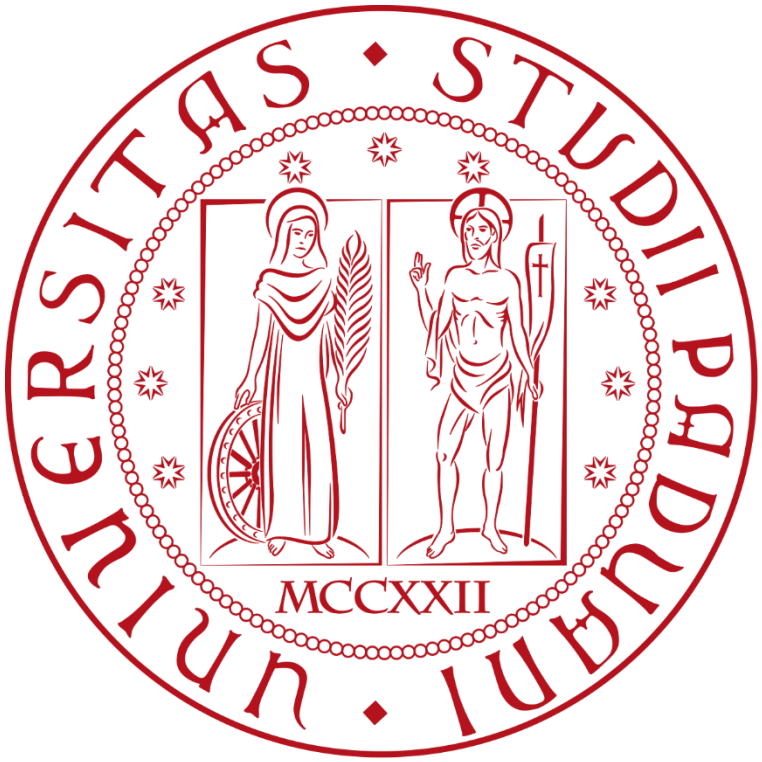
Università degli studi di Padova

Dipartimento di Matematica “Tullio Levi-Civita”

Corso di laurea in Informatica



ENGaming: la console di gaming a portata di tablet

Tesi di Laurea

Relatore:
Prof Claudio Enrico Palazzi

Laureando:
Gabriele Saracco

Sommario:

La tesi andrà ad analizzare il lavoro svolto durante lo stage nell'azienda Euronovate nello sviluppo dell'applicazione ENGaming per i tablet ESign 11 prodotti dall'azienda. Quest'applicazione, che è stata sviluppata usando il framework ElectronJS unito ad Angular, con linguaggio HTML, CSS e Javascript per permettere l'utilizzo di un browser integrato, affiancati al C++, utilizzato per la gestione del driver del tablet ed a Google Firebase, per la gestione in remoto dei record effettuati nei giochi, funzionerà come piccola stazione di Gaming a portata di tablet. In questo modo gli ESign 11, e successivi, prodotti da Euronovate inizieranno ad ampliare le loro funzionalità oltre a quella da schermo per firme digitali, ed inizieranno ad essere utilizzati in più occasioni per aumentare il proprio valore.

Indice

1. Introduzione:

In questa introduzione sarà presentata l'azienda in cui è stato svolto lo stage, il device utilizzato, l'ENSign 11 e i vari strumenti utilizzati.

1.1 Euronovate:

Euronovate Group è un'azienda multinazionale con oltre 150 dipendenti, leader nel mercato dell'implementazione e commercializzazione di soluzioni innovative per la trasformazione digitale, Certification Authority eIDAS compliant, e produttrice di oltre 50 prodotti proprietari Hardware e Software. Ha sede centrale a Mendrisio (Svizzera) e controllate con presenza diretta in quattro paesi:

- Italia (Padova, Reggio Emilia, Milano);
- Spagna (Barcellona, Madrid, Bilbao);
- Romania (Bucarest);
- Cina (Shanghai).

Euronovate Group è inoltre suddivisa in:

- Euronovate SA;
- eSignWorld;
- Víntegris.

1.1.1 Euronovate SA:

Fondata nel 2012 e con sede a Lugano (CH), è una società svizzera innovativa, leader in soluzioni di Digital Trasformation con approccio end-to-end, combinando soluzioni software, hardware e servizi di consulenza. L'obiettivo principale è aiutare ogni tipo di azienda ad eliminare tutti i processi e i documenti cartacei passando completamente al digitale, garantendo la stessa validità legale.

1.1.2 eSignWorld:

Società che opera nel settore della consulenza IT, processi e sistemi avanzati di firme elettroniche, fornitura, esercizio e manutenzione di sistemi informativi hardware e software, specializzata nel campo della produzione a filiera corta di tecnologia grafometrica. In particolare, eSignWorld fornisce soluzioni personalizzate e proprietarie nel campo della dematerializzazione documentale e dell'Information Communication Technology, garantendo la possibilità di visualizzare, elaborare e firmare elettronicamente qualsiasi tipo di documento. eSignworld offre soluzioni paperless con utilizzo di firma elettronica semplice e firma elettronica avanzata, un sistema di composizione e successiva conservazione di documenti in formato elettronico, nonché di firma dei documenti, attraverso un innovativo dispositivo di firma.

1.1.3 Víntegris:

Víntegris progetta, implementa e gestisce infrastrutture di sicurezza per istituzioni finanziarie e grandi società. Fornisce solide soluzioni su misura per ogni esigenza di business, integrando tecnologie ad alte prestazioni e soddisfacendo le esigenze di ogni azienda. La società definisce e realizza progetti ad alto valore aggiunto per la protezione delle informazioni, la sicurezza web, il controllo e la gestione degli accessi. Vengono utilizzate tecnologie tra le più robuste sul mercato (Docker, Kubernetes, AWS, HSM), integrandole nell'ambiente aziendale di ciascun cliente. L'essere consulenti ed integratori pone Víntegris in una posizione privilegiata nello sviluppo di nuovi prodotti: la conoscenza delle esigenze e delle criticità delle grandi aziende in materia di sicurezza delle informazioni, guida Víntegris verso la progettazione di prodotti che coprono il divario tra le soluzioni di sicurezza dei grandi produttori e le reali esigenze aziendali. In questo modo, l'azienda è in grado di anticipare le esigenze di mercato in segmenti critici come la gestione, il controllo e gli audit di certificati digitali e autenticazione dell'utente. Tutti i consulenti di Víntegris hanno una vasta esperienza nel campo della tecnologia dell'informazione e sono esperti nella progettazione, implementazione e gestione d'infrastrutture

di sicurezza delle informazioni. Inoltre, la maggior parte dei consulenti è in possesso di certificazioni riconosciute a livello internazionale, come CISA, CISSP e CISM, che garantiscono la loro esperienza e conoscenza nell'ambito della sicurezza delle informazioni.

1.2 ENSign 11:

L'ENSign 11 è un tablet versatile per firme grafometriche, con le sue differenti applicazioni, è lo strumento perfetto per la digitalizzazione delle firme a mano. L'ENSign è dotato delle seguenti caratteristiche:

- Pannello multi-tocco con isolamento nativo dello schermo;
- Connessione diretta ad un computer;
- Altamente compatto, con grande stabilità, alti livelli di sicurezza e design elegante;
- Cattura di parametri biometrici come pressione, accelerazione, velocità, ritmo e movimento in aria;
- Sistema proprietario di criptazione per ogni tipo di transazione.

Inoltre, la versatilità dell'ENSign 11 permette di trasformarlo in uno schermo aggiuntivo, semplicemente collegandolo tramite USB ad un computer e installando l'apposita applicazione ENSIGN 11 trayApp, in modo da avere tutte le funzionalità multimediali. Con lo schermo multitouch di ENSign 11 è possibile creare, scrivere, disegnare e mostrare contenuti multimediali personalizzati in tempo reale, che possono essere automaticamente condivisi attraverso videoconferenze o in presentazioni d'affari.

1.3 Strumenti di comunicazione:

Per mantenere una comunicazione tra me ed il tutor aziendale, durante il periodo di stage, ho utilizzato, a seconda dell'esigenza, i seguenti sistemi di comunicazione:

- Gmail: Per l'invio e scambio di informazioni o documenti formali ed importanti.
- Telegram: Per le comunicazioni informali e più veloci.
- Google Meet: Per svolgere sessioni di "Live coding" e riunioni riguardanti l'avanzamento del progetto o gli allineamenti con le tabelle di marcia e scadenze.

1.4 Strumenti di sviluppo:

Sistema operativo: Non essendoci richieste o imposizioni da parte dell'azienda su uno specifico sistema operativo da utilizzare durante lo sviluppo del progetto ho scelto di utilizzare Microsoft Windows 10, dal momento che trovo sia il sistema più stabile e funzionante presente nel mercato, in questo momento, oltre ad essere quello con cui abbia più familiarità e conoscenze al riguardo avendolo utilizzato dalla sua uscita.

Ambiente di sviluppo integrato: Anche per l'ambiente di sviluppo integrato non erano presenti obblighi o restrizioni da parte dell'azienda e mi è stata lasciata libera scelta, ed anche in questo caso la mia scelta è stata

influenzata da esperienza e stabilità delle opzioni, scelta che è ricaduta su Visual Studio Code. Vs Code, sviluppato da Microsoft per Windows, è un ottimo editor gratuito, che ho utilizzato maggiormente durante la mia carriera e che, Inoltre, grazie alle sue innumerevoli estensioni ed integrazioni, permette di adattarsi e rendere facile il proprio utilizzo da qualsiasi utente.

Database in remoto per il salvataggio dei record: Come metodo di salvataggio e visualizzazione dei record in un database remoto, e non in locale, l'azienda offriva delle opzioni tra cui scegliere, come un apposito database da loro hostato o Firebase. La mia scelta è ricaduta proprio su Firebase in quanto si tratta di una piattaforma di Google per il web app development che offre anche una funzione di database in tempo reale su cui leggere e salvare dei dati. Ho selezionato questo strumento perchè, oltre ad essere gratuito è anche di facile utilizzo ed implementazione con dell'ottima documentazione ufficiale presente online, oltre ad essere già utilizzato dall'azienda in altri progetti.

Sviluppo documenti del progetto: Per la stesura e modifica dei documenti, sempre non essendoci restrizioni o obblighi da parte dell'azienda, purché il formato finale fosse PDF, ho scelto di utilizzare Microsoft Word dal momento che lo conosco approfonditamente e che permette di scrivere documenti in linea con le mie necessità.

1.5 Organizzazione del documento:

Questo documento è organizzato nel seguente modo:

- **Primo capitolo:** Introduzione generale sull'azienda, sul dispositivo utilizzato (ENSign 11) e sugli strumenti con cui andrò a svolgere lo stage
- **Secondo capitolo:** Descrizione ed approfondimento sullo stage ed il suo svolgimento
- **Terzo capitolo:** Elenco i casi d'uso che saranno presenti in ENGaming ed i vari requisiti che dovrà soddisfare
- **Quarto capitolo:** Presentazione della struttura e composizione del prodotto finale di ENGaming, le caratteristiche di questo che soddisfano vari requisiti e in dettaglio come esso sia composto, in esso sono anche elencate tutte le tecnologie e classi utilizzate, in aggiunta alla spiegazione e motivazioni che hanno portato alla loro scelta
- **Quinto capitolo:** Nel quinto capitolo viene spiegato come è stato validato e verificato il raggiungimento degli obiettivi ed i requisiti da parte di ENGaming
- **Sesto capitolo:** Conclude la tesi dando una visione generale del progetto attraverso anche un mio parere personale sul cosa io abbia appreso e su quanto mi abbia lasciato quest'esperienza di stage.

2. Descrizione dello stage.

In questo capitolo spiegherò nel dettaglio come è stato strutturato lo stage ed a descrivere i vari periodi di esso.

2.1 Introduzione al progetto

L'azienda Euronovate è specializzata ed affermata nel mercato dei device di firma digitale grazie ai propri monitor da 10 pollici multitouch. Essi però, facilmente collegabili ad un computer tramite presa USB, possono essere utilizzati anche in altri campi, oltre a quello delle firme digitali, specialmente grazie alla loro funzione di multitouch. Questo progetto infatti apre le porte all'azienda, ed al loro ultimo tablet ENSign 11, al mondo dell'intrattenimento videoludico.

2.1.1 Requisiti e obiettivi

L'obiettivo dello stage, e del progetto, è quello di imparare a sviluppare applicazioni desktop e di gestire gli input ricevuti dai dispositivi a cui il computer è collegato. Inoltre, mi ha anche permesso di imparare ad operare ed a gestire il mio lavoro in base settimanale.

Alla fine dello stage avrò quindi prodotto un'applicazione software passando per tutte le fasi necessarie ad un suo corretto sviluppo quindi nello specifico: analisi, progettazione, codifica e testing. Il prodotto finale sarà funzionante su un ambiente desktop con collegato il device ENSign 11.

Le funzionalità richieste che andrà a sviluppare e completare nello stage saranno quindi:

- Visualizzazione, avvio e caricamento di giochi.
- Possibilità di giocare ai giochi tramite un controller virtuale che sarà presente a schermo ed adattato al tipo di gioco scelto.
- Salvataggio e visione dei punteggi effettuati dagli utenti in un database in remoto.

2.2 Pianificazione

2.2.1 Fasi dello stage

Lo stage è stato diviso, confrontandosi con il tutor aziendale, in diverse fasi di diversa durata, a seconda della complessità e delle risorse richieste per ogni fase. Queste sono le seguenti:

- Conoscenze generali: dal 10/07 al 13/07
- Formazione personale: dal 14/07 al 27/07
- Analisi dei requisiti: dal 28/07 al 02/08
- Progettazione tecnica: dal 03/08 al 11/08
- Codifica: dal 28/08 al 28/09
- Documentazione: dal 29/09 al 02/10
- Demo: dal 03/10 al 03/10

2.2.2 Conoscenze generali:

In questa prima fase dello stage mi sono focalizzato sull'installazione degli ambienti di sviluppo e di versionamento che saranno necessari per il tirocinio oltre ad iniziare a familiarizzare con essi. Infine, c'è stata, da parte dell'azienda, l'abilitazione dei loro strumenti aziendali che sarei andato ad utilizzare (account Git, indirizzo e-mail, Google drive ed AWS).

