Università degli Studi di Modena e Reggio Emilia
DIPARTIMENTO DI INGEGNERIA "ENZO FERRARI"

CORSO DI LAUREA TRIENNALE IN

INGEGNERIA INFORMATICA

Progettazione visuale di obiettivi in Minecraft: UltimateAdvancementGenerator

Prova finale di:
Davide Bilardello

Relatore:
Prof. Francesco Guerra

Anno Accademico 2022/2023

Sommario

La tesi si concentra sulla progettazione e implementazione di UltimateAdvancementGenerator, una web app per la creazione di obiettivi personalizzati nel videogioco Minecraft. Il contesto del progetto è legato all'ambiente multiplayer del videogioco e alla possibilità di creare server privati con l'aggiunta di plugin. Così nasce l’idea di semplificare la creazione di obiettivi in-game dal punto di vista grafico.

Nel corso della tesi, verranno esaminati nel dettaglio gli strumenti utilizzati, come sono stati utilizzati e applicati nel contesto dell’app. Come Angular per la creazione dell'interfaccia utente, TailwindCSS per lo stile e il layout, e Node.js con Express.js per lo sviluppo del backend. Jenkins viene impiegato per automatizzare il processo di integrazione continua, garantendo un flusso di sviluppo efficiente e privo di errori.

Indice

Capitolo 1

1 Il progetto, progettazione software e strumenti utilizzati

In questo capitolo viene descritto il progetto e qual è il suo scopo, la sua progettazione e, inoltre, verranno presentati i principali strumenti utilizzati per lo sviluppo della web app.

1.1 Il progetto

UltimateAdvancementGenerator è un web tool che permette di progettare graficamente gli advancements, d'ora in poi chiamati "obiettivi", del videogioco Minecraft, e successivamente scaricare il codice Java compatibile con UltimateAdvancementAPI e SpigotMC.

Minecraft ha una parte multigiocatore e permette la creazione di server privati con contenuti aggiuntivi, detti plugin. SpigotMC è un progetto open-source, scritto in Java, che fornisce un'API per la creazione di plugin per server Minecraft.

UltimateAdvancementAPI è un esempio di plugin che utilizza l'API di SpigotMC. Creato da Davide Bilardello e Francesco Goretti, il plugin aggiunge uno strato software attorno all'API SpigotMC per facilitare la creazione e gestione di obiettivi in gioco, specificando come devono esser visualizzati e come devo esser completati, tramite la scrittura di codice Java. Inoltre, aggiunge caratteristiche inesistenti all'API base data da SpigotMC, come ad esempio un diverso metodo di salvataggio dei progressi o la possibilità di condividere gli obiettivi tra un gruppo di giocatori.

L'obiettivo di UltimateAdvancementGenerator nasce quindi dall'esigenza di progettare prima visivamente, tramite un web tool, l'albero degli obiettivi, scaricare il codice Java ed eseguirlo tramite plugin lavorando assieme a UltimateAdvancementAPI. Lo strumento permette di scrivere meno codice ripetitivo, lasciando, così, solo l'implementazione della logica degli obiettivi da implementare successivamente.

Figura 1 Esempio di albero degli obiettivi di Minecraft creato tramite UltimateAdvancementAPI

1.2 Specifica dei requisiti

1.2.1 Requisiti funzionali

In questa sezione, verranno presentati e dettagliati i requisiti funzionali del software. Questi requisiti definiscono le specifiche funzioni e le operazioni che il software deve essere in grado di eseguire per soddisfare le esigenze degli utenti e dell'app.

Pagina principale che mostra il grafo degli obiettivi

Introduzione	**Attori coinvolti –** App
	Descrizione generale della funzione – Creazione di una pagina principale dove è presente il grafo degli obiettivi del tab selezionato.
Input	**Descrizione –** Chiunque può visualizzare questa pagina ed interagire con il grafo.
Descrizione	**Campi obbligatori -**
	Sequenza di operazioni – Questa è la pagina principale dell'app, di conseguenza viene mostrata a chiunque acceda al sito.
	Risposta ed eventuali anomalie se -
Output	L'utente può navigare nella pagina principale ed interagire con il grafo.

Pagina della selezione dei progetti

Introduzione	**Attori coinvolti** – App, Google Identity provider
	Descrizione generale della funzione – Creazione di una pagina dove l'utente può gestire e selezionare i suoi progetti.
Input	**Descrizione –** Solo chi è autenticato con il Google può visualizzare questa pagina.
Descrizione	**Campi obbligatori –** Il token d'accesso.
	Sequenza di operazioni – Una volta autenticato in automatico, o tramite il pulsante "Projects", verrà mostrata la pagina dei progetti.
	Risposta ed eventuali anomalie se – il token d'accesso non è valido.
Output	L'utente può selezionare un progetto.

Gestione e personalizzazione dei tab

Introduzione	**Attori coinvolti** - Utente
	Descrizione generale della funzione – Possibilità di modificare e personalizzare i tab.
Input	**Descrizione –** L'utente può interagire con la UI e modificare tutti gli aspetti del tab.
Descrizione	**Campi obbligatori -**
	Sequenza di operazioni – L'utente sarà in grado di: o Gestire i tab: creare/eliminare/duplicare un tab. o Personalizzare il tab, in particolare cambiare il suo "namespace", immagine di sfondo e le due proprietà "showToPlayers" "grantRootAdvancement".
	Risposta ed eventuali anomalie se -
Output	Personalizzazione dei tab.

Gestione e personalizzazione degli obiettivi

Introduzione	**Attori coinvolti –** Utente
	Descrizione generale della funzione – Possibilità di modificare e personalizzare gli obiettivi.
Input	**Descrizione –** L'utente può interagire con la UI e modificare tutti gli aspetti di un obiettivo.
Descrizione	**Campi obbligatori -**
	Sequenza di operazioni – L'utente sarà in grado di: o Gestire gli obiettivi: creare/eliminare/duplicare un obiettivo. o Personalizzare un obiettivo, in particolare, la sua icona, la forma, il titolo, la descrizione, le sue coordinate.
	Risposta ed eventuali anomalie se -
Output	Personalizzazione degli obiettivi.

Download del codice java

Introduzione	**Attori coinvolti** - App
	Descrizione generale della funzione – Download delle classi Java che andranno a visualizzare nel gioco il grafo progettato tramite la web app.
Input	**Descrizione –** Premere il pulsante della sezione “Download Java code”.
Descrizione	**Campi obbligatori –** La lista dei tab e il package del progetto Java.
	Sequenza di operazioni – Una volta inserito il package del progetto Java, l’utente potrà scaricare un archivio zip premendo l’apposito bottone.
	Risposta ed eventuali anomalie se – il package non è inserito.
Output	Archivio zip con le classi Java corrispondenti ai tab ed i loro obiettivi.

Login con Google

Introduzione	**Attori coinvolti** – App, Google Identity provider, Utente
	Descrizione generale della funzione – Possibilità di accedere tramite Google e rilasciare il permesso per scrivere sul Google Drive dell’utente.
Input	**Descrizione –** Qualunque utente non autenticato può effettuare il login con Google.
Descrizione	**Campi obbligatori** – Account Google
	Sequenza di operazioni – Una volta premuto il bottone per effettuare il login, l’utente dovrà rilasciare l’autorizzazione per l’utilizzo di Google Drive.
	Risposta ed eventuali anomalie se – l’autenticazione non va a buon fine.
Output	L’utente può navigare nella pagina dei progetti ed eseguire azioni addizionali una volta autenticato.

Sincronizzazione con il cloud Google Drive

Introduzione	**Attori coinvolti** – App, Utente e Google Identity provider
	Descrizione generale della funzione – Salvare e richiedere file di salvataggio.
Input	**Descrizione –** L'utente deve essere autenticato.
Descrizione	**Campi obbligatori –** Nome del file e token d'accesso.
	Sequenza di operazioni – Una volta selezionato il progetto, il file di salvataggio viene scaricato e sistemato per mostrare i tab. Se ci saranno delle modifiche, l'utente potrà salvarle premendo l'apposito pulsante.
	Risposta ed eventuali anomalie se – il token non è valido.
Output	L'utente può salvare e scaricare i salvataggi dei suoi progetti.

1.2.2 Requisiti non funzionali

In questa sezione, saranno delineati e descritti i requisiti non funzionali del software. Questi requisiti riguardano gli aspetti del sistema che non sono direttamente legati alle specifiche funzioni o operazioni.

Performance	
Descrizione	Grazie all'utilizzo del server-side rendering (SSR), le prestazioni della web app sono notevolmente ottimizzate. Questa metodologia consente di trasferire una parte delle operazioni di rendering al server, alleggerendo il carico di lavoro del browser. Questa ottimizzazione si traduce in un notevole beneficio per l'esperienza utente, rendendo l'applicazione perfettamente fruibile anche su dispositivi con minori capacità di elaborazione, come ad esempio i dispositivi mobili. Inoltre, L'uso del SSR accelera il tempo di caricamento iniziale delle pagine.

Usabilità	
Descrizione	L'app è stata progettata utilizzando Figma, prestando particolare attenzione all'usabilità e all'esperienza utente. L'interfaccia utente è stata progettata con l'obiettivo di rimanere semplici, enfatizzando segnali visivi mediante simboli e colori per segnalare eventuali problematiche. I pulsanti sono stati progettati in modo intuitivo, chiarendo il loro scopo e semplificando l'intera esperienza di navigazione. L'attenzione al design e alla chiarezza visiva mira a garantire un utilizzo fluido e senza intoppi, fornendo agli utenti un ambiente accattivante e user-friendly.

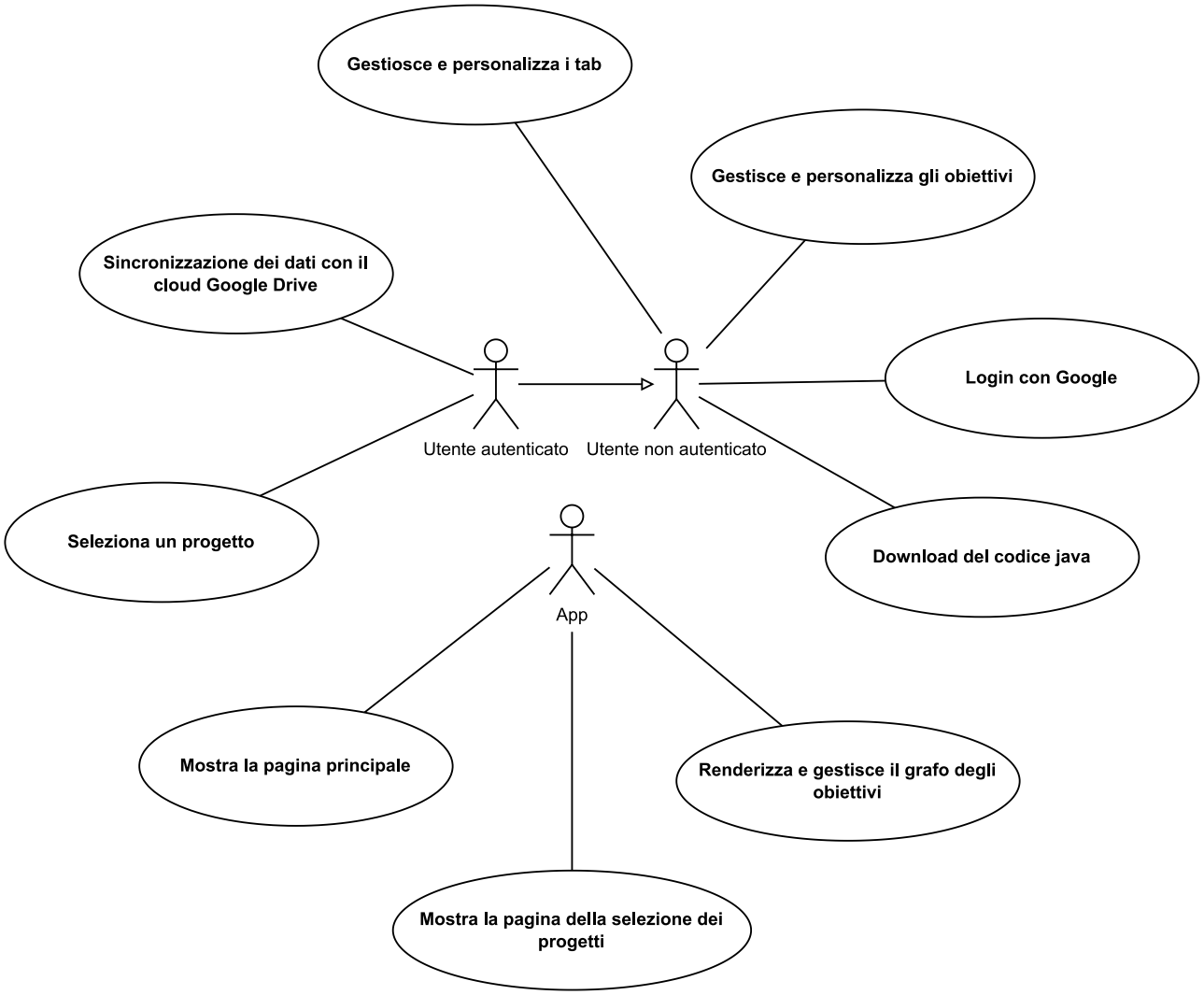
Compatibilità	
Descrizione	L'applicazione è stata sviluppata per offrire una totale compatibilità su diverse piattaforme web, assicurando un'esperienza uniforme su qualsiasi dispositivo dotato di un browser. Indipendentemente dall'hardware o dal sistema operativo in uso, l'applicazione è progettata per adattarsi in modo dinamico e garantire funzionalità ottimali.

Manutenzione e aggiornamento	
Descrizione	L'architettura dell'applicazione è stata concepita con una prospettiva di lungo termine, garantendo la sua adattabilità e facilitando gli aggiornamenti futuri. Considerando l'evoluzione costante del gioco Minecraft e della libreria UltimateAdvancementAPI, la web app sarà sempre allineata con le più recenti versioni e novità. Inoltre, l'applicazione consente l'integrazione agevole di nuove funzionalità in risposta alle esigenze degli utenti e all'evolversi del contesto di gioco.

