra diversi dettagli tecnici e considerazioni sulle performance.

Il sesto capitolo contiene un'analisi del lavoro svolto e le conclusioni tratte.

1.6.2 Convenzioni tipografiche

Riguardo la stesura del testo, relativamente al documento sono state adottate le seguenti convenzioni tipografiche:

- * gli acronimi, le abbreviazioni e i termini ambigui o di uso non comune menzionati vengono definiti nel glossario, situato alla fine del presente documento;
- * per la prima occorrenza dei termini riportati nel glossario viene utilizzata la nomenclatura "*parola(abbreviazione)*", mentre per ogni successiva occorrenza verrà utilizzata solamente l'abbreviazione di tale termine;
- * i termini in lingua straniera o facenti parti del gergo tecnico sono evidenziati con il carattere *corsivo*.

Capitolo 2

Descrizione dello stage

In questo capitolo è presente la lista degli obiettivi da raggiungere, la pianificazione delle ore di lavoro da effettuare e l'analisi preventiva dei rischi che potevano venire riscontrati durante lo svolgimento dello stage.

2.1 Obiettivi

Notazione

Si farà riferimento ai requisiti secondo le seguenti notazioni:

- * *O* per i requisiti obbligatori, vincolanti in quanto obiettivo primario richiesto dal committente;
- * *D* per i requisiti desiderabili, non vincolanti o strettamente necessari, ma dal riconoscibile valore aggiunto;
- * *F* per i requisiti facoltativi, rappresentanti valore aggiunto non strettamente competitivo.

Le sigle precedentemente indicate saranno seguite da una coppia sequenziale di numeri, identificativo del requisito.

Obiettivi fissati

Si prevede lo svolgimento dei seguenti obiettivi:

- * Obbligatori
 - O01: Acquisizione competenze sulle tematiche sopra descritte;
 - O02: Capacità di raggiungere gli obiettivi richiesti in autonomia seguendo il cronoprogramma;
 - O03: Portare a termine le implementazioni previste con una percentuale di superamento pari all'80%.
- * Desiderabili

- D01: Portare a termine le implementazioni previste con una percentuale di superamento pari al 100%.

* Facoltativi

- F01: Riuscire a renderizzare le maschere lato server usando Next.js/React.

2.2 Pianificazione del lavoro

Pianificazione settimanale

* **Prima Settimana (40 ore)**

- Incontro con persone coinvolte nel progetto per discutere i requisiti e le richieste relativamente al sistema da sviluppare;
- Verifica credenziali e strumenti di lavoro assegnati;
- Analisi del progetto da svolgere;
- Ripasso del linguaggio Java SE;
- Ripasso concetti Web (Servlet, servizi Rest, Json, ecc.).

* **Seconda Settimana (40 ore)**

- Studio principi generali di Spring Core (IOC, Dependency Injection);
- Studio di SpringBoot.

* **Terza Settimana (40 ore)**

- Studio Spring DataRest;
- Studio Spring Data/JPA.

* **Quarta Settimana (40 ore)**

- Ripasso linguaggio Javascript e studio TypeScript;
- Studio del [framework](#) Angular.

* **Quinta Settimana (40 ore)**

- Analisi e studio del progetto Voting-on-line;
- Progettazione ed implementazione della nuova maschera di login in Angular;
- Progettazione ed implementazione nuova maschera "Voto" in Angular.

* **Sesta Settimana (40 ore)**

- Ripasso [framework](#) React;
- Progettazione ed implementazione della nuova maschera di login in React;
- Progettazione ed implementazione nuova maschera "Voto" in React.

* **Settima Settimana (40 ore)**

- Analisi comparativa dei due [framework](#) utilizzati.

* **Ottava Settimana (40 ore)**

- Considerazioni finali e stesura elaborato.

2.3 Analisi preventiva dei rischi

Durante la fase di analisi iniziale sono stati individuati alcuni possibili rischi a cui si potrà andare incontro. Si è quindi proceduto a elaborare delle possibili soluzioni per far fronte a tali rischi.

1. Inesperienza tecnologica

Descrizione: alcune tecnologie da utilizzare risultano totalmente o parzialmente sconosciute.

Soluzione: studio e approfondimento attraverso tutorial online, corsi forniti dall'azienda e progetti di consolidamento.

2. Monitoraggio del lavoro e delle scadenze

Descrizione: per via dell'emergenza sanitaria dettata dal Covid-19, non sarà sempre possibile confrontarsi con il tutor aziendale e potrebbero emergere dei problemi nell'organizzazione del lavoro per seguire il crono-programma.

Soluzione: creazione di un foglio condiviso e di un repository per dichiarare le attività svolte giornalmente e per la condivisione del relativo codice prodotto. Gli strumenti utilizzati a tale scopo sono descritti nell'[apposita sezione](#).

3. Impossibilità di recarsi in ufficio

Descrizione: per via dell'emergenza sanitaria dettata dal virus Covid-19, risulterà impossibile lavorare ogni giorno nella sede aziendale.

Soluzione: utilizzo di diversi strumenti per lo "smart working" e la comunicazione asincrona, elencati nella [sezione apposita](#).

Capitolo 3

Analisi dei requisiti

In questo capitolo vengono descritte le funzionalità che il prodotto deve offrire elencando i casi d'uso e i requisiti individuati.

3.1 Casi d'uso

Per lo studio dei casi di utilizzo del prodotto sono stati creati dei diagrammi. I diagrammi dei casi d'uso¹ (UC) sono diagrammi di tipo UML dedicati alla descrizione delle funzioni o servizi offerti da un sistema, così come sono percepiti e utilizzati dagli attori che interagiscono col sistema stesso.

3.1.1 Attori principali

Gli attori principali individuati sono i seguenti:

- * **Utente non autenticato:** indica l'utente che non ha effettuato l'autenticazione attraverso la procedura di login. Questo attore non deve avere la possibilità di accedere agli strumenti di voto e a quelli di monitoraggio della piattaforma;
- * **Elettore:** indica l'utente che ha effettuato l'accesso alla piattaforma con un profilo da elettore e deve poter accedere agli strumenti di voto;
- * **Amministratore:** indica l'utente che ha effettuato l'accesso alla piattaforma con un profilo da amministratore e deve poter accedere agli strumenti di monitoraggio delle votazioni.

3.1.2 Elenco dei casi d'uso

¹ *Casi D'uso*. URL: https://www.math.unipd.it/~rcardin/swea/2021/Diagrammi%20Use%20Case_4x4.pdf.

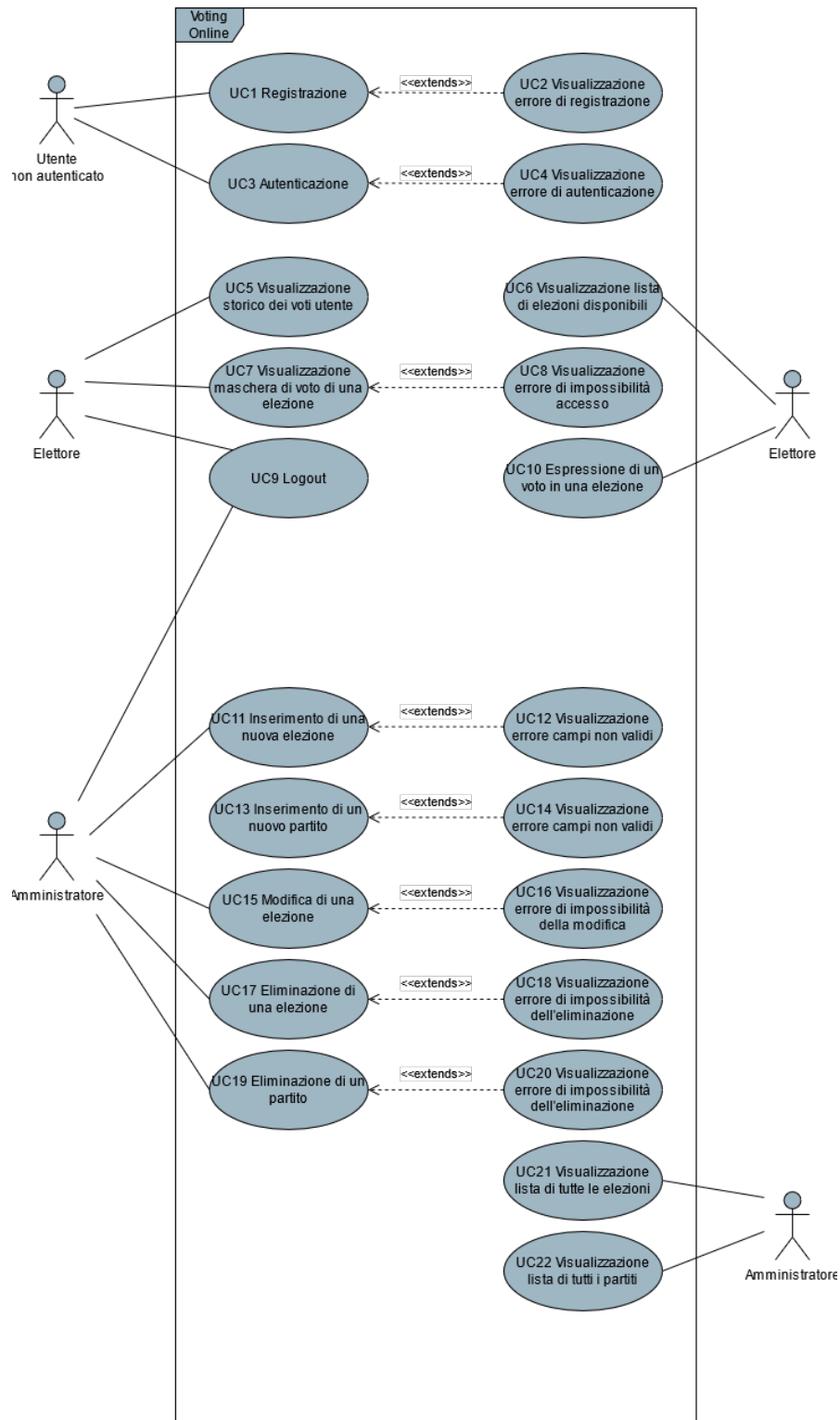


Figura 3.1: Scenario principale

UC1: Registrazione

Attori Principali: Utente non autenticato.

Precondizioni: L'utente non autenticato è all'interno della pagina di registrazione.

Descrizione: Viene effettuata la registrazione di un elettore nel sistema, inserendo i propri dati personali nella pagina dedicata.

Scenario Principale: L'utente non autenticato accede alla pagina di registrazione, il sistema rende disponibili i campi da compilare, l'utente inserisce l'indirizzo email [UC1.1], l'username [UC1.2], la password [UC1.3] e procede infine a confermare la registrazione.

Postcondizioni: La registrazione nel sistema è avvenuta con successo.

Estensioni:

1. **UC2:** se i campi non sono validi o l'email è già stata utilizzata nel sistema, viene mostrato un errore di registrazione all'utente non autenticato che potrà provare nuovamente a ripetere la procedura.

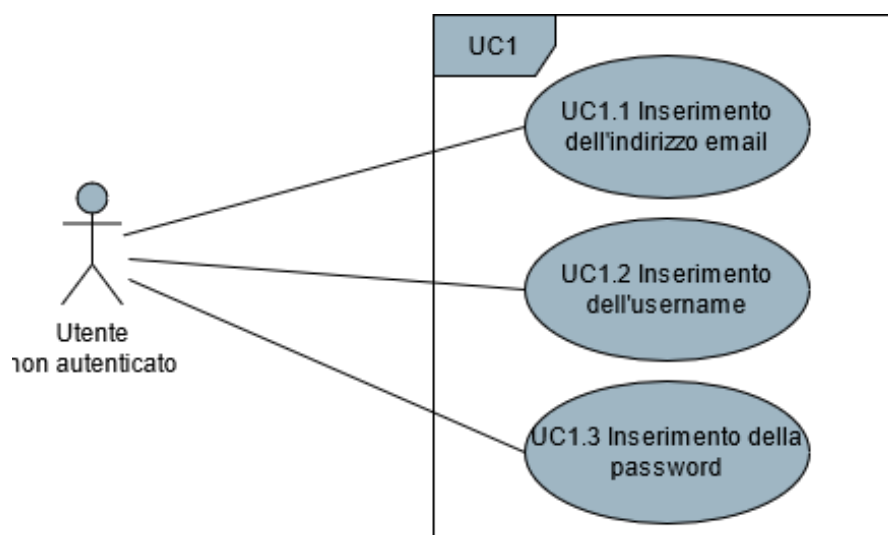


Figura 3.2: Use Case - UC1: Registrazione

UC1.1: Inserimento dell'indirizzo email

Attori Principali: Utente non autenticato.

Precondizioni: Il campo "email" risulta vuoto.

Descrizione: L'utente deve compilare il campo "email" per procedere alla registrazione.

Scenario Principale: L'utente inserisce il suo indirizzo email nell'apposito campo.

Postcondizioni: il campo "email" è stato compilato.

UC1.2: Inserimento dell'username

Attori Principali: Utente non autenticato.

Precondizioni: Il campo "username" risulta vuoto.

Descrizione: L'utente deve compilare il campo "username" per procedere alla registrazione.

Scenario Principale: L'utente inserisce l'username desiderato nell'apposito campo.

Postcondizioni: il campo "username" è stato compilato.

UC1.3: Inserimento della password

Attori Principali: Utente non autenticato.

Precondizioni: Il campo "password" risulta vuoto.

Descrizione: L'utente deve compilare il campo "password" per procedere alla registrazione.

Scenario Principale: L'utente inserisce la password desiderata nell'apposito campo.

Postcondizioni: il campo "password" è stato compilato.

UC2: Visualizzazione errore di registrazione

Attori Principali: Utente non autenticato.

Precondizioni: L'utente non autenticato ha inserito i campi ed ha provato ad effettuare la registrazione.

Descrizione: L'utente non autenticato visualizza un errore riguardante i campi di registrazione che ha inserito. Questi errori possono essere:

- * **campo non valido:** un campo risulta vuoto o con caratteri non validi;
- * **email già utilizzata:** l'email è già stata utilizzata da un altro elettore.

Scenario Principale: Il sistema riconosce uno o più errori nei campi inseriti dall'utente e vengono visualizzati nella pagina di registrazione.

Postcondizioni: Viene visualizzato un messaggio di errore nella pagina di registrazione e l'utente non risulta autenticato nel sistema.

UC3: Autenticazione

Attori Principali: Utente non autenticato.

Precondizioni: L'utente non autenticato è all'interno della pagina di autenticazione.

Descrizione: L'utente non autenticato, inserendo le proprie credenziali, viene autenticato alla piattaforma.

Scenario Principale: L'utente non autenticato accede alla pagina di autenticazione, il sistema rende disponibili i campi da compilare, l'utente inserisce l'indirizzo email [UC3.1], la password [UC3.2] e procede infine inviando la richiesta.

Postcondizioni: L'utente viene autenticato come elettore o amministratore dal sistema.

Estensioni:

1. **UC4:** se le credenziali inserite non vengono riconosciute dal sistema viene visualizzato un messaggio che informa l'utente dell'errore.

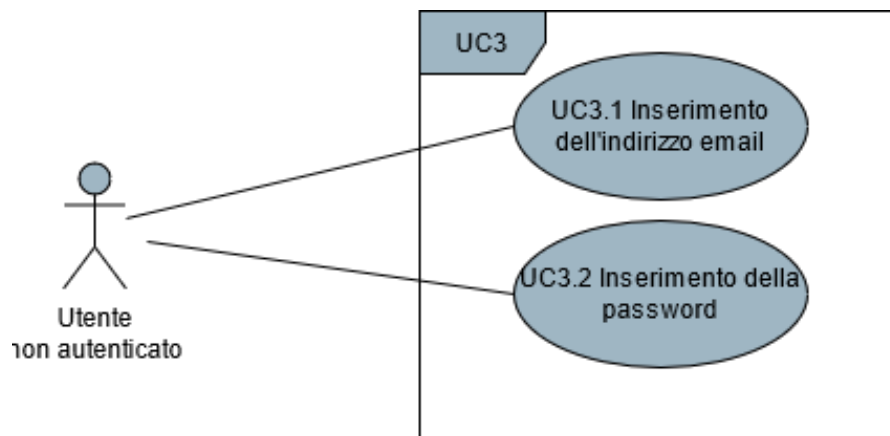


Figura 3.3: Use Case - UC3: Autenticazione

UC3.1: Inserimento dell'indirizzo email

Attori Principali: Utente non autenticato.