2.2.3 Formazione personale:

L'obiettivo principale di questa fase era, da parte mia, di familiarizzare, esercitarmi ed abituarmi all'utilizzo delle tecnologie sfruttate nello stage. In particolare ho sviluppato personalmente un'applicazione di test con ElectronJs per capire al meglio il suo funzionamento e risolvere eventuali problemi di configurazioni che mi potevano sfuggire. L'applicazione consisteva di semplici funzionalità come le tecniche di comunicazione per il passaggio di dati tra C++ e Typescript, oltre all'utilizzo del framework Angular per la creazione dell'UI dell'applicazione.

Infine, essendo il secondo studente a lavorare su ENGaming dopo aver familiarizzato con le tecnologie ho speso del tempo per vedere la situazione del progetto lasciata dal mio collega. In questo modo ho potuto vedere il codice già scritto e prendere nota degli eventuali problemi presenti nell'applicazione, oltre a quali requisiti fossero stati completati e quali fossero da perfezionare, così da sviluppare un'analisi dei requisiti e progettazione tecnica adeguati.

2.2.4 Analisi dei requisti:

Durante la fase di Analisi dei requisiti ho definito tutti i casi d'uso che l'applicativo dovrà prevedere, oltre alla tipologia di utenti che lo andrà ad utilizzare. Nell'analisi, oltre ai casi d'uso, ho anche inserito tutti i requisiti che l'applicazione dovrà avere (requisiti obbligatori), i requisiti aggiuntivi da rispettare nel caso mi avanzasse del tempo nel periodo di codifica, ma che non sono necessari al completamento del progetto (requisiti opzionali). Questa fase getta le basi per lo sviluppo del prodotto dal momento che contiene al suo interno la descrizione del comportamento in tutti i vari casi che si possono presentare; quindi, proprio per questo motivo ha dovuto ricevere l'approvazione del tutor una volta completata.

2.2.5 Progettazione tecnica:

Nel periodo della progettazione tecnica ho individuato e definito tutte le componenti e classi che sarebbero andate a comporre il prodotto finale. In particolare ho individuato: l'architettura che sarà utilizzata, le varie classi utilizzate, i loro collegamenti ed i vari dati che verranno tra loro scambiati.

2.2.6 Codifica:

Nella fase di codifica ho aggiornato l'applicativo lasciato dal mio collega adattandolo a ciò che avevo progettato nelle fasi precedenti. In questa fase, la più lunga dello stage, ho dovuto lavorare seguendo le norme di codifica aziendali. Ogni qualvolta che avevo completato delle parti di prodotto mi sono accertato che esse funzionassero come desiderato tramite dei test manuali, essendo i vari casi limitati e semplici da riprodurre.

2.2.7 Documentazione:

Nella fase di documentazione ho steso i documenti necessari all'utilizzo e manutenzione del prodotto, questi sono il manuale dello sviluppatore ed il manuale utente:

- **Manuale dello sviluppatore:** Questo manuale, dedicato a chi in futuro manterrà e lavorerà all'applicazione contiene tutte le informazioni relative all'ambiente di sviluppo, i possibili miglioramenti, problemi noti da risolvere e qualsiasi indicazione o accorgimento da sapere.
- **Manuale utente:** Questo manuale contiene tutte le informazioni e requisiti necessari per far funzionare l'applicazione, oltre a tutte le sue funzionalità ed istruzioni.

2.2.8 Demo:

Nella fase finale di demo ho mostrato l'applicazione che ho realizzato all'azienda, in modo che potessero vederlo e valutarlo, oltre ad analizzare come possano integrarlo nei propri prodotti.

2.3 Supervisione e controllo:

Nelle prime fasi di progetto, quelle di preparazione, mi sono confrontato con il tutor aziendale solo al termine di ognuna delle attività da lui assegnate, per poter discutere dei miei possibili dubbi o difficoltà riscontrate.

Nelle fasi successive, invece, ho comunicato con lui in maniera giornaliera aggiornandolo sul mio lavoro e sulla situazione del prodotto; al termine di ogni settimana erano previsti degli incontri per poter definire le attività dello sprint successivo.

3. Analisi dei requisiti

3.1 Casi d'uso

In questo capitolo elencherò e descriverò tutti i casi d'uso che ho individuato per il progetto di ENGaming. Ogni caso d'uso sarà diviso in più sezioni per la sua descrizione e spiegazione, esse sono:

- **Descrizione:** Questa sezione spiega nel dettaglio il caso d'uso che si sta analizzando
- **Scenario principale:** Questa sezione descrive lo scenario principale in cui il caso d'uso viene applicato
- **Precondizione:** Questa sezione descrive la condizione dell'applicazione precedentemente all'avvenire del caso d'uso
- **Postcondizione:** Questa sezione descrive la condizione dell'applicazione una volta terminato il caso d'uso

UC 1 - Selezione di un videogioco

- **Descrizione:** Cliccato sull'immagine di uno dei videogiochi presenti nel menù questo viene avviato mostrando la propria pagina introduttiva, con comandi, pulsante di avvio e di chiusura
- **Scenario principale:** Il cliente clicca su una delle immagini o gif a schermo di ENGaming ed il corrispettivo videogioco viene avviato
- **Precondizioni:** Il cliente si trova nel menù principale di ENGaming
- **Postcondizioni:** Il gioco selezionato viene avviato e mostrato a schermo

UC 2 - Visualizzazione delle classifiche

- **Descrizione:** Cliccato il pulsante per aprire le classifiche, nel menù principale, la sezione "Leaderboard" dell'app viene mostrata a schermo
- **Scenario principale:** Il cliente clicca sul pulsante per aprire le classifiche che verranno poi visualizzate
- **Precondizioni:** Il cliente si trova nel menù principale di ENGaming
- **Postcondizioni:** La sezione delle classifiche dell'app viene visualizzata a schermo

UC 3 - Cambio della classifica visualizzata

- **Descrizione:** Nella pagina delle classifiche (vedi UC 2) premendo sulle frecce, poste a destra e sinistra del nome del gioco della classifica che si sta visualizzando, ci verrà mostrata quella del gioco precedente o successivo
- **Scenario principale:** Il cliente preme sulle frecce poste in alto nella pagina delle classifiche e gli viene mostrata quella del gioco precedente o successivo
- **Precondizioni:** Il cliente si trova nella pagina delle classifiche
- **Postcondizioni:** A schermo compare la classifica con i punteggi di un altro gioco presente in ENGaming

UC 4 - Avvio di un gioco

- **Descrizione:** Dalla pagina introduttiva di un gioco (vedi UC 1) premendo sul pulsante di avvio del gioco, questo verrà avviato, mostrando a schermo il corrispettivo controller con i comandi (se presente)
- **Scenario principale:** Il cliente, una volta scelto il videogioco che desidera giocare, dalla rispettiva pagina introduttiva potrà avviare il gioco premendo sul corrispettivo pulsante
- **Precondizioni:** Il cliente ha precedentemente selezionato un gioco dal menù principale
- **Postcondizioni:** Viene avviata una partita del gioco, mostrando a schermo l'eventuale controller di questo

UC 5 - Uscita dal videogioco dal menù principale

- **Descrizione:** Dalla pagina introduttiva di un gioco (vedi UC 1) premendo sul pulsante “Esci” si verrà riportati nel menù di ENGaming chiudendo il processo del videogioco
- **Scenario principale:** Il cliente che desidera chiudere il gioco appena aperto e provarne un altro cliccherà su “Esci” e sarà reindirizzato al menù principale
- **PrecondizionI:** L’utente si deve trovare nella pagina introduttiva selezionato un gioco
- **Postcondizioni:** Il cliente una volta premuto “Esci” sarà riportato al menù principale di ENGaming

UC 6 - Chiusura di ENGaming

- **Descrizione:** Dal menù principale di ENGaming premendo sul pulsante “Power” l’applicazione sarà chiusa ed il tablet tornerà ad avere la funzione di schermo aggiuntivo
- **Scenario principale:** Il cliente che avrà finito di giocare potrà chiudere l’applicazione premendo sul pulsante “Power”, il quale terminerà il processo di ENGaming
- **Precondizioni:** Il cliente dovrà trovarsi nel menù principale di ENGaming
- **Postcondizioni:** Il processo di ENGaming sarà terminato ed il tablet tornerà ad essere uno schermo

UC 7 - Usare il controller di gioco

- **Descrizione:** Iniziata la partita ad un gioco che richiede l’utilizzo del controller (vedi UC 4) l’utente potrà premere sui pulsanti dello specifico controller implementato per il gioco, visibili a schermo, per comandare il personaggio che sta utilizzando
- **Scenario principale:** Il cliente, avviata una partita, comanderà il proprio personaggio all’interno del gioco premendo sui pulsanti del controller a schermo per inviare i propri input al gioco
- **Precondizione:** Il cliente deve aver avviato la partita ad un gioco che richiede l’utilizzo del controller
- **Postcondizione:** Gli input premuti dal cliente saranno registrati ed inviati al gioco sottoforma di comandi

UC 8 - Usare il pennino dell’ENSign 11 come controller

- **Descrizione:** Iniziata la partita ad un gioco che riceverà gli input a schermo dal pennino dell’ENSign 11 (vedi UC 4), premendo e muovendo questo sullo schermo i relativi movimenti saranno registrati come input
- **Scenario principale:** Il cliente, avviata una partita, premerà o muoverà, il pennino a schermo per registrare i propri comandi da inviare come input al gioco
- **Precondizione:** Il cliente deve aver avviato una partita ad un gioco che richiede l’utilizzo del pennino
- **Postcondizione:** I movimenti del pennino saranno registrati ed inviati al gioco come input

UC 9 - Mettere in pausa la propria partita

- **Descrizione:** Durante una partita ad un videogioco lo si può mettere in pausa premendo sull’apposito pulsante, visibile a schermo, per riavviarlo o tornare al menù principale di ENGaming
- **Scenario principale:** Il cliente che vorrà riavviare la propria partita ad un gioco, o uscire da esso per per tornare al menù principale, premerà sul pulsante di pausa a schermo che gli aprirà il menù di pausa
- **Precondizione:** Il cliente dovrà aver avviato la partita ad un videogioco
- **Postcondizione:** A schermo sarà mostrato il menù di pausa per riavviare o uscire dal gioco

UC 10 - Riavviare la propria partita

- **Descrizione:** Dal menù di pausa (vedi UC 9) si può decidere di riavviare la propria partita al videogioco che si sta giocando premendo sull'apposito pulsante "Riavvia"
- **Scenario principale:** Il cliente, insoddisfatto dei propri risultati in una partita, potrà, dopo aver messo in pausa il gioco, riavviare il gioco cliccando sul pulsante "Riavvia" che riavvierà il gioco a cui stava giocando
- **Precondizione:** Il cliente deve essere nel menù di pausa durante una partita
- **Postcondizione:** Il gioco sarà riavviato ed a schermo comparirà il corrispettivo menù principale

UC 11 - Uscita dal videogioco dal menù di pausa

- **Descrizione:** Dal menù di pausa (vedi UC 9) si può premere sul pulsante "Esci dal gioco" per chiudere il videogioco ed essere riportati nel menù principale di ENGaming
- **Scenario principale:** Il cliente, una volta che vorrà uscire dal gioco a cui sta giocando, dal menù di pausa premerà sul pulsante "Esci dal gioco" che chiuderà il gioco e visualizzerà sullo schermo il menù principale di ENGaming
- **Precondizione:** Il cliente deve aver messo in pausa il videogioco a cui stava giocando
- **Postcondizione:** A schermo sarà visualizzato il menù principale di ENGaming

UC 12 - Salvare il proprio punteggio

- **Descrizione:** Quando si uscirà da un gioco attraverso il menù di pausa (vedi UC 11), e si sarà tornati nel menù principale di ENGaming, verrà visualizzata una schermata che chiederà all'utente se vorrà o meno salvare il proprio punteggio nelle classifiche, premendo sul pulsante "Salva il punteggio" verrà richiesto l'inserimento del proprio nominativo in formato arcade (AAA), ed una volta fatto ciò il nuovo punteggio, collegato al nominativo, sarà salvato nella classifiche del gioco nel server
- **Scenario principale:** Una volta che il cliente sarà uscito da un gioco potrà salvare il proprio punteggio nelle classifiche premendo sul pulsante "Salva il punteggio", una nuova pagina, sarà quindi mostrata dove dovrà inserire il proprio nome in formato arcade, ed una volta confermato il nominativo il punteggio sarà inserito nella classifica mentre al cliente verrà mostrata il menù principale di ENGaming
- **Precondizione:** Il cliente dovrà uscire da una partita tramite il menù di pausa
- **Postcondizione:** Il punteggio ed il nominativo saranno salvati sulla classifica del videogioco

UC 13 - Visualizzazione degli ENSign 11 collegati alla rete

- **Descrizione:** Dal menù principale di ENGaming ci dev'essere un pulsante che mostrerà una nuova schermata con tutti gli ENSign 11 attualmente connessi alla stessa rete ed il loro nominativo
- **Scenario principale:** Il cliente, che vorrà sfidare un altro utente ad una partita multigiocatore, premendo un pulsante nel menù principale di ENGaming vedrà una lista di tutti gli altri tablet collegati alla stessa rete con cui può interagire
- **Precondizione:** Il cliente deve essere nella pagina principale di ENGaming
- **Poscondizione:** Al cliente viene mostrata una nuova schermata con una lista di tutti gli altri dispositivi ENSign collegati alla stessa rete