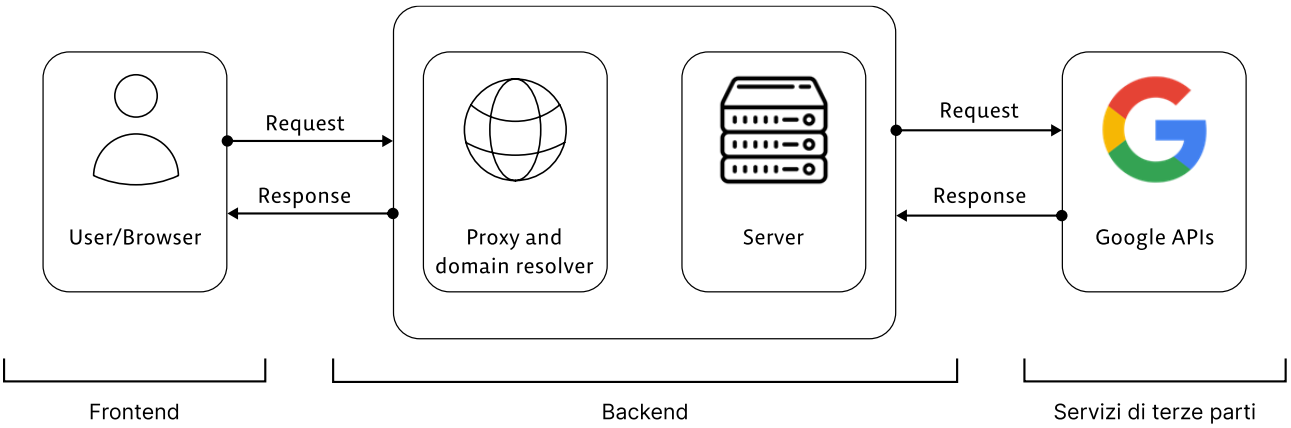
Integrità dei dati e Conservazione dei dati	
Descrizione	I dati generati dall'applicazione vengono salvati sul Google Drive dell'utente autenticato, garantendo in tal modo sia l'integrità che la sicurezza dei dati. Questa implementazione non solo contribuisce a preservare la coerenza e l'affidabilità delle informazioni memorizzate, ma assicura anche una gestione sicura e a lungo termine delle stesse. La scelta di utilizzare Google Drive come archivio dei dati offre un ulteriore strato di sicurezza, sfruttando le robuste misure di sicurezza di Google.

1.2.3 Diagramma dei casi d’uso

Verranno presentati i casi d'uso mediante un diagramma UML. Il diagramma illustra le interazioni tra gli attori (“*Utente autenticato*”, “*Utente non autenticato*”, “*App*”) e le funzionalità della web app. I casi d'uso rappresentano, quindi, le diverse funzionalità offerte dal sistema e forniscono una panoramica chiara delle attività e dei processi che coinvolgono gli utenti e il software.



1.3 Architettura dell'applicazione



L'architettura utilizzata dalla web app è di tipo *client-server*. Rappresentato da un modello di distribuzione del software in cui le responsabilità di elaborazione sono divise generalmente divise tra il client (il dispositivo dell'utente) e il server (un computer remoto):

- Il *frontend* di un'applicazione web rappresenta la componente lato client che consente agli utenti di interagire con il servizio backend direttamente attraverso il loro browser. All'interno del browser, il codice frontend gestisce le richieste degli utenti e presenta le informazioni richieste. In questo contesto, la progettazione UI/UX assume un ruolo centrale, insieme a elementi come dashboard informative, notifiche, il layout della pagina e tutti gli elementi interattivi che concorrono a definire l'esperienza utente complessiva.
- Il *server* gestisce uno o più applicazioni web e servizi. Utilizzando l'HTTP, il server interpreta le richieste degli utenti provenienti attraverso un browser, elabora tali richieste, per poi consegnare il contenuto richiesto all'utente finale. La sua responsabilità si estende al recupero dei dati dal database, all'esecuzione di calcoli complessi e alla generazione dinamica delle pagine web. Va notato che questo server backend opera dietro un proxy e un resolver DNS, migliorando la sicurezza, la gestione del traffico e ottimizzando le prestazioni garantendo una maggiore stabilità del servizio.
- Le *API di Google* offrono la possibilità di potenziare i servizi forniti dall'applicazione, inclusi un sistema di autenticazione sicuro mediante OAuth 2.0 e l'integrazione con Google Drive per il salvataggio dei dati.

1.4 Angular

> “Angular is a web framework that empowers developers to build fast, reliable applications.
>
> Maintained by a dedicated team at Google, Angular provides a broad suite of tools, APIs, and libraries to simplify and streamline your development workflow. Angular gives you a solid platform on which to build fast, reliable applications that scale with both the size of your team and the size of your codebase.”
>
> - Dal sito uffiale angular.dev

Angular fornisce funzionalità per facilitare lo sviluppo web, come ad esempio, routing, gestione delle dipendenze, creazione di forms, animazioni, testing…

In particolare, offre la funzionalità per la creazione di componenti isolati e riutilizzabili, consentendo agli sviluppatori di suddividere l'interfaccia utente in componenti indipendenti e modulari.

I componenti in Angular sono identificati da quattro file:

- HTML file che contiene la struttura del componente.
- CSS file che contiene lo stile del componente.
- Typescript file che contiene la logica del componente e come interagisce con gli altri.
- Typescript file per il testing.

Mentre Angular SSR (Server-side rendering) abilita il rendering lato server. È un processo che prevede il rendering delle pagine sul server, con generazione del contenuto HTML. Una volta che il contenuto HTML viene consegnato al browser, Angular inizializza l'applicazione e utilizza i dati contenuti nell'HTML.

I principali vantaggi dell'SSR rispetto al client-side rendering sono:

- Migliori prestazioni: L'SSR può migliorare le prestazioni delle applicazioni web fornendo al client un HTML completamente renderizzato, che il browser può analizzare e visualizzare prima ancora di scaricare il JavaScript dell'applicazione. Questo può essere particolarmente vantaggioso per gli utenti che utilizzano connessioni a bassa larghezza di banda o dispositivi mobili.
- Migliore SEO: L'SSR può migliorare l'ottimizzazione per i motori di ricerca (SEO) delle applicazioni web, facilitando l'indicizzazione del contenuto dell'applicazione da parte dei motori di ricerca.

1.5 CI/CD, Docker e Jenkins

Nell' ingegneria del software. CI/CD è la combinazione di Continuous Integration, integrazione continua, e Continuous Deployment, consegna continua.

Integrazione continua
L'integrazione continua (CI) è una pratica fondamentale nei contesti di sviluppo software basati su sistemi di controllo di versione, come GitHub. Questo approccio prevede l'allineamento frequente degli ambienti di lavoro (branch) degli sviluppatori con l'ambiente condiviso (repository), garantendo una collaborazione fluida e continua.

Il CI si basa sull'idea che siano già stati preparati test automatici e gli sviluppatori eseguono questi test immediatamente prima di rilasciare le loro contribuzioni nell'ambiente condiviso. Questa metodologia assicura che le modifiche apportate non introducano errori nel software esistente, contribuendo così a mantenere la stabilità del sistema nel tempo.

Questo paradigma è spesso applicato in ambienti dotati di sistemi di build automatica e/o esecuzione automatica di test, come ad esempio utilizzando Jenkins. Questi strumenti automatizzano i processi di compilazione e test, rendendo il flusso di sviluppo più efficiente e riducendo il rischio di errori.

I principi su cui si basa questa pratica sono:

- Mantenere un repository del codice sorgente.
- Automatizzare le build.
- Rendere le build auto-testanti.
- Tutti eseguono commit alla baseline tutti i giorni.
- Ogni commit far partire una build.
- Fare in modo che le build siano veloci.
- Eseguire i test in un clone dell'ambiente di produzione.
- Fare in modo che sia facile prendere le ultime versioni dei pacchetti.
- Ognuno può vedere i risultati delle ultime build.
- Automatizzare i rilasci.

Consegna continua
La consegna continua (CD) è un approccio dinamico nel quale i team sviluppano software in cicli brevi, garantendo che il prodotto possa essere rilasciato in modo affidabile in qualsiasi momento attraverso una pipeline automatizzata, simile all'ambiente di produzione. L'obiettivo primario è costruire, testare e rilasciare il software con maggiore velocità e frequenza, riducendo contemporaneamente costi, tempi e rischi associati all'implementazione di modifiche. Questo approccio facilita l'introduzione di aggiornamenti incrementali alle applicazioni in produzione.

La CD segue una pipeline di distribuzione articolata in tre fasi chiave: visibilità, feedback e distribuzione continua.

*Visibilità***:** Tutti gli aspetti del sistema di distribuzione, compresi building, deploy, test e rilascio, sono resi visibili a tutti i membri del team. Questa trasparenza promuove la collaborazione, consentendo a ciascun membro di avere una visione chiara del processo e contribuire efficacemente al miglioramento continuo.

*Feedback***:** I membri del team ricevono tempestivamente segnalazioni sugli eventuali problemi appena si verificano. Questa prontezza nel feedback permette al team di risolvere gli inconvenienti in modo rapido ed efficiente, migliorando la qualità complessiva del software.

*Distribuzione continua***:** Grazie a un processo completamente automatizzato, è possibile distribuire e rilasciare qualsiasi versione del software in qualsiasi ambiente senza intervento manuale. Questa automazione non solo riduce il margine di errore, ma garantisce anche una distribuzione rapida e affidabile del software, consentendo al team di mantenere un flusso di lavoro continuo e efficiente.

Docker

Docker è una potente piattaforma software progettata per agevolare la creazione, il testing e la distribuzione rapida di applicazioni. La sua caratteristica distintiva è l'utilizzo di unità standardizzate denominate *container*, che racchiudono tutto ciò che è necessario per l'esecuzione corretta di un'applicazione, compresi librerie, strumenti di sistema, codice e runtime.

In termini pratici, Docker offre una modalità uniforme per eseguire il codice, agendo come un sistema operativo per container. Analogamente a come le macchine virtuali virtualizzano i server hardware, i container di Docker virtualizzano il sistema operativo di un server. Questa virtualizzazione semplifica il processo di distribuzione del codice, standardizza il funzionamento delle applicazioni, ottimizza il trasferimento del codice e consente un uso più efficiente delle risorse, contribuendo a risparmiare denaro.

Docker semplifica la realizzazione e l'esecuzione di architetture di microservizi distribuite, agevola la distribuzione del codice attraverso pipeline di integrazione e distribuzione continue standardizzate, facilita la creazione di sistemi di elaborazione dati altamente scalabili e supporta la creazione di piattaforme completamente gestite per gli sviluppatori.

La sintassi semplice di Docker facilita il controllo delle risorse, e la sua diffusione implica la presenza di un ricco ecosistema di strumenti e applicazioni pronte all'uso.

Jenkins

Jenkins è un server open source per CI/CD, è implementato in Java e offre una versatilità eccezionale attraverso la sua compatibilità multipiattaforma, fornendo pacchetti dedicati per Linux, Mac OS X e Windows.

La forza di Jenkins risiede nella sua capacità di automatizzare le varie fasi del ciclo di vita del software, coprendo dall'assemblaggio al testing fino alla distribuzione. Le attività possono essere programmate per eseguirsi in risposta a trigger specifici, come un commit nel repository del codice. Questo permette di avviare automaticamente test e analizzare i risultati, ottenendo un feedback immediato sulla qualità del codice.

Jenkins ottiene ulteriore potenza grazie a un vasto ecosistema di plugin sviluppati e condivisi dalla comunità di utenti e sviluppatori. Questi plugin estendono le funzionalità di base di Jenkins, offrendo una flessibilità e un adattamento a diverse esigenze progettuali.

Le pipeline di Jenkins sono definite nel file chiamato Jenkinsfile, scritto in un linguaggio specifico chiamato "Pipeline DSL" (Domain-Specific Language). Questo file può essere integrato direttamente nel repository del codice sorgente del progetto, consentendo una gestione e versionamento della configurazione della pipeline insieme al codice stesso. Questa integrazione agevola la coerenza e la tracciabilità tra la configurazione della pipeline e lo sviluppo del codice, migliorando l'efficienza complessiva del processo di sviluppo e integrazione.

1.6 Node.js e Express.js

Node.js è un ambiente runtime di JavaScript, è un progetto open-source che opera su diverse piattaforme, rendendo accessibile la sua potenza a sviluppatori di tutto il mondo. La sua natura cross-platform consente la creazione di applicazioni server-side e di networking ad alte prestazioni e scalabili su una vasta gamma di sistemi operativi.

La scelta di adottare un'architettura I/O event-driven e non bloccante consente a Node.js di gestire in modo ottimale le richieste, rendendolo particolarmente indicato per applicazioni real-time che richiedono una risposta istantanea agli eventi.

Express.js è un framework di backend per Node.js. Si inserisce in questo contesto, offrendo rapidità e solidità nella creazione di applicazioni scalabili. Il suo sistema di routing strutturato e le funzioni semplificate facilitano lo sviluppo e l'estensione del framework. La versatilità di Express.js emerge nella sua capacità di adattarsi a una varietà di scenari applicativi, permettendo agli sviluppatori di personalizzare e potenziare il framework per rispondere in modo ottimale alle esigenze specifiche di ogni progetto.

1.7 Altri strumenti

1.7.1 TailwindCSS

Tailwind CSS è un framework CSS open source. La caratteristica principale di questa libreria è che, a differenza di altri framework CSS, non fornisce una serie di classi predefinite per elementi come pulsanti o tabelle. Al contrario, crea un elenco di classi CSS "di utilità" che possono essere utilizzate per creare lo stile di ogni elemento, mescolandole tra loro.

Inoltre, ha un file di configurazione dove è possibile estendere le potenzialità di Tailwind, e facilita enormemente la creazione di pagine web che si adattano a qualsiasi dimensione dello schermo.

1.7.2 Figma

Figma è un editor di grafica vettoriale e uno strumento di prototipazione. È principalmente basato sul Web, con funzionalità offline aggiuntive abilitate dalle applicazioni desktop per macOS e Windows. Il set di funzionalità di Figma si concentra sull'uso nell'interfaccia utente e sulla progettazione dell'esperienza utente, con particolare attenzione alla collaborazione in tempo reale.

1.7.3 Git e GitHub

Git è un sistema di controllo di versione distribuito (DVCS) che consente di tenere traccia delle modifiche apportate al codice sorgente di un progetto software. Git è uno strumento gratuito e open source che è diventato lo standard de facto per il controllo di versione del software.

Utilizza un modello di repository distribuito, che significa che ogni sviluppatore ha una copia completa del repository. Questo modello consente agli sviluppatori di lavorare in modo indipendente e di collaborare in modo efficace.