Precondizioni: Il campo "email" risulta vuoto.

Descrizione: L'utente deve compilare il campo "email" per procedere all'autenticazione.

Scenario Principale: L'utente inserisce il proprio indirizzo email nell'apposito campo.

Postcondizioni: il campo "email" è stato compilato.

UC3.2: Inserimento della password

Attori Principali: Utente non autenticato.

Precondizioni: Il campo "password" risulta vuoto.

Descrizione: L'utente deve compilare il campo "password" per procedere all'autenticazione.

Scenario Principale: L'utente inserisce la propria password nell'apposito campo.

Postcondizioni: il campo "password" è stato compilato.

UC4: Visualizzazione errore di autenticazione

Attori Principali: Utente non autenticato.

Precondizioni: L'utente non autenticato ha inserito i campi ed ha provato ad effettuare l'autenticazione.

Descrizione: L'utente non autenticato visualizza un messaggio di errore che lo informa che i dati da lui inseriti durante il login non sono riconosciuti dal sistema.

Scenario Principale: L'utente tenta di effettuare il login usando credenziali non presenti nel sistema.

Postcondizioni: Viene visualizzato un messaggio di errore nella pagina di autenticazione e l'utente non risulta autenticato nel sistema.

UC5: Visualizzazione storico dei voti dei voti dell'utente

Attori Principali: Elettore.

Precondizioni: L'elettore è all'interno della pagina corrispondente alla sua dashboard personale.

Descrizione: L'elettore può visualizzare, nel caso abbia partecipato ad almeno un'elezione, lo storico dei propri voti.

Scenario Principale: L'elettore può visualizzare lo storico dei propri voti nella forma di una lista di singole votazioni passate [UC5.1].

Postcondizioni: L'elettore visualizza lo storico dei voti.

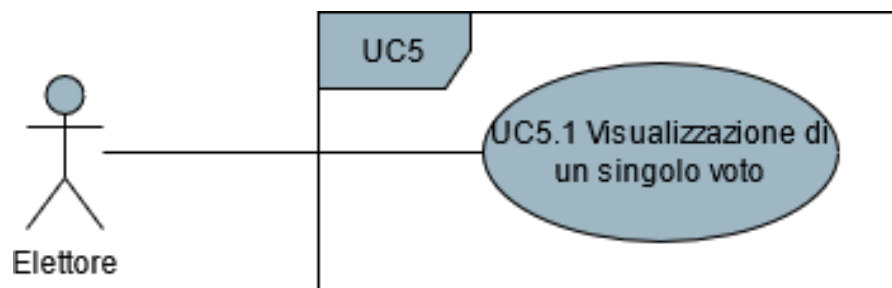


Figura 3.4: Use Case - UC5: Visualizzazione storico dei voti dei voti dell'utente

UC5.1: Visualizzazione di un singolo voto

Attori Principali: Elettore.

Precondizioni: L'elettore è all'interno della pagina corrispondente alla sua dashboard personale.

Descrizione: L'elettore può visualizzare le informazioni corrispondenti ad un singolo voto appartenente al suo storico.

Scenario Principale: L'elettore visualizza le seguenti informazioni caratterizzanti di

un voto appartenente al suo storico: il nome dell'elezione, la data di inizio e di fine dell'elezione, di espressione del voto, il partito e il candidato votato.

Postcondizioni: L'elettore visualizza tutti i dati che caratterizzano un suo voto passato.

UC6: Visualizzazione lista delle elezioni disponibili

Attori Principali: Elettore.

Precondizioni: L'elettore è all'interno della pagina corrispondente alla sua dashboard personale.

Descrizione: L'elettore può visualizzare, nel caso sia presente almeno un'elezione aperta nella data corrente, un elenco di elezioni.

Scenario Principale: L'elettore può visualizzare le elezioni disponibili nella forma di una lista di singole votazioni [UC6.1].

Postcondizioni: L'elettore visualizza le elezioni disponibili nella dashboard personale.

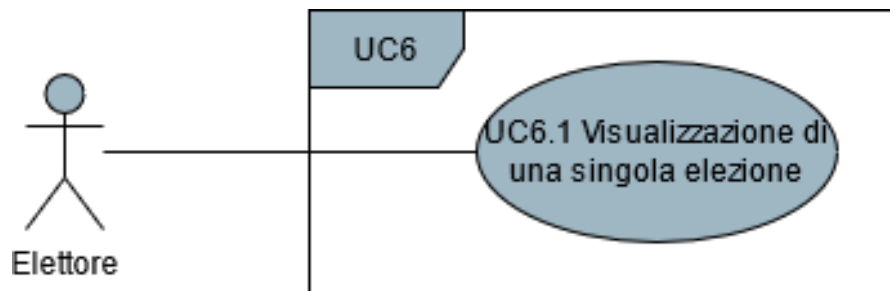


Figura 3.5: Use Case - UC6: Visualizzazione lista delle elezioni disponibili

UC6.1: Visualizzazione di una singola elezione

Attori Principali: Elettore.

Precondizioni: L'elettore è all'interno della pagina corrispondente alla sua dashboard personale.

Descrizione: L'elettore può visualizzare le informazioni corrispondenti ad una singola elezione presente nell'elenco delle elezioni disponibili.

Scenario Principale: L'elettore visualizza le seguenti informazioni caratterizzanti di un'elezione disponibile: il nome, la tipologia, la data di inizio e di fine della votazione.

Postcondizioni: L'elettore visualizza tutti i dati che caratterizzano una singola elezione disponibile.

UC7: Visualizzazione maschera di voto di una elezione

Attori Principali: Elettore.

Precondizioni: L'elettore visualizza l'elenco delle elezioni disponibili ed è presente almeno una voce nella lista.

Descrizione: L'elettore accede alla maschera di voto per esprimere la propria preferenza cliccando su un'elezione disponibile, rispetto alla quale visualizzerà le informazioni necessarie per procedere alla votazione.

Scenario Principale: L'elettore, cliccando su una delle elezioni disponibili nella lista [UC6], accede alla maschera di voto di tale elezione, della quale visualizza il relativo nome, le istruzioni per esprimere la propria preferenza con successo e la lista dei partiti partecipanti [UC7.1].

Postcondizioni: L'elettore visualizza le informazioni necessarie per esprimere il proprio voto nell'apposita pagina. **Estensioni:**

1. **UC8:** se si clicca su un'elezione per la quale l'elettore ha già espresso una preferenza viene visualizzato un messaggio che avvisa l'utente dell'errore.

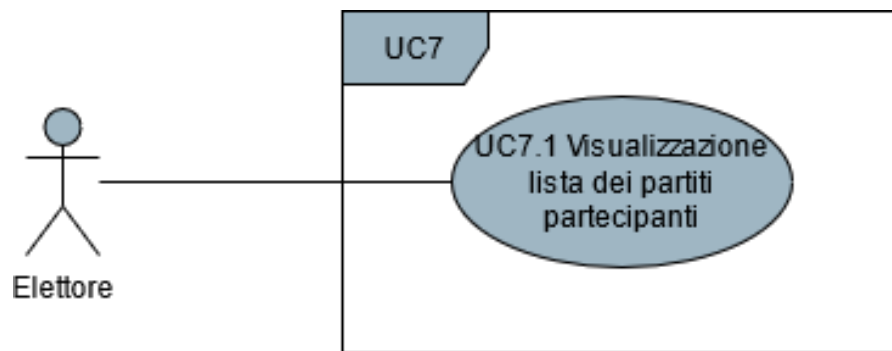


Figura 3.6: Use Case - UC7: Visualizzazione maschera di voto di una elezione

UC7.1: Visualizzazione lista dei partiti partecipanti

Attori Principali: Elettore.

Precondizioni: L'elettore è all'interno della pagina corrispondente alla maschera di voto per una elezione.

Descrizione: L'elettore può visualizzare, nel caso sia presente almeno un partito partecipante, una lista di partiti.

Scenario Principale: L'elettore può visualizzare i partiti partecipanti nella forma di una lista di singoli partiti [UC7.1.1].

Postcondizioni: L'elettore visualizza tutti i partiti partecipanti che possono essere votati nella maschera di voto relativa ad una specifica elezione.

UC7.1.1: Visualizzazione di un singolo partito

Attori Principali: Elettore.

Precondizioni: L'elettore è all'interno della pagina corrispondente alla maschera di voto per una elezione e sta visualizzando la lista dei partiti partecipanti.

Descrizione: L'elettore può visualizzare le informazioni corrispondenti ad un singolo partito presente nell'elenco dei partiti partecipanti.

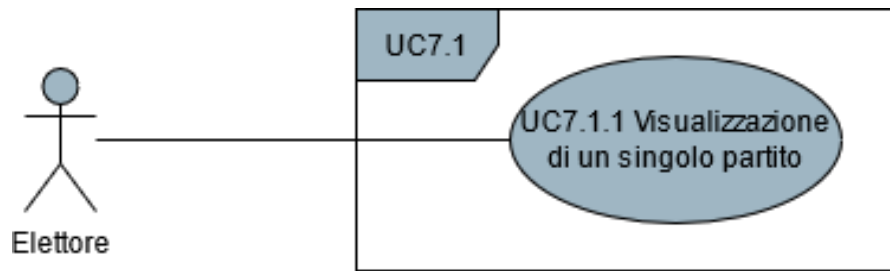


Figura 3.7: Use Case - UC7.1: Visualizzazione lista dei partiti partecipanti

Scenario Principale: L'elettore visualizza le seguenti informazioni caratterizzanti del partito partecipante: il logo, il nome e il nominativo del candidato associato al partito.

Postcondizioni: L'elettore visualizza tutti i dati che caratterizzano un singolo partito partecipante all'elezione.

UC8: Visualizzazione errore di impossibilità di accesso all'elezione

Attori Principali: Elettore.

Precondizioni: L'elettore visualizza l'elenco delle elezioni disponibili ed ha provato a cliccare su una delle voci presenti nella lista.

Descrizione: L'elettore visualizza un messaggio di errore riguardante l'impossibilità di accedere alla maschera di voto relativa all'elezione selezionata. Questo errore indica che è stata già effettuata una preferenza per questa elezione.

Scenario Principale: Il sistema riconosce un errore al tentativo di accesso da parte dell'elettore ad un'elezione e viene visualizzato il relativo messaggio di errore nella dashboard personale.

Postcondizioni: Viene visualizzato un messaggio di errore nella dashboard personale dell'elettore e quest'ultimo non ha la possibilità di accedere alla maschera di voto relativa all'elezione selezionata.

UC9: Logout

Attori Principali: Elettore, Amministratore.

Precondizioni: L'utente, che ha precedentemente effettuato la procedura di login, è attualmente autenticato nella piattaforma.

Descrizione: Viene effettuato il logout di un'utente autenticato, che può essere un elettore o amministratore.

Scenario Principale: L'utente richiede il logout tramite un bottone dedicato.

Postcondizioni: L'utente non è più autenticato nel sistema.

UC10: Espressione di un voto in un'elezione

Attori Principali: Elettore.

Precondizioni: L'elettore si trova nella pagina relativa alla maschera di voto di una specifica elezione.

Descrizione: L'elettore può esprimere il proprio voto in una determinata elezione selezionando il partito desiderato e confermando infine la propria scelta.

Scenario Principale: L'elettore, visualizzando le informazioni contenute nella maschera di voto [UC7] dell'elezione in questione, può procedere ad esprimere la propria preferenza selezionando uno tra i partiti partecipanti [UC10.1] e confermando successivamente la propria decisione [UC10.2].

Postcondizioni: L'elettore ha votato per un partito partecipante ad una specifica elezione.

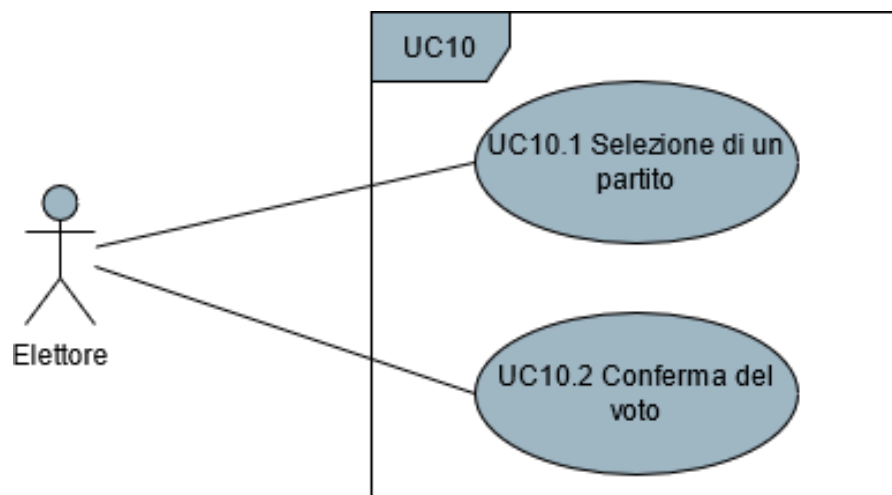


Figura 3.8: Use Case - UC10: Espressione di un voto in un'elezione

UC10.1: Selezione di un partito

Attori Principali: Elettore.

Precondizioni: L'elettore si trova nella pagina relativa alla maschera di voto di una specifica elezione.

Descrizione: L'elettore deve selezionare esattamente un partito dalla lista dei partiti partecipanti per poter continuare con la procedura di voto.

Scenario Principale: L'elettore, visualizzando la lista di partiti partecipanti contenuta nella maschera di voto [UC7.1] dell'elezione in questione, deve selezionare esattamente uno tra i partiti partecipanti per esprimere la propria preferenza.

Postcondizioni: L'elettore ha selezionato un partito partecipante all'elezione in questione.

UC10.2: Conferma del voto

Attori Principali: Elettore.

Precondizioni: L'elettore si trova nella pagina relativa alla maschera di voto di una specifica elezione ed ha selezionato un partito partecipante.

Descrizione: L'elettore deve confermare il partito scelto come preferenza per concludere la votazione.

Scenario Principale: L'elettore, dopo aver selezionato un partito partecipante come preferenza [UC10.1], deve cliccare sull'apposito pulsante per richiedere la conclusione della votazione e, visualizzando un apposito riquadro di riepilogo, deve confermare la propria decisione per concludere la procedura di voto.

Postcondizioni: L'elettore ha confermato la preferenza del partito partecipante scelto all'elezione in questione.

UC11: Inserimento di una nuova elezione

Attori Principali: Amministratore.

Precondizioni: L'amministratore si trova nella pagina corrispondente alla admin dashboard.

Descrizione: L'amministratore può inserire una nuova elezione nella piattaforma inserendo gli appositi dati.

Scenario Principale: L'amministratore per inserire una nuova elezione deve compilare i campi dati in modo da descriverne le seguenti informazioni: il nome [UC11.1], la tipologia [UC11.2], la data di inizio [UC11.3] e di fine [UC11.4]. E' necessario, infine, che vengano selezionati i partiti partecipanti da un apposito elenco [UC11.5].

Postcondizioni: L'amministratore ha inserito una nuova elezione nel sistema con successo.

Estensioni:

1. **UC12:** se i campi non sono validi viene mostrato un errore di inserimento all'amministratore che potrà provare nuovamente a ripetere la procedura modificando i campi che hanno causato l'errore.

UC11.1: Inserimento del nome

Attori Principali: Amministratore.

Precondizioni: Il campo "nome" risulta vuoto.

Descrizione: L'amministratore deve compilare il campo "nome" per procedere all'inserimento.

Scenario Principale: L'amministratore inserisce il nome dell'elezione nell'apposito campo.

Postcondizioni: il campo "nome" è stato compilato.

UC11.2: Inserimento della tipologia

Attori Principali: Amministratore.