UC 14 - Inviare una sfida multigiocatore

- **Definizione:** Dalla lista degli ENSign collegati alla rete (vedi UC 13) si potrà cliccare su uno di questi per poter inviare all'utente che lo sta utilizzando una sfida multigiocatore ad un gioco
- **Scenario principale:** Il cliente che vorrà sfidare un altro utente ad una partita clicca sul nome dell'ENSign che vuole sfidare e, grazie ad una nuova schermata che comparirà a schermo, sceglierà a quale gioco sfidarlo, fatto ciò, premerà sul pulsante conferma per inviare la sfida
- **Precondizione:** Il cliente si trova nella schermata con la lista di tutti gli ENSign collegati alla rete
- **Postcondizione:** Una richiesta di sfida viene inviata all'ENSign selezionato

UC 15 - Accettare una sfida multigiocatore

- **Definizione:** Ricevuta una richiesta di sfida da un altro tablet sarà possibile accettarla premendo sul pulsante "Accetta" a schermo, a quel punto ad entrambi gli ENSign verrà caricato il gioco ed avviata la partita
- **Scenario principale:** Il cliente che riceve la notifica della sfida premerà sul pulsante accetta per accettarla e nel suo schermo, come a quello dell'avversario che ha inviato la sfida, si avvierà il gioco e la partita a cui si sfideranno
- **Precondizione:** Un altro cliente deve aver inviato una richiesta di sfida al nostro ENSign
- **Postcondizione:** La sfida viene accettata ed ad entrambi gli ENSign viene avviato il gioco e la partita

UC 16 - Rifiutare una sfida multigiocatore

- **Definizione:** Ricevuta la notifica di una sfida multigiocatore sarà possibile rifiutarla premendo sull'apposito pulsante, a quel punto all'ENSign che aveva inviato la richiesta sarà inviato un messaggio di errore
- **Scenario principale:** Il cliente che viene sfidato non vuole partecipare alla sfida multigiocatore; quindi premerà sul pulsante "Rifiuta" che gli comparirà a schermo, fatto ciò un messaggio di errore "Sfida rifiutata" sarà inviato al tablet che aveva inviato la sfida il quale gli comunicherà che non è stata accettata.
- **Precondizione:** Un altro cliente deve aver inviato una richiesta di sfida al nostro ENSign
- **Postcondizione:** La sfida viene rifiutata ed un messaggio di errore viene inviato all'ENSign che aveva effettuato la richiesta

UC 17 - Modificare le impostazioni di ENGaming

- **Definizione:** Da uno specifico pulsante presente nella home di ENGaming, chiamato "impostazioni" sarà possibile modificare le impostazioni di questo, come il volume dei giochi, ed applicare le modifiche
- **Scenario principale:** Il cliente che vorrà modificare le impostazioni di ENGaming premerà sul pulsante "Impostazioni" presente nel menù principale da cui si aprirà una nuova pagina contenente tutte le impostazioni modificabili nell'applicazione
- **Precondizione:** Il cliente deve trovarsi nel menù principale di ENGaming
- **Postcondizione:** Le modifiche apportate ai parametri presenti nelle impostazioni saranno salvate ed applicate ad ENGaming

3.2 Requisiti

In questo capitolo ho elencato tutti i requisiti relativi ad ENGaming. Questi sono stati individuati da parte mia insieme al tutor aziendale. Ognuno dei requisiti avrà quattro caratteristiche ad esso associato:

- **Codice:** Valore univoco per ogni requisito che serve ad identificarlo. La struttura dei codici sarà composta da 3 lettere ed un numero. La prima lettera "R" indica il fatto che il codice sia assegnato ad un requisito del progetto. La seconda lettera "F", "Q" o "V" indica se il requisito è Funzionale, Qualitativo o di Vincolo. La terza lettera "O" o "D" indica se il requisito sia Obbligatorio o Desiderabile, mentre il numero alla fine del codice indica quale requisito di quella categoria si stia identificando.
- **Importanza:** Livello di importanza di ogni requisito per stabilirne la priorità durante la fase di codifica. I due valori che questo campo può assumere sono "Obbligatorio" o "Desiderabile".
- **Descrizione:** Essa sarà la descrizione effettiva di ciò che ogni requisito deve soddisfare.
- **Fonti:** Questa sezione elenca le fonti dalle quali il requisito è stato individuato e scelto. Nel caso in cui come fonte sia indicata la parola "Interno" vuol dire che il motivo per cui il requisito è presente è stato scelto arbitrariamente da me e/o il tutor aziendale, e che esso non derivi da nessuno dei casi d'uso.

Oltre a tutte queste caratteristiche i requisiti sono divisi in 2 gruppi principali:

- **Requisiti funzionali:** Sono i requisiti che descrivono il funzionamento dell'applicazione.
- **Requisiti non funzionali:** I requisiti non funzionali sono a loro volta divisi in 2 categorie, qualitativi e di vincolo. Questi descrivono, rispettivamente i requisiti riguardanti la qualità del progetto ed i vari vincoli che dovranno essere rispettati.

3.2.1 Requisiti funzionali

Codice	Importanza	Descrizione	Fonti
RFO1	Obbligatorio	Il cliente deve poter selezionare un gioco all'interno di ENGaming.	**UC 1**
RFO2	Obbligatorio	Il cliente deve poter visualizzare le classifiche dei vari giochi presenti e cambiare quella visualizzata.	**UC 2, UC 3**
RFO3	Obbligatorio	Selezionato il gioco il cliente deve poterlo avviare dalla rispettiva pagina introduttiva presente a schermo.	**UC 4**
RFO4	Obbligatorio	Il cliente, dalla pagina introduttiva di un gioco, deve poter tornare al menù principale di ENGaming.	**UC 5**
RFO5	Obbligatorio	Il cliente deve poter chiudere ENGaming dal suo menù principale e far ritornare il tablet ENSign 11 allo stato di schermo aggiuntivo.	**UC 6**
RFO6	Obbligatorio	Avviata una partita ad un gioco il cliente deve poter controllare il proprio personaggio tramite l'apposito controller mostrato a schermo, se supportato, o tramite il pennino.	**UC 7, UC8**
RFO7	Obbligatorio	Durante una partita il cliente deve poter premere un apposito pulsante a schermo per mettere in pausa il gioco.	**UC 9**
RFO8	Obbligatorio	Dal menù di pausa di un gioco il cliente deve avere la possibilità di riavviarlo tornando al menù principale di questo.	**UC 10**

RFO9	Obbligatorio	Dal menù di pausa il cliente deve poter chiudere il gioco e tornare al menù principale di ENGaming.	**UC 11**
RFO10	Obbligatorio	Una volta uscito da un gioco, il cliente deve avere la possibilità di salvare il proprio punteggio nelle classifiche.	**UC 12**
RFD11	Desiderabile	Il cliente deve poter vedere una lista con gli altri ENSign 11 che sono collegati alla rete con cui può comunicare.	**UC 13**
RFD12	Desiderabile	Il cliente deve poter sfidare un altro ENSign 11, collegato alla stessa rete, ad una partita multigiocatore.	**UC 14**
RFD13	Desiderabile	Il cliente che riceverà una notifica di sfida da un altro ENSign 11 collegato alla rete deve poter accettare o rifiutare la sfida.	**UC 15, UC 16**
RFO14	Obbligatorio	Il cliente deve poter modificare le impostazioni di ENGaming da un'apposita pagina chiamata "Impostazioni" presente nel menù principale.	**UC 17**

3.2.2 Requisiti qualitativi

Codice	**Importanza**	**Descrizione**	**Fonti**
RQO1	Obbligatorio	Devono essere stesi i manuali utente e sviluppatore.	**Interno**
RQO2	Obbligatorio	Deve essere consegnato il codice prodotto in formato sorgente utilizzando i sistemi di versionamento offerti dall'azienda (AWS).	**Interno**
RQO3	Obbligatorio	Deve essere presente una ux con uno stile grafico aggiornato ed appropriato al progetto in tutti i menù ed impostazioni di ENGaming.	**Interno**
RQO4	Obbligatorio	I controller a schermo devono supportare multipli tocchi contemporanei da parte dei clienti per non creare limitazioni o problemi durante l'esperienza di gioco.	**Interno**

3.2.3 Requisiti di vincolo

Codice	**Importanza**	**Descrizione**	**Fonti**
RVO1	Obbligatorio	Gli utenti finali dovranno utilizzare macchine con sistema operativo Windows 10.	**Interno**
RVO2	Obbligatorio	Il prodotto deve essere sviluppato usando Node.js, ElectronJS, Angular e C++.	**Interno**
RVO3	Obbligatorio	ENGaming deve già contenere una serie di giochi funzionanti al suo interno.	**Interno**
RVO4	Obbligatorio	Deve essere usato AWS per l'hosting della repository contenente ENGaming	**Interno**

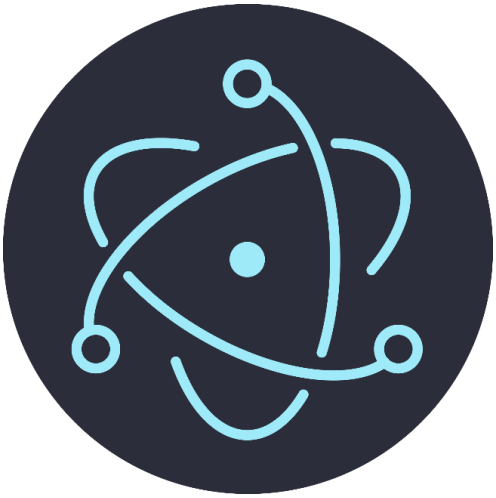
4. Progettazione e codifica:

4.1 Tecnologie e strumenti

In questa sezione ho elencato tutte le varie tecnologie che ho usato per lo sviluppo di ENGaming, con la loro relativa descrizione e spiegazione di come queste siano state impiegate nel progetto e del perché siano state scelte ed utilizzate.

4.1.1 Tecnologie e strumenti di ENGaming

Electron



Electron è il framework su cui si basa l'intera applicazione desktop di ENGaming. Esso permette di sviluppare GUI di applicazioni desktop utilizzando tecnologie web. Inoltre, permette anche una comunicazione tra il backend e frontend dell'applicazione tramite l'invio di messaggi attraverso i canali di Inter-Process Communication, un sistema di comunicazione in esso integrato. Il suo utilizzo è stato imposto dall'azienda, per degli ottimi motivi; infatti, essendo un framework open source offre molta libertà e personalizzazione del framework, se necessarie, ed è anche un'applicazione pensata specialmente per essere facile ed intuitiva nell'utilizzo, e tali caratteristiche rendono estremamente agevole lo sviluppo di applicazioni desktop tramite di essa anche perché facilita di molto l'integrazione di qualsiasi tecnologia web sia necessaria. Tutto questo è permesso grazie alla combinazione di Chromium, per il rendering, e Node.js per il runtime.

Angular

Angular è il framework dell'applicazione che si occupa della creazione e gestione della parte visiva di questa. Il suo funzionamento si basa sulla creazione di "Componenti" dove ciascuna componente è un elemento o pagina di ENGaming.

Ognuna di queste componenti è caratterizzata dalla presenza di 3 elementi fondamentali: un file HTML, che descrive e compone la struttura del componente, un foglio di stile che gestisce la grafica ed un file TypeScript in cui è contenuto e sviluppato il comportamento del componente. Oltre a tutto questo Angular offre anche la possibilità di trasferire dati tra le sue componenti.

Anch'esso è stato selezionato da parte dell'azienda per lo sviluppo della creazione, ed anche per esso è facile capirne il motivo, è anch'esso un framework open source con documentazioni ed esempi di implementazioni chiari e dettagliati per apprenderne al meglio il funzionamento. Infine, grazie al suo funzionamento basato sulle componenti, rende molto pratico concentrare la propria attenzione nello sviluppo di una singola parte alla volta, rendendo poi estremamente facile l'unione di tutte le componenti necessarie e richieste in una singola pagina.

TypeScript

La necessità dell'utilizzo di TypeScript, che è un'estensione di JavaScript che permette la tipizzazione delle variabili, deriva dalle tecnologie scelte e descritte precedentemente, ElectronJs ed Angular, dal momento che il codice TypeScript necessario per la definizione del comportamento delle varie componenti all'interno della parte Angular sarà poi codificato e trasformato in codice JavaScript che potrà essere letto dalla parte in Electron dell'applicazione.

C++

C++, linguaggio di programmazione che non necessita di particolari introduzioni, è utilizzato per la gestione della logica del device attraverso un apposito driver (fornito dall'azienda). Tramite esso vengono rilevate tutte le interazioni che si hanno con l'ENSign 11 specialmente grazie alla sua estrema velocità. Infatti, dal momento che ad ogni interazione vengono ricevuti 250 o 100 pacchetti, rispettivamente se con l'utilizzo del pennino o con le dita, al secondo la necessità di un linguaggio estremamente veloce che sia in grado di elaborare molto rapidamente tale mole di dati, è ovvia.

Una volta che i dati degli input sono stati elaborati e decodificati dal driver in C++ essi sono trasmessi al resto dell'applicazione grazie al loro intermediario: Node-API.

Node-API

Node-API è un toolkit presente in Node che funziona da intermediario tra il codice scritto in C/C++ e quello scritto in Javascript. Grazie ad esso il driver, che è scritto in C++ riesce a comunicare ed interfacciarsi con la parte del programma che è stata sviluppata in Electron. Anche per questa tecnologia, quindi, la motivazione che ha portato alla sua scelta è ovvia, infatti senza di essa sarebbe impossibile comunicare gli input rilevati dal tablet, e dal suo driver, al resto dell'applicazione.