Git offre una serie di funzionalità che lo rendono un potente strumento per il controllo di versione del software, tra cui:

- *Tracciamento delle modifiche*: tiene traccia di tutte le modifiche apportate al codice sorgente di un progetto. Questo può essere utile per ripristinare versioni precedenti del codice o per identificare la fonte di un bug.
- *Collaborazione*: consente agli sviluppatori di lavorare in modo indipendente e di collaborare. Questo può essere utile per i progetti di grandi dimensioni o per i progetti che coinvolgono pochi sviluppatori.

GitHub principalmente è un servizio di hosting di repository Git, però inoltre consente di tenere traccia di Issue, richieste di tipo “Pull”, strumenti di CI/CD e molto altro.

Capitolo 2

2 Frontend con Angular

Questo capitolo descrive l'intero sviluppo del frontend della web app, dalla sua progettazione alla sua realizzazione.

La web app, prima di venir interamente sviluppata, è stata progettata graficamente tramite l'applicazione Figma. È un passaggio fondamentale nel processo di sviluppo di una app, perché permette di ben definire tutte le funzionalità e come queste verranno visualizzate tramite la UI.

2.1 Progetto template su git UIElements

Il progetto Github UIElements è un progetto Angular contenente tutti gli elementi grafici di base, come ad esempio testi, bottoni e altro; con lo stile progettato precedentemente su Figma. È un progetto template perché verrà usato come base per tutti i siti web che vogliono mantenere lo stesso stile grafico. Ogni elemento grafico di base è tradotto come componente Angular, permettendo il riutilizzo dei componenti all'interno dei siti web derivati.

Di seguito verranno presentati i principali componenti:

2.1.1 TypographyComponent

Figura 2 Esempio di TypographyComponent

Descrizione

Il componente TypographyComponent gestisce lo stile del testo. In base ai valori presi in input sceglie le classi Tailwind da applicare. È progettato per standardizzare e semplificare l'aspetto visivo del testo in vari contesti, garantendo uniformità e coerenza nel design complessivo.

Valori di input

- *ng-content*: elemento HTML a cui sarà applicato lo stile. Esempio: una stringa o un div.

- *typeText*: è richiesto un valore dalla enumerazione TextType che specifica il contesto e la dimensione dell'utilizzo. Esempio: TextType.h, TextType.p.
- *fontText*: è richiesto un valore dalla enumerazione TextFont che specifica il font grafico. Esempio: TextFont.BespokeRegular, TextFont.Kola.

2.1.2 SelectComponent

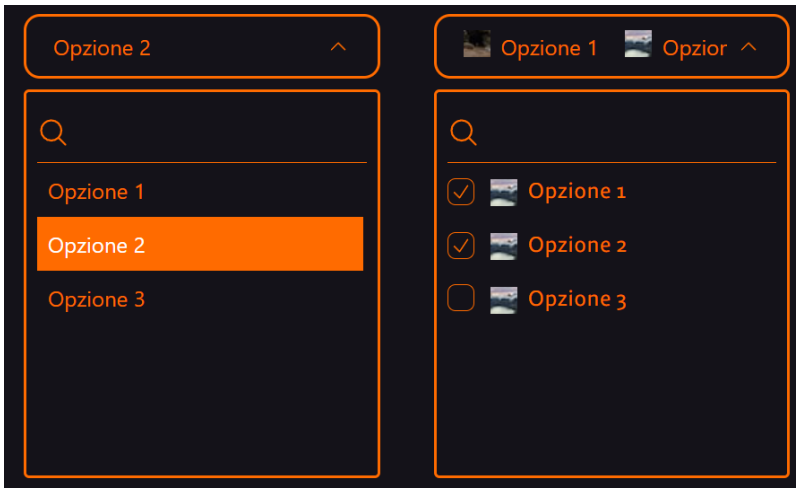


Figura 3 Esempio di SelectComponent

Descrizione

Un SelectComponent permette agli utenti di selezionare un'opzione da un insieme predefinito di scelte. Esiste in due versioni: a selezione singola e a selezione multipla. Quando il menu a discesa è chiuso, il componente mostra quale opzione è selezionata. Mentre quando il menu a discesa è aperto elenca tutte le opzioni disponibili e permette la ricerca tra le stesse.

Se le opzioni da mostrare fossero troppe la web app potrebbe appesantirsi. Quindi, il componente mostra una lista virtuale.

Valori di input

- *options*: lista delle opzioni.
- *selected*: lista degli elementi selezionati. Se il componente è a singola selezione, verrà utilizzato solo il primo elemento della lista.
- *hasImage*: se sono presenti le immagini nelle opzioni.
- *isMultiselector*: se il componente è a multipla selezione.
- *multiselectorLimit*: limite di opzioni selezionati per il componente a multipla selezione.
- *disabledOptions*: le opzioni disabilitate e quindi non selezionabili.
- *isDisabled*: se il componente è disabilitato e quindi non interagibile.

2.1.3 InputComponent

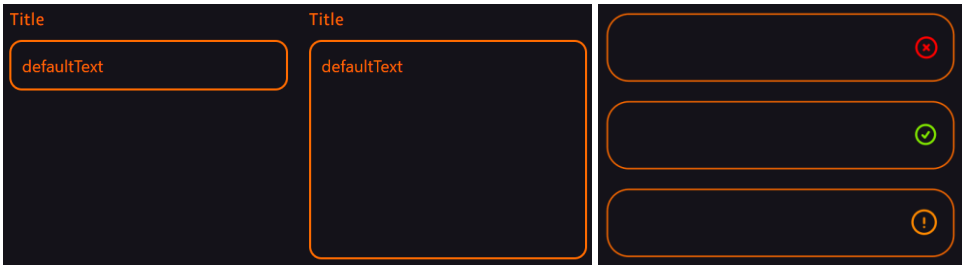


Figura 4 Esempio di InputComponent

Descrizione

Il componente InputComponent consentire agli utenti di inserire dati attraverso l'interazione con la tastiera. È di due tipi: a linea singola o a linea multipla.

Valori di input

- *text*: il contenuto del campo di testo.
- *title*: il titolo presente sopra il componente.
- *isTextArea*: se il componente è a linea multipla.
- *inputType:* tipo di dato accettato dal componente. Esempio: “text”, “number”.
- *isValid:* se il componente deve mostrare l’icona di approvazione.
- *isWarning:* se il componente deve mostrare l’icona di attenzione.
- *isError:* se il componente deve mostrare l’icona di errore.
- *isDisabled:* se il componente è disabilitato e quindi non interagibile.

2.1.4 CheckBoxComponent

Figura 5 Esempio di CheckBoxComponent

Descrizione

Il componente CheckBoxComponent è progettato per consentire agli utenti di fare una scelta binaria attraverso l'interazione con una casella di controllo. Questo componente è comunemente utilizzato per consentire agli utenti di selezionare o deselezionare una determinata opzione o impostazione.

Questo componente viene utilizzato all’interno del SelectComponent nella sua versione a selezione multipla.

Valori di input

- *isChecked*: se il componente è selezionato.
- *canSelect*: se il componente può passare dallo stato di “non selezionato” a “selezionato”.
- *canDeselect*: se il componente può passare dallo stato di “selezionato” a “non selezionato”.

- *isDisabled:* se il componente è disabilitato e quindi non interagibile.

2.1.5 ButtonComponent

Figura 6 Esempio di ButtonComponent

Descrizione

Il componente ButtonComponent si presenta come un elemento cliccabile che reagisce al click dell'utente.

Valori di input

- *ng-content*: il contenuto del bottone.
- *isDisabled:* se il componente è disabilitato e quindi non interagibile.
- *rounded_class*: stringa che equivale alla classe che si vuole applicare con Tailwind per indicare il raggio della forma del bottone.

2.1.6 RadioButtonsComponent

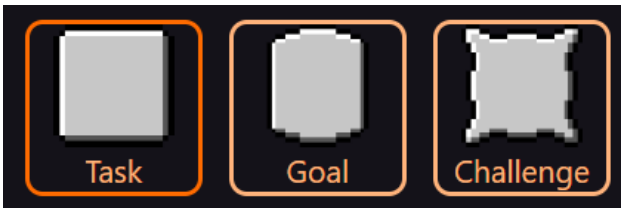


Figura 7 Esempio di RadioButtonsComponent

Descrizione

Il componente RadioButtonsComponent consente agli utenti di fare una scelta esclusiva tra un insieme di opzioni.

Valori di input

- *options*: lista delle opzioni.
- *selected*: elemento selezionato.
- *isDisabled:* se il componente è disabilitato e quindi non interagibile.

2.2 Progetto principale e i suoi componenti

Il progetto UltimateAdvancementGenerator su Github è una copia del progetto UIElements. Sono stati utilizzati gli elementi UI di base per creare l'intera web app e sono state aggiunte le caratteristiche necessarie per la gestione del grafo degli obiettivi.

2.2.1 Classi principali

Tab, Advancement e FormattedText sono astrazioni che rappresentano oggetti presenti nel gioco di Minecraft.

Tab

I tab sono raggruppamenti di obiettivi e ogni tab ha il suo grafo. La definizione della classe è la seguente:

```
export interface Tab {

  //array degli obiettivi, poi visualizzati come grafo
  advancements: Advancement[]

  //identificativo univoco del tab
  namespace: string

  //sfondo del tab
  background: Option

  //se il tab deve essere mostrato automaticamente ai giocatori che entrano nel
  //gioco
  showToPlayers: boolean

  //se la radice del grafo deve essere marcato come "completato" automaticamente
  //ai giocatori che entrano nel gioco
  grantRootAdvancement: boolean

  //mappa <Cell, Advancement>. La classe Cell indica un nodo all'interno di un
  //grafo mxGraph
  advancementsMap: Map<Cell, Advancement>

}
```

Advancement

Advancement rappresenta un obiettivo del gioco.

```
export interface Advancement {

    //nodo di maxGraph che rappresenta l'obiettivo
    node: Cell | null;

    //titolo dell'obeittivo
    title: string;

    //descrizione dell'obiettivo
```

```
    description: string;

    //coordinata x dell'obiettivo
    x: number;

    //coordinata y dell'obiettivo
    y: number;

    //progressione massima dell'obiettivo
    maxProgression: number

    //se l'obiettivo deve mostrare un avviso nella UI di gioco una
    //volta completato
    showToast: boolean

    //se l'obiettivo deve mostrare un avviso nella chat di gioco una volta
    //completato
    announceChat: boolean

    //l'icona dell'obiettivo
    icon: Option

    //la cornice dell'obiettivo
    frame: Option

    //tipologia della visibilità dell'obiettivo
    visibility: Option

    //tipo dell'obiettivo
    type: Option

    //obiettivo a cui è collegato lo stesso, "null" se è la radice del grafo
    parent: Advancement | null

    //identificativo univoco dell'obiettivo all'interno del tab
    key: string

    //il tab dell'obiettivo
    tab: Tab

}
```

FormattedText

Titoli e descrizioni degli obiettivi possono essere suddivisi in frammenti di testo con diverse caratteristiche. L'utente ha la possibilità di inserire stringhe che successivamente verranno sistemate per mostrare il testo come se fosse visualizzato nel gioco. Queste stringhe verranno frammentate dal divisore "§?", dove il punto interrogativo identifica una caratteristica del frammento, come il suo colore o tipo di testo (rosso, grassetto, italic…).

Esempio: "§cTestoRosso §aTestoVerde" verrà mostrato con il primo frammento impostato con il colore rosso e il secondo frammento impostato con il colore verde.

```
Export interface FormattedText {

  //testo del frammento
  text: string
```

```
  //colore del frammento
  color: string

  //colore predefinito del frammento, la variabile "colore" viene impostata
  //inizialmente con questo valore
  defaultColor: string

  //se il testo è in grassetto
  isBold: boolean

  //se il testo è in corsivo
  isItalic: boolean

  //se il testo è sottolineato
  isUnderline: boolean

  //se il testo ha una barra in mezzo
  isStrike: boolean

  //se il testo è sostituito con caratteri casuali
  isObfuscated: boolean

  //frammento precedente, potrebbe influenzare le componenti della classe
  previousText: FormattedText | null

}
```

2.2.2 Gestione del grafo con maxGrap

"maxGraph is a fully client side JavaScript diagramming library"

- Descrizione di maxGraph sul progetto Github

maxGraph è una libreria TypeScript che permette di mostrare e interagire con diagrammi vettoriali.

A grandi linee, fornisce:

- *Nodi*, o vertici, di solito rappresentati come rettangoli.
- *Lati* che possono essere linee o frecce che normalmente collegano due nodi tra di loro.
- *Elementi sovrapposti*, ovvero elementi grafici aggiuntivi posizionati al di sopra di nodi o lati.
- Gestione dello sfondo della sezione del grafo.
- Algoritmi automatici per la gestione della disposizione del grafo, come ingrandimento, spostamento e centramento.
- Gestione interna ad eventi per interazioni con gli elementi del grafo, come selezione e hover.

Il progetto è in alpha ed è ancora in sviluppo.

È il successore di mxGraph, libreria JavaScript per la creazione di diagrammi ormai non più mantenuta e aggiornata. È alla base del software web draw.io.

Problemi con l'integrazione di maxGraph

Problema 1

La libreria è totalmente client-side, ovvero che viene generata e gestita dal browser e le sue risorse. Va, però, in conflitto con Angular SSR: il server non ha conoscenza dei riferimenti che un browser possiede.

Soluzione:

- Mostrare il grafo degli obiettivi appena il browser deve mostrare la pagina tramite il metodo afterNextRender().
- Importare le classi della libreria solamente quando sono necessari.

```
afterNextRender( async () => {
  //codice che viene eseguito solo sul browser

  //import del gestore del grafo
  const {GraphManager} = await import("./classes/graph/GraphManager");

  //codice che utilizza il gestore
});
```

Problema 2

Importare componenti o utilizzare eventi asincroni, non sono totalmente compatibili con l'aggiornamento automatico di Angular e di conseguenza la UI del sito non viene aggiornata.

Soluzione:

Utilizzare NgZone, classe di utilità per aggiornare la UI del sito dopo eventi asincroni.

```
ngZone.run( async () => {
  //alla fine di questo codice asincrono aggiorna la UI
})
```

2.2.3 Componenti principali

HeaderComponent

Figura 8 HeaderComponent

HeaderComponent gestisce la parte di UI superiore della web app.

Sono presenti i seguenti elementi:

- Il logo dell'app.
- Bottone per tornare nella sezione "Progetti". Sarà abilitato solo per gli utenti collegati con un account Google, visto che potranno salvare diversi salvataggi come progetti.