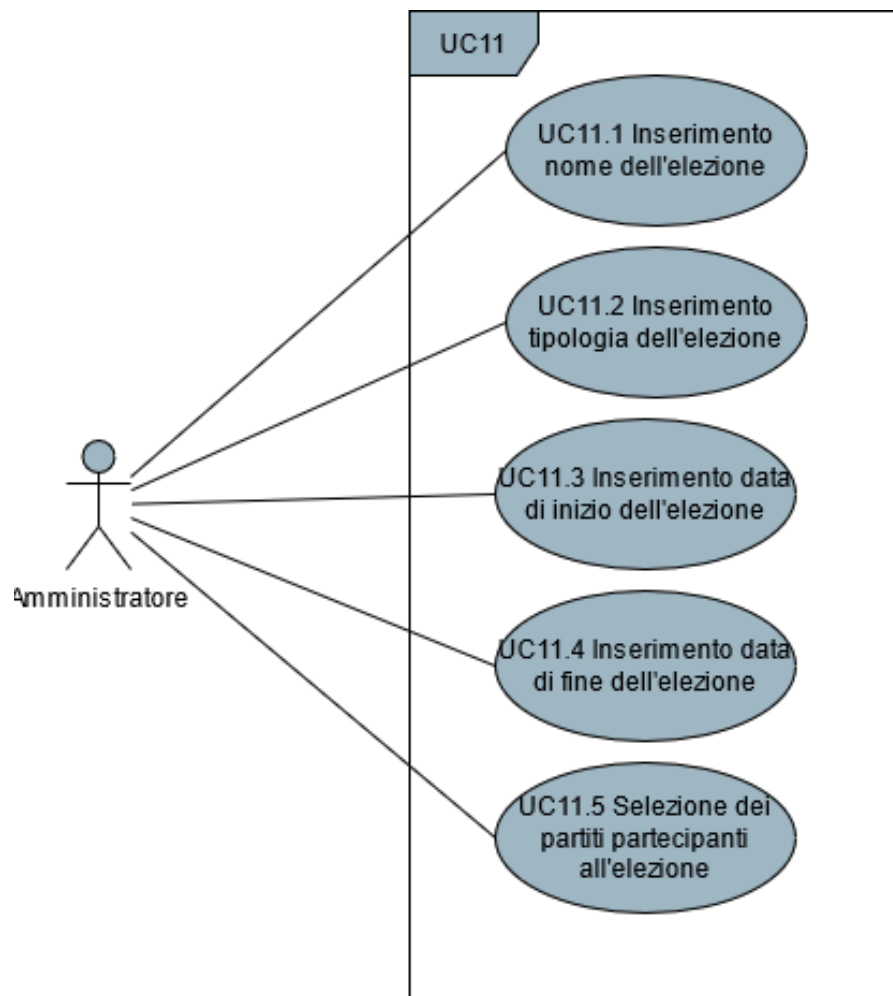


Figura 3.9: Use Case - UC11: Inserimento di una nuova elezione

Precondizioni: Il campo “tipologia” risulta vuoto.

Descrizione: L’amministratore deve compilare il campo “tipologia” per procedere all’inserimento.

Scenario Principale: L’amministratore inserisce la tipologia dell’elezione nell’apposito campo.

Postcondizioni: il campo “tipologia” è stato compilato.

UC11.3: Inserimento della data di inizio

Attori Principali: Amministratore.

Precondizioni: Il campo “data di inizio” risulta vuoto.

Descrizione: L’amministratore deve compilare il campo “data di inizio” per procedere all’inserimento.

Scenario Principale: L’amministratore inserisce la data di inizio dell’elezione nel-

l'apposito campo.

Postcondizioni: il campo "data di inizio" è stato compilato.

UC11.4: Inserimento della data di fine

Attori Principali: Amministratore.

Precondizioni: Il campo "data di fine" risulta vuoto.

Descrizione: L'amministratore deve compilare il campo "data di fine" per procedere all'inserimento.

Scenario Principale: L'amministratore inserisce la data di fine dell'elezione nell'apposito campo.

Postcondizioni: il campo "data di fine" è stato compilato.

UC11.5: Selezione dei partiti partecipanti all'elezione

Attori Principali: Amministratore.

Precondizioni: Nessun partito risulta selezionato nell'elenco dei partiti partecipanti.

Descrizione: L'amministratore può decidere di selezionare nessuno o tutti i partiti presenti nel sistema in modo da renderli partecipanti all'elezione. La selezione dei partiti partecipanti è modificabile in seguito utilizzando la funzionalità di modifica dell'elezione [UC15].

Scenario Principale: L'amministratore decide quali partiti aggiungere all'elezione in questione selezionandoli nell'apposito elenco che contiene tutti i partiti presenti nella piattaforma.

Postcondizioni: L'elenco dei partiti partecipanti è stato compilato da parte dell'amministratore.

UC12: Visualizzazione errore per campi non validi

Attori Principali: Amministratore.

Precondizioni: L'amministratore ha compilato i campi ed ha provato ad effettuare l'inserimento di una nuova elezione.

Descrizione: L'amministratore visualizza un errore riguardante uno o più campi che ha compilato e che risultano invalidi perchè vuoti o perchè c'è un errore nella formattazione del contenuto.

Scenario Principale: Il sistema riconosce uno o più errori nei campi inseriti dall'amministratore e vengono visualizzati nell'apposita form di inserimento.

Postcondizioni: Viene visualizzato un messaggio di errore nella form e la nuova elezione non risulta inserita nel sistema.

UC13: Inserimento di un nuovo partito

Precondizioni: L'amministratore si trova nella pagina corrispondente alla admin

dashboard.

Descrizione: L'amministratore può inserire un nuovo partito nella piattaforma inserendo gli appositi dati.

Scenario Principale: L'amministratore per inserire un nuovo partito deve compilare i campi dati in modo da descriverne le seguenti caratteristiche: il nome [UC13.1] e il nominativo del candidato [UC13.2]. E' necessario, infine, che venga inserito il logo del partito in formato immagine (PNG/JPEG) [UC13.3].

Postcondizioni: L'amministratore ha inserito un nuovo partito nel sistema con successo.

Estensioni:

1. **UC14:** se i campi non sono validi viene mostrato un errore di inserimento all'amministratore che potrà provare nuovamente a ripetere la procedura modificando i campi che hanno causato l'errore.

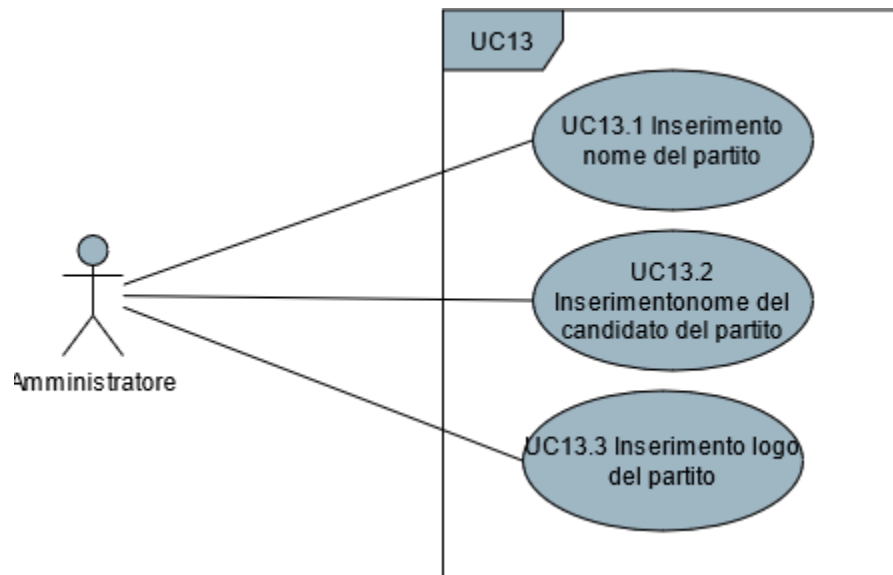


Figura 3.10: Use Case - UC13: Inserimento di un nuovo partito

UC13.1: Inserimento del nome

Attori Principali: Amministratore.

Precondizioni: Il campo "nome" risulta vuoto.

Descrizione: L'amministratore deve compilare il campo "nome" per procedere all'inserimento.

Scenario Principale: L'amministratore inserisce il nome del partito nell'apposito campo.

Postcondizioni: il campo "nome" è stato compilato.

UC13.2: Inserimento del candidato

Attori Principali: Amministratore.

Precondizioni: Il campo “candidato” risulta vuoto.

Descrizione: L'amministratore deve compilare il campo “candidato” inserendo il nominativo del candidato per il partito in modo da procedere all'inserimento.

Scenario Principale: L'amministratore inserisce il nominativo del candidato del partito nell'apposito campo.

Postcondizioni: il campo “candidato” è stato compilato.

UC13.3: Inserimento del logo

Attori Principali: Amministratore.

Precondizioni: Il campo nel quale inserire il logo del partito risulta vuoto.

Descrizione: L'amministratore deve inserire il logo del partito in formato immagine (PNG/JPEG) per procedere all'inserimento.

Scenario Principale: L'amministratore inserisce il logo del partito nell'apposito campo.

Postcondizioni: Il logo del partito è stato inserito nell'apposito campo.

UC14: Visualizzazione errore per campi non validi

Attori Principali: Amministratore.

Precondizioni: L'amministratore ha compilato i campi ed ha provato ad effettuare l'inserimento di un nuovo partito.

Descrizione: L'amministratore visualizza un errore riguardante uno o più campi che ha compilato e che risultano invalidi perchè vuoti o perchè c'è un errore nella formattazione del contenuto.

Scenario Principale: Il sistema riconosce uno o più errori nei campi inseriti dall'amministratore e vengono visualizzati nell'apposita form di inserimento.

Postcondizioni: Viene visualizzato un messaggio di errore nella form e il nuovo partito non risulta inserito nel sistema.

UC15: Modifica di una elezione

Attori Principali: Amministratore.

Precondizioni: L'amministratore si trova nella pagina corrispondente alla admin dashboard e visualizza le informazioni di dettaglio di un'elezione.

Descrizione: L'amministratore può modificare un'elezione presente nella piattaforma compilando gli appositi campi dati.

Scenario Principale: L'amministratore per modificare un'elezione esistente deve, rendendo editabili le sue informazioni attraverso l'apposito pulsante, compilare i campi dati in modo da descrivere almeno una delle seguenti informazioni: il nome [UC15.1], la tipologia [UC15.2], la data di inizio [UC15.3] e di fine [UC15.4]. E' possibile, inoltre, modificare i partiti partecipanti alterando le voci selezionate di un apposito elenco [UC15.5] e, infine, la modifica deve essere confermata da un apposito pulsante.

Postcondizioni: L'amministratore ha modificato un'elezione presente nel sistema con successo.

Estensioni:

1. **UC16:** nel caso in cui un'elezione risulti cominciata o già terminata non è possibile modificarla, verrà quindi mostrato un messaggio esplicativo all'amministratore, il quale non potrà procedere con la modifica.

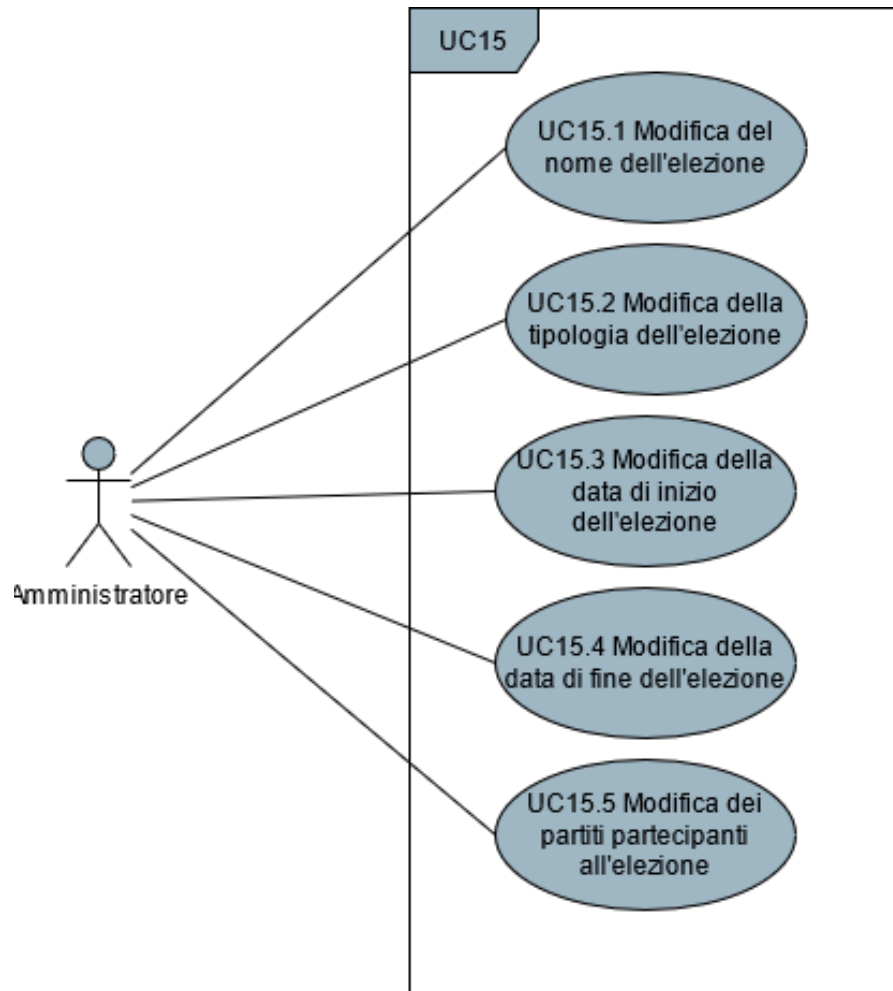


Figura 3.11: Use Case - UC15: Modifica di una elezione

UC15.1: Modifica del nome

Attori Principali: Amministratore.

Precondizioni: Il campo “nome” risulta occupato dall’informazione attuale dell’elezione.

Descrizione: L'amministratore può compilare il campo “nome” per procedere alla

modifica di tale informazione.

Scenario Principale: L'amministratore inserisce il nuovo nome dell'elezione nell'apposito campo.

Postcondizioni: il campo "nome" è stato modificato.

UC15.2: Modifica della tipologia

Attori Principali: Amministratore.

Precondizioni: Il campo "tipologia" risulta occupato dall'informazione attuale dell'elezione.

Descrizione: L'amministratore può compilare il campo "tipologia" per procedere alla modifica di tale informazione.

Scenario Principale: L'amministratore inserisce la nuova tipologia dell'elezione nell'apposito campo.

Postcondizioni: il campo "tipologia" è stato modificato.

UC15.3: Modifica della data di inizio

Attori Principali: Amministratore.

Precondizioni: Il campo "data di inizio" risulta occupato dall'informazione attuale dell'elezione.

Descrizione: L'amministratore può compilare il campo "data di inizio" per procedere alla modifica di tale informazione.

Scenario Principale: L'amministratore inserisce la nuova data di inizio dell'elezione nell'apposito campo.

Postcondizioni: il campo "data di inizio" è stato modificato.

UC15.4: Modifica della data di fine

Attori Principali: Amministratore.

Precondizioni: Il campo "data di fine" risulta occupato dall'informazione attuale dell'elezione.

Descrizione: L'amministratore può compilare il campo "data di fine" per procedere alla modifica di tale informazione.

Scenario Principale: L'amministratore inserisce la nuova data di fine dell'elezione nell'apposito campo.

Postcondizioni: il campo "data di fine" è stato modificato.

UC15.5: Modifica dei partiti partecipanti all'elezione

Attori Principali: Amministratore.

Precondizioni: L'elenco dei partiti partecipanti contiene le informazioni attuali riguardo ai partiti.

Descrizione: L'amministratore può decidere di modificare l'elenco selezionando nessuno o tutti i partiti presenti nel sistema in modo da renderli partecipanti all'elezione.

La selezione dei partiti partecipanti è modificabile finché non arriva la data di inizio dell'elezione.

Scenario Principale: L'amministratore decide quali partiti aggiungere o rimuovere dall'elezione in questione selezionandoli nell'apposito elenco che contiene tutti i partiti presenti nella piattaforma.

Postcondizioni: L'elenco dei partiti partecipanti è stato modificato da parte dell'amministratore.

UC16: Visualizzazione errore per impossibilità della modifica

Attori Principali: Amministratore.

Precondizioni: L'amministratore visualizza le informazioni riguardanti un'elezione già cominciata o conclusa ed ha provato a cliccare sul pulsante per abilitare la loro modifica.

Descrizione: Viene mostrato all'amministratore un messaggio che indica che non è possibile modificare un'elezione già cominciata o conclusa.

Scenario Principale: Il sistema mostra un messaggio esplicativo dell'errore all'amministratore.

Postcondizioni: Viene visualizzato un messaggio di errore nella admin dashboard e non viene permessa la modifica dell'elezione all'amministratore.

UC17: Eliminazione di una elezione

Attori Principali: Amministratore.

