**Università degli Studi di Padova**

Dipartimento di Matematica "Tullio Levi-Civita"

Corso di Laurea in Informatica


# Creazione di un'interfaccia grafica in Vue.js per l'embed di contenuti

*Tesi di laurea triennale*

*Relatore*

Prof.ssa Ombretta Gaggi

*Laureanda*

Anna Cisotto Bertocco
Matricola 1170706

Anno Accademico 2022-2023

# Sommario

Il presente documento ha lo scopo di descrivere dettagliatamente il lavoro svolto dalla laureanda Anna Cisotto Bertocco durante l'attività di stage, della durata di trecentododici ore, presso l'azienda THRON S.p.A.
Obiettivo principale del progetto di stage è stato realizzare un'interfaccia grafica per consentire la condivisione di un contenuto tramite codice di *embed* utilizzando il *framework* Vue.js.
Tale interfaccia è stata progettata per consentire la condivisione di tutti i tipi di contenuto supportati dalla piattaforma THRON DAM.
In secondo luogo è stato richiesto di implementare la funzionalità di applicazione di filtri su un contenuto di tipo immagine, consentendo all'utente di poter impostare manualmente i livelli di contrasto, saturazione, luminosità, nitidezza e sfocatura.
Infine tutte le componenti implementate sono state opportunamente documentate e il loro corretto funzionamento è stato verificato tramite test di unità e di accettazione.

# Indice

# Elenco delle figure

# Elenco delle tabelle

# Capitolo 1

# Introduzione

*Il seguente capitolo fornisce una panoramica dell'azienda presso cui si è svolta l'attività di stage e delle principali metodologie e strumenti di sviluppo utilizzati all'interno del contesto aziendale e per il progetto di stage.*

## 1.1 L'azienda

THRON S.p.A è una *software house*, con sede a Piazzola sul Brenta, il cui prodotto principale è THRON DAM PLATFORM, ovvero un Digital Asset Management (DAM) per controllare, gestire e distribuire su qualsiasi canale o sistema contenuti e informazioni di prodotto, evitando duplicazioni e perdite di dati. Punti cardine dell'azienda THRON sono infatti la gestione centralizzata dei contenuti e la semplicità di poterli adattare e distribuire sui vari canali in modo performante.
La piattaforma supporta contenuti di tipo immagine, anche a 360 gradi, video, audio, *playlist* [1], URL, documenti, cartelle, *tag* e dati di prodotto.
La piattaforma DAM prevede poi anche l'integrazione con espansioni THRON (*Photoshooting*, *Workflow*, PIM e *Brand Portal*) che hanno lo scopo di aiutare il cliente in tutte le altre fasi di distribuzione di un contenuto sul mercato.[1]
Attualmente l'azienda è composta da circa cinquanta dipendenti, suddivisi in *team* a seconda della loro specifica area di competenza. In particolare, per l'attività di stage la laureanda è stata inserita come sviluppatrice *front-end* all'interno del *team* Contenuti, che si occupa di gestire quelle che sono le componenti strettamente inerenti alla condivisione e gestione dei contenuti da parte del cliente finale.

## 1.2 Metodologie di sviluppo

All'interno del contesto aziendale vengono adottate metodologie di sviluppo *agile*[2], al fine di garantire maggior sinergia tra i vari *team* aziendali e di conseguenza un'efficienza maggiore all'interno di tutta l'azienda.
In particolare viene utilizzato il *framework* Scrum[3], che ha lo scopo di definire una serie di riunioni, strumenti e ruoli per una consegna efficiente di un progetto.

[1] Le *playlist* possono essere composte da immagini, audio o video

I principi cardine su cui si basa Scrum sono:

* **Trasparenza:** al fine di evitare errori di comunicazione o colli di bottiglia di informazioni, è necessario che gli aspetti significativi di un processo seguano uno standard comune e vengano frequentemente condivisi con i responsabili di progetto;
* **Riflessione:** i membri di un *team* e i responsabili di progetto devono frequentemente ispezionare i progressi effettuati al fine di poter fare delle stime e pianificazioni delle attività future ed evitare potenziali errori;
* **Adattamento:** i membri di un *team* possono ridefinire le priorità delle attività in base ai cambiamenti delle esigenze del cliente.

Per garantire i sovracitati principi fondamentali, lo sviluppo di un progetto viene suddiviso in rapidi blocchi di lavoro chiamati *Sprint*, della durata tipicamente di due settimane. Per ogni *Sprint* viene definito un elenco delle attività da svolgere, chiamato *Sprint Backlog*, scelte tra quelle preventivamente definite nel *Product Backlog*, ovvero l'elenco dinamico di caratteristiche, requisiti, miglioramenti e risoluzioni di problemi che è necessario completare per la buona riuscita di un progetto. Al termine di ogni *Sprint* viene prodotto un incremento *software*, ovvero un prodotto effettivamente utilizzabile. In Figura 1.1 vengono illustrate le attività principali di uno *Sprint* in Scrum. I *team*