- Simbolo che indica lo stato del salvataggio cloud su Google Drive. Abilitato solo per gli utenti collegati con un account Google.
- Bottone per scaricare il codice Java del progetto.
- Bottone per modificare le impostazioni.
- Bottone per eseguire il login e controllare le informazioni dell'account.

TabSettingsComponent

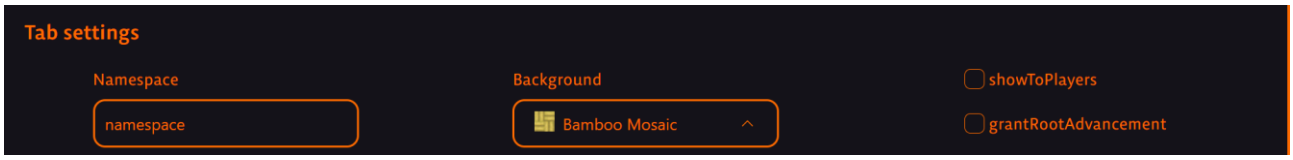


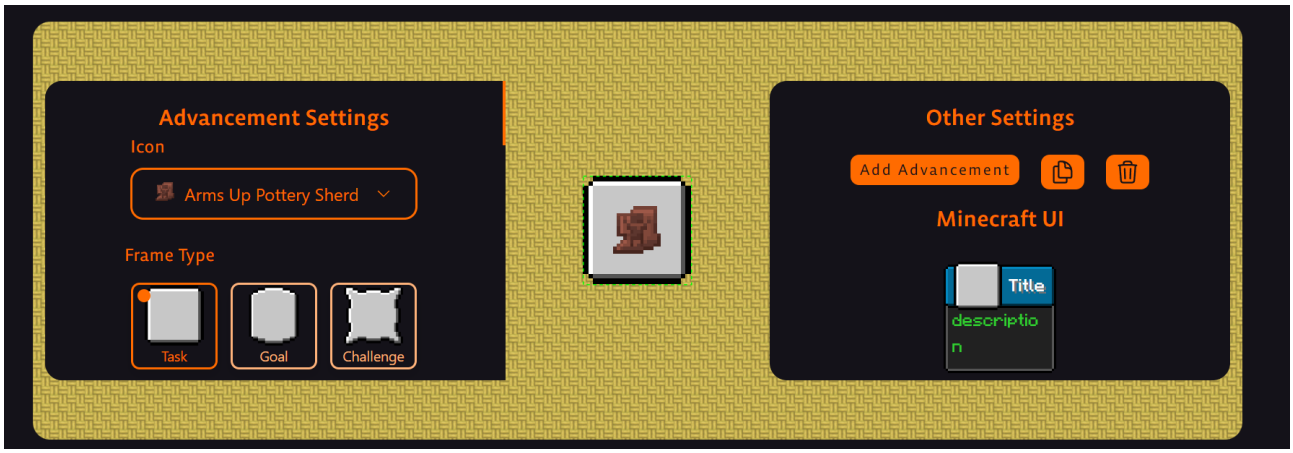
Figura 9 TabSettingsComponent

TabSettingsComponent gestisce la parte delle impostazioni del tab selezionato.

Sono presenti i seguenti elementi:

- Un campo per immettere l'identificativo univoco, detto "Namespace".
- Un selettore per lo sfondo.
- 2 CheckBoxComponent, uno indica se il tab va mostrato automaticamente ad un giocatore che entra nel gioco, mentre l'altro se l'obiettivo radice deve essere marcato come "completato" automaticamente ai giocatori che entrano nel gioco.

BoardComponent



BoardComponent gestisce il grafo degli obiettivi. Permette la visualizzazione e la navigazione tra i nodi. Inizialmente viene mostrato un solo obiettivo, detto "RootAdvancement" o obiettivo radice. Una volta selezionato un obiettivo, questo si mostrerà al centro dello schermo con altre due sezioni a comparsa sui lati:

La sezione di sinistra contiene:

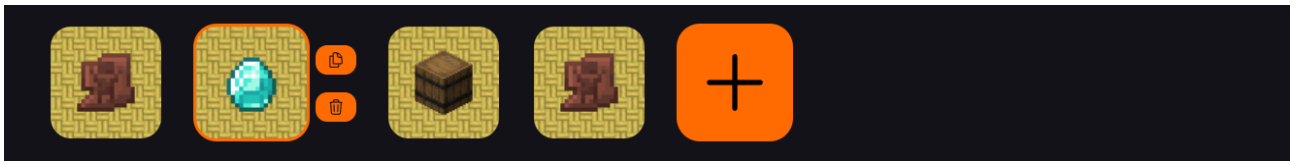
- Selettore per impostare l'icona dell'obiettivo.
- Selettore per impostare la cornice dell'obiettivo.
- Campo d'inserimento per l'identificativo univoco dell'obiettivo (*key*).

- Campo d’inserimento per la massima progressione dell’obiettivo.
- Campo d’inserimento per la coordinata x dell’obiettivo.
- Campo d’inserimento per la coordinata y dell’obiettivo.
- Campo d’inserimento per la visibilità dell’obiettivo.
- Selettore binario se l'obiettivo deve mostrare un avviso nella UI di gioco una volta completato.
- Selettore binario se l'obiettivo deve mostrare un avviso nella chat di gioco una volta completato.
- Selettore per il tipo di obiettivo.
- Selettore per il genitore dell’obiettivo.
- Campo d’inserimento per il titolo dell’obiettivo.
- Campo d’inserimento per la descrizione dell’obiettivo.

La sezione di destra contiene:

- Bottone per aggiungere un obiettivo figlio da collegare con il selezionato.
- Bottone per duplicare l’obiettivo.
- Bottone eliminare l’obiettivo.
- Visualizzazione simile a quella nel gioco del titolo e della descrizione dell’obiettivo.

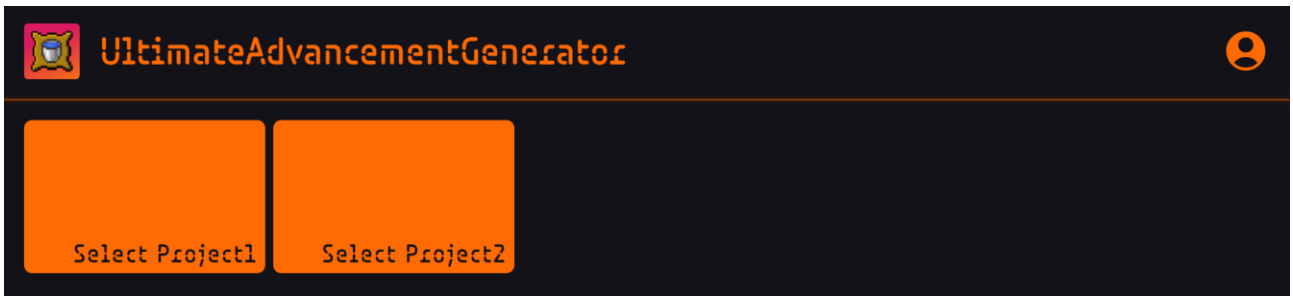
TabSelectorComponent



TabSelectorComponent gestisce i tab. Nello specifico permette:

- Aggiungere un nuovo tab.
- Selezionare un tab.
- Eliminare un tab.
- Duplicare un tab.

ProjectsPageComponent



La pagina permette di selezionare un progetto. Una volta selezionato, se precedentemente erano stati salvati dei dati relativi ai tab, questi vengono richiesti e scaricati. Dopo aver selezionato un progetto, si verrà reindirizzati alla pagina principale ed il grafo verrà mostrato. Prima di accedere a questa pagina, il client verifica se l’utente è autenticato.

2.2.4 Servizi

In Angular, un servizio è una classe che fornisce una particolare funzionalità per l'applicazione e sono utilizzati per organizzare e condividere logica e dati che devono essere accessibili in tutta l'applicazione. Questi servizi possono essere iniettati all'interno dei componenti per fornire funzionalità riutilizzabili e mantenere la separazione delle responsabilità.

AdvancementService

AdvancementService dà un punto unico d'accesso alle informazioni che devono essere condivise tra tutti i componenti dell'applicazione. In particolare, gestisce:

- Lista dei tab.
- Tab e obiettivo selezionato in quel momento dall'utente.
- Liste di Option, come ad esempio quella che contiene tutte le icone usabili da un obiettivo.
- Due array di FormattedText, uno per il titolo e uno per la descrizione dell'obiettivo selezionato.
- Dati in formato JSON da salvare in cloud, contenente tutte le proprietà dei tab e degli obiettivi.

ExporterService

Questo servizio prendendo in input i tab dell'AdvancementService, genera il codice Java dei tab e permette lo scaricamento dell'archivio zip contenente tutte le classi.

La generazione del codice Java viene effettuato tramite l'utilizzo di stringhe multilinea template, dove vengono inseriti i dati propri dell'obiettivo tramite il "template literals" di JavaScript (${}).

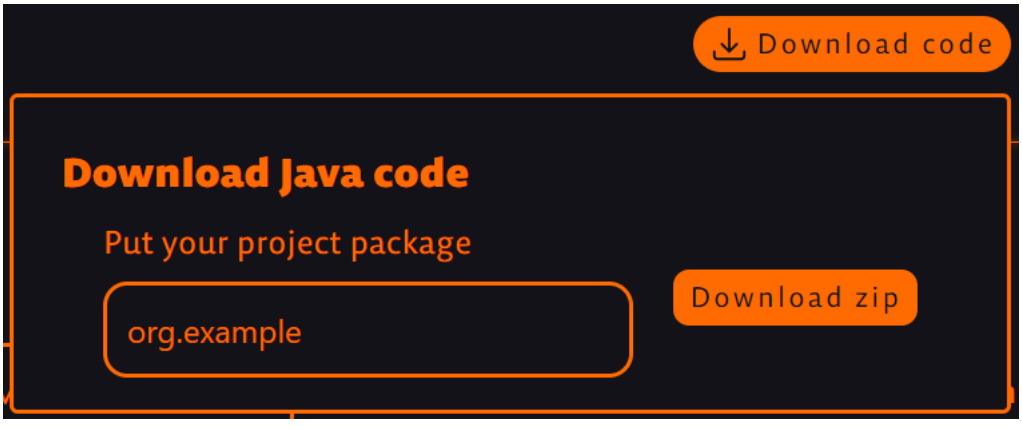


Figura 10 Sezione di scaricamento del codice Java

Per facilitare l'inserimento del codice scaricato nel progetto Java, è richiesto l'inserimento del package.

La struttura dello zip che viene scaricato è il seguente:

- *advs* è una cartella e contiene:
  - *una sottocartella per ogni tab* contenente una classe per ogni suo obiettivo.

    - *AdvancementTabNamespaces.java* è una classe che contiene in maniera statica tutti gli identificativi univoci dei tab registrati.
- *initTabs* non è una classe Java, ma contiene un metodo per inizializzare tutti i tab e i suoi obiettivi tramite UltimateAdvancementAPI.
- *readme.md* contiene una guida su come importare il codice nel progetto Java e alcuni link utili.

AuthService

Questo servizio gestisce l'interazione con il server backend e salva le informazioni attorno rilevanti.

In particolare, si occupa di:

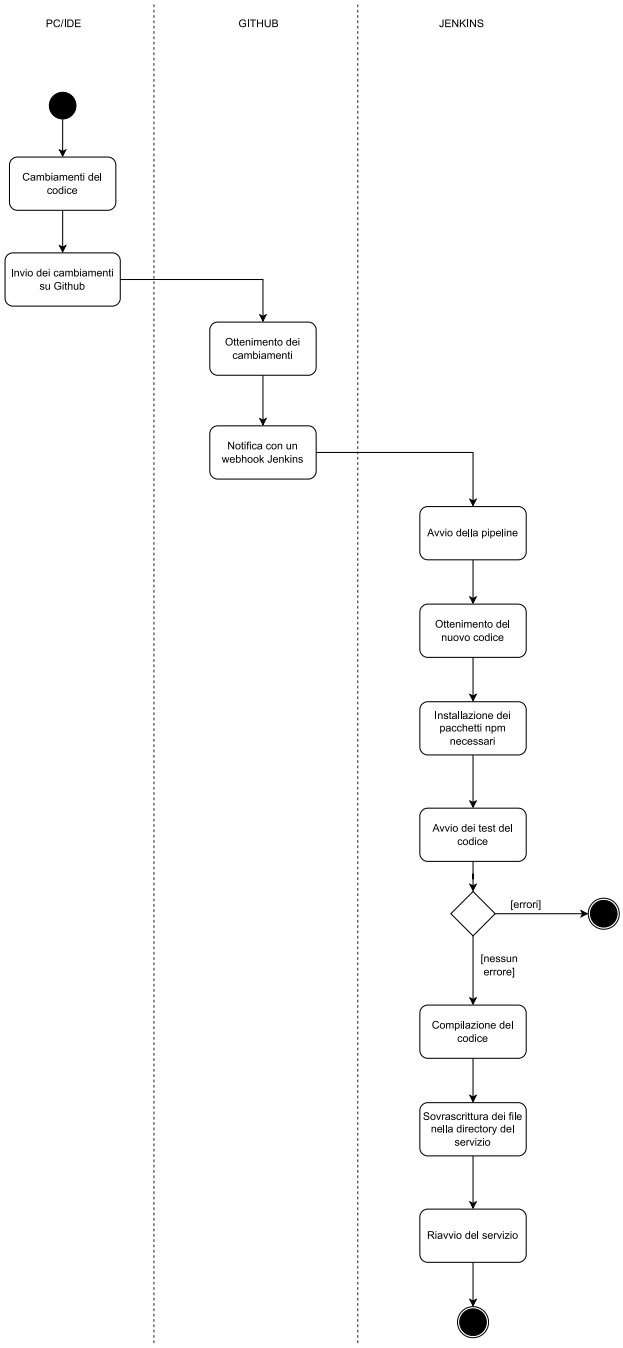
- gestire il nome del progetto scelto dall'utente autenticato.
- gestire le richieste *getToken* e *isValid* per l'ottenimento di un token di autenticazione con Google e per l'ottenimento delle informazioni sulla validità del token salvato nel *local storage* del browser.
- gestione delle richieste *getFile* e *saveFile* per l'ottenimento e il salvataggio dei dati sul cloud Google Drive dell'utente.
- dell'ottenimento del nome da mostrare nella sezione profilo della web app.

Capitolo 3

3 Pipeline per la CI/CD

Questo capitolo descrive come è stata realizzata una pipeline per l'hosting automatica per ogni ultimo commit effettuato nel repository di Github. Questo processo verrà utilizzato solo con codice durante lo sviluppo; infatti, il sito sarà hostato in un dominio di test.