Precondizioni: L'amministratore si trova nella pagina corrispondente alla admin dashboard e visualizza le informazioni di dettaglio di un'elezione.

Descrizione: L'amministratore può eliminare un'elezione presente nella piattaforma cliccando l'apposito pulsante.

Scenario Principale: L'amministratore per eliminare un'elezione esistente deve premere l'apposito pulsante situato contestualmente alle informazioni dell'elezione stessa.

Postcondizioni: L'amministratore ha eliminato un'elezione presente nel sistema con successo.

Estensioni:

1. **UC18:** nel caso in cui un'elezione risulti cominciata o già terminata non è possibile eliminarla, verrà quindi mostrato un messaggio esplicativo all'amministratore, il quale non potrà procedere con l'eliminazione.

UC18: Visualizzazione errore per impossibilità dell'eliminazione

Attori Principali: Amministratore.

Precondizioni: L'amministratore visualizza le informazioni riguardanti un'elezione già cominciata o conclusa ed ha provato a cliccare sul pulsante per eliminarla.

Descrizione: Viene mostrato all'amministratore un messaggio che indica che non è possibile eliminare un'elezione già cominciata o conclusa.

Scenario Principale: Il sistema mostra un messaggio esplicativo dell'errore all'amministratore.

Postcondizioni: Viene visualizzato un messaggio di errore nella admin dashboard e non viene permessa l'eliminazione dell'elezione all'amministratore.

UC19: Eliminazione di un partito

Attori Principali: Amministratore.

Precondizioni: L'amministratore si trova nella pagina corrispondente alla admin dashboard e visualizza le informazioni di dettaglio di un partito.

Descrizione: L'amministratore può eliminare un partito presente nella piattaforma cliccando l'apposito pulsante.

Scenario Principale: L'amministratore per eliminare un partito esistente deve premere l'apposito pulsante situato contestualmente alle informazioni del partito stesso.

Postcondizioni: L'amministratore ha eliminato un partito dal sistema con successo.

Estensioni:

1. **UC20:** nel caso in cui un partito risulti presente in un'elezione già cominciata o terminata non è possibile eliminarlo, verrà quindi mostrato un messaggio esplicativo all'amministratore, il quale non potrà procedere con l'eliminazione.

UC20: Visualizzazione errore per impossibilità dell'eliminazione

Attori Principali: Amministratore.

Precondizioni: L'amministratore visualizza le informazioni riguardanti un partito partecipante in un'elezione già cominciata o conclusa ed ha provato a cliccare sul pulsante per eliminarlo.

Descrizione: Viene mostrato all'amministratore un messaggio che indica che non è possibile eliminare un partito facente parte di un'elezione già cominciata o conclusa.

Scenario Principale: Il sistema mostra un messaggio esplicativo dell'errore all'amministratore.

Postcondizioni: Viene visualizzato un messaggio di errore nella admin dashboard e non viene permessa l'eliminazione del partito all'amministratore.

UC21: Visualizzazione lista di tutte le elezioni

Attori Principali: Amministratore.

Precondizioni: L'amministratore è all'interno della pagina corrispondente all'admin dashboard.

Descrizione: L'amministratore può visualizzare, nel caso sia stata inserita nella piattaforma almeno un'elezione, un elenco di elezioni.

Scenario Principale: L'amministratore può visualizzare tutte le elezioni precedentemente inserite nella piattaforma con la forma di una lista di singole votazioni [UC21.1].

Postcondizioni: L'amministratore visualizza le elezioni presenti nella piattaforma sulla admin dashboard.

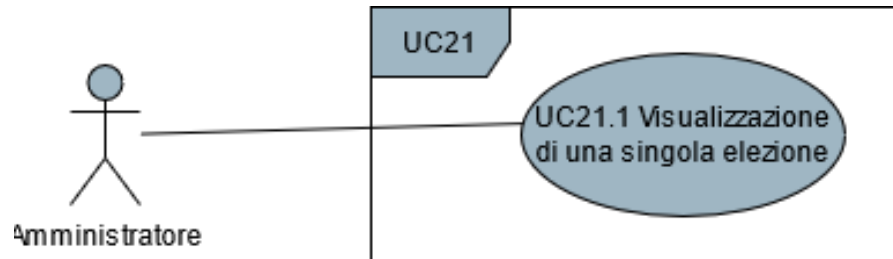


Figura 3.12: Use Case - UC21: Visualizzazione lista di tutte le elezioni

UC21.1: Visualizzazione di una singola elezione

Attori Principali: Amministratore.

Precondizioni: L'amministratore è all'interno della pagina corrispondente all'admin dashboard.

Descrizione: L'amministratore può visualizzare le informazioni corrispondenti ad una singola elezione presente nell'elenco di tutte le elezioni.

Scenario Principale: L'amministratore visualizza le seguenti informazioni caratterizzanti di un'elezione: il nome, la tipologia, la data di inizio e di fine della votazione.

Postcondizioni: L'amministratore visualizza tutti i dati che caratterizzano una singola elezione.

UC22: Visualizzazione lista di tutti i partiti

Attori Principali: Amministratore.

Precondizioni: L'amministratore è all'interno della pagina corrispondente all'admin dashboard.

Descrizione: L'amministratore può visualizzare, nel caso sia stato inserito nella piattaforma almeno un partito, un elenco di partiti.

Scenario Principale: L'amministratore può visualizzare tutti i partiti precedentemente inseriti nella piattaforma con la forma di una lista di singoli partiti [UC22.1].

Postcondizioni: L'amministratore visualizza tutti i partiti presenti nella piattaforma sulla admin dashboard.

UC22.1: Visualizzazione di un singolo partito

Attori Principali: Amministratore.

Precondizioni: L'amministratore è all'interno della pagina corrispondente all'admin dashboard.

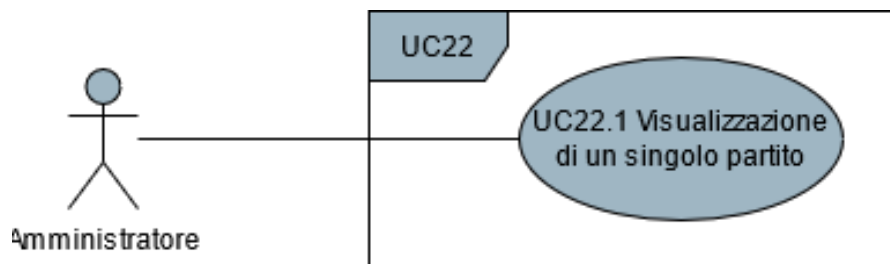


Figura 3.13: Use Case - UC22: Visualizzazione lista di tutti i partiti

Descrizione: L'amministratore può visualizzare le informazioni che caratterizzano un singolo partito presente nell'elenco di tutti i partiti.

Scenario Principale: L'amministratore visualizza le seguenti informazioni caratterizzanti di un'elezione disponibile: il nome, il logo e il nominativo del candidato del partito.

Postcondizioni: L'amministratore visualizza tutti i dati che caratterizzano un singolo partito.

3.2 Tracciamento dei requisiti

Da un'attenta analisi dei requisiti e degli use case effettuata sul progetto è stata stilata la tabella che traccia i requisiti in rapporto agli use case.

Sono stati individuati diversi tipi di requisiti e si è quindi fatto utilizzo di un codice identificativo per distinguerli.

Il codice dei requisiti è così strutturato R(F/Q/V)(N/D/O) dove:

R = requisito

F = funzionale

Q = qualitativo

V = di vincolo

N = obbligatorio (necessario)

D = desiderabile

Z = opzionale

Le fonti dei requisiti possono essere le seguenti:

- * UC[numero]: indica un caso d'uso;
- * Committente: indica il capo progetto.

Nelle tabelle 3.1, 3.2 e 3.3 sono riassunti i requisiti e il loro tracciamento con gli use case delineati in fase di analisi.

Tabella 3.1: Tabella del tracciamento dei requisiti funzionali

Requisito	Descrizione	Fonte
RFN-1	Il sistema deve permettere la registrazione di un nuovo elettore	UC1
RFN-1.1	Il sistema deve permettere l'inserimento dell'indirizzo email in fase di registrazione	UC1.1
RFN-1.2	Il sistema deve permettere l'inserimento dell'username in fase di registrazione	UC1.2
RFN-1.3	Il sistema deve permettere l'inserimento della password in fase di registrazione	UC1.3
RFN-2	Il sistema deve mostrare un errore nel caso in cui i campi non siano validi in fase di registrazione	UC2
RFN-3	Il sistema deve permettere l'autenticazione di un utente registrato come elettore o amministratore	UC3
RFN-3.1	Il sistema deve permettere l'inserimento dell'indirizzo email in fase di autenticazione	UC3.1
RFN-3.2	Il sistema deve permettere l'inserimento della password in fase di autenticazione	UC3.2
RFN-4	Il sistema deve mostrare un errore nel caso in cui i dati inseriti negli appositi campi non vengano riconosciuti in fase di autenticazione	UC4
RFN-5	Il sistema deve permettere di visualizzare lo storico dei voti ad un elettore	UC5
RFN-5.1	Il sistema deve permettere di visualizzare le informazioni di ogni singolo voto appartenente allo storico di un elettore	UC5.1
RFN-6	Il sistema deve permettere di visualizzare tutte le elezioni disponibili ad un elettore	UC6
RFN-6.1	Il sistema deve permettere di visualizzare le informazioni di ogni singola votazione appartenente alle elezioni disponibili di un elettore	UC6.1
RFN-7	Il sistema deve permettere di visualizzare la maschera di voto di una specifica elezione ad un elettore	UC7
RFN-7.1	Il sistema deve permettere di visualizzare la lista dei partiti partecipanti nella maschera di voto di una specifica elezione	UC7.1
RFN-7.1.1	Il sistema deve permettere di visualizzare le informazioni di ogni singolo partito partecipante nella maschera di voto di una specifica elezione	UC7.1.1
RFN-8	Il sistema deve mostrare un errore nel caso in cui l'elettore provi ad accedere alla maschera di voto di un'elezione per la quale ha già espresso il proprio voto	UC8
RFN-9	Il sistema deve permettere l'uscita dalla piattaforma da parte di un utente precedentemente autenticato come elettore o amministratore	UC9

RFN-10	Il sistema deve permettere ad un elettore l'espressione della propria preferenza all'interno della maschera di voto di una specifica elezione	UC10
RFN-10.1	Il sistema deve permettere ad un elettore di selezionare esattamente un partito dalla lista dei partiti partecipanti nella maschera di voto	UC10.1
RFD-10.2	Il sistema deve permettere ad un elettore di confermare la scelta del partito selezionato, visualizzando un riquadro di riepilogo	UC10.2
RFN-11	Il sistema deve permettere ad un amministratore l'inserimento di una nuova elezione nella piattaforma	UC11
RFN-11.1	Il sistema deve permettere ad un amministratore l'inserimento del nome in fase di inserimento di una nuova elezione	UC11.1
RFN-11.2	Il sistema deve permettere ad un amministratore l'inserimento della tipologia in fase di inserimento di una nuova elezione	UC11.2
RFN-11.3	Il sistema deve permettere ad un amministratore l'inserimento della data di inizio in fase di inserimento di una nuova elezione	UC11.3
RFN-11.4	Il sistema deve permettere ad un amministratore l'inserimento della data di fine in fase di inserimento di una nuova elezione	UC11.4
RFN-11.5	Il sistema deve permettere ad un amministratore la selezione dei partiti partecipanti, tra quelli presenti nella piattaforma, in fase di inserimento di una nuova elezione	UC11.5
RFN-12	Il sistema deve mostrare un errore nel caso in cui i campi compilati in fase di inserimento di una nuova elezione dall'amministratore risultino invalidi perchè vuoti o con errori di formattazione del contenuto	UC12
RFN-13	Il sistema deve permettere ad un amministratore l'inserimento di un nuovo partito nella piattaforma	UC13
RFN-13.1	Il sistema deve permettere ad un amministratore l'inserimento del nome in fase di inserimento di un nuovo partito	UC13.1
RFN-13.2	Il sistema deve permettere ad un amministratore l'inserimento del nominativo del candidato in fase di inserimento di un nuovo partito	UC13.2
RFN-13.3	Il sistema deve permettere ad un amministratore l'inserimento del logo in fase di inserimento di un nuovo partito	UC13.3
RFN-14	Il sistema deve mostrare un errore nel caso in cui i campi compilati in fase di inserimento di un nuovo partito dall'amministratore risultino invalidi perchè vuoti o con errori di formattazione del contenuto	UC14
RFN-15	Il sistema deve permettere ad un amministratore la modifica di un'elezione presente nella piattaforma	UC15

RFN-15.1	Il sistema deve permettere ad un amministratore l'inserimento del nuovo nome in fase di modifica di una elezione	UC15.1
RFN-15.2	Il sistema deve permettere ad un amministratore l'inserimento della nuova tipologia in fase di modifica di una elezione	UC15.2
RFN-15.3	Il sistema deve permettere ad un amministratore l'inserimento della nuova data di inizio in fase di modifica di una elezione	UC15.3
RFN-15.4	Il sistema deve permettere ad un amministratore l'inserimento della nuova data di fine in fase di modifica di una elezione	UC15.4
RFN-15.5	Il sistema deve permettere ad un amministratore l'aggiornamento dei partiti partecipanti, sempre tra quelli presenti nella piattaforma, in fase di modifica di una elezione	UC15.5
RFN-16	Il sistema deve mostrare un errore nel caso in cui l'amministratore provi a modificare un'elezione che risulti già cominciata o terminata	UC16
RFN-17	Il sistema deve permettere ad un amministratore l'eliminazione di un'elezione presente nella piattaforma	UC17
RFN-18	Il sistema deve mostrare un errore nel caso in cui l'amministratore provi ad eliminare un'elezione che risulti già cominciata o terminata	UC18
RFN-19	Il sistema deve permettere ad un amministratore l'eliminazione di un partito presente nella piattaforma	UC19
RFN-20	Il sistema deve mostrare un errore nel caso in cui l'amministratore provi ad eliminare un partito che risulti partecipante in un'elezione già cominciata o terminata	UC20
RFN-21	Il sistema deve permettere ad un amministratore la visualizzazione di una lista contenente tutte le elezioni presenti nella piattaforma	UC21
RFN-21.1	Il sistema deve permettere ad un amministratore di visualizzare le informazioni di ogni singola elezione appartenente alla lista di tutte le elezioni presenti nella piattaforma	UC21.1
RFN-22	Il sistema deve permettere ad un amministratore la visualizzazione di una lista contenente tutti i partiti presenti nella piattaforma	UC22
RFN-22.1	Il sistema deve permettere ad un amministratore di visualizzare le informazioni di ogni singolo partito appartenente alla lista di tutti i partiti presenti nella piattaforma	UC22.1

RFD-23	Il sistema deve permettere ad un elettore di visualizzare il proprio username nella dashboard personale	Committente
--------	---	-------------

Tabella 3.2: Tabella del tracciamento dei requisiti qualitativi

Requisito	Descrizione	Fonte
RQN-1	Il codice sorgente della piattaforma deve essere pubblicato e versionato usando lo strumento Git	Committente
RQZ-2	La piattaforma deve garantire prestazioni di caricamento e SEO migliori utilizzando il rendering lato server con il framework Next.js	Committente

Tabella 3.3: Tabella del tracciamento dei requisiti di vincolo

Requisito	Descrizione	Fonte
RVN-1	La piattaforma deve essere sviluppata utilizzando il linguaggio di programmazione Typescript	Committente
RVN-2	La piattaforma deve essere sviluppata utilizzando il framework Angular	Committente
RVN-3	La piattaforma deve essere sviluppata utilizzando la libreria React.js	Committente
RVZ-4	La piattaforma deve effettuare il rendering del FE lato server utilizzando il framework Next.js	Committente

Capitolo 4

Progettazione e codifica

In questo capitolo vengono analizzati in modo dettagliato i moduli di [BE](#) e [FE](#) del progetto Voting-Online, descrivendo per entrambi le tecnologie utilizzate, la progettazione effettuata ed illustrando infine le maschere implementate.

4.1 Backend

Il progetto di stage proposto dall'azienda Sync Lab, prevedendo le prime tre settimane di formazione ed approfondimento riguardo al linguaggio Java e ai moduli fondamentali del [framework](#) Spring, mi ha permesso di realizzare, oltre al Frontend con Angular e React, anche il Backend per un consolidamento dello studio più efficace. Grazie a questa implementazione è stato possibile riutilizzare gli stessi servizi web [Representational State Transfer \(REST\)](#) per fornire i dati a entrambi i [FE](#), senza dover trovare soluzioni diverse allo stesso problema per le due tecnologie sopraccitate.