Electron-Forge

Electron-Forge è uno strumento utilizzato per la creazione dei pacchetti per la distribuzione di applicazioni Electron specifiche per il sistema operativo su cui il progetto viene compilato. In questo modo l'applicazione può essere compilata indipendentemente dal sistema operativo che si sta usando, questo permette ad ENGaming, nel caso si voglia, di essere un'applicazione multipiattaforma. Il suo utilizzo nel progetto è quindi necessario per permettere ad ENGaming di essere distribuito una volta finito.

Firebase

Firebase è una piattaforma online, sviluppata da Google, per la creazione di applicazioni per dispositivi mobili e web. Essa offre vari strumenti al suo interno, per il progetto è stata utilizzata solo la funzionalità di "Realtime Database" che permette di salvare, modificare o leggere dei dati in remoto facilmente evitando così problemi di consistenza. Ho scelto di utilizzare Firebase, tra le varie proposte presentatemi dall'azienda, perché dopo averlo esaminato ho trovato molto facile ed intuitiva la console ed interfaccia web, per poterci operare e gestirlo in remoto; è stata anche molto veloce e semplice la sua integrazione all'interno delle applicazioni. Inoltre, all'interno del database, i dati salvati usano una struttura tramite indici rendendo così molto semplice il passaggio dall'utilizzo di un file json, usato fino ad ora per salvare i punteggi e le classifiche in locale, all'utilizzo di questa tecnologia.

4.1.2 Tecnologie e strumenti dei giochi

HTML, CSS & JavaScript

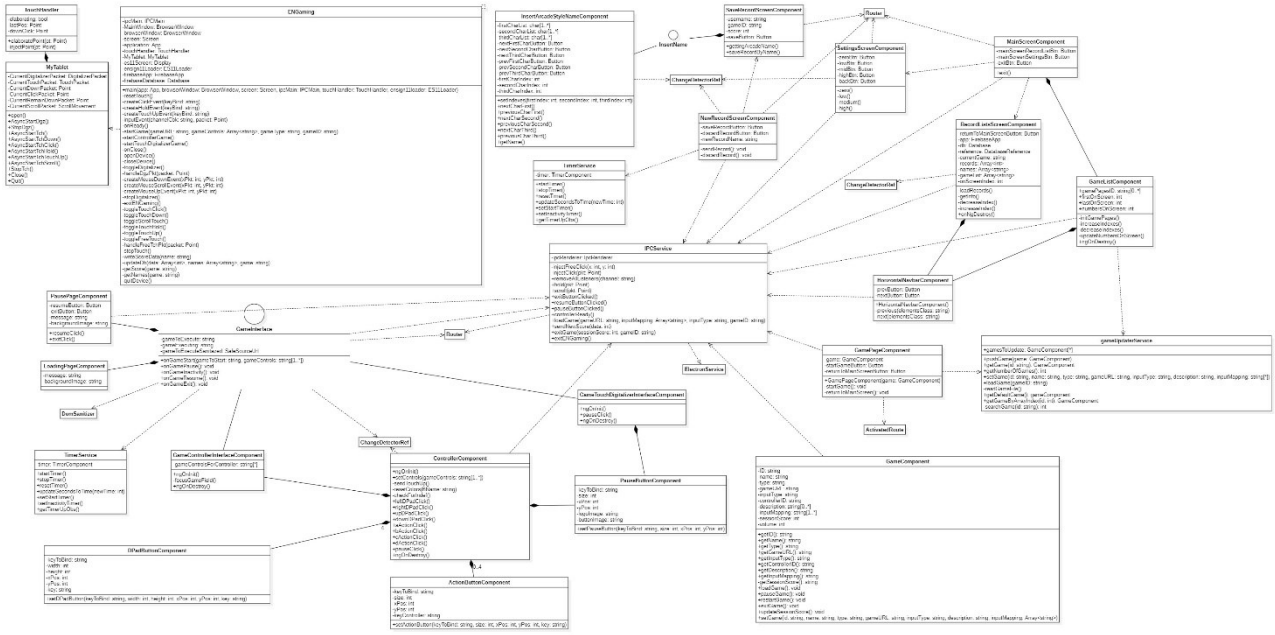
I giochi presenti in ENGaming sono eseguibili sui browser, per questo motivo sono costituiti da una componente HTML, una CSS ed una JavaScript. La parte HTML è utilizzata per la struttura del gioco, mentre il CSS viene utilizzato per lo stile di esso ed il JavaScript si occupa del gioco vero e proprio. Però, affinché i giochi possano interfacciarsi con l'applicazione, per esempio per salvare il punteggio effettuato, e scambiare con essa dei dati è necessario che questi possano inviargli dei messaggi.

WebAssemly

Alcuni giochi in ENGaming, come Doom, hanno prima bisogno di passare attraverso WebAssemly per poter essere eseguibili sul browser. Le modifiche da effettuare su di essi, affinché il gioco funzioni correttamente, dipendono però dal singolo gioco, ma sicuramente ci sarà bisogno di possedere i codici sorgenti per poterle identificare. Infine, per poter funzionare, questi giochi saranno poi letti da un file JavaScript, attraverso il file di tipo wasm. Purtroppo, però, con questa tecnologia non si può ricevere il punteggio ottenuto tramite messaggio a causa delle loro limitazioni, quindi per essi i punteggi ottenuti non saranno salvabili.

4.2 Progettazione:

4.2.1 Classi utilizzate:



In questa immagine sono mostrate tutte le classi utilizzate e come siano relazionate e collegate tra loro. Esse possono essere anche suddivise e raggruppate tra loro a seconda del linguaggio di programmazione utilizzato per la loro implementazione, in particolare:

Electron:

- ENGaming

C++:

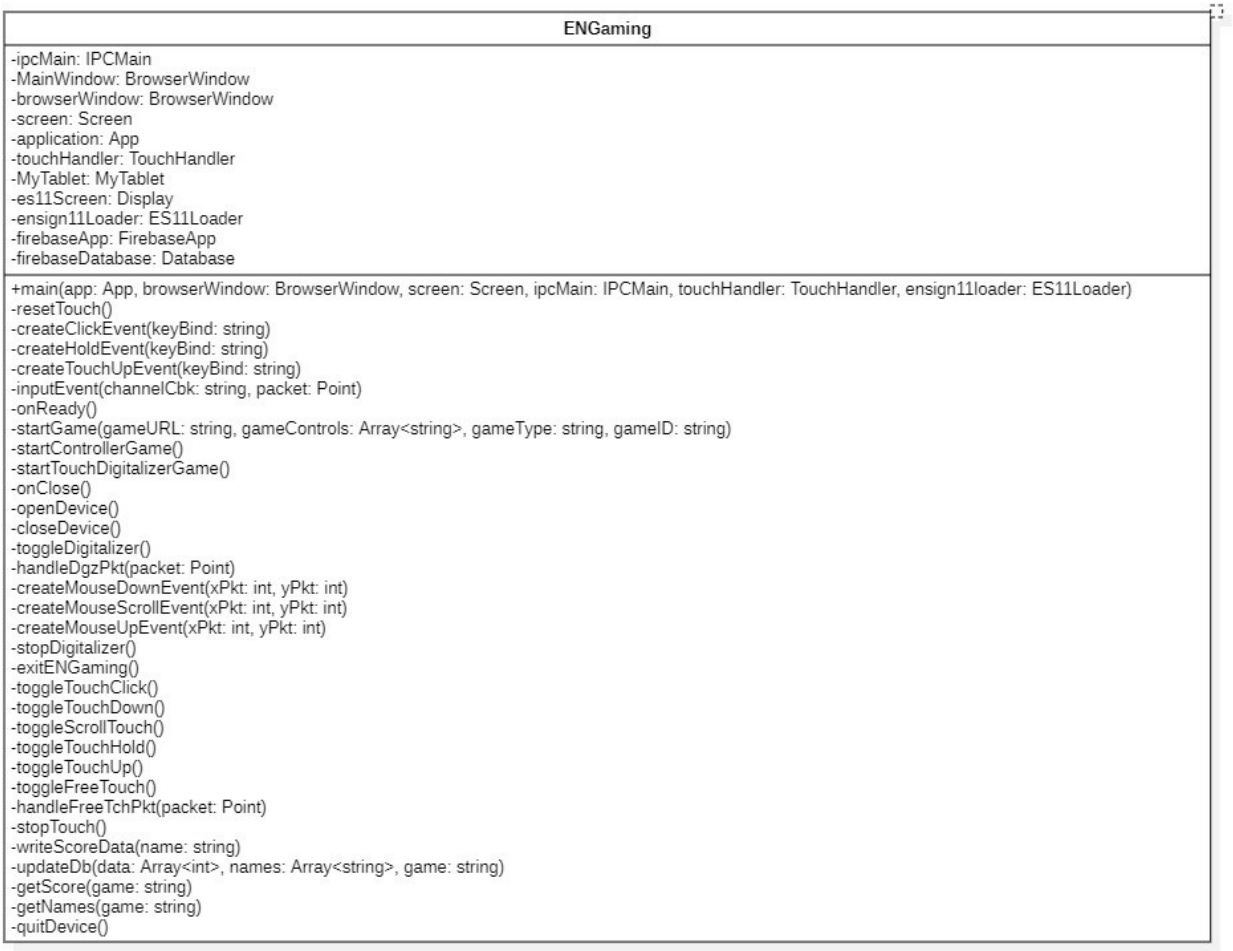
- DigitalizerPacket
- MyTablet
- Point
- ScrollMovement
- TouchHandler
- TouchPacket

Angular:

- Ogni classe contenente “Component”
- Ogni classe contenente “Service”
- ActivatedRoute
- ChangeDetectorRef
- DomSanitizer
- GameInterface
- InsertName
- Router

Ora avremo un’analisi più dettagliata delle classi in base al loro funzionamento.

4.2.2 Engaming



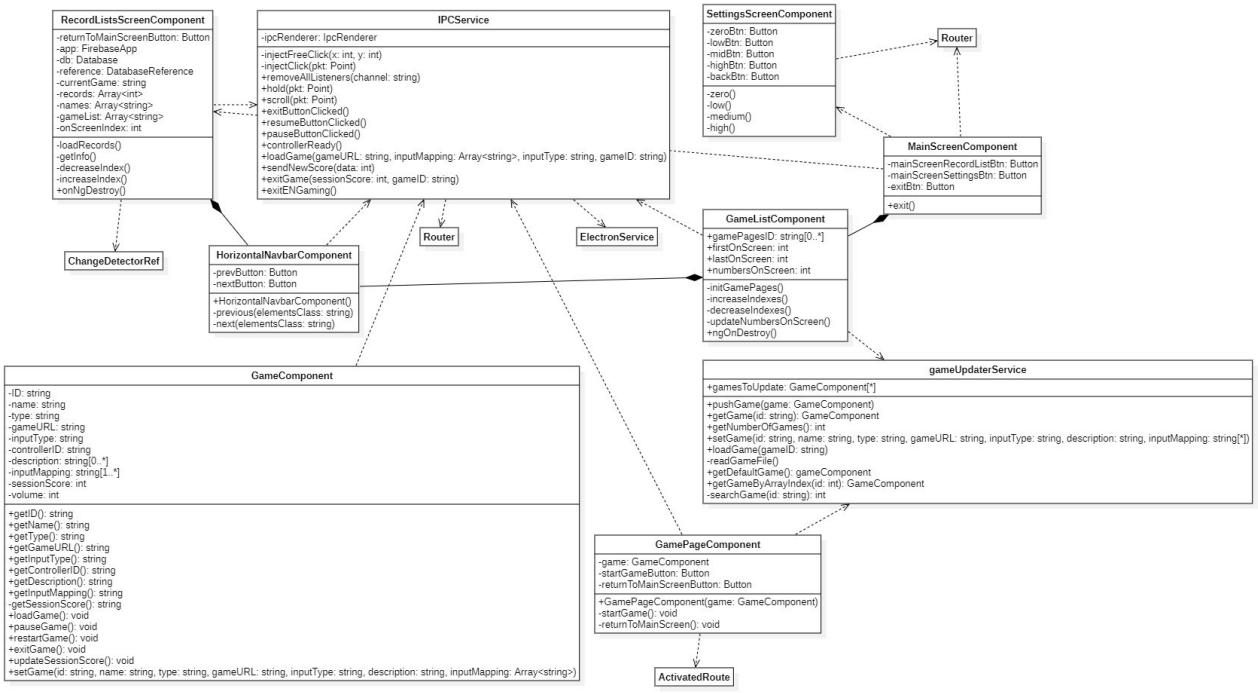
La classe ENGaming si occupa dell'avvio dell'applicazione. Sviluppata in Electron, contiene i componenti necessari per la creazione della finestra nell'ENSign 11 (Screen e BrowserWindow) e per la comunicazione con la parte in Angular (IPCMain). Ed infine, contiene anche i moduli per la comunicazione con ENSign 11.