3.1 Progettazione tramite activity diagram



3.2 Jenkinsfile

Il Jenkinsfile contiene il seguente testo:

```
//tag iniziale del Jenkinsfile
pipeline{

    agent any
    tools {
        //speicifico la versione di Node.js da utilizzare
        nodejs '18.18.2'
    }

    stages {

        //scarica la repository da Github
        stage('Checkout') {
            steps {
                echo 'Cleaning...'
                cleanWs()
                checkout scm
            }
        }

        //installazione di Angular e le altre dipendenze
        stage('Setup') {
            steps {
                echo 'Setup npm...'
                sh 'npm install @angular/cli'
                sh 'npm i'
            }
        }

        //esegui i test e successivamente esegui la build
        stage('Testing and building') {
            steps {
                echo 'Testing and building...'
                sh 'ng test'
                sh 'ng build'
            }
        }
    }
    post {
            //se tutti gli step precedenti sono andati a buon fine esegui...
            success {

                //sposta i file della build in una cartella condivisa con il
                //sistema
                echo 'Moving files...'
                sh 'cp -r ./dist/ultimate-advancement-generator
                /home/root/test.devheim'

                //chiama il servizio generatoreloader.service per riavviare
                //il server della web app
                echo 'Reloading server...'
                sh 'curl https://devheim.space/reload'
            }
        }
}
```

3.3 Systemd Service

La web app è hostata su un server Ubuntu e i processi che mantengono su il sito sono dei Systemd Service. Tramite un file di configurazione puoi dire al sistema operativo di gestire un tuo processo come se fosse un servizio di sistema, in modo che il processo venga inizializzato all'avviso del sistema e possa esser gestito in maniera standard dai comandi forniti da Systemd.

3.3.1 generatorjenkins.service

Questo servizio avvia il server di Angular SSR per l'accesso alla web app. Infatti, una volta che arriva una richiesta di visualizzazione da un browser, non è direttamente il server web Apache2 a rispondere, ma la richiesta viene inoltrata internamente al servizio appena citato.

3.3.2 generatoreloader.service

Jenkins essendo in un container Docker non può direttamente interagire con i servizi Systemd del sistema e per questo non può riavviare il generatorjenkins.service una volta che ha finito la fase di build della web app.

Jenkins è obbligato a mandare una richiesta http ad uno script Typescript con Express.js che ascolta le richieste all'endpoint */reload*. Questo poi effettuerà il riavvio di generatorjenkins.service.

3.3.3 backend.service

Questo servizio avvia il server backend, esponendo i suoi endpoint per l'autenticazione tramite Google e il salvataggio/ottenimento dei dati salvati su Google Drive.

Capitolo 4

4 Backend con Express.js

Il web tool permette il salvataggio su Google Drive dei tab in modo da poter tornare sulla web app successivamente e apportare modifiche agli obiettivi.

4.1 google-api-nodejs-client e OAuth2 client

> *"Google's officially supported Node.js client library for accessing Google APIs. Support for authorization and authentication with OAuth 2.0, API Keys and JWT (Service Tokens) is included."*
> - Descrizione di google-api-nodejs-client sul progetto Github

google-api-nodejs-client è un insieme di librerie, fornite da Google, da utilizzare su un backend con Node.js. Una delle librerie fornite è OAuth2 per i servizi Google. Viene utilizzato dalla web app perché permette di effettuare chiamate API e accedere a servizi per conto di un determinato utente. Nel caso dell'app, l'utente visita il sito, accede con il proprio account Google e fornisce l'autorizzazione per accedere e modificare file all'interno del suo Google Drive. In particolare una cartella chiamata "appDataFolder", usata dagli sviluppatori per contenere i dati delle proprie app.

Per poter utilizzare questo servizio è stato registrato il progetto nel sito *console.developers.google.com,* dashboard dei servizi di Google per sviluppatori. Dove è anche possibile scaricare le credenziali client che deve utilizzare il backend per effettuare le richieste.

L'autenticazione OAuth 2.0 di Google coinvolge diversi passaggi e richiede l'interazione tra il backend, il browser e i server Identity provider di Google. Una panoramica del processo di autenticazione è descritta dal seguente sequence diagram:

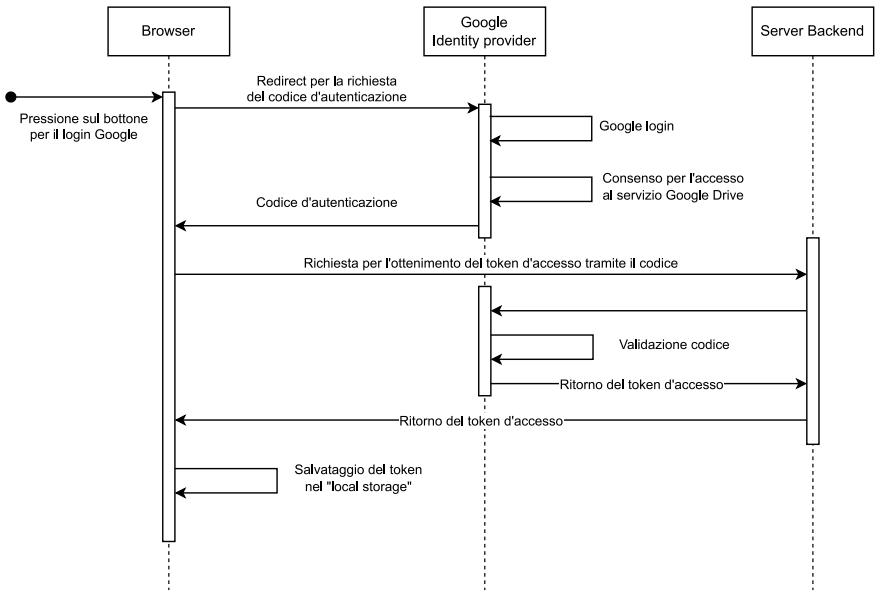


Figura 11Sequence diagram OAuth 2.0

4.2 Backend

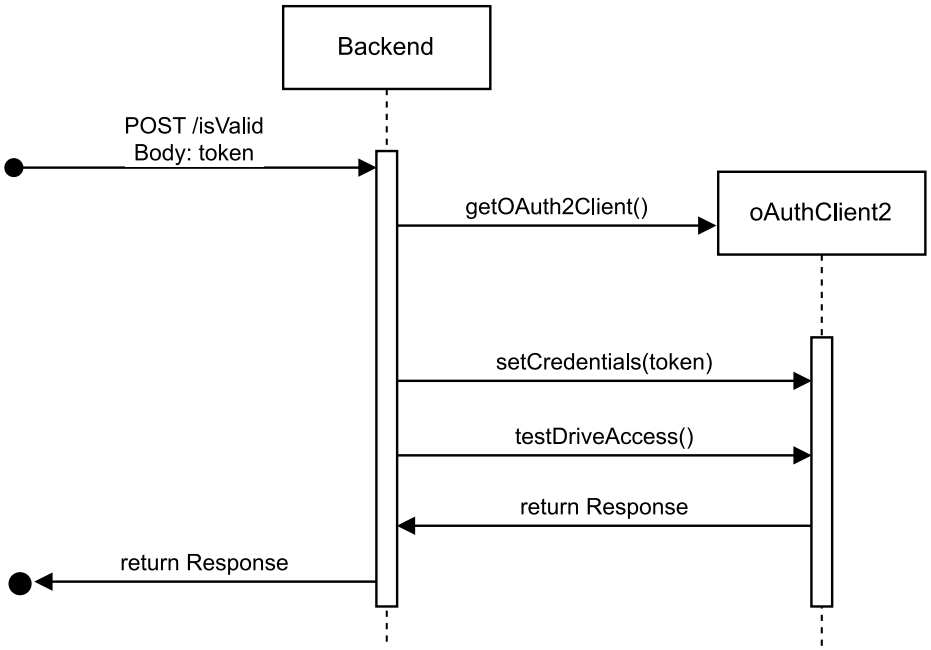
Di seguito verrà descritto com'è composto il progetto e gli endpoint utilizzati dal server Express.js. Il backend deve di gestire i dati da salvare e gestire gli accessi con i servizi di Google.

4.2.1 Il progetto

Il progetto è suddiviso in quattro file, ognuno con le sue responsabilità:

- *index.ts*: inizializza e configura il server Express.js.
- *server.ts*: gestisce gli endpoint HTTPS.
- *GoogleAPI.ts*: contiene i metodi che lavorano con le API di Google.
- *.env*: contiene le variabili d'ambiente. Nello specifico, la porta del servizio, chiave privata e certificato per le comunicazioni sotto SSL con HTTPS, e le credenziali per accedere alle API di Google.

4.2.2 isValid



Parametri

- Token d'accesso.

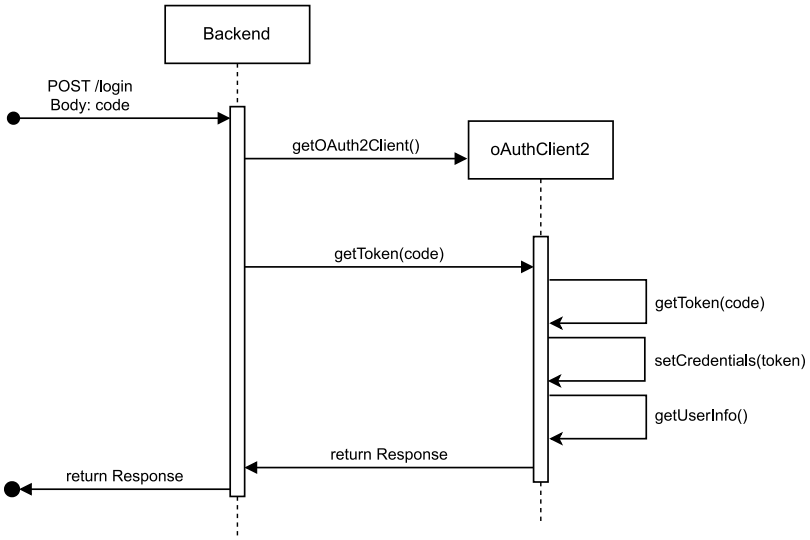
Descrizione

Questa richiesta ritorna se il token passato come parametro è ancora valido ed utilizzabile per richieste che richiedono un token d'autenticazione.

Risposte http

- (200 – OK) se il token è valido.
- (400 – Bad Request) se il token non è valido.

4.2.3 login



Parametri

- Codice d'autenticazione: codice generato da Google dopo aver effettuato il login.

Descrizione

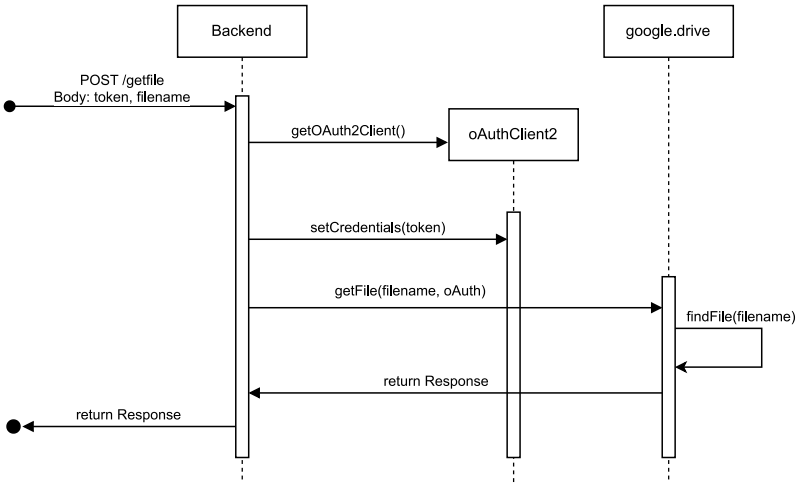
Questa richiesta è divisa nei seguenti passi:

- Generazione di un token d'autenticazione dato il codice come input.
- Ottenimento dello di altre informazioni, come lo "user id", il nome e l'email.

Risposte http

- (200 – OK) se ottenimento del token e delle informazioni sono andati a buon fine. La risposta ritorna i valori ottenuti.
- (400 – Bad Request) se il codice d'autenticazione è errato.
- (401 – Unauthorized) se l'ottenimento delle informazioni non è andata a buon fine.

4.2.4 getfile



Parametri

- Token d'autenticazione: token per accedere alle API e risorse di Google.
- Nome del file con i dati richiesti.

Descrizione

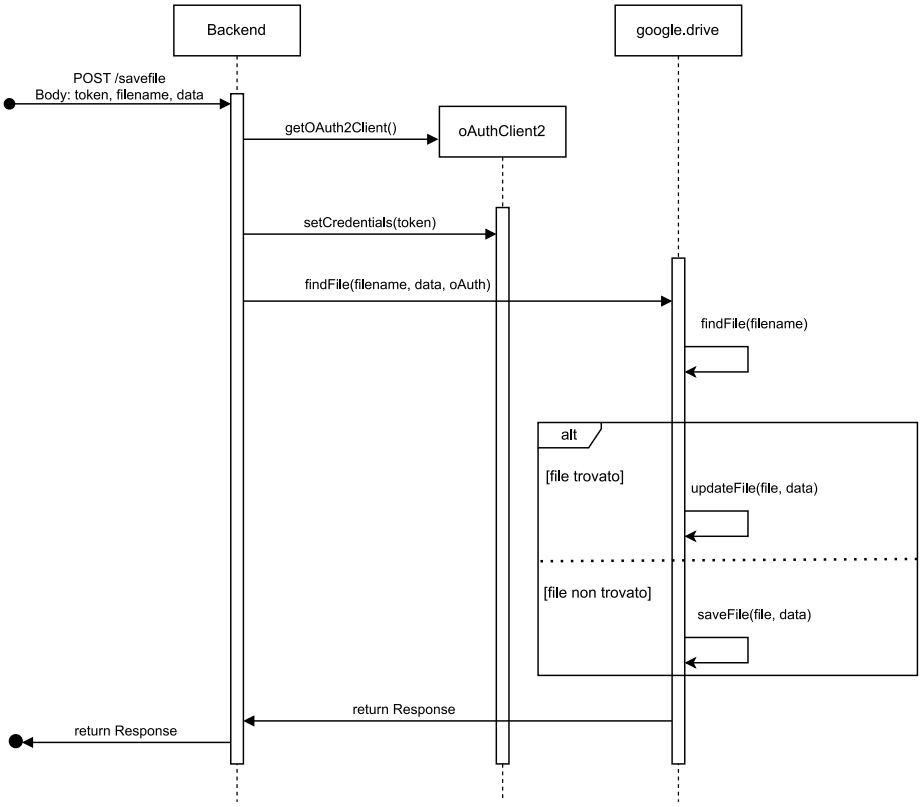
Questa richiesta è divisa nei seguenti passi:

- Autenticazione nei servizi Google e Drive API.
- Ricerca del file con i dati richiesti.