4.1.1 Tecnologie

Di seguito viene data una breve panoramica delle tecnologie utilizzate per lo sviluppo del Backend.

Maven

In informatica Apache Maven¹ è uno strumento di gestione di progetti software basati su Java e build automation.

Maven usa un costrutto conosciuto come *Project Object Model* (POM); un file XML che descrive le dipendenze fra il progetto e le varie versioni di librerie necessarie nonché le dipendenze fra di esse. In questo modo si separano le librerie dalla directory di progetto utilizzando questo file descrittivo per definirne le relazioni. Maven effettua automaticamente il download di librerie Java e plug-in Maven dai vari repository definiti scaricandoli in locale o in un repository centralizzato lato sviluppo. Questo permette di recuperare in modo uniforme i vari file JAR e di poter spostare il progetto indipendentemente da un ambiente all'altro avendo la sicurezza di utilizzare sempre le stesse versioni delle librerie.

¹Apache Maven Website. URL: <https://maven.apache.org/>.



Figura 4.1: Logo di Apache Maven

Il vantaggio derivante dall'utilizzo di questo strumento risulta molto rilevante in quanto la *build automation* permette l'automatizzazione dell'intero processo di sviluppo software, il quale normalmente è composto da diverse fasi, riducendo quindi il carico di lavoro del programmatore e diminuendo le possibilità di errore da parte dello stesso.

Spring Framework



Figura 4.2: Logo del framework Spring

In informatica Spring² è un **framework open source** per lo sviluppo di applicazioni su piattaforma Java, nato con l'intento di gestire la complessità nello sviluppo di applicazioni enterprise. Il **framework** in questione utilizza lo strumento Maven per gestire il concetto di **Inversion of Control**, il quale è stato reso popolare negli anni proprio da Spring.

Il **framework** Spring risulta un'ottima soluzione per lo sviluppo di applicazioni Java in quanto ha diverse caratteristiche importanti a suo favore:

- * E' *leggero* in quanto ha una struttura estremamente modulare e risulta quindi possibile utilizzare diverse funzionalità solo quando necessario, aggiungendone di nuove in modo incrementale senza sconvolgere il progetto in sviluppo;
- * E' *semplice* in quanto offre diversi moduli per una semplice risoluzione di vari problemi nello sviluppo software, come per esempio l'interfacciamento ad un Database o la gestione dell'autenticazione ed autorizzazione degli utenti nella propria applicazione;
- * E' *facile da testare* in quanto il **framework** nasce con la concezione che il codice di qualità debba essere facilmente testato, mettendo quindi a disposizione diversi strumenti nativi per varie tipologie di testing.

²Spring Framework Website. URL: <https://spring.io/projects/spring-framework>.

4.1.2 Descrizione

Il **BE** consiste in un servizio web **REST** realizzato utilizzando il linguaggio di programmazione Java e il **framework** Spring. Il progetto di base, creato sulla piattaforma Spring Initializr³, ha consentito di creare un'applicazione SpringBoot⁴ che, utilizzando i moduli Spring Data-JPA⁵ e Data-REST⁶, permette l'accesso ai dati necessari ai **FE** attraverso appositi **Endpoint**. La persistenza dei dati in questione avviene attraverso l'utilizzo di un Database, gestito con il **Database Management System (DBMS)** MySQL e hostato attraverso la piattaforma WAMP. La configurazione per l'interfacciamento di Spring al DB è disponibile nel file **application.properties**.

4.1.3 Architettura

Nella figura 4.3 viene raffigurato il diagramma dei package, il quale permette di visualizzare la struttura del modulo di Backend.

Di seguito vengono descritti i principali package:

- * **controller**: contiene gli **Endpoint** aggiuntivi e personalizzati necessari per fornire le funzionalità di sign-up e login in modo corretto. Ogni controller ha un riferimento con almeno uno strato di persistenza (repository), del quale può utilizzare i metodi offerti dal **framework** per l'interazione con le risorse presenti nel Database;
- * **repository**: contiene la dichiarazione delle interfacce necessarie al modulo Spring Data-REST per fornire automaticamente tutti gli **Endpoint** utilizzabili per interagire con le risorse gestite dal **framework**. Nel caso dell'**UserRepository** è stata bloccata l'esportazione del metodo di salvataggio di un nuovo utente offerto da Spring, in modo da rendere disponibile una diversa versione della funzionalità tramite il controller **UserController** presente nel package controller;
- * **model**: contiene le classi che rappresentano le entità gestite da Spring, che quindi corrispondono alle tabelle del Database, mentre i rispettivi campi dati corrispondono agli attributi;
- * **dto**: contiene i Data Transfer Object, ovvero gli oggetti usati per trasferire dati tra il Backend e il Frontend. Questi oggetti vengono utilizzati quindi come contenitori di dati in modo da serializzare le informazioni inoltrate attraverso il protocollo HTTP sia in entrata che in uscita al modulo di Backend;
- * **security**: contiene la definizione di un filtro utilizzato per effettuare una configurazione di sicurezza di base nel progetto. L'utilità del filtro consiste principalmente nel controllare la provenienza delle richieste HTTP, in modo da rispondere con le risorse richieste solo all'indirizzo corrispondente al **FE**, e nell'impostare correttamente l'header delle risposte fornite dal server.

³Spring Initializr Website. URL: <https://start.spring.io/>.

⁴SpringBoot. URL: <https://spring.io/projects/spring-boot>.

⁵Spring Data JPA. URL: <https://spring.io/projects/spring-data-jpa>.

⁶Spring Data REST. URL: <https://spring.io/projects/spring-data-rest>.

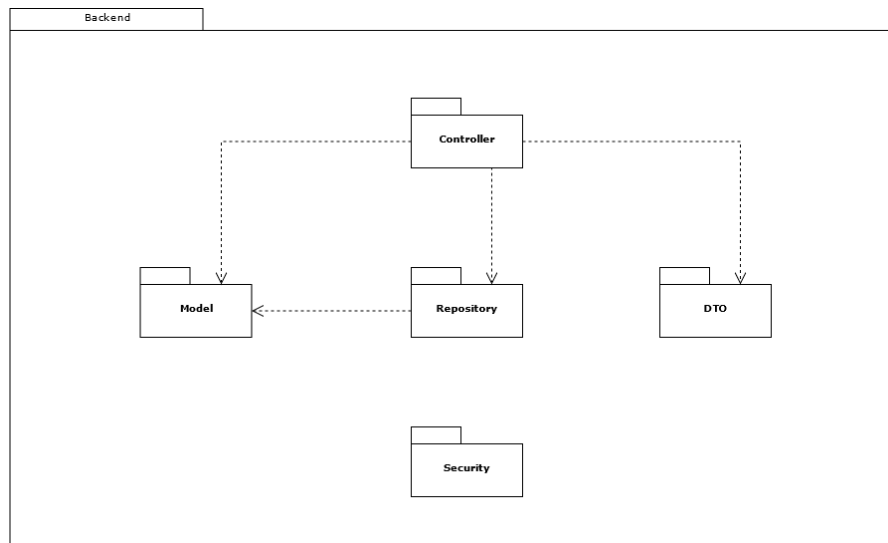


Figura 4.3: Modulo Backend

4.1.4 Design pattern utilizzati

- * **Dependency Injection:** Il dependency injection è un pattern integrato di default nel [framework](#) Spring per migliorare l'efficienza e la modularità del codice. Questo design pattern viene implementato da Spring in vari modi, in particolare nel modulo di Backend è stata utilizzata l'annotazione **Autowired**. Questa annotazione consente alle classi di dichiarare tra i campi dati di quali servizi o repository si ha bisogno e in fase di inizializzazione tali dipendenze gli vengono fornite dall'esterno. Questo pattern migliora molto anche la testabilità del codice in quanto permette l'utilizzo di servizi di mock evitando di, per esempio, inviare chiamate HTTP al server vero e proprio, sostituendo dei componenti con una loro alternativa fittizia;
- * **Singleton**⁷: Il singleton è un pattern già integrato con Spring che ha lo scopo di garantire un'unica istanza di una determinata classe in tutto l'applicativo. Il [framework](#) in questione lo implementa utilizzando l'annotazione **Bean**, la quale appunto garantisce l'esistenza di un'unica istanza della classe target e consente di iniettare quest'istanza ad ogni classe che la dichiara come dipendenza;

4.2 Tecnologie per il Frontend

Di seguito viene data una breve panoramica delle tecnologie utilizzate per lo sviluppo del Frontend.

Per quanto riguarda **Angular**, **React** e **Next.js** viene effettuata una trattazione in modo approfondito nel [capitolo successivo](#).

⁷Johnson R. Gamma E. Vlissides J. *Design Patterns: Elements of Reusable Object-Oriented Software*. 1997.

JavaScript



Figura 4.4: Logo di Javascript

JavaScript è un linguaggio di programmazione orientato agli oggetti e agli eventi, comunemente utilizzato nella programmazione Web lato client (esteso poi anche al lato server) per la creazione, in siti web e applicazioni web, di effetti dinamici interattivi tramite funzioni di script invocate da eventi innescati a loro volta in vari modi dall'utente sulla pagina web in uso.

Le funzioni di script, utilizzati dunque nella logica di presentazione, possono essere opportunamente inserite in file HTML, in pagine JSP o in appositi file separati con estensione *.js* poi richiamati nella logica di business.

Le caratteristiche principali di JavaScript sono:

- * essere un linguaggio interpretato;
- * la sintassi è relativamente simile a quella dei linguaggi C;
- * è un linguaggio debilmente tipizzato e debolmente orientato agli oggetti.

Typescript



Figura 4.5: Logo di Typescript

TypeScript è un linguaggio di programmazione [open source](#) sviluppato da Microsoft e consiste in un Super-set di JavaScript che basa le sue caratteristiche su ECMAScript 6. Il linguaggio estende la sintassi di JavaScript in modo che qualunque programma scritto in JavaScript sia anche in grado di funzionare con TypeScript senza nessuna modifica. Il linguaggio nasce dal crescente bisogno di un linguaggio frontend per lo sviluppo di applicazioni JavaScript su larga scala e dalla necessità di sicurezza e robustezza, sia da parte di sviluppatori interni a Microsoft sia da parte di clienti e sviluppatori indipendenti. Typescript è destinato ad essere compilato in JavaScript per poter essere interpretato da qualunque web browser o app.

Il linguaggio di programmazione in questione permette di aggiungere o rendere più flessibili e potenti varie caratteristiche di Javascript:

- * Firma dei metodi;

- * Classi;
- * Interfacce;
- * Moduli;
- * Operatore " $=>$ " che permette di definire le funzioni anonime;
- * Tipi di dato.

4.3 Frontend con Angular

4.3.1 Descrizione

Il **FE** in questione consiste in un'applicazione web realizzata con Angular, utilizzando molte delle *best practices*⁸ del **framework**, e il linguaggio di programmazione Typescript. Il progetto di base, creato utilizzando lo strumento **ng** della Angular CLI⁹, ha permesso di sviluppare una piattaforma che fornisce agli utenti elettori ed amministratori tutte le funzionalità necessarie a soddisfare i requisiti rilevati nell'**analisi dei requisiti**, utilizzando i dati forniti dal **BE** descritto nella precedente sezione.

Infine è necessario citare il fatto che il progetto Voting-Online, non avendo come focus principale la sicurezza nello sviluppo della piattaforma, ha consentito di implementare una gestione della sessione utente utilizzando il **Local Storage**, in modo da poter archiviare e recuperare semplicemente diverse informazioni ed utilizzarle per il corretto rendering lato client delle pagine web.

4.3.2 Architettura di Angular

L'architettura di un'applicazione Angular si basa su alcuni concetti fondamentali. L'elemento costitutivo di base è il **component**, ovvero una classe che si occupa di gestire la vista, logica, e stile di un'intera pagina o di una determinata porzione di **Graphical User Interface (GUI)**, come descritto in modo più dettagliato nel capitolo successivo. Ogni component può a sua volta contenere diversi component figli, avendo in questo modo la possibilità di creare maschere in modo modulare ed altamente manutenibile.

Altri elementi fondamentali del **framework** in questione risultano i **servizi**, ovvero delle classi che forniscono delle funzionalità non direttamente collegate alle viste e possono essere iniettati nei component come dipendenze, e i **moduli**, i quali consistono in una collezione di servizi e configurazioni dedicate ad uno o più componenti.

Un'applicazione Angular è definita da un insieme di moduli ed ha sempre almeno un modulo radice che abilita il bootstrap e in genere ha molti più moduli di funzionalità.

4.3.3 Architettura dell'applicazione sviluppata

Nella figura 4.6 viene raffigurato il diagramma dei package del modulo di **FE**, il quale permette di visualizzare la struttura del progetto e contiene i seguenti elementi fondamentali:

⁸Angular Best Practices. URL: <https://angular.io/guide/styleguide>.

⁹Angular CLI. URL: <https://angular.io/cli>.

- * **components:** contiene tutti i component usati per lo sviluppo dell'applicazione, sia quelli «*principali*», scanditi nel file di configurazione delle rotte *routing.module.ts* e utilizzati per renderizzare intere pagine web, sia quelli «*secondari*», utilizzati per renderizzare delle sezioni di pagina e contenuti in quelli principali;
- * **services:** contiene delle classi corrispondenti ai servizi Angular, i quali in particolare effettuano le chiamate HTTP al [BE](#), fornendo l'accesso ai dati necessari ai component;
- * **lib:** contiene le funzioni di supporto per standardizzare le chiamate al [BE](#);
- * **classes:** contiene i tipi utilizzati dai component e dai service;
- * **assets:** contiene file come immagini ed icone, utilizzate dalle viste.

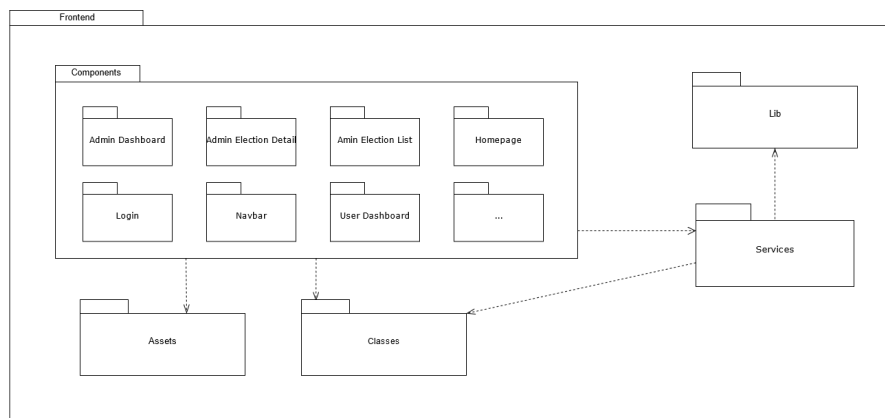


Figura 4.6: Modulo Frontend realizzato con Angular

4.3.4 Design pattern utilizzati

- * **Dependency Injection:** consiste in un design pattern per il quale è sufficiente dichiarare le dipendenze di cui un componente necessita e, quando il componente verrà istanziato, un iniettore si prenderà carico di risolvere le dipendenze (attuando dunque l'inversione del controllo). Il pattern in questione implementato da Angular è di tipo **constructor**, ovvero prevede che in ogni componente vengano dichiarati nel costruttore di quali servizi ha bisogno e in fase di inizializzazione tali dipendenze gli vengono fornite dall'esterno;
- * **Singleton**¹⁰: Il singleton è un pattern integrato di default in Angular che ha lo scopo di garantire un'unica istanza di una determinata classe in tutto l'applicativo. Il [framework](#) in questione lo implementa nei **servizi**, i quali possono venire iniettati come unica istanza in diversi componenti, permettendo di condividere funzionalità e stato;

¹⁰Gamma E., *Design Patterns: Elements of Reusable Object-Oriented Software*.

- * **Decorator**¹¹: Il decorator è un design pattern integrato di default in Angular, utile per estendere la funzionalità degli oggetti mantenendo intatte le funzionalità base degli stessi. Questo pattern è implementato in Angular come **class decorator** per indicare se una classe è di tipo component o module oppure come **property decorator** per indicare se una variabile è visibile all'esterno della classe in input o output, il tutto senza dover scrivere codice aggiuntivo.