I metodi al suo interno sono divisi nelle seguenti categorie:

- Metodi per interagire con l'ENSign 11:
    - openDevice: apre il device
    - closeDevice: chiude il device
    - toggleDigitalizer: riceve i pacchetti con la struttura definita in DigitalizerPacket
    - stopDigitalizer: ferma la ricezione di pacchetti e le interazioni con il digitalizer
    - toggleTouchClick: riceve un click
    - toggleTouchDown: riceve un "down", evento che corrisponde ad un dito appoggiato allo schermo
    - toggleScrollTouch: riceve uno "scroll", evento che corrisponde al movimento di un dito sullo schermo
    - toggleTouchHold: riceve un "hold", evento che corrisponde ad un dito che viene mantenuto appoggiato sullo schermo
    - toggleTouchUp: riceve un "touch up", evento che corrisponde al sollevamento di un dito dallo schermo
    - toggleFreeTouch: riceve i pacchetti con la struttura definita in TouchPacket
    - stopTouch: ferma la ricezione di pacchetti e le interazioni con lo schermo del device

    - quitDevice: ferma l'istanza di ENGaming e rende il device di nuovo libero di essere utilizzato da altri applicativi.
- Metodo per l'avvio dei giochi:
    - startGame: inizializza un gioco dopo averne identificato la tipologia
    - startControllerGame: avvia un gioco che richiede l'uso del controller
    - startTouchDigitalizerGame: avvia un gioco che richiede l'uso delle dita o del digitalizer.
- Metodi per l'interazione con giochi che richiedono l'uso di un controller:
    - inputEvent: permette di interagire con il corrispondente componente del controller che viene premuto
    - createClickEvent: crea l'evento "click" nella pagina del gioco
    - createHoldEvent: crea l'evento "hold" nella pagina del gioco
    - createTouchUpEvent: crea l'evento "touch up" nella pagina del gioco.
- Metodi per interagire con un gioco che utilizza il dito o il digitalizer:
    - handleFreeTchPkt: decide che evento sollevare a seconda del pacchetto ricevuto
    - handleDgzPkt: decide che evento sollevare del digitalizer a seconda del pacchetto ricevuto
    - createMouseDownEvent: crea l'evento di appoggio del dito o digitalizer nel gioco
    - createMouseScrollEvent: crea l'evento di movimento del dito o digitalizer nel gioco
    - createMouseUpEvent: crea l'evento di sollevamento del dito o digitalizer nel gioco.
- Metodi per interagire con il database in remoto
    - writeScoreData: aggiorna i punteggi dei giochi in locale
    - updateDb: salva i nuovi punteggi nel database in remoto
    - getScore: ottieni i punteggi dal database in remoto
    - getNames: ottiene i nomi dei giocatori dal database in remoto.
- resetTouch: ripristina il touch del device una volta terminata l'esecuzione di un gioco.
- exitENGaming: chiude l'applicazione.

4.2.3 Schermata iniziale



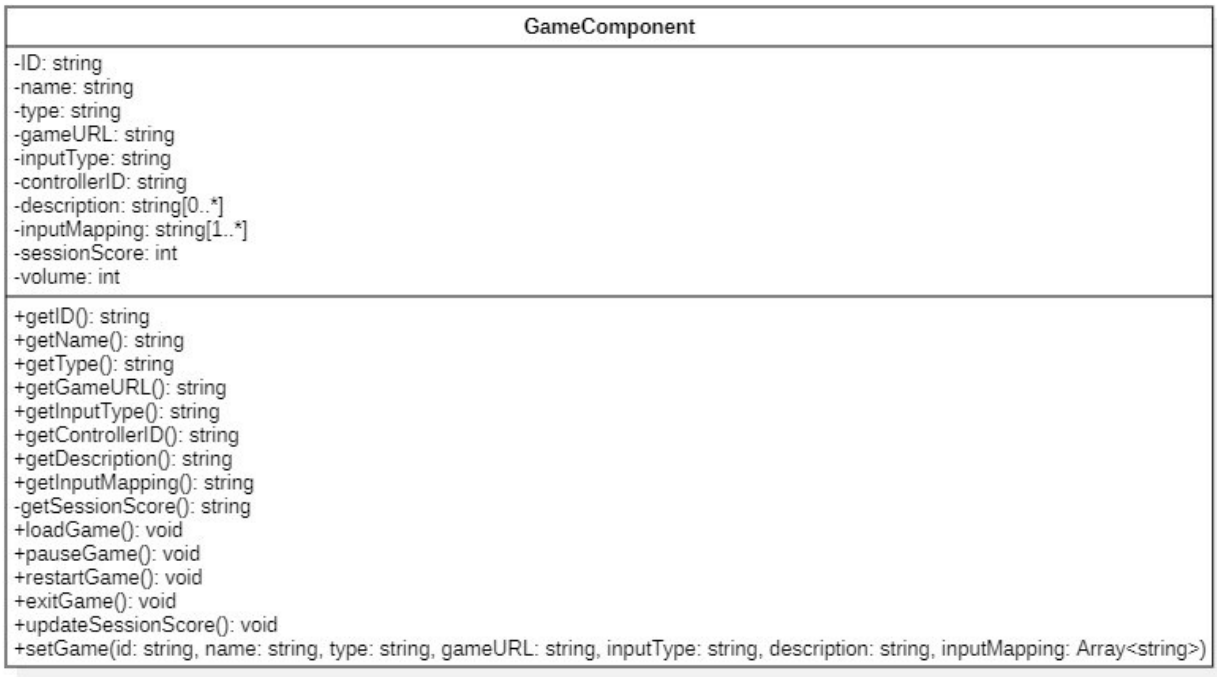
All'avvio dell'applicazione la schermata principale presenterà 4 elementi:

- L'elenco dei giochi disponibili, ottenuto da GameListComponent
- Il pulsante per accedere alla pagina di RecordListScreenComponent
- Il pulsante per accedere alla pagina di SettingsScreenComponent
- Il pulsante per uscire dall'applicazione.

Per far comunicare tutte queste diverse pagine e sezioni la componente della schermata principale, MainScreenComponent, ho utilizzato il servizio IPCService, implementato in Electron, che permette di inviare e ricevere messaggi negli appositi canali IPC. Inoltre:

- RecordListScreenComponent contiene un riferimento al database in remoto di Firebase da cui ottiene le informazioni riguardo i record del gioco attualmente selezionato a schermo. Tramite la HorizontalNavbarComponent naviga tra una classifica all'altra.
- SettingsScreenComponent è la pagina relativa alle impostazioni dell'applicazione, in essa possono essere selezionati / modificati i valori di questi, come il volume, che saranno poi comunicati aggiornati ad ENGaming.
- GameListComponent è formato da una HorizontalNavBarComponent, per la gestione della visualizzazione dei giochi disponibili, e da una GamePageComponent, cioè da una pagina relativa di almeno un gioco.
- Ciascun GamePageComponent corrisponde ad un singolo GameComponent. Le informazioni in essa sono caricate da GameUpdaterService. In essa sono presenti le informazioni ed i comandi relativi ad ogni gioco ed i pulsanti per avviarlo o tornare al menù principale.

4.2.4 GameComponent



GameComponent è la classe contenente tutte le informazioni riguardanti un gioco all'interno di ENGaming. Queste informazioni sono sia quelle relative al suo avvio che quelle dei suoi controlli, ed oltre a ciò, sono presenti anche tutte le informazioni descrittive del gioco.

Per essere rappresentato ogni gioco avrà bisogno di:

- **ID:** corrisponde al codice utilizzato per identificare un gioco
- **name:** corrisponde al nome del gioco
- **type:** specifica la tipologia del gioco
- **gameURL:** contiene l'indirizzo in cui trovare la finestra del gioco
- **inputType:** indica la tipologia di input che utilizza il gioco, può essere "controller" o "touchDigit", rispettivamente touch o digitalizer
- **controllerID:** contiene l'ID del controller specifico utilizzato da quel gioco
- **description:** contiene una descrizione del gioco
- **inputMapping:** in questo campo è contenuta la mappatura dei controlli che sono utilizzati nel gioco
- **volume:** contiene il livello del volume che viene utilizzato.

Grazie all'utilizzo di IPCService essa può sia inviare che ricevere gli eventi riguardanti il gioco in esecuzione, come gli input dell'utente sullo schermo, e grazie all'utilizzo di una variabile, chiamata sessionScore, viene registrato il punteggio più alto ottenuto nella sessione di gioco in modo che possa venire salvato nelle classifiche.
Per il salvataggio e la memorizzazione dei giochi è utilizzato un file games.json la cui struttura è la seguente:

{"games": {

"id": "catMario",

"name": "Cat Mario",

"type": "Action",

"gameURL": "https://www.cat-mario.com/",

"inputType": "controller",

"controllerID": "NintendoDS",

"description": "Clone di Super Mario con un gatto come personaggio",

"inputMapping": "["salto: A","left: D-Pad left","right: D-Pad right"]"

},

{

"id": "subwaySurfers",

"name": "Subway Surfers",

"type": "Endless",

"gameURL": "https://subway-surfers.me/",

"inputType": "touchDigit",

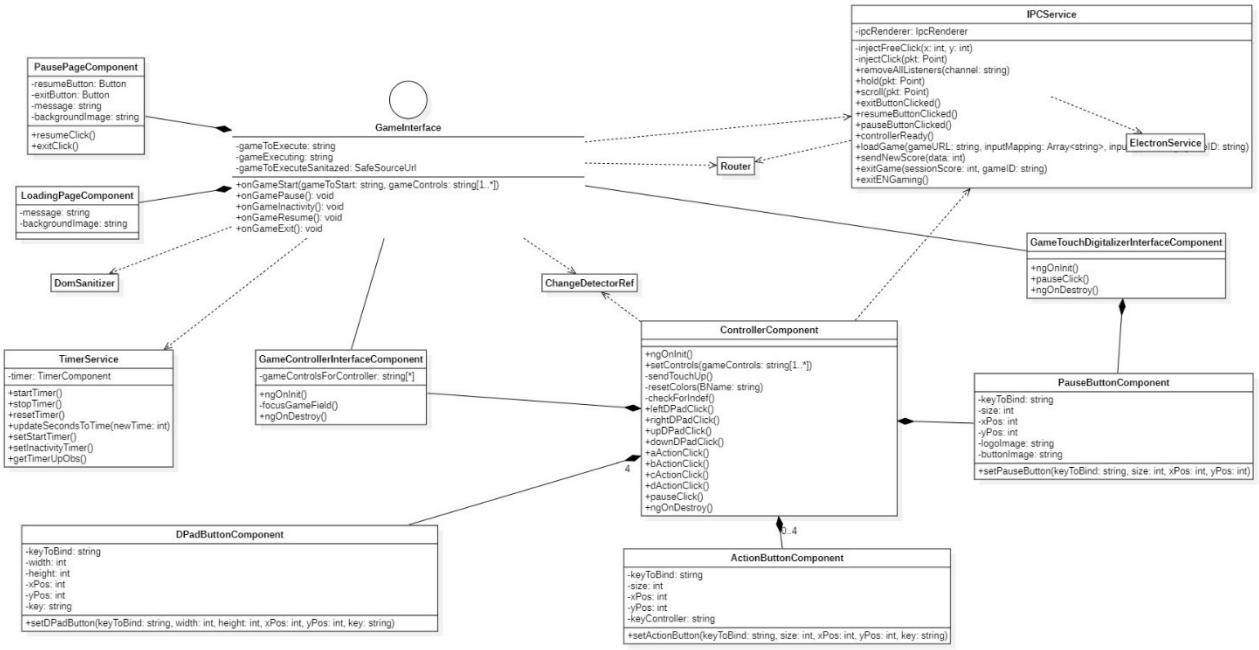
"description": "Gioco di corsa senza fine ambientato in una metropolitana",

"inputMapping": "["inizia: tap","left: swipe-left","right: swipe-right","jump: swipe-up","roll: swipe-down"]"

}

}

4.2.5 Interfaccia di gioco



Dal momento che ho dovuto implementare due differenti tipologie di gioco, che però hanno degli elementi in comune, ho utilizzato l'interfaccia GameInterface che contiene tutte le informazioni relative alla corretta visualizzazione di un gioco comuni ed entrambe le interfacce utilizzate, spiegate successivamente, che si differenziano per la tipologia di gioco avviato. Queste componenti sono:

- un TimerComponent per la gestione della schermata di caricamento e lo stato di inattività
- un LoadingPageComponent per la visualizzazione della schermata di caricamento
- un PausePageComponent per visualizzare le opzioni disponibili durante la pausa di un videogioco, con la possibilità di riavviarlo o chiuderlo.
- IPCService per far comunicare le varie componenti nei rispettivi canali
- la classe Router per essere reinderizzati da un componente ad un altro aggiornando la vista tramite il metodo navigate
- DomSanitizer che "pulisce" un URL rendendo il suo contenuto visualizzabile in un iframe oppure un object
- ChangeDetectorRef che permette di aggiornare la vista tramite il metodo detectChanges

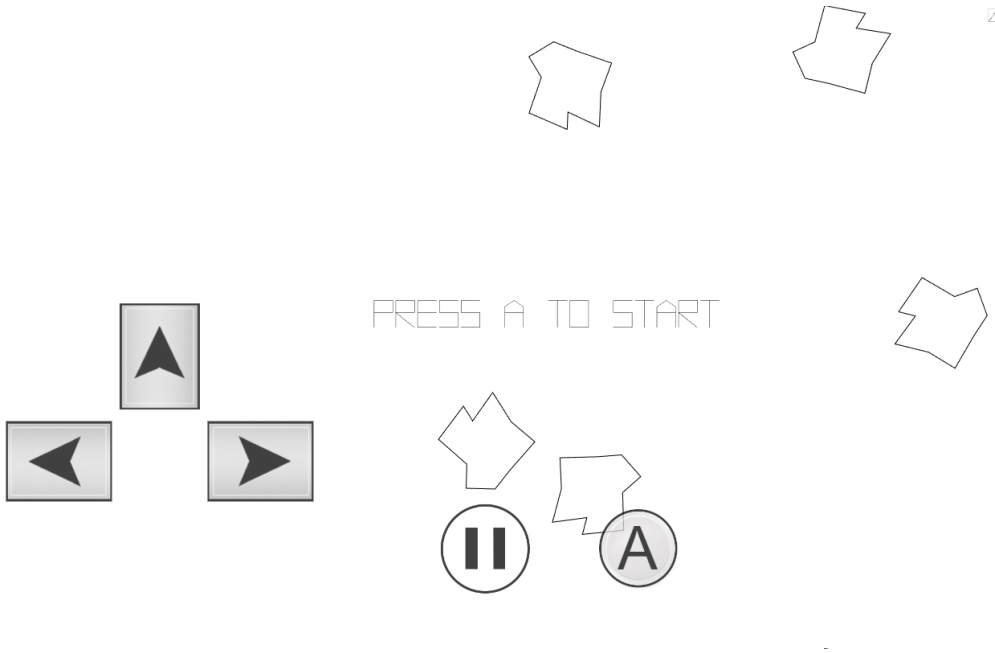
4.2.6 GameControllerInterfaceComponent

Questa componente rappresenta l'interfaccia di gioco per i giochi che utilizzano il controller. Essa, oltre ad avere gli elementi di GameInterface dal momento che implementa l'interfaccia, è composta da un controller ed un pulsante di pausa, tutto diviso in 3 parti:

- **Prima parte:** Composta da 4 pulsanti "DPadButtonComponent" viene utilizzata per gestire i movimenti all'interno del gioco
- **Seconda parte:** Composta da 4 pulsanti "ActionButtonComponent" viene utilizzata per gestire le azioni compiute all'interno del gioco
- **Terza parte:** È composta da un "PauseButtonComponent" ed è dedicata a mettere in pausa il gioco.