Risposte http

- (200 – OK) se tutti i passi sono andati a buon fine. La risposta ritorna il file dati.
- (400 – Bad Request) se il file non viene trovato.
- (401 – Unauthorized) se non è stato possibile autenticarsi ai servizi Google o Drive API.

4.2.5 savefile



Parametri

- Token d'autenticazione: token per accedere alle API e risorse di Google.
- Dati da salvare.
- Nome del file dove salvare i dati.

Descrizione

Questa richiesta è divisa nei seguenti passi:

- Autenticazione nei servizi Google e Drive API.

- Ricerca del file con il nome, se è presente viene aggiornato con i dati richiesti, se non è presente viene creato con i dati richiesti.

Risposte http

- (200 – File updated) se il file viene trovato all'interno di Google Drive.
- (200 – File created) se il file non viene trovato all'interno di Google Drive.
- (400 – Bad request) se non è stato possibile creare/aggiornare il file.
- (401 – Unauthorized) se non è stato possibile autenticarsi ai servizi Google o Drive API.

Conclusioni

La realizzazione di questa web app rappresenta un capitolo significativo nel mio percorso accademico. Attraverso questo progetto, ho avuto l'opportunità di applicare in modo concreto le conoscenze acquisite durante i corsi universitari, abbracciando ambiti che spaziano dall'Ingegneria del Software ai Sistemi Operativi. La passione intrinseca per il progetto ha favorito una crescita significativa delle mie competenze, spaziando dalla progettazione grafica all'implementazione di soluzioni software complesse.

La decisione di gestire in modo completo l'intero progetto, compresi il design e la progettazione software, ha rappresentato molto nella mia crescita a livello professionale. In particolare, ritengo che le competenze acquisite attraverso l'uso degli strumenti selezionati siano altamente richieste dal mercato del lavoro.

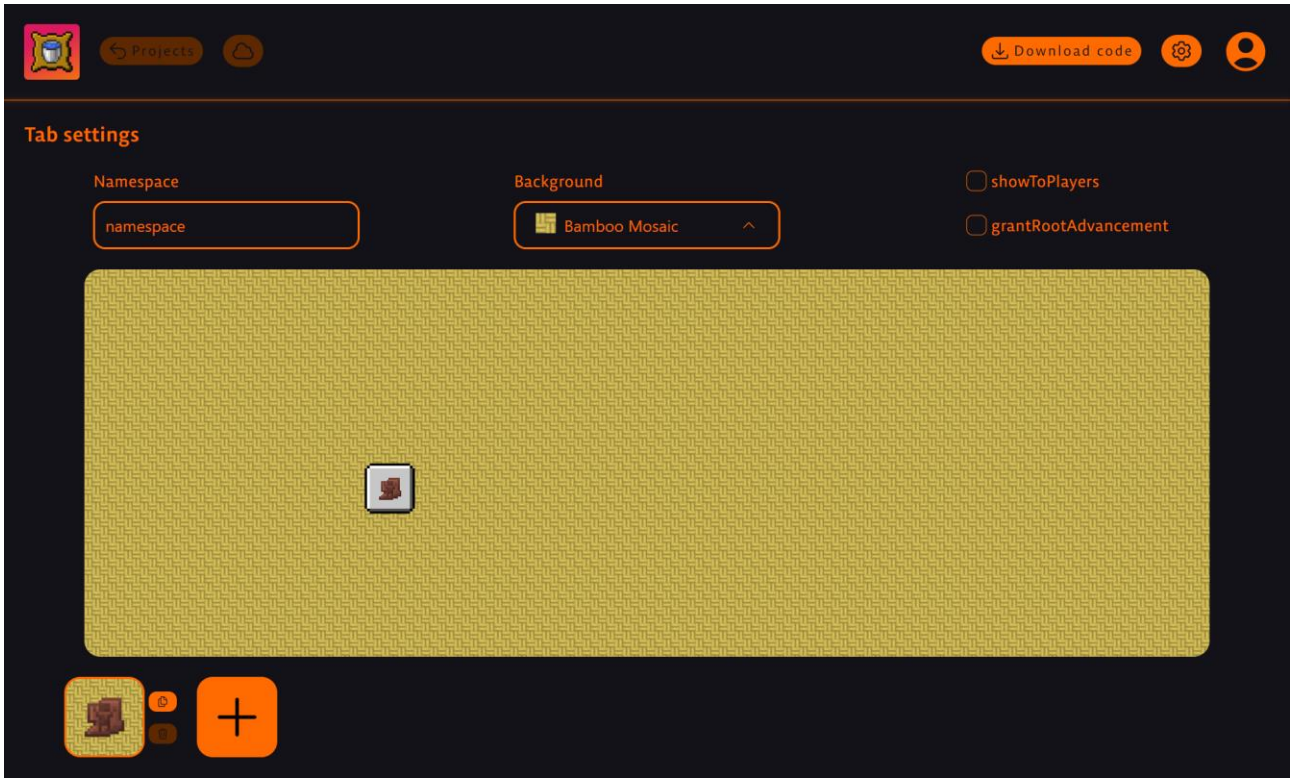
In primo luogo, l'adozione di Angular si è dimostrata un'opzione strategica, considerando la sua robustezza e la qualità degli aggiornamenti forniti da Google. Questo framework non solo ha contribuito al successo del progetto, ma ha anche consolidato la mia preferenza nei confronti di Angular rispetto ad altri strumenti simili.

L'utilizzo di TailwindCSS ha notevolmente semplificato l'approccio alla scrittura del CSS, migliorando l'efficienza del processo di styling dell'applicazione. Inoltre, l'integrazione delle API di Google per il login e lo storage ha rappresentato un passo importante nella realizzazione di funzionalità essenziali, con la consapevolezza che ulteriori integrazioni verranno sviluppate in futuro per ampliare le capacità dell'app.

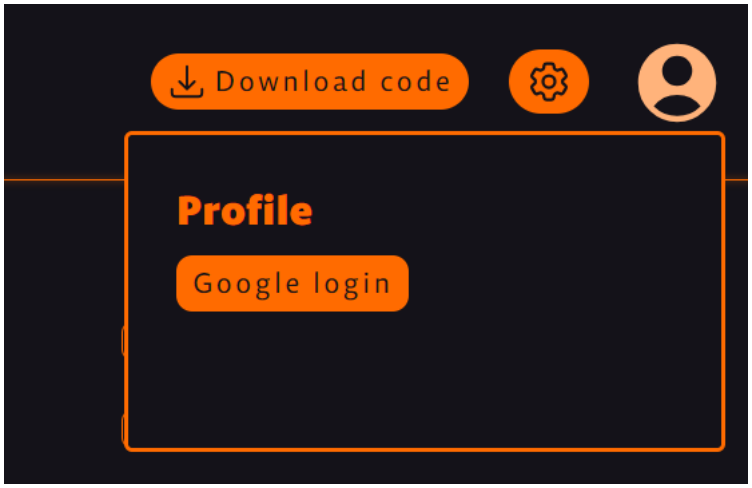
Questi strumenti hanno dato una solida base per il suo futuro sviluppo: l'applicazione sarà soggetta a costanti aggiornamenti, introducendo nuove funzionalità e correzioni per garantire la massima efficienza e soddisfare le esigenze in evoluzione degli utenti. Questa prospettiva di sviluppo continuo rappresenta un impegno costante verso una risorsa di valore nel tempo.

Esempio dell’utilizzo dell’applicazione

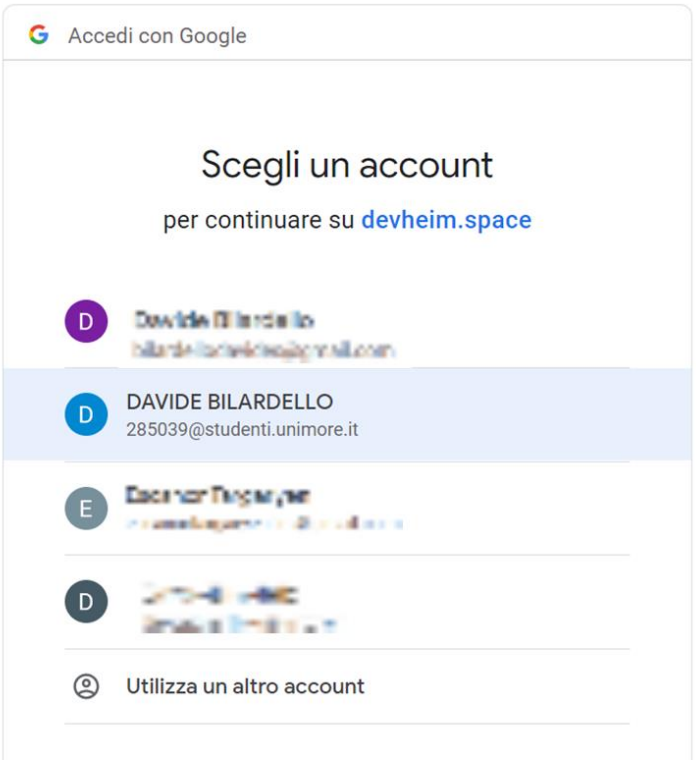
Passo 1 – Accedere alla pagina dell’applicazione
Il sito web appena aperto si presenterà nel seguente modo:



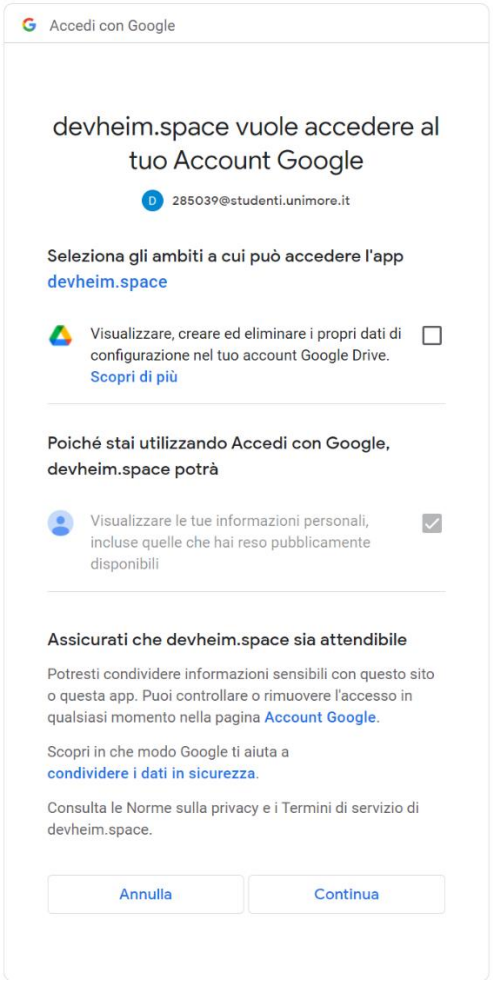
Passo 2 – Autenticarsi con Google
Autenticarsi con Google è obbligatorio se si vuole salvare i dati e le modifiche effettuate ai tab. Premere sull’icona del profilo e successivamente premere su “Google login”.



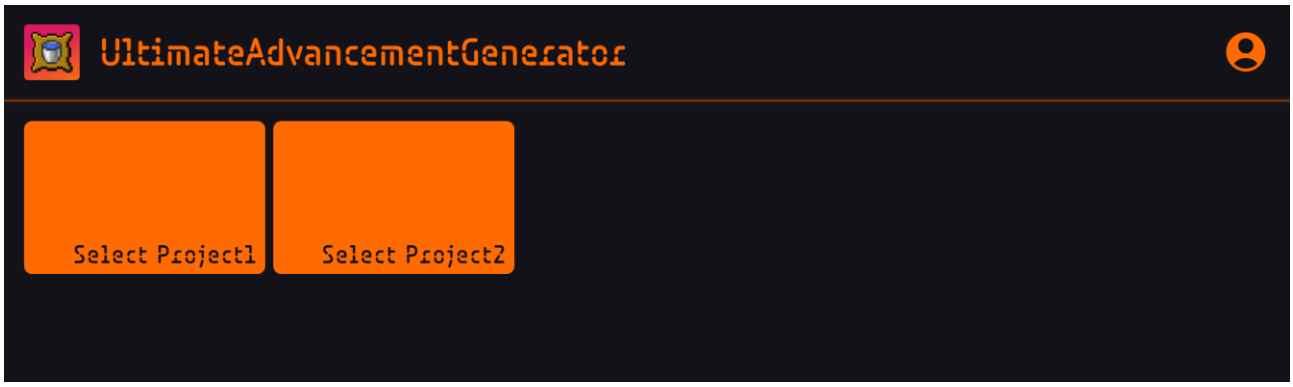
Ora si verrà rediretti sul sito Google per completare il login.



Bisognerà ora dare il permesso d'accesso alla cartella "*appDataFolder*" dell'account Google Drive. Successivamente premere su "Conferma".