4.4 Frontend con React

4.4.1 Descrizione

Il **FE** in questione consiste in un'applicazione web realizzata utilizzando la libreria **React** per la creazione delle interfacce utente ed il **framework Next.js** per la gestione del rendering del sito.

Il progetto di base, creato utilizzando lo strumento **create-next-app** del package manager NPM¹², ha consentito lo sviluppo di una piattaforma che, allo stesso modo dell'applicazione sviluppata con Angular, fornisce agli utenti elettori ed amministratori tutte le funzionalità necessarie a soddisfare i requisiti rilevati nell'**analisi dei requisiti**, sfruttando i dati forniti dal **BE** descritto precedentemente.

Infine è necessario citare il fatto che il progetto Voting-Online, non avendo come focus principale la sicurezza nello sviluppo della piattaforma, ha consentito di implementare una gestione della sessione utente utilizzando i **Cookie**, in modo da poter archiviare e recuperare semplicemente diverse informazioni ed utilizzarle per il corretto rendering lato server delle pagine web.

4.4.2 Architettura dell'applicazione sviluppata

Nella figura 4.7 viene raffigurato il diagramma dei package del modulo di **FE**, il quale permette di avere una chiara visione della struttura del progetto creato seguendo il tutorial ufficiale¹³ del **framework Next.js**. Gli elementi fondamentali nell'architettura sono i seguenti:

- * **pages**: Next.js utilizza un sistema di routing basato sul file system, quindi i componenti React contenuti nei file dentro alla cartella pages renderizzano le pagine web presenti nella piattaforma;
- * **components**: contiene tutti i componenti React utilizzati dalle pages per renderizzare parti della **GUI** dell'applicativo;
- * **images**: contiene tutte le immagini utilizzate dalle viste;
- * **services**: contiene tutte le funzioni che consentono di effettuare le chiamate HTTP al **BE** e che forniscono i dati necessari ai component e alle pages;
- * **classes**: contiene i tipi utilizzati dai component e dai service;
- * **styles**: contiene l'unico file style.css, il quale applica lo stile alla piattaforma con scope globale;

¹¹Gamma E., *Design Patterns: Elements of Reusable Object-Oriented Software*.

¹²Node Package Manager. URL: <https://www.npmjs.com/>.

¹³Next.js Tutorial. URL: <https://nextjs.org/docs/getting-started>.

- * **lib**: contiene le funzioni di supporto utilizzate dai service per standardizzare le chiamate al **BE**.

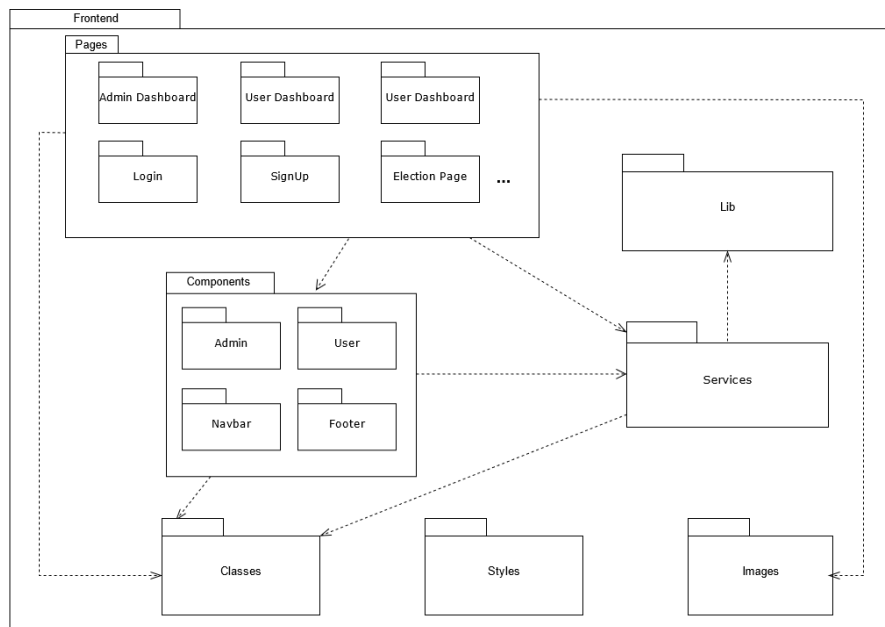


Figura 4.7: Modulo Frontend realizzato con Next.js e React

4.4.3 Il pre-rendering di Next.js

La piattaforma, utilizzando Next.js, effettua il **pre-rendering** delle pagine web lato server, ovvero genera e fornisce direttamente codice HTML, senza farlo generare al browser client. Ad ogni pagina HTML generata viene associato il codice JavaScript minimo necessario. Successivamente, quando una pagina viene caricata dal browser, il suo codice JavaScript viene eseguito e rende la pagina completamente interattiva. Nella figura 4.8 viene raffigurato uno schema generale per mostrare l'interazione tra diversi elementi del modulo in questione, in relazione al pre-rendering effettuato dalla piattaforma.

Il **FE** utilizza il **Server-Side Rendering** (SSR), attraverso la funzione **getServerSideProps** per effettuare il fetch dei dati dal **BE**, nelle pagine in cui è importante avere i dati sempre aggiornati, come per esempio le dashboard dell'admin e dell'utente.

Viene invece utilizzato lo **Static-Site Generation** (SSG) nelle pagine in cui o non serve effettuare il fetch di dati o non è importante averli aggiornati ad ogni richiesta, come per esempio la homepage o la maschera di login.

NB: Next.js effettua solamente il SSR in ambiente di sviluppo, mentre effettua anche il SSG una volta effettuato il deploy dell'applicativo.

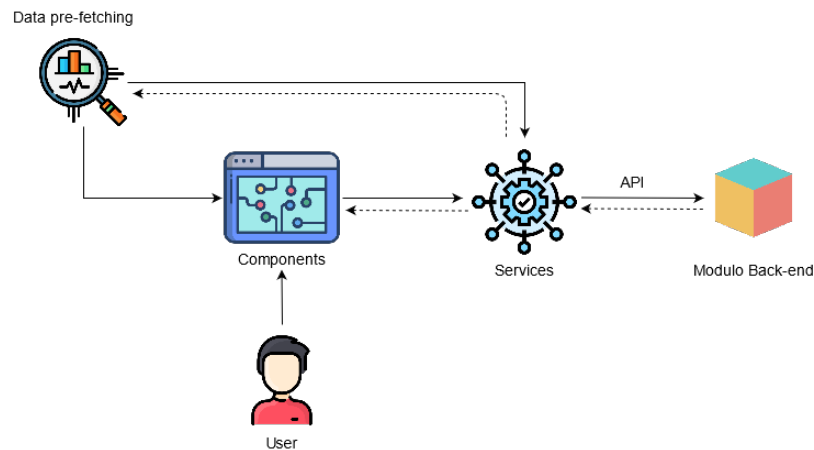


Figura 4.8: Schema generale del Frontend in Next.js e React

4.4.4 Design pattern utilizzati

* **React Best Practices:** le buone pratiche di React, consigliate nel sito ufficiale¹⁴ della libreria e dalla community, sono svariate e consentono di creare interfacce utente con codice corretto, leggibile, modulare e manutenibile. Le best practices utilizzate in particolare nel progetto in questione sono le seguenti:

- scomporre la **GUI** in una gerarchia di component, seguendo il principio di singola responsabilità tipico della programmazione ad oggetti;
- identificare lo **stato minimo indispensabile** necessario per ottenere tutte le funzionalità desiderate e decidere in quale component farlo giacere. Il principio base, valido in generale per qualsiasi ambito della programmazione, consiste nel DRY (ovvero «Don't Repeat Yourself»);
- utilizzare la **destrutturazione degli oggetti**, soprattutto nel caso di quelli passati tramite props, in modo da rendere il codice più snello e leggibile;
- utilizzare i **functional component** e i **react hooks** in quanto consentono una gestione più rapida e leggibile dello stato rispetto ai class component, anche se dal punto di vista di React le due tipologie di component sono equivalenti.

4.5 Maschere prodotte

Con lo scopo di effettuare un paragone dettagliato tra Angular e React le maschere **FE** sono state realizzate in modo che fossero il più possibile coincidenti. Per questo motivo nell'attuale sezione, essendo le interfacce utente praticamente uguali sia a livello di funzionalità che di stile, viene elencato un unico insieme di maschere per illustrare il prodotto software, risultato del progetto in questione. Le maschere prodotte hanno permesso di soddisfare la totalità dei requisiti riscontrati nell'**analisi dei requisiti** per entrambe le tecnologie sopracitate, arrivando ad avere un funzionamento completo della piattaforma che verrà descritta in seguito.

¹⁴React Best Practices. URL: <https://reactjs.org/docs/thinking-in-react.html>.

Inoltre, prima di elencare le diverse maschere del sito, risulta utile notare la presenza in ogni pagina di una **navigation bar** dinamica in base alla tipologia di utente autenticato nella parte superiore dell'ipertesto: infatti se l'utente risulta loggato con delle credenziali da elettore ha la possibilità di accedere, tramite l'apposito link, alla dashboard personale per esprimere il proprio voto in un'elezione disponibile o visualizzare lo storico dei propri voti, oltre a visualizzare il proprio username che conferma l'esito positivo alla procedura di autenticazione. Se l'utente risulta invece autenticato come amministratore può accedere alla dashboard per la gestione delle elezioni e dei partiti nella piattaforma. In entrambi i casi è comunque disponibile un pulsante per effettuare il logout dalla piattaforma.

Infine è possibile notare un **footer** comune nella parte inferiore di ogni pagina dell'applicazione web.

4.5.1 Homepage

La pagina di benvenuto nella piattaforma è costituita da:

- * un contenuto statico per accogliere l'utente con una breve spiegazione delle funzionalità disponibili nell'applicazione;
- * due pulsanti per indirizzare l'utente alle pagine di autenticazione o di registrazione, visualizzabili solamente dall'utente non autenticato.

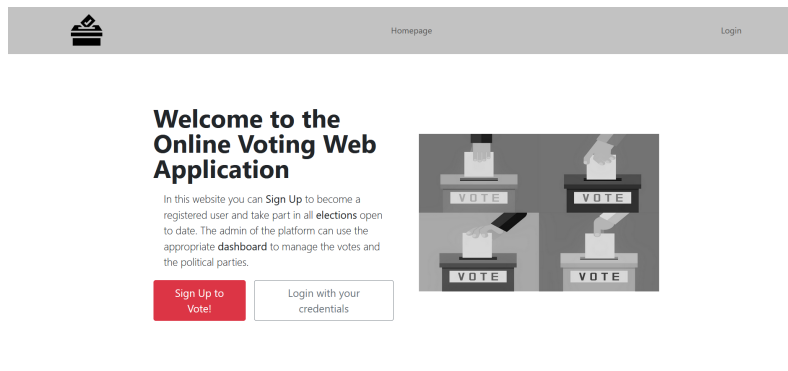
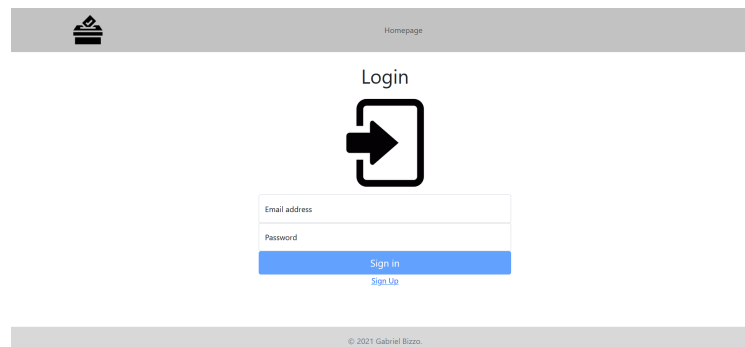


Figura 4.9: Homepage

4.5.2 Autenticazione

La pagina di autenticazione richiede che l'utente non autenticato inserisca l'indirizzo email e la password che ha utilizzato per registrarsi nella piattaforma per poterci accedere.

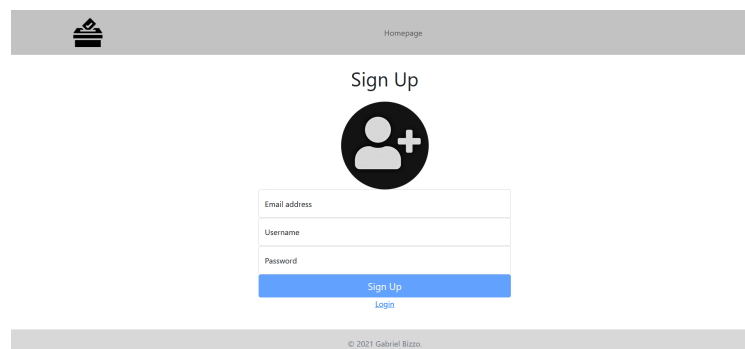


The screenshot shows a web page with a grey header bar containing a logo on the left and the text "Homepage" on the right. The main content area is white and features the word "Login" at the top. Below it is a large icon of a square with a right-pointing arrow. Underneath the icon is a form with two input fields labeled "Email address" and "Password". Below these fields is a blue button labeled "Sign in" and a link labeled "Sign Up". At the bottom of the page is a grey footer bar with the text "© 2021 Gabriel Bizzo."

Figura 4.10: Pagina di autenticazione

4.5.3 Registrazione

La pagina di registrazione richiede che l'utente non autenticato inserisca l'indirizzo email, l'username e la password che desidera utilizzare all'interno della piattaforma.



The screenshot shows a web page with a grey header bar containing a logo on the left and the text "Homepage" on the right. The main content area is white and features the words "Sign Up" at the top. Below it is a large icon of a person with a plus sign. Underneath the icon is a form with three input fields labeled "Email address", "Username", and "Password". Below these fields is a blue button labeled "Sign Up" and a link labeled "Login". At the bottom of the page is a grey footer bar with the text "© 2021 Gabriel Bizzo."

Figura 4.11: Pagina di Registrazione

4.5.4 Admin Dashboard

L'amministratore, dopo aver effettuato la procedura di autenticazione nel sistema, può accedere alla pagina di monitoraggio della piattaforma che gli permette di gestire tutte le elezioni e i partiti in essa presenti.

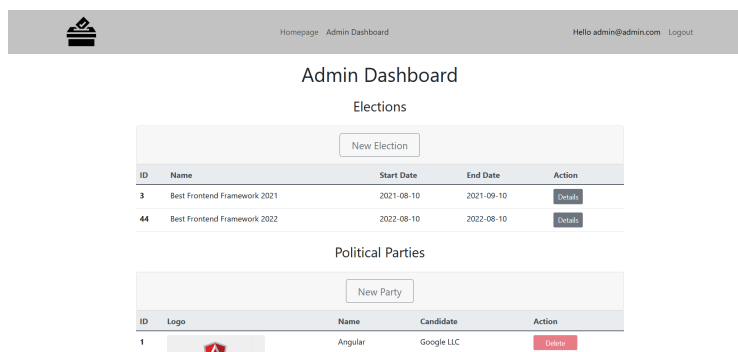


Figura 4.12: Pagina della Admin Dashboard

Elezioni

L'amministratore può visualizzare le informazioni di ogni elezione presente nella piattaforma nell'apposita tabella e ha la possibilità di eseguire diverse operazioni:

- * *visualizzare i dettagli* di una singola elezione cliccando sul pulsante "Details", facendo così comparire una finestra in sovrapposizione che elenca il nome, la tipologia, la data di inizio e fine ed i partiti partecipanti dell'elezione desiderata;
- * *eliminare* l'elezione della quale si stanno visualizzando i dettagli cliccando sull'apposito pulsante, il quale risulta abilitato solo nel caso in cui l'elezione non risulti già cominciata o terminata;
- * *modificare* i dati dell'elezione della quale si stanno visualizzando i dettagli cliccando sull'apposito pulsante per rendere editabili i campi contenenti le informazioni. Il bottone che abilita la modifica risulta abilitato e permette l'operazione in questione solamente nel caso in cui l'elezione risulta ancora da cominciare.