Il controller che sarà utilizzato viene configurato prima dell'avvio a seconda di quanti e quali comandi siano necessari al gioco che si sta avviando, gli eventuali pulsanti aggiuntivi che non verranno utilizzati all'interno del gioco non saranno mostrati a schermo per evitare confusione da parte del giocatore. Il design e layout dei controller, inoltre, è pensato per il device posizionato in orizzontale su una superficie e, dal momento che

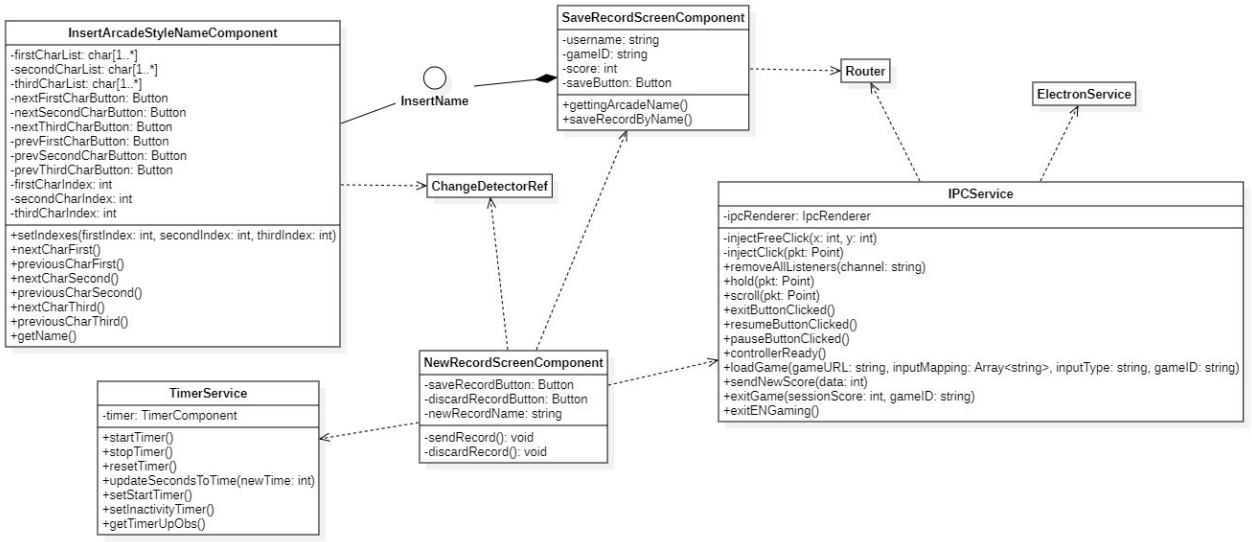
questa sarà la posizione in cui si troverà sempre l'ENSign, non sarà necessario prenderlo in mano o spostarlo per poter giocare. In particolare, i pulsanti di movimento sono posizionati nel lato sinistro inferiore dello schermo, e quelli di azione nel lato destro inferiore, in questo modo viene facilitato l'utilizzo di entrambe le mani per il gioco senza che queste coprano una quantità eccessiva dello schermo bloccando la visione dello schermo all'utente. Un esempio del layout di un controller è mostrato nell'immagine che segue:



4.2.7 GameDigitalizerInterfaceComponent

Questa componente rappresenta l'interfaccia di gioco per quelli che non utilizzano il controller, che quindi utilizzeranno touch o digitalizer. Per questa categoria di giochi non è necessario avere definita la tipologia di input dal momento che entrambe le modalità ottengono e producono lo stesso risultato, in questo modo l'utente potrà interagire con il tablet nella maniera che più preferisce e gli è comoda. La classe, che implementa sempre l'interfaccia GameInterface, contiene una sola sezione aggiuntiva in cui è presente un PauseButtonComponent, per mettere in pausa il gioco, che è di base posizionato in alto a destra dello schermo, per invadere il meno possibile lo schermo durante il gioco. Una dimostrazione di questo layout è presente nell'immagine che segue:

4.2.8 Salvataggio record



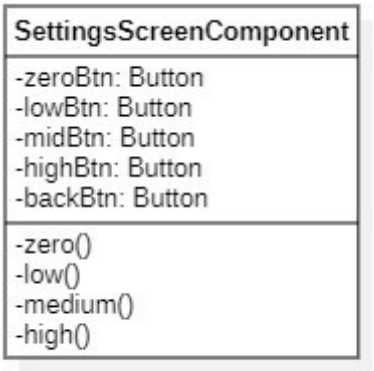
Quando, al termine di una partita, viene registrato un nuovo record l'applicazione mostra la pagina NewRecordScreenComponent dove viene si può scegliere se salvare o no il punteggio appena ottenuto. Se viene scelto di salvarlo verrà chiesto all'utente di inserire il proprio username, lungo 3 caratteri, come si usa fare nelle sale giochi, tramite la classe InsertArcadeStyleNameComponent. Una volta inserito il nome le informazioni riguardo al nuovo record, che sono all'interno della classe SaveRecordScreenComponent, sono inviate alla classe ENGaming, sempre tramite i canali di comunicazioni offerti da IPCService, che procederà ad aggiornare le classifiche nel Realtime Database di Firebase a cui è ENGaming è collegato.

Infine viene utilizzato un TimerComponent, tramite la classe TimerService, durante la procedura di salvataggio per gestire l'eventuale l'inattività all'interno della schermata nel caso di allontanamento dall'ENSign da parte dell'utente.

Nel Realtime Databse di firebase il salvataggio dei dati avviene in maniera simile ad un file json, infatti sono presenti degli indici, uno per ogni gioco presente nell'applicazione che permette di salvarne i record, ai quali sono collegati, nuovamente tramite indice con i numeri da 1 a 10, tutti i 10 migliori punteggi effettuali su quel gioco con annesso lo username dell'utente che ha ottenuto il punteggio.

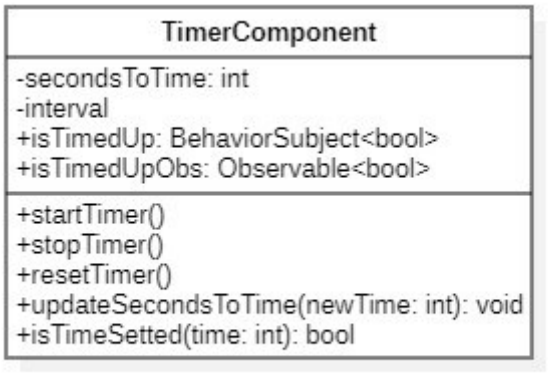
Un'immagine in cui è mostrato come sono salvati i record nel database è presentata di seguito:

4.2.9 Impostazioni



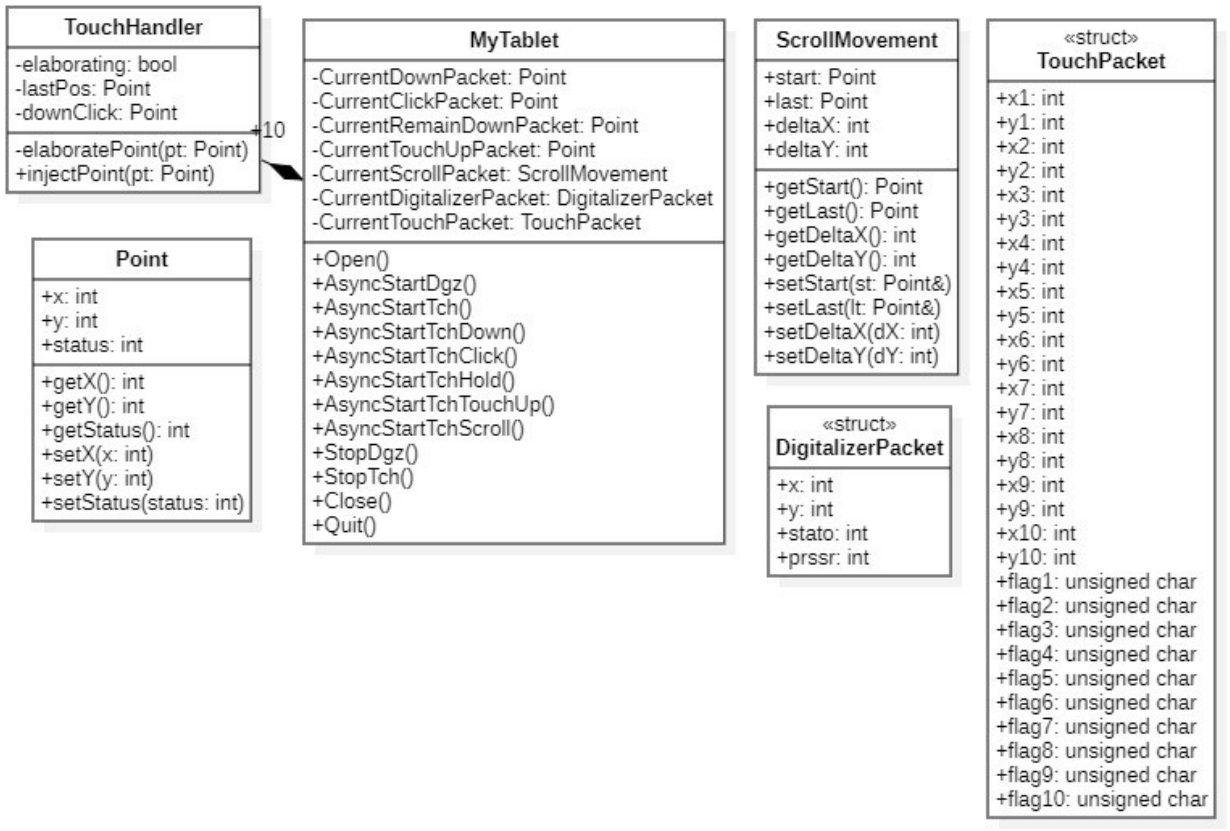
Per il menù delle impostazioni ho utilizzato una classe apposita che raccoglie al suo interno tutte le varie impostazioni modificabili in ENGaming e che, al variare di una di queste, invia un segnale alla classe ENGaming tramite l'IPCService contenente l'impostazione modificate ed il nuovo valore assegnatogli che sarà salvato. In questo esempio, e nello stato attuale dell'applicazione, la pagina delle impostazioni contiene solo i pulsanti ed i metodi per cambiare il volume dell'applicazione.

4.2.10 TimerComponent



La componente TimerComponent, utilizzata all'interno degli altri componenti tramite TimerService, gestisce l'utilizzo del tempo. Essa serve per contare, dato un periodo di tempo passatogli, espresso in millisecondi, quando questo si sia esaurito e notificare l'applicazione al raggiungimento di esso, la notifica dello scadere del tempo dell'intervallo avviene tramite un Observable. L'utilizzo di questa classe è principalmente collegato allo stato di inattività dell'applicazione, infatti quando questo si presenta, presupponendo che il precedente utente si sia allontanato dal tablet ed abbia finito di utilizzarlo, grazie al TimerComponent una volta scaduto il tempo l'applicazione torna alla sua schermata principale pronta per essere usata di nuovo, presumibilmente da un nuovo utente.

4.2.11 ENSign 11



Il driver utilizzato dall'ENSign 11 ha in esso le funzioni che servono per interagire con esso tramite la classe MyTablet. Affinchè questa classe possa ricevere e comprendere gli input effettuati sul tablet viene affiancato dalla classe TouchHandler che contiene tutte le informazioni dei vari click effettuati.

MyTablet contiene 10 diverse istanze della classe TouchHandler dal momento che il device offre la possibilità di gestire al massimo 10 tocchi simultanei, ed ognuna di esse memorizza se sullo schermo avviene un evento, o input, tra le seguenti categorie:

- **Down:** è il primo pacchetto che viene inviato dal driver, ed indica che il dito è appena stato appoggiato sullo schermo
- **Click:** indica che il dito è rimasto appoggiato sullo schermo per un breve lasso di tempo
- **Scroll:** indica che il dito, che è rimasto appoggiato allo schermo, si trova a delle nuove coordinate la cui distanza, dal punto iniziale, supera la tolleranza del click, utilizzata per non confondere dei click con scroll durante l'utilizzo
- **Hold:** indica che il dito è ancora appoggiato allo schermo senza essersi spostato di una distanza superiore alla tolleranza
- **Touch up:** è l'ultimo pacchetto inviato dal driver, indica che il dito è stato alzato dallo schermo.