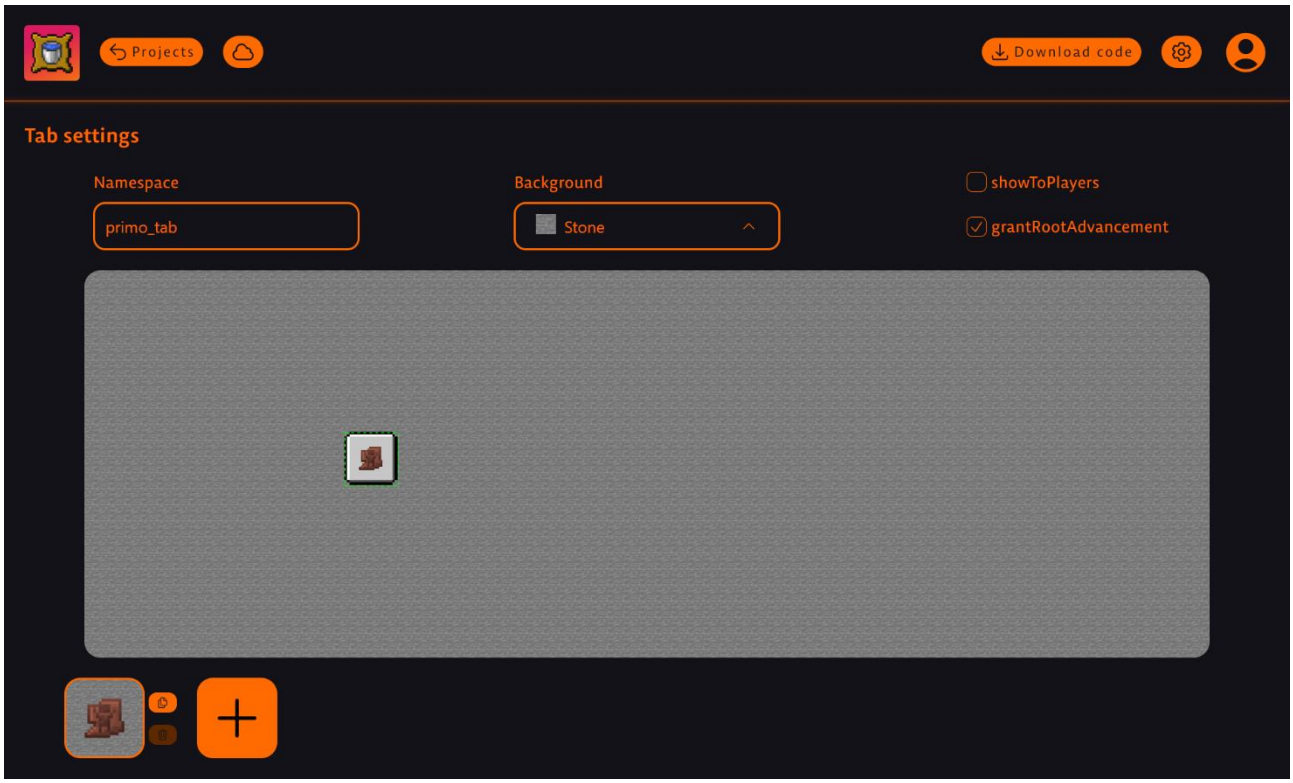


Passo 3 – Scegliere un progetto
Una volta autenticato, si potrà scegliere un progetto dalla pagina dei progetti.



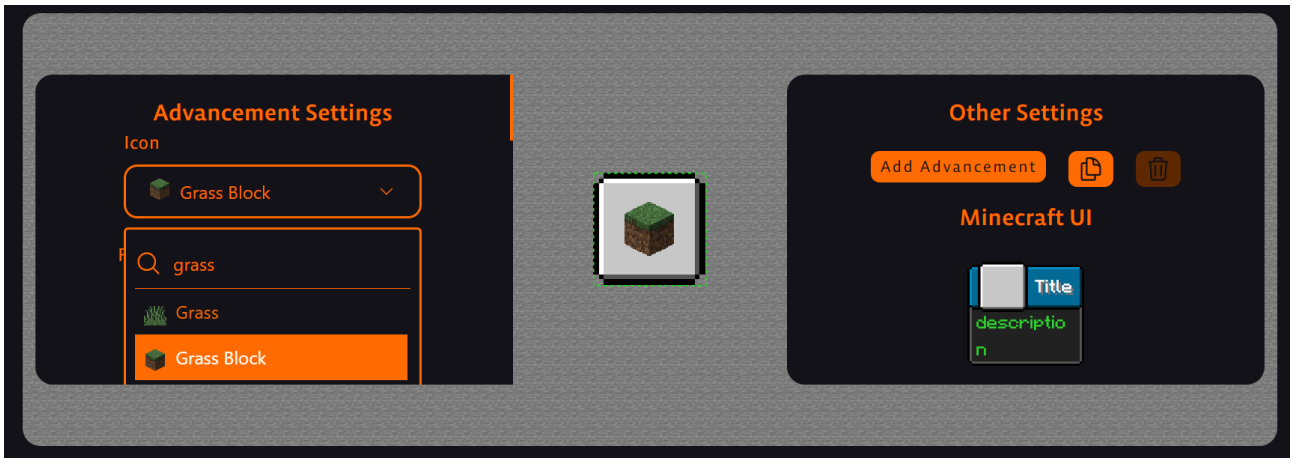
Passo 4 – Configurazione del tab e dei propri obiettivi
Modificare il tab ed impostare:

- Il suo nome univoco con “primo_tab”.
- Lo sfondo del tab in “Stone”, ovvero “Pietra”.
- Selezionare la checkbox “grantRootAdvancement” per fare in modo che quando un giocatore entrerà nel mondo di gioco, il tab sarà subito visualizzabile e l’obietto radice verrà segnato come “completato”.



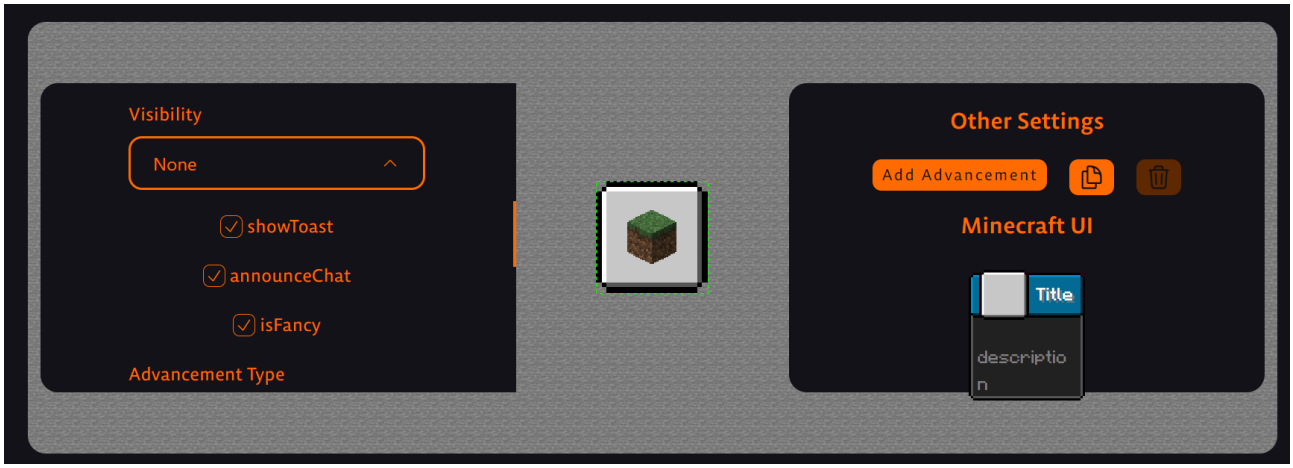
Ora selezionare l’obiettivo radice per far apparire le sezioni di personalizzazione.

Cambiare la sua icona in “Grass Block”, ovvero “Blocco d’erba”.

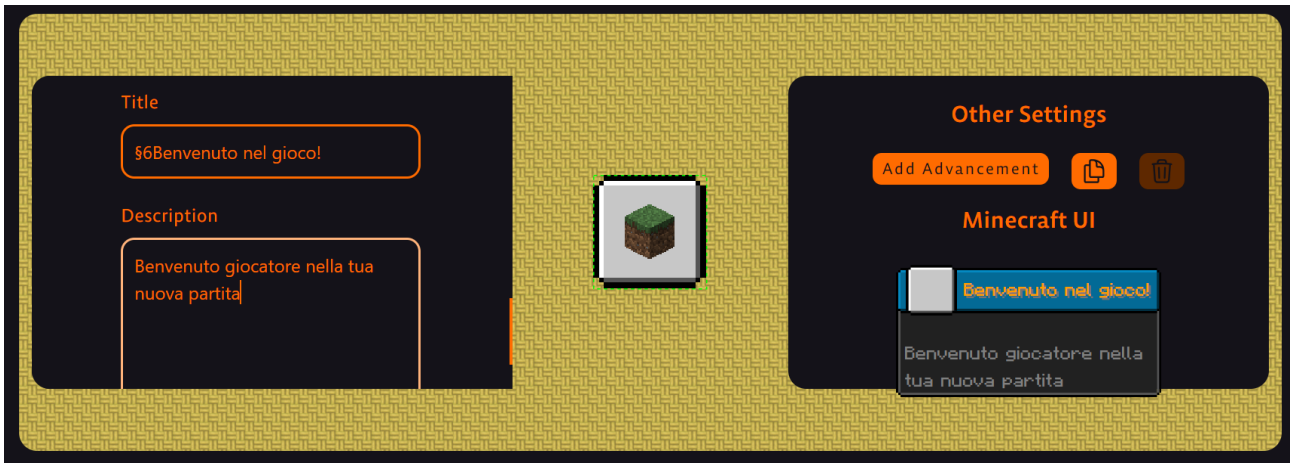


Selezionare le checkbox:

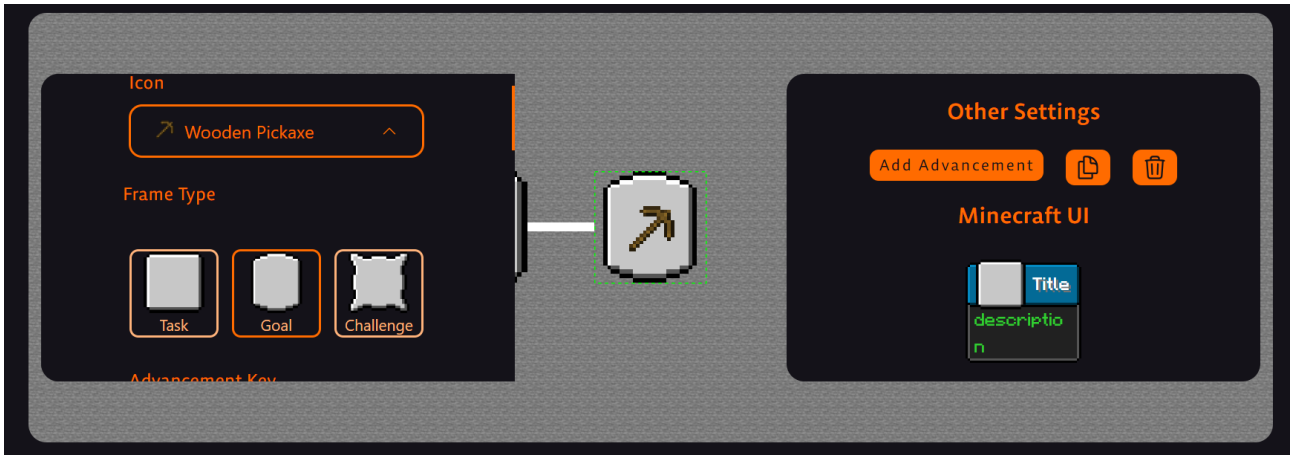
- “showToast” invierà un messaggio di tipo toast (ovvero apparirà una piccola notifica in alto a destra) quando l’obiettivo sarà segnato come completato.
- “announceChat” invierà un messaggio nella chat di gioco quando l’obiettivo sarà segnato come completato.
- “isFancy” che modifica la descrizione, in particolare, aggiunge una riga vuota prima di tutta la descrizione e la colora di grigio chiaro.



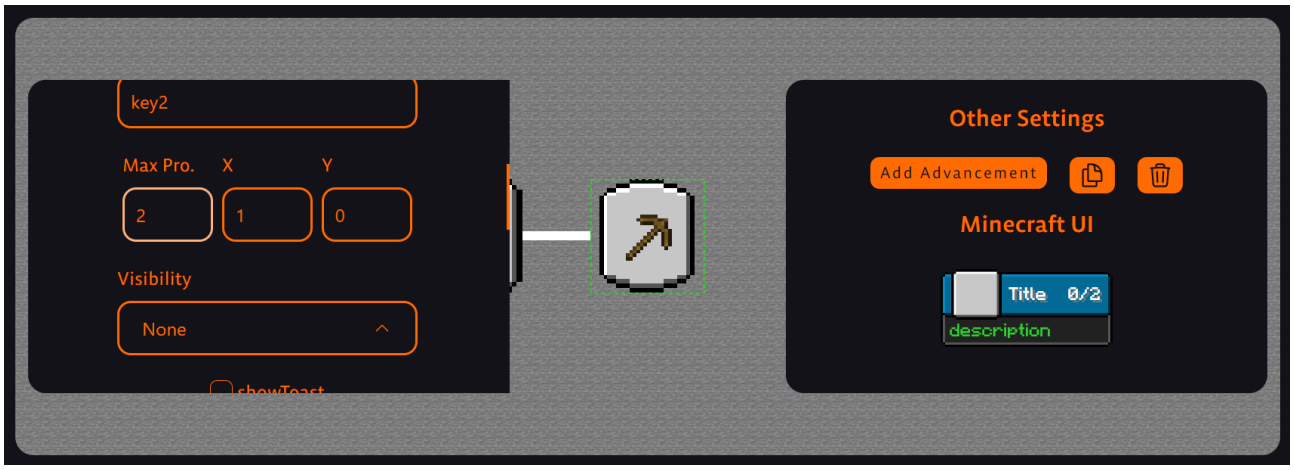
Modificare titolo e descrizione come si vede in figura. I campi a sinistra ci permettono di digitare ciò che si vuole, mentre a destra ci sarà un’anteprima del testo formattato simile a come sarà mostrato in gioco.



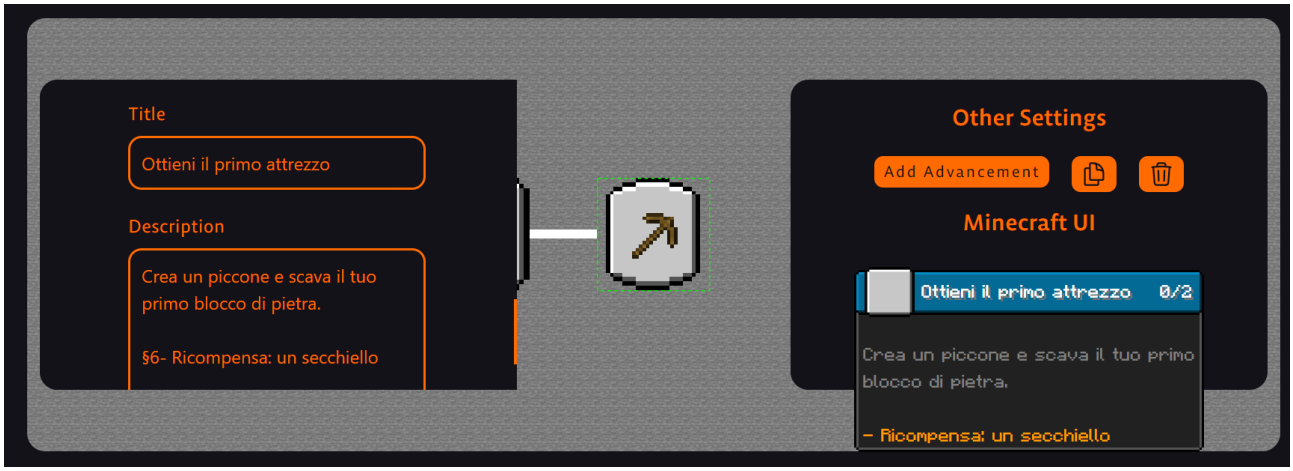
Premere su “Add advancement”, ovvero “aggiungi obiettivo”, e verrà creato un nuovo obiettivo collegato con il precedente. Il precedente obiettivo verrà chiamato “Advancement Parent”, ovvero obiettivo padre. Modificare la sua icona in “Wooden Pickaxe”, ovvero “Piccone di legno”, e modificare anche la sua cornice in quella tonda.