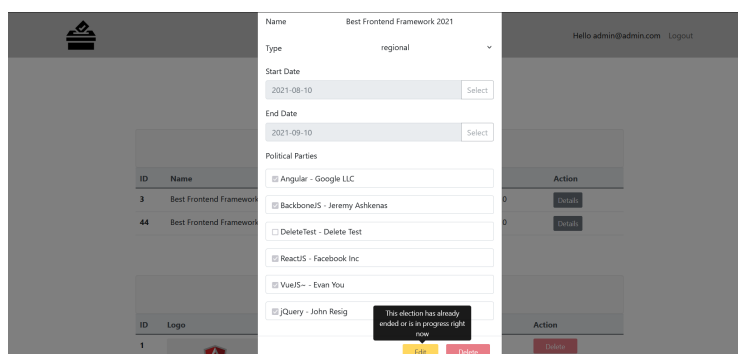


Figura 4.13: Finestra contenente i dettagli di un'elezione

L'amministratore ha la possibilità, inoltre, di *aggiungere* una nuova elezione nella piattaforma compilando il form che compare cliccando il pulsante "New Election". I campi da compilare consistono nel nome, tipologia, data di inizio, data di fine ed elenco dei partiti partecipanti. Nel caso in cui i campi vengano lasciati vuoti o compilati in modo errato degli appositi messaggi verranno visualizzati in modo da consentire, in

seguito ad una modifica dei campi compilati erroneamente, un ulteriore tentativo di inserimento all'utente.

Figura 4.14: Form di creazione di una nuova elezione

Partiti

L'amministratore, in modo simile alle elezioni, ha la possibilità di effettuare diverse operazioni per la gestione dei partiti presenti nella piattaforma:

- * *visualizzare le informazioni* di ogni partito nell'apposita tabella, in particolare il nome, logo e nominativo del candidato del partito;
- * *eliminare* un partito dalla piattaforma cliccando sull'apposito pulsante, il quale risulta abilitato solo nel caso in cui il partito non risulti già presente in un'elezione;
- * *aggiungere* un nuovo partito nella piattaforma compilando il form che compare cliccando sul pulsante "New Party". Il funzionamento del form risulta uguale a quello per creare una nuova elezione, tuttavia i campi richiesti in questo caso sono il nome, logo e nominativo del candidato del partito.

4.5.5 User Dashboard

L'elettore, dopo aver effettuato la procedura di autenticazione nel sistema, può accedere alla dashboard personale che gli permette di esprimere la propria preferenza in un'elezione disponibile e di controllare il proprio storico dei voti.

Name	Type	Start Date	End Date	You have already voted for this election
Best Frontend Framework 2021	regional	2021-08-10	2021-09-10	<input type="button" value="Vote"/>
Best Frontend Framework 2022	municipal	2022-08-10	2022-08-10	<input type="button" value="Vote"/>

Election Name	Election Start Date	Election End Date	Vote Date	Voted Party and Candidate
Best Frontend Framework 2021	2021-08-10	2021-09-10	2021-08-10	ReactJS - Facebook Inc

Figura 4.15: Pagina della User Dashboard

Elezioni

L'elettore può visualizzare le informazioni di ogni elezione disponibile, ovvero aperta alle votazioni nel momento di consultazione della piattaforma, nell'apposita tabella e ha la possibilità di accedere alla maschera di voto di una specifica elezione solamente nel caso in cui non ha già espresso la propria preferenza nella votazione in questione.

Storico dei voti

L'elettore ha la possibilità di consultare le proprie preferenze espresse in elezioni passate nell'apposita tabella, visualizzando in particolare le seguenti informazioni: il nome, la data di inizio e fine, la data di espressione del voto e il partito partecipante votato.

4.5.6 Maschera di voto di un'elezione

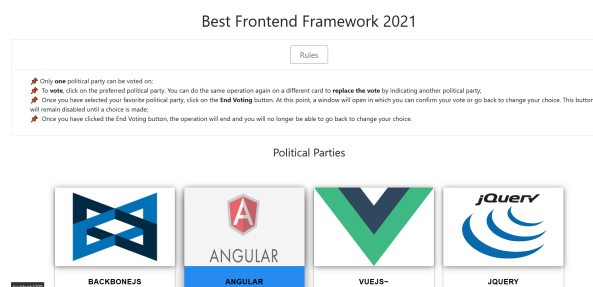


Figura 4.16: Maschera di voto di un'elezione

L'elettore, dopo aver effettuato la procedura di autenticazione nel sistema, viene indirizzato alla maschera di voto dell'elezione in questione. In questa pagina l'utente può eseguire diverse operazioni:

- * visualizzare una lista, che compare cliccando il pulsante "Rules", contenente le *regole* per esprimere correttamente il proprio voto nella piattaforma;
- * visualizzare l'elenco dei *partiti partecipanti* all'elezione e, cliccando su uno di essi, esprimere la propria preferenza;
- * *confermare il voto* cliccando sull'apposito pulsante e, una volta letta la finestra che compare in sovrimpressione contenente l'informazione del partito selezionato, premendo il tasto "Confirm".

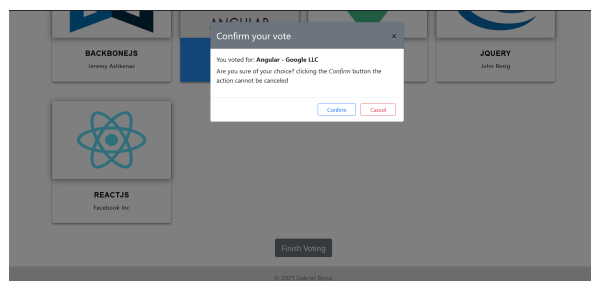


Figura 4.17: Pop-up di conferma della preferenza dell'elettore

Capitolo 5

Confronto tra Angular e React

In questo capitolo viene effettuato il confronto tra Angular e React. Inizialmente viene presentata una panoramica introduttiva sulle tecnologie in questione, proseguendo poi con un paragone tra i dettagli tecnici e le performance, illustrando infine una riflessione conclusiva.

5.1 Introduzione

5.1.1 Angular

Angular¹ è una piattaforma [open source](#) per lo sviluppo di applicazioni web con [licenza MIT](#), evoluzione di AngularJS, sviluppata principalmente da Google.



Figura 5.1: Angular Logo

Angular, essendo un [framework](#) molto vasto, fornisce diversi strumenti e tecnologie oltre alla possibilità di sviluppo di interfacce utente, tra le quali citiamo la gestione delle rotte, la dependency injection e il two way data-binding. Per questo motivo il [framework](#) offre agli sviluppatori maggiori soluzioni ai problemi comuni dello sviluppo

¹Angular Website. URL: <https://angular.io/>.

di applicazioni web rispetto a React, avendo come rovescio della medaglia il fatto di creare dei progetti molto più pesanti a livello di memoria. Quest'ultimo aspetto viene approfondito successivamente nell' [apposita sezione](#).

Risulta utile citare, infine, che questa tecnologia è arrivata attualmente alla versione 12.1.4 e viene utilizzata da diverse aziende importanti per lo sviluppo del loro software proprietario.

Alcuni esempi di piattaforme create con Angular sono i seguenti: Gmail, MS Office, Paypal, Overleaf.

5.1.2 React

React² è una libreria JavaScript [open source](#) per la creazione di interfacce utente in applicazioni web, sviluppata e mantenuta principalmente da Facebook.



Figura 5.2: React Logo

React, essendo una libreria per la creazione di UI, tende a offrire meno strumenti e soluzioni a problemi comuni dello sviluppo di applicazioni web rispetto ad Angular, dovendosi affidare a librerie esterne create da diverse aziende o dalla community, con il vantaggio di avere un progetto più snello e leggero a livello di memoria. Questo aspetto viene approfondito successivamente nell' [apposita sezione](#).

Come fatto per la precedente tecnologia, è utile citare il fatto che React sia arrivato attualmente alla versione 17.0.2 e venga utilizzato da diverse aziende importanti per lo sviluppo delle loro piattaforme.

Di seguito vengono riportati alcuni esempi di software sviluppato utilizzando React: Airbnb, Discord, Instagram.

5.1.3 I componenti

Tutti i [framework](#) per il frontend di applicazioni web hanno un obiettivo comune: rendere lo sviluppo web più agevole e trasparente, andando a semplificare molte operazioni che, eseguite con gli strumenti di base come HTML, CSS e Javascript, risultano molto complesse e richiedono molto tempo.

La soluzione a questo problema proposta da entrambe le tecnologie in esame, seppur gestita in maniera differente, consiste nei **component**.

²React Website. URL: <https://it.reactjs.org/>.

Un component corrisponde ad un mattoncino di base per la costruzione di applicazioni web complesse. Ogni component ha il controllo di una determinata sezione di schermo, acquisendone la responsabilità di renderizzare i dati ricevuti e gestire l'interazione con l'utente. Il fatto che ogni component possa averne altri al suo interno crea un modello di programmazione che consente una forte modularità del codice, favorendo il test di unità e la leggibilità del codice stesso.

5.2 Dettagli tecnici

5.2.1 Linguaggio di programmazione

Angular utilizza come linguaggio di programmazione predefinito **Typescript**, in quanto la creazione di un progetto avviene attraverso lo strumento dell'Angular CLI³ **ng** e quest'ultimo provvede a creare un workspace che comprende una configurazione di base del linguaggio in questione.

React invece consiste in una libreria Javascript e, utilizzando lo strumento **create-react-app** con il package manager npm⁴ per la creazione di un progetto, la configurazione base utilizza proprio il linguaggio **Javascript**. E' tuttavia possibile configurare un nuovo progetto con il linguaggio Typescript utilizzando lo stesso tool con un ulteriore flag: **npx create-react-app my-app-name -template typescript**.

5.2.2 Gestione dei componenti

Come è stato descritto nell'introduzione, i component risultano i mattoni di base delle applicazioni web create con entrambe le tecnologie in esame.

Angular implementa questi componenti utilizzando tre file che consentono di configurare diverse caratteristiche:

- * **file.component.ts**: un file Typescript che contiene l'implementazione di una classe nella quale si possono inserire lo stato e le funzionalità offerte all'utente attraverso funzioni che possono agire in modo asincrono ed event-driven;
- * **file.component.html**: un file che contiene codice HTML (con la possibilità di estenderne le funzionalità attraverso gli attributi direttive⁵) e che corrisponde al template della vista che viene renderizzata dal browser;
- * **file.component.css**: un file CSS che contiene lo stile applicabile al template della vista del singolo componente.

React invece implementa questi componenti in un unico file **component.js** (o **component.tsx** se il progetto è stato configurato per Typescript) attraverso l'utilizzo del codice **JSX**, ovvero un'estensione di Javascript (non obbligatoria in quanto è possibile utilizzare anche esclusivamente codice Javascript).

Il codice JSX consiste in una sorta di fusione tra un linguaggio di template e il linguaggio Javascript, consentendo di creare "elementi react" che possono essere successivamente

³ Angular CLI. URL: <https://angular.io/cli>.

⁴ Node Package Manager. URL: <https://www.npmjs.com/>.

⁵ Direttive Angular. URL: <https://angular.io/guide/attribute-directives>.

renderizzati nel DOM.

I component possono essere definiti attraverso l'utilizzo delle classi javascript (implementando un costruttore che consente di settare lo stato e la funzione `render()`) o attraverso la definizione di una funzione (in questo caso si definisce "componente funzionale").

5.2.3 Gestione dello stato

Lo stato, citato nella sezione precedente, corrisponde a dei dati appartenenti al componente che, se modificati, causano una nuova renderizzazione anche della vista del componente stesso o dei suoi "figli".

Angular permette la gestione dello stato attraverso il **two way data-binding**, ovvero la possibilità, attraverso un'apposita sintassi, di legare in due direzioni i dati definiti nella classe (modello) con i dati usati dal template HTML (vista). In questo modo se i dati cambiano nel modello le modifiche vengono riflesse nella vista effettuando un nuovo rendering del componente e, nel verso opposto, se l'utente interagisce con la vista i dati del modello vengono aggiornati automaticamente.

React invece permette la gestione dello stato solamente attraverso un **one way data-binding**, ovvero se i dati (modello) vengono modificati allora tale modifica viene riflessa anche nella vista effettuando un nuovo rendering del componente, ma non accade il contrario. Infatti per modificare i dati dello stato è necessario creare apposite funzioni che reagiscano agli eventi innescati dagli elementi presenti nel DOM e che modifichino i dati del modello.

5.2.4 Gestione dello stile

Lo stile da applicare al template HTML è supportato da entrambe le tecnologie, seppur con qualche lieve differenza.

Angular, come descritto nella sezione relativa ai componenti, utilizza l'apposito **file.component.css** per applicare lo stile con scope locale (ovvero sul singolo componente) al template HTML definito in un file separato.

In *React* invece viene utilizzato principalmente un approccio in cui si usa un **unico file di stile** con codice CSS con scope globale (ovvero su tutta l'applicazione web). La libreria tuttavia mette a disposizione la possibilità di creare diversi **moduli CSS** in modo da poter definire gli stessi nomi di classe in file differenti senza il rischio di una collisione tra nomenclature.

5.3 Performance

5.3.1 Memoria

Come accennato nella sezione introduttiva, Angular è un **framework** completo, il quale offre molti più strumenti rispetto a React per lo sviluppo di applicazioni web. Questo aspetto, anche se può sembrare un netto punto a favore nei confronti di Angular, va analizzato in modo da valutare pro e contro.

Angular, attraverso lo strumento apposito della [Angular CLI](#) alla versione 12.1.4, crea un progetto di base dal peso di circa **344MB** (sul sistema operativo Windows 10 Home). Nel progetto è presente una configurazione di base del linguaggio TypeScript, oltre che a strumenti utili come, ad esempio, la gestione delle rotte, il two way data-binding e le librerie jasmine/karma per effettuare unit testing. Tutti questi ottimi strumenti, utili ad uno sviluppatore con l'intento di creare una piattaforma partendo da basi solide, possono tuttavia risultare superflui per creare delle applicazioni web più semplici o a scopo prototipale, rendendo solamente il progetto più pesante a livello di memoria.

React invece, attraverso l'apposito strumento del [Node Package Manager](#) alla versione 7.19.1, crea un progetto di partenza dalle dimensioni su disco di circa **215MB** (sul sistema operativo Windows 10 Home). Come si può notare il workspace creato da React risulta molto più leggero in memoria, avvicinandosi alla metà delle dimensioni rispetto a quello generato con Angular, offrendo certamente meno servizi allo sviluppatore. Infatti risultano disponibili, oltre alla libreria jest per effettuare unit testing, pochi altri servizi necessari al funzionamento dell'applicazione web. Questo aspetto consente, specialmente ad uno sviluppatore esperto, di integrare attraverso il proprio package manager solamente le funzionalità di cui si ha bisogno in modo modulare, anche se ciò potrebbe risultare problematico ai programmatori novizi.

5.3.2 Site Rendering

Entrambe le tecnologie, per fornire al browser il documento da visualizzare a schermo, utilizzano di base il **Client-Side Rendering (CSR)**. Con una soluzione di rendering lato client, il server esegue il rendering di una pagina vuota con un dei riferimenti ai file Javascript dell'applicazione web. La pagina vuota viene inviata al browser client, che inizia a eseguire l'app, compila il tutto, quindi effettua le chiamate [Application Program Interface \(API\)](#) necessarie e a visualizzare il contenuto della pagina.

Questa modalità di rendering consente di dare poco carico di lavoro al server in quanto invia una pagina vuota al client, tuttavia presenta problemi di diverse tipologie, ad esempio nel caso in cui l'applicazione web è di grandi dimensioni e il client ha una connessione lenta, poiché subirà un notevole ritardo nel caricamento e vedrà la pagina renderizzata solo quando sarà presente tutto il contenuto (causando un lungo caricamento con una pagina vuota alla prima navigazione).

Per ovviare a questo problema sono disponibili principalmente due tecniche di pre-rendering:

- * **Server-Side Rendering (SSR)**: corrisponde alla capacità di un'applicazione di contribuire alla visualizzazione della pagina web sul server invece di renderizzarla nel browser. Ciò significa che se di un'applicazione viene eseguito il rendering lato server, il suo contenuto corrisponde a codice HTML che viene generato in anticipo dal server e successivamente passato al browser del client per essere visualizzato dall'utente. Con il rendering lato client è diverso, in quanto bisogna navigare su una pagina prima che vengano recuperati i dati dal server, il che significa che l'utente dovrebbe aspettare alcuni secondi prima di ricevere il contenuto della pagina per poterla finalmente renderizzare. Questa modalità di rendering effettua il fetch delle risorse necessarie per creare il contenuto della pagina web ad ogni richiesta da parte del client e ha alcuni vantaggi, tra i quali la possibilità di visualizzare quasi istantaneamente le pagine web (al costo di un maggior carico di lavoro al server) e un miglioramento delle performance SEO.