Le varie tipologie di punti, rappresentati da oggetti, che MyTablet può inviare tramite i rispettivi metodi sono di quattro diversi tipi:

- **Point:** oggetto contenente le coordinate di un punto ed il suo stato, anche detto flag, esso viene utilizzato per tutti gli eventi indicati prima tranne per lo Scroll

- **ScrollMovement:** oggetto contenente i punti (della classe Point) corrispondi all'inizio ed alla fine di un movimento sullo schermo, oltre alle distanze tra queste, questo oggetto viene utilizzato per gli eventi di Scroll
- **TouchPacket:** questo oggetto contiene in esso tutte le coordinate e lo stato dei 10 tocchi sullo schermo, viene utilizzato durante i giochi che richiedono l'utilizzo di un controller
- **DigitalizerPacket:** questo oggetto gestite i tocchi sullo schermo effettuati dal digitalizer, contiene le coordinate, lo stato di esso e la pressione con cui sta avvenendo il tocco, viene utilizzato durante i giochi che non richiedono l'uso del controller.

4.3 Servizi Angular

Infine, il programma utilizza dei servizi specifici, offerti da Angular, per la sua corretta esecuzione, essi sono:

- **Router:** permette di essere reinderizzati da un componente all'altro, quando richiesto, aggiornato la vista a schermo tramite il metodo nagivate
- **ChangeDetectorRef:** aggiorna la vista quando delle modifiche sono rilevate dal metodo detectChanges
- **ActivatedRoute:** questo servizio permette di conoscere l'URL della pagina corrente
- **ElectronService:** permette di utilizzare i canali di comunicazione IPC e delle altre funzioni presenti di base in Electron
- **DomSanitazer:** questo servizio "igienizza" un URL in modo che il suo contenuto sia visualizzato all'interno di un iframe o object.

4.4 Scelte progettuali

Oltre a tutto quello già spiegato e descritto ci sono altre scelte progettuali che, a mio avviso, vale la pena siano discusse, ma che non appartengono a nessuna di queste sezioni.

- L'architettura utilizzata dall'applicativo è MVVM, Model-View-View-Model, questa è stata scelta dal mio precedente collega, e quindi non era modificabile, se non rifacendo completamente l'applicativo che sarebbe risultata solo una perdita di tempo, anche perché avrei personalmente effettuato la stessa scelta.
- Grazie agli Inter-Process Communication (IPC) è possibile far comunicare tra loro tutti gli elementi dell'applicazioni affinché sia presente un canale di comunicazione tra essi. Questo è estremamente importante, come si può vedere dalla progettazione, perché praticamente tutte le componenti utilizzano questo metodo di comunicazione ed è necessario tra le componenti Angular ed Electron. Per avere un'idea di come essi funzionino basta sapere che il modo in cui questi comunicano, è molto simile al comportamento di un Observer.
- L'utilizzo di un database in remoto, invece di uno in locale, per il salvataggio dei dati è stato introdotto per dare ad ENGaming un maggiore senso di sfida e competizione tra i vari utenti durante il gioco, che non si limiti al singolo dispositivo che stanno utilizzando, ma si espanda a tutti i vari giocatori che utilizzano l'applicazione.

4.5 Codifica:

Per lo sviluppo dell'applicazione sono state utilizzate, oltre a quelle già citate in precedenza, le seguenti versioni delle tecnologie già elencate:

- **NodeJS:** v. 18.16.0
- **Npm:** v. 9.5.1
- **TypeScript:** v. 4.9.5
- **C++:** v. 17
- **AngularCLI:** v. 15.2.8

- **Electron:** v. 25.0.1
- **Git:** v. 2.41.0

Mentre per il versionamento del codice sorgente ho utilizzato l'apposita repository creata per me da parte dell'azienda di Amazon Web Services (AWS), dove ho caricato e salvato il codice.

4.5.1 Ambiente di sviluppo

Per la gestione della cartella di sviluppo ho deciso di utilizzare la suddivisione effettuata dal mio collega, dal momento che divideva molto bene il codice relativo alle varie tecnologie utilizzate, permettendomi inoltre di familiarizzare più velocemente con esso e trovare più facilmente una parte di codice al bisogno. Essa è ordinata ed organizzata nel seguente modo:

- src_angular
  - app
    - COMPONENTI
    - «servizi»
    - «interfacce»
  - assets
    - fonts
    - games
      - giochi
    - previews
      - «immagini»
    - games.json
    - «altri file»
  - index.html
  - main.ts
  - «altri file»
- src_electron
  - App.ts
  - ENGaming.ts
- src_module
  - ES11LIB
  - ES11LOADER
  - LIBUSB
  - «altri file»

Dove ognuna di queste sezioni rappresenta:

- COMPONENTI: Indica tutte le cartelle dei componenti Angular che compongono l'applicazione
- «servizi»: indica tutti i file dei servizi, dotati di un singolo file TypeScript, per la definizione dei loro comportamenti
- «interfacce»: indica tutti i file delle interfacce, dotati solo di un file TypeScript, per la definizione delle stesse
- GIOCHI: indica le cartelle dono sono presenti tutti i giochi presenti nell'applicazione con i loro relativi sorgenti
- «immagini»: indica le immagini che sono utilizzate all'interno dell'applicazione
- ES11LIB: indica la cartella che contiene i driver che permettono il funzionamento di ENSign11, questi driver sono stati forniti dall'azienda
- ES11LOADER: indica la cartella con i file relativi alla gestione dell'ENSign11

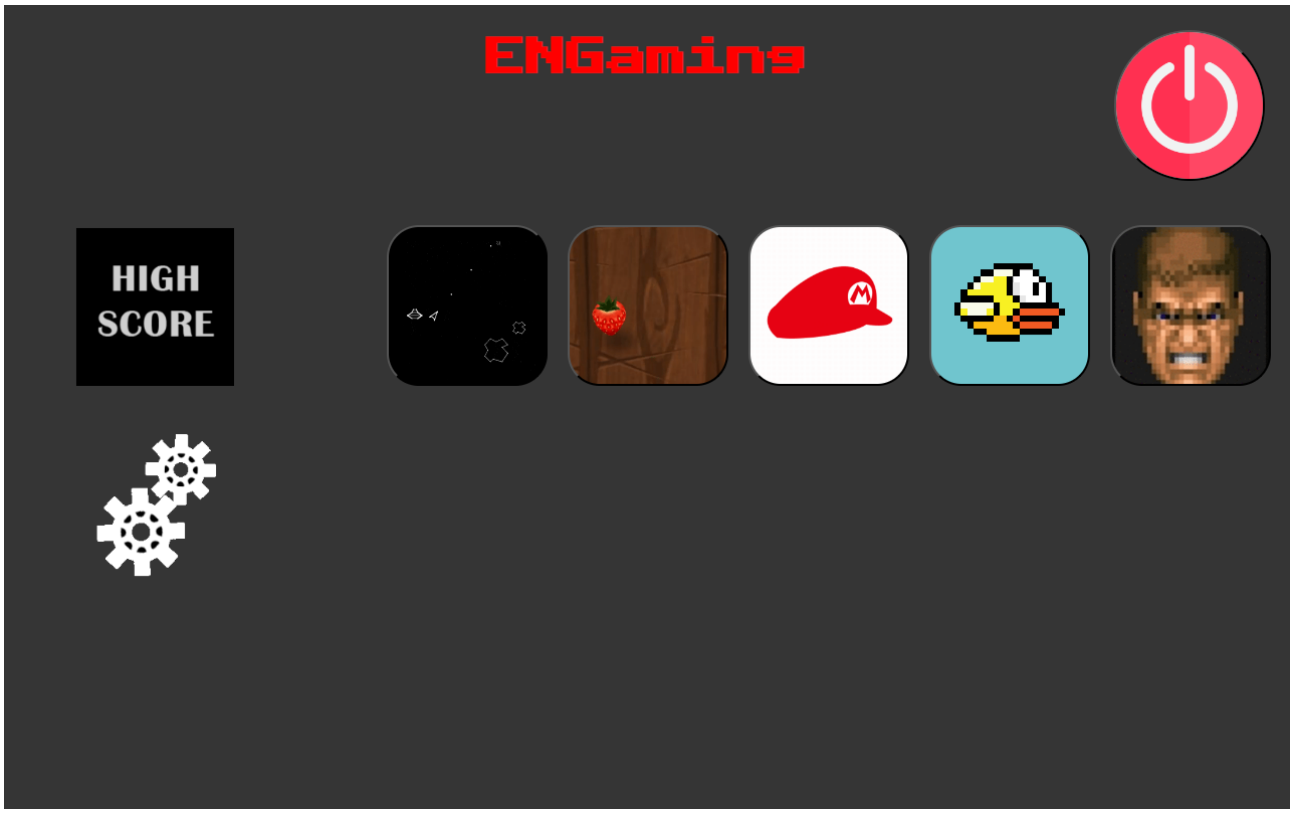
- LIBUSB: indica la cartella dov'è contenuta la libreria usata dai file presenti in ES11LIB
- «altri file»: indica gli altri file presenti nelle cartelle ma meno importanti, come il file .gitignore

Allo stesso livello di queste cartelle principali si trovano anche tutti i file di configurazione, come package.json, angularjson, tsconfig.json e forge.config.js.

4.6 Panoramica del prodotto

In questa sezione mostrerò uno screenshot delle varie sezioni e pagine dell'applicazione, per poi spiegare e descrivere ciò che si vede. Si partirà dalla schermata iniziale.

4.6.1 Schermata iniziale



Nella schermata iniziale possiamo trovare le componenti essenziali e principale dell'applicazione, cioè:

- Il nome dell'applicazione
- L'elenco dei giochi disponibili, mostrati tramite una loro icona o gif
- Un'icona per accedere alle classifiche dei record e visualizzare i migliori per ogni gioco
- Un'icona per accedere alle impostazioni dell'applicazione e modificarle
- Un pulsante per uscire dell'applicazione.

Questo tipo di visualizzazione avvicina ENGaming ai menù presenti nei dispositivi mobili a cui tutti siamo abituati in modo che l'utente si trovi in un ambiente dove potersi orientare facilmente senza sentirsi immediatamente spaesato o perso all'avvio di ENGaming e così possa familiarizzarci velocemente.

4.6.2 Visualizzazione record

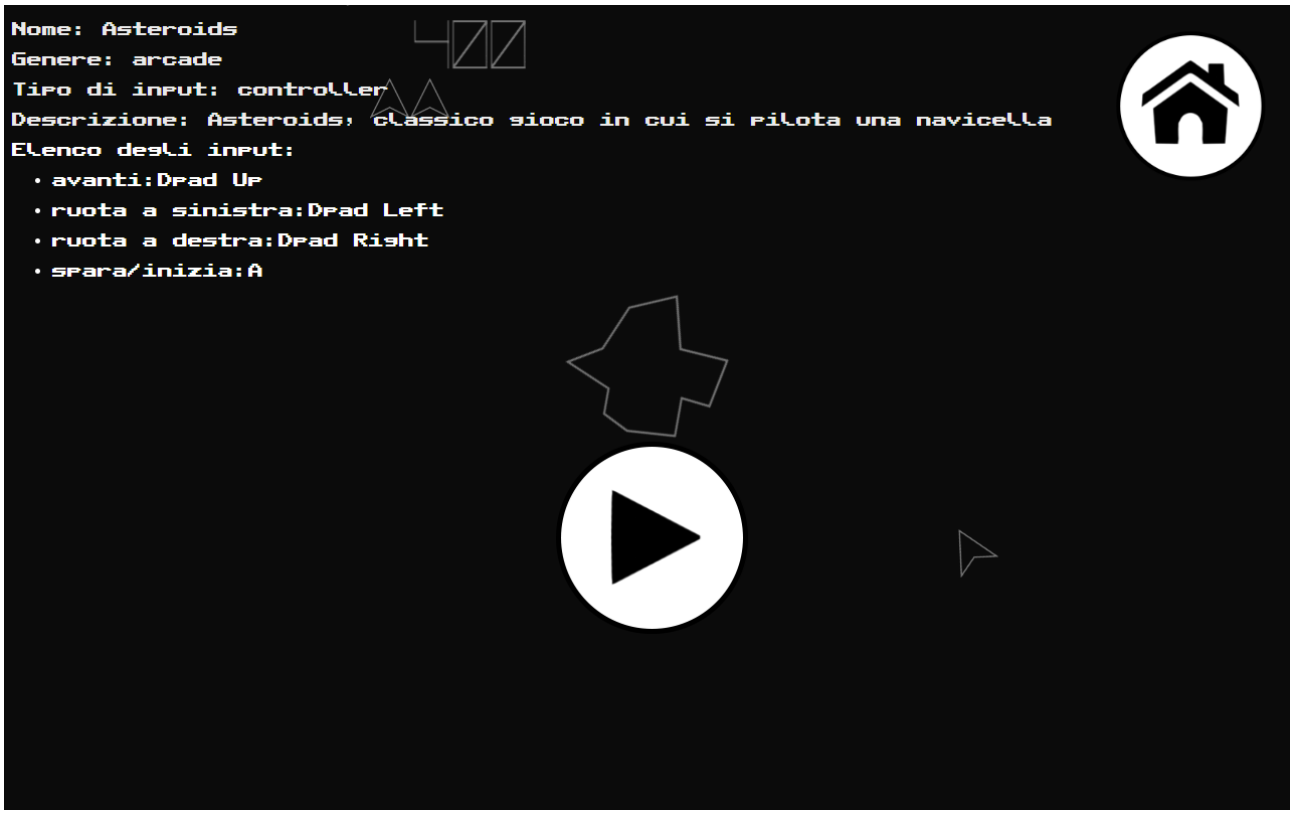
Tramite la gif animata nel menù principale per accedere alle classifiche si viene portati alla pagina che li mostra, la classifica selezionata di default per questa versione di ENGaming è quella del gioco “Asteroids” in quanto è il primo gioco dell’elenco a disposizione sull’applicazione che permette il salvataggio dei punteggi.