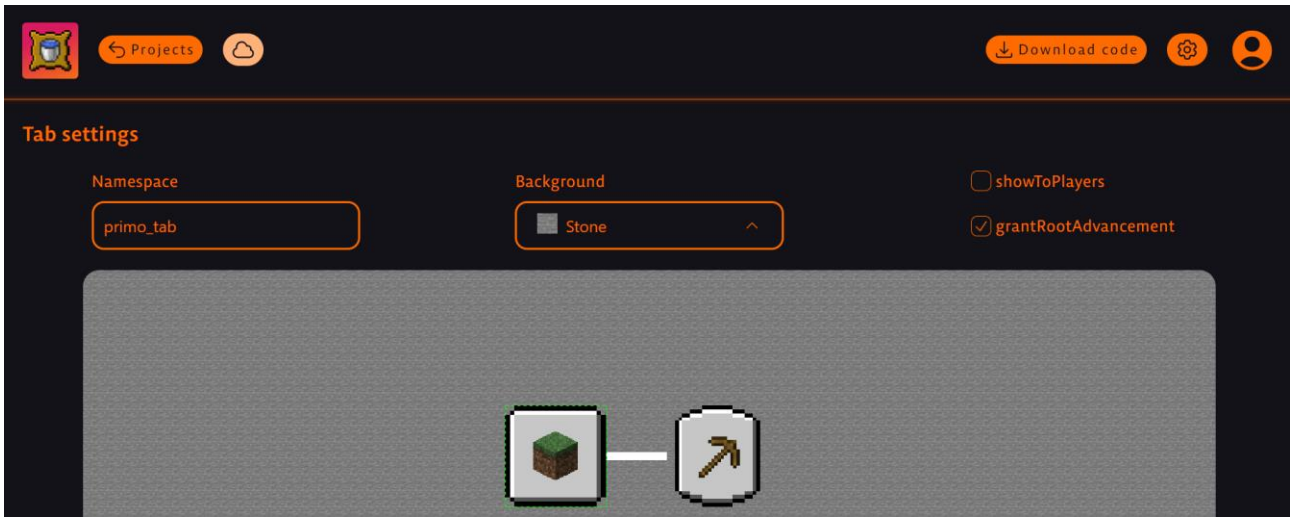
Modificare la sua progressione massima da “1” a “2”. Questo indicherà che l’obiettivo dovrà essere ripetuto 2 volte per esser marcato come completato.



Modificare il titolo e descrizione come nell’immagine:

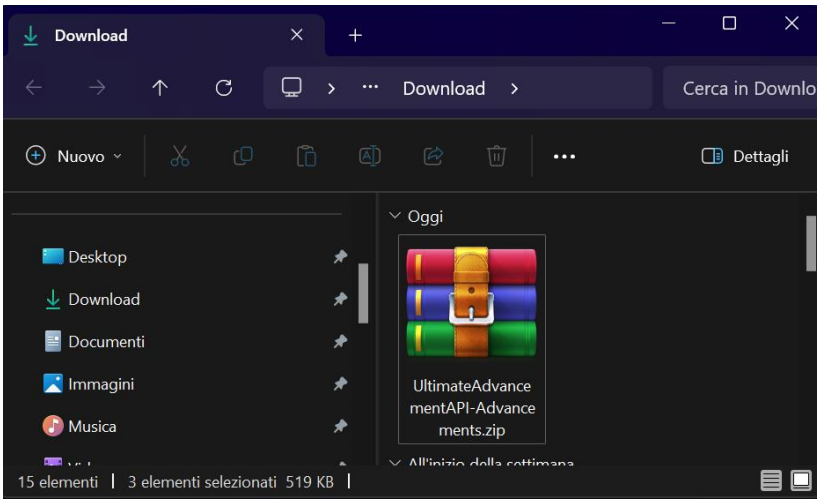


Passo 5 – Salvataggio delle modifiche su Google Drive
Premere sul simbolo della nuvola per salvare le modifiche.



Passo 6 – Download del codice Java e test nel gioco
Aprire il pannello “Download code” ed inserire il package del progetto Java. Premere il tasto “Download zip”.

Verrà Scaricato uno zip contente un progetto Java con le classi generate che andranno a replicare nel gioco quello che precedentemente è stato progettato con la web app.



Dopo aver configurato il progetto Java, avviato il server ed il gioco, il tab e gli obiettivi ci appariranno in questo modo:

Glossario

Minecraft · Videogioco sandbox creato e sviluppato da Mojang Studios. Rilasciato per la prima volta nel 2011, Minecraft ha guadagnato una vasta popolarità grazie alla sua natura aperta e alla libertà di creazione offerta ai giocatori. Ambientato in un mondo generato proceduralmente, il gioco consente agli utenti di esplorare, costruire e interagire con il loro ambiente. Con una varietà di modalità di gioco, inclusa la modalità sopravvivenza e la modalità creativa, Minecraft ha conquistato una vasta base di fan in tutto il mondo. Il gioco è noto per il suo stile grafico pixelato e il focus sulla creatività, incoraggiando i giocatori a costruire strutture elaborate e esplorare mondi virtuali senza limiti.

Plugin · Un plugin è un modulo di software aggiuntivo progettato per estendere le funzionalità di un'applicazione o di un sistema. Nei contesti di Minecraft, i plugin sono spesso utilizzati per personalizzare l'esperienza di gioco su server.

API · In informatica con il termine Application Programming Interface API (ing. interfaccia di programmazione di un'applicazione) si indica un insieme di strumenti, protocolli e definizioni che consente a diverse applicazioni software di comunicare e interagire tra loro. Nell'ambito dei plugin per Minecraft, come quelli utilizzati su piattaforme come SpigotMC, l'API fornisce un'interfaccia standardizzata che consente agli sviluppatori di creare estensioni e modificare il comportamento del software principale.

UI · UI sta per "User Interface" (Interfaccia Utente) ed è un termine utilizzato nel contesto del design e dello sviluppo software per riferirsi alla parte visibile di un'applicazione, programma o sistema che permette all'utente di interagire con esso. L'UI comprende tutti gli elementi visivi, grafici e interattivi che costituiscono l'esperienza utente mentre utilizza un'applicazione o un sito web.

Framework · Un framework è un insieme di strumenti, librerie, convenzioni e linee guida che forniscono una struttura predefinita per lo sviluppo di software. L'obiettivo principale di un framework è semplificare il processo di sviluppo, fornendo un ambiente organizzato e standardizzato in cui gli sviluppatori possono costruire applicazioni.

div · Un è un elemento HTML (Hypertext Markup Language) utilizzato per creare un contenitore o un "divisione" all'interno di una pagina web. La sua funzione principale è quella di raggruppare e organizzare altri elementi HTML, consentendo di applicare stili, regole CSS o manipolare il contenuto con script JavaScript in modo più agevole.

Hover · Il termine "hover" si riferisce a un comportamento interattivo nelle interfacce utente, particolarmente comune in contesti web. Quando si dice che un elemento è "in hover" o "hovered", significa che il cursore del mouse si trova sopra di esso senza necessariamente cliccarlo. L'evento "hover" è innescato quando il cursore entra nell'area di un elemento, come un link o un pulsante, ma non è ancora stato cliccato.

Apache2 · Apache HTTP Server, comunemente noto come Apache, è un server web open source ampiamente utilizzato per distribuire contenuti su Internet. Creato e mantenuto dalla Apache Software Foundation, Apache è uno dei server web più popolari e affidabili al mondo. È compatibile con una vasta gamma di sistemi operativi, tra cui Linux, Unix, Windows e altri.

Endpoint · In informatica e sviluppo software, un "endpoint" è un punto di comunicazione all'interno di un sistema o di una rete. Un endpoint rappresenta un URL specifico a cui è possibile inviare richieste e da cui è possibile ricevere risposte. Gli endpoint definiscono le risorse o le operazioni disponibili attraverso un'API.

JSON · Acronimo di JavaScript Object Notation, è un formato leggero di scambio dati utilizzato comunemente per la trasmissione di informazioni strutturate tra un server e un client, o tra componenti software. La sua sintassi consiste principalmente in coppie chiave-valore, dove le chiavi sono stringhe e i valori possono essere stringhe, numeri, booleani, array, oggetti o null.

HTTP e HTTPS · Acronimo di Hypertext Transfer Protocol, è un protocollo di comunicazione utilizzato per il trasferimento di informazioni su una rete, in particolare su Internet. È il protocollo di base che regola la comunicazione tra il client e il server, permettendo il recupero di risorse come pagine web, immagini e altri contenuti. HTTPS è la sua versione sicura.

HTML · Acronimo di Hypertext Markup Language, è un linguaggio di markup utilizzato per la creazione e la strutturazione di documenti ipertestuali su Internet. HTML utilizza una sintassi basata su "tag" (etichette) che indicano al browser come interpretare e presentare il testo, le immagini, i link, i form e altri elementi all'interno di una pagina web.

CSS · Acronimo di Cascading Style Sheets, è un linguaggio di stile utilizzato per definire l'aspetto e il layout di documenti HTML. La principale funzione di CSS è quella di separare la struttura del documento (definita in HTML) dalla sua presentazione visuale, consentendo ai progettisti web di apportare modifiche stilistiche senza alterare il contenuto del documento.

Rendering · Processo di generazione e visualizzazione di grafica o immagini a partire dai dati sottostanti. Nel contesto del web, il termine "rendering" è spesso utilizzato per descrivere il processo in cui i browser web interpretano e mostrano pagine HTML, CSS e JavaScript, trasformandoli in una rappresentazione visuale sulla schermata del dispositivo dell'utente.

Building · In informatica, il termine "building" è comunemente associato al processo di compilazione di un programma o di un'applicazione da parte di un compilatore. Questa fase è parte integrante dello sviluppo del software e coinvolge la trasformazione del codice sorgente scritto dall'utente in un formato eseguibile che può essere eseguito dal computer.

Deploy · Si riferisce al processo di distribuzione e installazione di un'applicazione software su un ambiente operativo o una piattaforma di destinazione. Il deploy è una fase critica nello sviluppo del software, in cui il codice sorgente, dopo essere stato scritto, testato e compilato, viene reso disponibile per l'utilizzo effettivo da parte degli utenti o dei clienti.

Repository · È un archivio dove vengono conservati e gestiti i file sorgente, la documentazione e altri elementi di un progetto software. I repository sono utilizzati per facilitare la collaborazione tra membri del team di sviluppo, tracciare le modifiche apportate al codice e gestire versioni diverse di un'applicazione o di un software.

Grafica vettoriale · È un tipo di rappresentazione grafica basata su oggetti geometrici definiti da equazioni matematiche, noti come vettori. In contrasto con la grafica “raster”, che utilizza pixel per descrivere un'immagine, la grafica vettoriale mantiene la sua qualità indipendentemente dalle dimensioni, poiché gli oggetti sono definiti in modo proporzionale e basato su formule geometriche.

Sitografia

Angular. *Overview*. Tratto da angular.dev: https://angular.dev/overview

Angular. *ssr*. Tratto da angular.dev: https://angular.dev/guide/ssr

AWS. *docker*. Tratto da aws.amazon: https://aws.amazon.com/it/docker/

Google. *google-api-nodejs-client*. Tratto da github.com: https://github.com/googleapis/google-api-nodejs-client

Kinsta. *express-js*. Tratto da kinsta.com: https://kinsta.com/it/knowledgebase/cos-e-express-js/

Kinsta. *node.js*. Tratto da kinsta.com: https://kinsta.com/it/knowledgebase/node-js/

maxGraph. *maxgraph*. Tratto da github.com: https://github.com/maxGraph/maxGraph

Wikipedia. *CI/CD*. Tratto da wikipedia.org: https://en.wikipedia.org/wiki/CI/CD

Wikipedia. *Continuous_delivery*. Tratto da wikipedia.org: https://en.wikipedia.org/wiki/Continuous_delivery

Wikipedia. *Integrazione_continua*. Tratto da wikipedia.org: https://it.wikipedia.org/wiki/Integrazione_continua

Wikipedia. *Tailwind_CSS*. Tratto da wikipedia.org: https://en.wikipedia.org/wiki/Tailwind_CSS

Ringraziamenti

Desidero esprimere profondi ringraziamenti a coloro che hanno contribuito in modo significativo alla mia carriera universitaria:

Ringrazio di cuore la mia **famiglia** per il costante sostegno, la comprensione e l'incoraggiamento durante il mio percorso accademico. Il vostro aiuto è stata la linfa vitale che ha alimentato il mio impegno e successo.

Un caloroso ringraziamento va ai miei **amici**, veri pilastri del mio percorso. Ezio, Nicolas, Alberto, Alessio, Giacomo, Salvatore, Simone e Riccardo. Il vostro supporto morale e le risate condivise hanno reso il cammino più leggero e piacevole. La vostra presenza ha reso ogni sfida più gestibile.

Vorrei esprimere la mia gratitudine a **JEMORE** per il prezioso contributo offerto durante due anni di progetti. Le competenze acquisite e il tempo trascorso con persone eccezionali hanno arricchito il mio bagaglio professionale e personale in modo significativo.

Un ringraziamento speciale a **Francesco**, mio compagno di progetti di Minecraft. Con lui, abbiamo concepito l'idea alla base di questo progetto, trasformando un'idea iniziale in una realtà concreta.

Infine, desidero esprimere la mia profonda gratitudine al **Prof. Guerra** per la sua guida, il suo sostegno e l'ispirazione fornita nel corso del mio percorso accademico. La sua competenza e dedizione hanno contribuito in modo significativo alla mia crescita professionale e personale.

Ogni contributo, grande o piccolo, ha reso possibile il completamento di questa tesi. Grazie di cuore a tutti coloro che hanno fatto parte di questo straordinario viaggio accademico.