- * **Static-Site Generation (SSG)**: consiste in un'applicazione software che crea pagine HTML da modelli o componenti e da una determinata fonte di contenuto. La generazione statica del sito è simile al server-side rendering con l'eccezione che si esegue il rendering delle pagine in fase di build anziché ad ogni richiesta da parte del client. Questo significa che le pagine dell'applicazione web vengono generate al momento della build e il contenuto del sito non cambia a meno che non vengano aggiunti nuovi contenuti o component e venga effettuato nuovamente la build. Un'ulteriore possibilità (implementabile ad esempio nel [framework Next.js](#)), consiste nell'effettuare una nuova build del sito dopo un determinato regolare intervallo di tempo, in modo da dare la possibilità al server di capire se sono presenti modifiche nelle risorse utilizzate per il contenuto delle pagine web.

React di base fornisce esclusivamente il Client-Side Redering, tuttavia è una libreria integrata dal [framework Next.js](#)⁶, il quale offre sia Server-Side Rendering che Static-Site Generation e una soluzione ibrida tra queste, come citato precedentemente.



Figura 5.3: Next.js Logo

Anche *Angular* fornisce di base esclusivamente il Client-Side Rendering, tuttavia è possibile integrare le tecnologie **Angular Universal**⁷ per permettere il Server-Side Rendering e **Scully**⁸ per offrire lo Static-Site Generation.



Figura 5.4: Angular Universal Logo

5.3.3 Rendering della UI

Parlando di performance di applicazioni web la dimensione non risulta un problema particolare, soprattutto quando si tratta di software di grandi dimensioni. Ciò che influisce maggiormente sulle prestazioni di runtime è la modalità di manipolazione del Document Object Model (DOM), ovvero la rappresentazione dell'ipertesto strutturata come modello orientato agli oggetti. Infatti la manipolazione del DOM risulta il cuore del web moderno e interattivo anche se, sfortunatamente, è anche molto più lento

⁶ *Next.js Website*. URL: <https://nextjs.org/>.

⁷ *Angular Universal Website*. URL: <https://angular.io/guide/universal>.

⁸ *Scully Website*. URL: <https://scully.io/>.

della maggior parte delle operazioni JavaScript. Per questo motivo è fondamentale analizzare come le due tecnologie implementano l'interazione con il DOM per valutarne le performance.

Angular utilizza e manipola direttamente il **DOM reale** e sfrutta il data-binding bidirezionale, quindi tutte le modifiche apportate al modello vengono replicate anche nella vista. Durante la traduzione di un'applicazione pesante, potrebbero quindi rallentare le prestazioni.

React invece utilizza il **Virtual DOM**, ovvero una copia più leggera del DOM realmente utilizzato per rendere dinamica l'applicazione. Per ogni oggetto DOM c'è un corrispondente oggetto DOM virtuale che corrisponde a una sua rappresentazione con le stesse proprietà, ma senza il potere di cambiare direttamente ciò che è presente a schermo. Mentre la manipolazione del DOM è lenta, quella del DOM virtuale è molto più veloce perché nulla viene renderizzato sullo schermo.

L'utilità del DOM virtuale consiste nel fatto che quando viene eseguito il rendering di un elemento JSX, ogni singolo oggetto DOM virtuale viene aggiornato molto rapidamente. Una volta che il DOM virtuale è stato aggiornato, React confronta il DOM virtuale con un'istantanea dello stesso che è stata scattata subito prima dell'aggiornamento. Effettuando questa operazione React scopre esattamente quali oggetti DOM virtuali sono cambiati (processo di Diffing). Questo permette a React di aggiornare solamente gli oggetti necessari nel DOM reale, senza effettuare inutili rendering di oggetti che sono rimasti inalterati.

Grazie a questa tipologia di innovazione nell'interazione con il DOM React può vantare di avere delle prestazioni nel rendering della user interface migliori rispetto ad Angular.

5.4 Conclusioni

Angular:

- * consiste in un [framework](#) ricco e completo, molto basato su Typescript;
- * è un [framework](#) opinionated, ovvero limita o guida in modo molto stringente nel modo di fare le cose, risultando uno svantaggio per gli sviluppatori che vogliono costruire soluzioni diverse da quelle standard ma un vantaggio in quanto limita molto la probabilità di errori nel codice;
- * possiede un'ottima documentazione dei suoi strumenti;
- * richiede un certo tempo per essere appreso in modo efficace in quanto è necessario studiare un intero ecosistema;
- * consente di dare una struttura solida al progetto sin dalla sua creazione e risulta adatto per applicazioni di grandi dimensioni senza un eccessivo quantitativo di aggiornamenti a schermo.

React:

- * consiste in una libreria per sviluppo di interfacce grafiche, molto basata su Javascript, quindi molto leggera ma dipendente da librerie terze per effettuare molte operazioni necessarie allo sviluppo web;

- * è unopinionated, quindi risulta molto flessibile e potente per sviluppatori esperti ma con probabilità più alta di commettere errori nel codice rispetto ad Angular;
- * può vantare una community più vasta ed attiva rispetto ad Angular;
- * richiede un tempo di apprendimento minore rispetto ad Angular rendendo disponibili meno strumenti da imparare;
- * consente di effettuare progetti in modo più libero e veloce, risultando particolarmente utile quindi per creare prototipi e molto appetibile per utenti principianti che possiedono una base di Javascript.

In conclusione all'analisi effettuata in questo documento è possibile affermare che nel dibattito Angular vs React non c'è una scelta migliore. Ogni soluzione ha i suoi vantaggi e svantaggi, si adatta a diverse tipologie di progetto e gruppi di lavoro. Personalmente durante lo sviluppo del progetto Voting-Online ho preferito usare Angular in quanto, una volta superata la fase iniziale di apprendimento, mi ha fornito tutti gli strumenti di cui avevo bisogno per creare l'applicazione web, mentre per effettuare le stesse operazioni con React mi sono trovato più volte a dover installare diverse librerie scritte da utenti della community che risultavano a volte limitate o comunque scarsamente documentate.

Capitolo 6

Conclusioni

6.1 Raggiungimento degli obiettivi

Gli [obiettivi prefissati](#) all'inizio dello stage risultano tutti soddisfatti. Di seguito viene illustrata la tabella riassuntiva.

Tabella 6.1: Tabella di riepilogo dello stato degli obiettivi

Obiettivo	Stato
O01	Soddisfatto
O02	Soddisfatto
O03	Soddisfatto
D01	Soddisfatto
F01	Soddisfatto

6.2 Analisi del lavoro svolto

A causa dell'emergenza sanitaria da Covid-19 lo stage è stato effettuato in gran parte da remoto, utilizzando gli appositi [strumenti organizzativi](#) senza riscontrare particolari problemi. Questa modalità di svolgimento del lavoro, anche se avrebbe potuto creare dei potenziali problemi logistici, mi ha permesso di guadagnare maggiore autonomia nell'apprendimento teorico e nell'organizzazione e svolgimento di attività.

Per quanto riguarda l'implementazione della piattaforma gli [strumenti di sviluppo](#) sono risultati più che sufficienti per portare a termine il lavoro. In particolare ho avuto la possibilità di approfondire l'IDE Visual Studio Code, il quale offre un'enorme quantità di estensioni utili per uno sviluppo software efficace ed efficiente, soprattutto riguardo all'auto-completamento del codice per Typescript.

Infine i prodotti attesi al termine del progetto Voting-Online, ovvero la piattaforma di e-Voting e il documento tecnico contenente il confronto tra Angular e React, sono risultati all'altezza delle aspettative da parte del capo progetto.

6.3 Valutazione personale

L'attività di stage mi ha permesso di apprendere nuove tecnologie e di approfondirne altre già conosciute. Sono rimasto molto soddisfatto dalla velocità di apprendimento dei

framework Angular e Spring, i quali mi hanno permesso di apprezzare particolarmente le conoscenze acquisite in vari corsi di studio durante il periodo universitario. Infatti le conoscenze riguardo la programmazione ad oggetti, derivanti dal linguaggio C++ imparato durante il corso di *Programmazione ad Oggetti* e Java imparato durante il corso di *Altri paradigmi di programmazione*, mi hanno permesso un rapido apprendimento di Spring, mentre le conoscenze derivanti dal corso di *Tecnologie Web* mi hanno permesso una buona realizzazione delle maschere **FE** utilizzando Angular. Un'ulteriore merito va dato inoltre al corso di *Ingegneria del Software* il quale, grazie al progetto didattico, mi ha dato le competenze necessarie al lavoro collaborativo in un team di sviluppo e alla stesura del documento tecnico.

In conclusione l'attività di stage presso l'azienda Sync Lab è risultata molto stimolante e costruttiva, permettendomi una crescita personale importante grazie a questo primo assaggio del mondo del lavoro, che risulta molto diverso rispetto a quello accademico. Questa opportunità mi ha permesso di conoscere meglio le mie capacità nell'ambito lavorativo, permettendomi un miglioramento nella gestione delle tempistiche e nella capacità di lavorare in gruppo.

Glossario

API in informatica con il termine *Application Programming Interface API* (ing. interfaccia di programmazione di un'applicazione) si indica ogni insieme di procedure disponibili al programmatore, di solito raggruppate a formare un set di strumenti specifici per l'espletamento di un determinato compito all'interno di un certo programma. La finalità è ottenere un'astrazione, di solito tra l'hardware e il programmatore o tra software a basso e quello ad alto livello semplificando così il lavoro di programmazione. [63](#)

BE in informatica con il termine *Backend* si intende l'insieme delle applicazioni e dei programmi con cui l'utente non interagisce direttamente ma che sono essenziali al funzionamento del sistema. [63](#)

Cookie Gli HTTP cookie sono un tipo particolare di magic cookie (una sorta di gettone identificativo) e vengono utilizzati dalle applicazioni web lato server per archiviare e recuperare informazioni a lungo termine sul lato client. [42](#)

DBMS In informatica, un *Database Management System* (abbreviato in DBMS o Sistema di gestione di basi di dati) è un sistema software progettato per consentire la creazione, la manipolazione e l'interrogazione efficiente di database.. [63](#)

Endpoint Una comunicazione *endpoint* è un nodo di comunicazione rilevabile la cui portata può essere variata per restringere o ampliare la zona di ricerca. Gli endpoint favoriscono uno strato di astrazione programmabile per cui sistemi software e/o sottosistemi possono comunicare tra di loro, inoltre i mezzi di comunicazione sono disaccoppiati dai sottosistemi di comunicazione. [37](#)

ER In informatica, nell'ambito della progettazione dei database, il modello entità-relazione (o modello entità-associazione; più comune modello E-R) è un modello teorico per la rappresentazione concettuale e grafica dei dati a un alto livello di astrazione. [2](#), [61](#)

framework Un framework, termine della lingua inglese che può essere tradotto come struttura o quadro strutturale, in informatica e specificamente nello sviluppo software, è un'architettura logica di supporto (spesso un'implementazione logica di un particolare design pattern) sulla quale un software può essere progettato e realizzato, spesso facilitandone lo sviluppo da parte del programmatore. [iii](#), [2](#), [6](#), [33](#), [35–38](#), [40–42](#), [51](#), [52](#), [54](#), [56](#), [57](#), [60](#)

FE in informatica con il termine *Frontend* si intende la parte visibile all'utente di un programma e con cui egli può interagire, tipicamente un'interfaccia utente. [63](#)

GUI L'interfaccia grafica, nota anche come GUI (dall'inglese Graphical User Interface), in informatica è un tipo di interfaccia utente che consente l'interazione uomo-macchina in modo visuale utilizzando rappresentazioni grafiche (es. widget) piuttosto che utilizzando i comandi tipici di un'interfaccia a riga di comando. [63](#)

IoC L'*Inversion of Control* è un principio architetturale basato sul concetto di invertire il controllo del flusso di sistema rispetto alla programmazione tradizionale. Nella programmazione tradizionale la logica di tale flusso è definita esplicitamente dallo sviluppatore, che si occupa tra le altre cose di tutte le operazioni di creazione, inizializzazione ed invocazione dei metodi degli oggetti. L'IoC invece inverte il controllo. [36](#), [63](#)

licenza MIT La Licenza MIT (MIT License in inglese) è una licenza di software libero creata dal Massachusetts Institute of Technology (MIT). [51](#)

Local Storage Il *Local Storage* fornisce accesso all'interfaccia del W3C Web Storage. Quest'ultima fornisce alle applicazioni web metodi e protocolli per l'archiviazione dei dati lato client. L'archiviazione Web supporta l'archiviazione dei dati persistenti, simile ai cookie ma con una capacità notevolmente migliorata. [40](#)

open source Con open source (in italiano sorgente aperto), in informatica, si indica un tipo di software o il suo modello di sviluppo o distribuzione. Un software open source è reso tale per mezzo di una licenza attraverso cui i detentori dei diritti favoriscono la modifica, lo studio, l'utilizzo e la redistribuzione del codice sorgente. [36](#), [39](#), [51](#), [52](#)

REST *Representational state transfer* (REST) è uno stile architetturale (di architettura software) per i sistemi distribuiti. L'architettura REST si basa su HTTP. Il funzionamento prevede una struttura degli URL ben definita che identifica univocamente una risorsa o un insieme di risorse e l'utilizzo dei metodi HTTP specifici per il recupero di informazioni (GET), per la modifica (POST, PUT, PATCH, DELETE) e per altri scopi (OPTIONS, ecc.). [63](#)

UML in ingegneria del software *UML, Unified Modeling Language* (ing. linguaggio di modellazione unificato) è un linguaggio di modellazione e specifica basato sul paradigma object-oriented. L'*UML* svolge un'importantissima funzione di "lingua franca" nella comunità della progettazione e programmazione a oggetti. Gran parte della letteratura di settore usa tale linguaggio per descrivere soluzioni analitiche e progettuali in modo sintetico e comprensibile a un vasto pubblico. [2](#), [9](#), [63](#)

Acronimi

API [Application Program Interface](#). 55, 61

BE [Backend](#). [iii](#), 2, 4, 35, 37, 40–43, 61

DBMS [Database Management System](#). 37, 61

FE [Frontend](#). [iii](#), 2, 4, 33, 35, 37, 40, 42–44, 60, 61

GUI [Graphical User Interface](#). 40, 42, 44, 62

IoC [Inversion of Control](#). 62

REST [Representational State Transfer](#). 35, 37, 62

UML [Unified Modeling Language](#). 62

Bibliografia

Riferimenti bibliografici

Gamma E. Vlissides J., Johnson R. *Design Patterns: Elements of Reusable Object-Oriented Software*. 1997 (cit. alle pp. 38, 41, 42).

Siti web consultati

Angular Best Practices. URL: <https://angular.io/guide/styleguide> (cit. a p. 40).

Angular CLI. URL: <https://angular.io/cli> (cit. alle pp. 40, 53).

Angular Universal Website. URL: <https://angular.io/guide/universal> (cit. a p. 56).

Angular Website. URL: <https://angular.io/> (cit. a p. 51).

Apache Maven Website. URL: <https://maven.apache.org/> (cit. a p. 35).

Casi D'uso. URL: https://www.math.unipd.it/~rcardin/swea/2021/Diagrammi%20Use%20Case_4x4.pdf (cit. a p. 9).

Directive Angular. URL: <https://angular.io/guide/attribute-directives> (cit. a p. 53).

Next.js Tutorial. URL: <https://nextjs.org/docs/getting-started> (cit. a p. 42).

Next.js Website. URL: <https://nextjs.org/> (cit. a p. 56).

Node Package Manager. URL: <https://www.npmjs.com/> (cit. alle pp. 42, 53).

React Best Practices. URL: <https://reactjs.org/docs/thinking-in-react.html> (cit. a p. 44).

React Website. URL: <https://it.reactjs.org/> (cit. a p. 52).

Scully Website. URL: <https://scully.io/> (cit. a p. 56).

Spring Data JPA. URL: <https://spring.io/projects/spring-data-jpa> (cit. a p. 37).

Spring Data REST. URL: <https://spring.io/projects/spring-data-rest> (cit. a p. 37).

Spring Framework Website. URL: <https://spring.io/projects/spring-framework> (cit. a p. 36).

Spring Initializr Website. URL: <https://start.spring.io/> (cit. a p. 37).

SpringBoot. URL: <https://spring.io/projects/spring-boot> (cit. a p. 37).