In questa pagina possiamo trovare:

- Un pulsante per tornare al menù principale
- Due pulsanti per visualizzare la classifica del gioco precedente o successivo
- L’indicizzazione della tabella contenente i record
- I vari record ottenuti dagli utenti, divisi in:
    - Posizione
    - Nome utente
    - Punteggio

Questa visualizzazione rappresenta fedelmente quella utilizzata nei cabinati all’interno delle sale giochi, a cui l’applicazione vuole ispirarsi, in cui i punteggi migliori erano elencati verticalmente con informazioni come: posizione nella classifica, nome (sempre composto da 3 sole lettere) e punteggio.

4.6.3 Pagina per l’avvio di un gioco



Una volta scelto un gioco dal menù principale si apre la pagina di avvio di questo dove, oltre ad uno screenshot del gioco di sottofondo, sono elencati in alto a sinistra i vari comandi ed istruzioni di questo, dove inizialmente si ripone l’attenzione di un utente aprendo una nuova pagina, oltre ai pulsanti per avviarlo o tornare al menù principale, posizionati invece al centro dello schermo per evidenziare che siano il contenuto principale della pagina.

Le informazioni minime presenti in questa pagina sono:

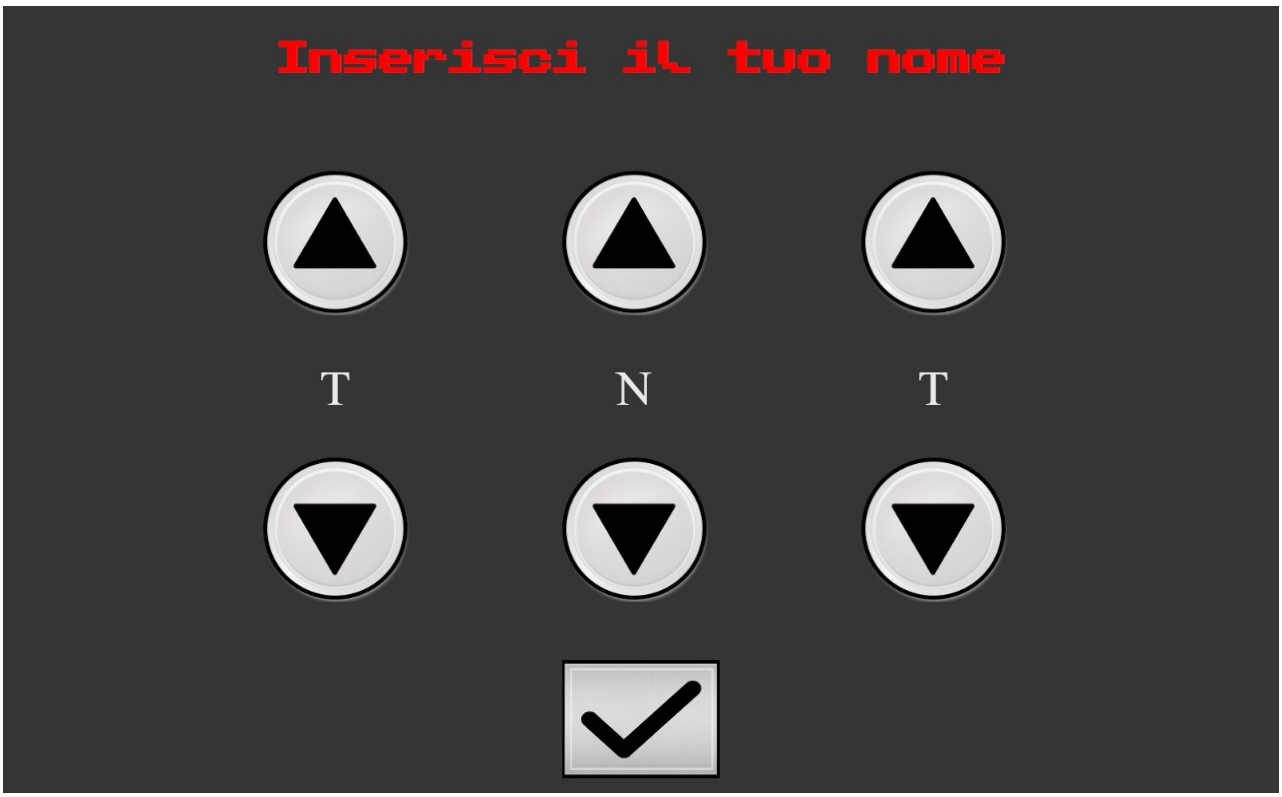
- Il nome del gioco
- Il genere del gioco
- La tipologia di input che può essere utilizzata nel gioco
- Una sua breve descrizione
- L’elenco degli input o gesture utilizzati da esso

4.6.4 Schermata di pausa

La schermata di pausa, presente all'interno di ognuno dei giochi, contiene molto semplicemente un pulsante per ritornare al gioco ed un pulsante per tornare al menù principale. La scelta di posizionare il pulsante per il ritorno al menù sulla destra si basa sul fatto che, nei videogiochi, in questo tipo di scelte l'opzione di uscita o corrispondente ad un'interruzione è sempre sulla destra.

4.6.5 Salvataggio record & inserimento nome

Una volta terminata una partita viene posta la domanda all'utente se voglia salvare o meno il punteggio più alto ottenuto giocando o meno tramite una schermata con la suddetta domanda ed i pulsanti relativi usati per rispondere. Nel caso si scelga di non salvare il record il punteggio è scartato e l'applicazione torna nel menù principale, mentre se si decide di salvare il punteggio si viene reindirizzati alla pagina per l'inserimento del nome in formato arcade che poi sarà inviato al programma per essere salvato nel database.



In questa schermata si è presentati con i 3 caratteri da cui sarà composto il nostro username e dei pulsanti per cambiare ognuno dei 3 caratteri con il precedente o successivo nell'alfabeto. Nel caso una delle lettere sia "A" scegliendo di passare a quella precedente verrà selezionata la lettera "Z", e viceversa, per semplificare la scelta nel caso si voglia selezionare velocemente una lettera nel lato opposto dell'alfabeto.

Questa tipologia di input, oltre a ricordare le sale giochi, permette di scegliere facilmente il proprio username senza utilizzare periferiche esterne o tastiere virtuali lasciando comunque abbastanza possibilità agli utenti di scegliere un nome che li identifichi e differenzi dagli altri.

4.7 Gestione degli errori

Gli errori che possono verificarsi in ENGaming sono limitati, dal momento che sono assenti configurazioni da parte dell'utente. Ma comunque sono state implementate delle accortezze per migliorare l'esperienza con esso.

4.7.1 Gestione del non collegamento del device

Quando l'applicazione viene avviata viene cercato un dispositivo ENSign 11 a cui collegarsi per farla funzionare. Nel caso questo non venga trovato verrà sollevata un'eccezione che mostrerà un errore all'utente per poi terminare il programma.

4.7.2 Gestione dell'inattività

Nel caso non vengano ricevuti input da parte dell'applicazione per 10 secondi sarà attivato un timer di durata 4 minuti, che sarà resettato quando un nuovo input sarà registrato. Nel caso, però, il timer raggiunga lo zero verrà terminata qualsiasi attività si stesse compiendo venendo riportati nel menù principale e nel caso si stesse salvando un record il punteggio effettuato sarà purtroppo perso. Questa funzionalità, oltre ad essere stata implementata per evitare una schermata statica dell'applicazione, ma per portare l'applicazione nel menù principale dove sono presenti più gif che attireranno più facilmente l'attenzione facendo utilizzare nuovamente ENGaming, è anch'essa un riferimento alla stessa gestione dell'inattività nei giochi arcade, in cui dopo si veniva sempre portati nel menù principale dopo un periodo di inattività.

5. Verifica e validazione:

5.1 Verifica

Per la verifica dell'applicazione non ho ritenuto necessari l'utilizzo di test automatici, dal momento che essendo le varie situazioni e comportamenti dell'applicazioni semplici e limitati, sono stato in grado di riprodurre e verificare tutte le singole parti manualmente, sia osservando le risposte a schermo nell'applicativo, sia servendomi di strumenti di debug se necessari.

5.2 Validazione

La fase di validazione, invece, è stata fatta in presenza del tutor aziendale, il quale, dopo una mia presentazione e descrizione, ha potuto con mano visionare il mio lavoro e il prodotto risultante, collaudandolo personalmente, che ha poi personalmente accettato come prodotto finale.

6. Conclusioni:

6.1 Raggiungimento degli obiettivi ed analisi del prodotto finale

Al termine del periodo di stage sono riuscito a completare 11 requisiti totali, ma comunque completando tutti quelli obbligatori. Questo è stato dovuto sia al fatto che quelli desiderabili erano requisiti ben più complessi rispetto ai restanti, individuati e pensati nel caso tutti gli altri fossero completati prima del previsto e senza troppe difficoltà, sia per il fatto, come già accennato precedentemente, che abbia lavorato come secondo a questo progetto e, una volta avuto il codice e l'applicativo nelle mie mani, abbia trovato e dovuto sistemare dei problemi e bug di estrema importanza lasciati dal mio precedente collega che non ha potuto sistemare per assenza di tempo. Uno dei più importanti era la corretta gestione di più input contemporanei, che è una delle funzionalità principali per cui ENGaming è stata pensata in primo luogo, per cui non poteva essere mal implementata o con errori.

Codice Requisito	Stato
RFO1	Completato
RFO2	Completato
RFO3	Completato
RFO4	Completato
RFO5	Completato
RFO6	Completato
RFO7	Completato
RFO8	Completato
RFO9	Completato
RFO10	Completato
RFD11	Non completato
RFD12	Non completato
RFD13	Non completato
RFO14	Completato

Inoltre, la risoluzione di questi problemi era più urgente rispetto all'aggiunta delle funzionalità desiderabili, perché come accennato prima, ed anche parlandone con il tutor, era più importante e prioritario avere le funzionalità ed i comportamenti di base e di vanto di ENGaming completamente funzionanti invece di aggiungerne altri in maniera parziale e senza avere delle basi solide.

6.2 Conoscenze ed abilità acquisite

Dal momento che la maggior parte delle tecnologie utilizzate sono state per me nuove e mai usate prima, come Angular ed Electron, sicuramente da questo stage ho appreso moltissime nuove informazioni e conoscenze, anche molto interessanti, di cui farò tesoro nel corso degli anni. In particolare, mi ha affascinato ed interessato molto quanto facilmente Electron permetta lo sviluppo di applicazioni contenenti tecnologie web su desktop ed ho trovato notevole il modo in cui esso riesca ad integrare al meglio Angular per la gestione delle varie pagine e componenti, oltre ad esser stato sorpreso da quanto la divisione di Angular in "Componenti" delle varie parti del progetto abbia reso molto facile e semplice lavorarci e permettere loro di integrarsi.

Anche aver utilizzato TypeScript come linguaggio è stata una piacevolissima sorpresa ed esperienza, avendo un background principalmente composto dall'utilizzo di Java e C++, l'utilizzo di TypeScript e la sua similitudine con JavaScript hanno dato una piacevole e stimolante ventata d'aria fresca alla mia esperienza da programmatore.

Un’altra interessante e sorprendente tecnologia utilizzata è sicuramente quella di Node-API e come esse permettano di far interagire facilmente codice C++ in ambito web, rendendo disponibili le sue performance e velocità molto elevate, necessarie per l’elaborazione della grandissima mole di pacchetti ricevuti in input per questa applicazione.

Infine, sono rimasto molto sorpreso ed affascinato da Firebase Google, in quanto è stato facilissimo e velocissimo da integrare al progetto tramite la sua documentazione o forum online, e che oltre al Realtime Database, che ho utilizzato in questo progetto, offre molti altri servizi molto utili gratuitamente, sicuramente sarà uno strumento che non esiterò a riutilizzare in futuro quando ne avrò bisogno.

6.3 Valutazione Personale

Quest’esperienza di stage ha sicuramente fatto crescere moltissimo le mie conoscenze ed abilità come sviluppatore; infatti, non solo mi ha insegnato come funzioni il mondo del lavoro ed una realtà aziendale che non conoscevo prima, ma mi ha anche insegnato come utilizzare tecnologie diverse nello stesso progetto e sfruttarle al meglio, come individuare queste tecnologie e capire come farle comunicare tra di loro in maniera efficiente e funzionale. Le nuove tipologie di problemi e difficoltà che ho incontrato durante il mio percorso hanno anche aumentato le mie capacità di problem solving ed adattamento a situazioni inaspettate, oltre ad aiutarmi a gestire al meglio la pressione delle date di scadenza che dovevo rispettare. Infine, una delle cose sicuramente più importanti e che più ho apprezzato dello stage, è stato sicuramente l’ambiente di lavoro, il sentirmi parte di un gruppo che lavorava in ambiti diversi ma allo stesso dispositivo, l’aiuto ricevuto dai dipendenti dell’azienda ed il tempo passato in ufficio.

Bibliografia

Ambiente di Sviluppo Integrato. url: https://www.redhat.com/it/topics/middleware/what-is-ide
AWS. url: https://aws.amazon.com/it/what-is-aws/?nc1=f_cc
C++. url: https://www.w3schools.com/cpp/cpp_intro.asp
CSS. url: https://www.treccani.it/enciclopedia/web-design_(XXI-Secolo)/
Electron Forge. url: https://www.electronforge.io/
ENSign 11 Signature Pad. url: https://www.euronovategroup.com/solutions/product-map/ensign-11-ensign-nfc-hardware
Firebase Google: https://firebase.google.com
Gmail. url: https://www.google.com/intl/it/gmail/about
Google Meet. url: https://meet.google.com
HTML. url: https://www.treccani.it/enciclopedia/html/
Javascript. url: https://www.treccani.it/enciclopedia/javascript_%28Enciclopedia-della-Matematica%29/
Node-API. url: https://nodejs.org/api/n-api.html#node-api
Telegram. url: https://telegram.org/
Visual Studio Code. url: https://code.visualstudio.com/
WebAssembly. url: https://webassembly.org/
Windows 10. url: https://support.microsoft.com/en-us/windows#WindowsVersion=Windows_10