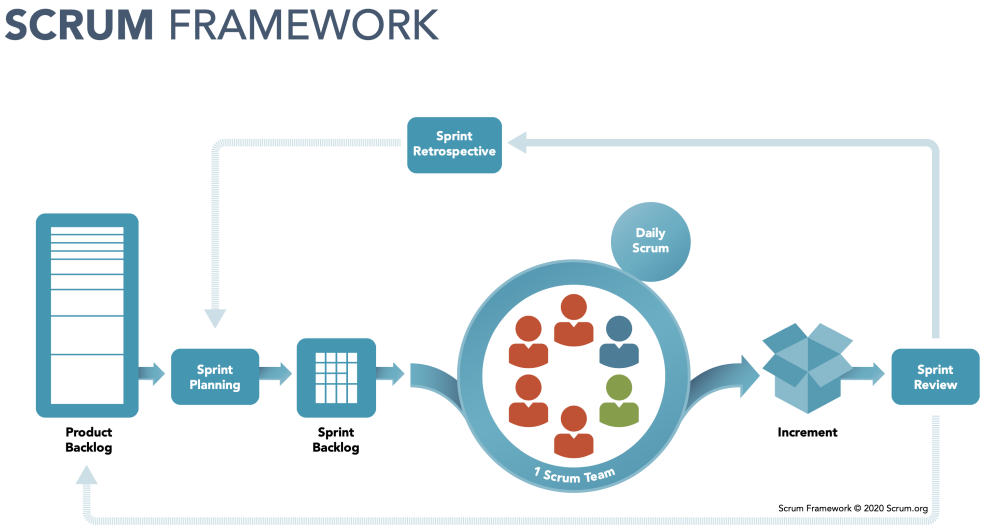


**Figura 1.1:** Struttura framework Scrum

di Scrum sono tipicamente composti da poche persone, che devono avere competenze diverse e formarsi a vicenda per evitare colli di bottiglia nella consegna del lavoro. Le figure principali all'interno del team sono lo *Scrum Master*, che si occupa di garantire l'efficienza dell'intero *team* pianificando le risorse e i tempi degli *Sprint*, il *Product Owner*, che ha il compito di fornire al *team* una guida chiara sulle funzionalità da implementare e stabilendo quando e con quale frequenza le nuove versioni saranno rilasciate, ed infine gli sviluppatori. [4]
Nel contesto aziendale THRON, il lavoro viene suddiviso in *Sprint* di due settimane e ciascun *team* si aggiorna quotidianamente sui progressi effettuati tramite una riunione

denominata *Daily Scrum*. Vengono inoltre effettuate riunioni a cadenza settimanale, denominate *Competence*, durante le quali gli sviluppatori appartenenti ai diversi *team* ma aventi la stessa area di competenza (*front-end* o *back-end*) discutono dei progressi fatti durante la settimana e si aggiornano riguardo possibili migliorie future e innovazioni tecnologiche da poter adottare.
Il tracciamento di tutte le segnalazioni e decisioni prese durante le riunioni viene fatto utilizzando *Jira*[5], una *suite* ottimizzata per la gestione dei progetti sviluppati con metodologie *agile*.

## 1.3 Strumenti di sviluppo

Al fine di agevolare le attività, di sviluppo e non, all'interno dell'azienda e in particolare per il progetto di stage sono stati utilizzati i seguenti strumenti a supporto dello svolgimento del lavoro:

* **Microsoft 365** per la stesura della documentazione richiesta per il progetto, la gestione delle riunioni a calendario e della email aziendale [6];
* **AWS** (Amazon Web Services) per la gestione della *repository* contenente il progetto e il *deploy* dello stesso [7];
* **Fork** per gestire il versionamento del codice sorgente [8];
* **Confluence** per la gestione della documentazione interna all'azienda [9];
* **Figma** per la condivisione dei *mockup* grafici tra il *team* di *design* e gli sviluppatori [10].

# Capitolo 2

# Descrizione del progetto di stage

*Il seguente capitolo descrive nel dettaglio le principali caratteristiche e funzionalità del progetto di stage, assieme alle motivazioni che hanno portato alla scelta di tale progetto da parte dell'azienda.*
*Vengono poi riportati gli obiettivi definiti in fase di pianificazione del lavoro del progetto di stage.*

## 2.1 Introduzione al progetto

Il progetto di stage è consistito nello sviluppo di un'interfaccia grafica in Vue.js per consentire la condivisione su molteplici canali di contenuti tramite codice di incorporamento, anche chiamato codice di *embed*, ovvero un frammento di codice HTML per incorporare un contenuto multimediale in una pagina web. In particolare, tale interfaccia è stata pensata in sostituzione dell'attuale GUI utilizzata all'interno di THRON DAM PLATFORM, accessibile dalla *Dashboard* della piattaforma.
Tutte le tipologie di contenuto supportate, ovvero immagini, video, audio, *playlist*, documenti e URL, sono state condivise e presentate utilizzando il THRON *Content Experience Player*, per consentire all'utente di visualizzare facilmente tutte le modifiche effettuate su un contenuto in tempo reale. Per ogni contenuto infatti viene data la possibilità all'utente di personalizzare le seguenti proprietà principali:

* ***Context Experience Template*:** l'utente può applicare un *template* grafico al *player*, personalizzando ad esempio l'aspetto delle scritte e dei controlli;
* **Disabilitare *tracking*:** l'utente può scegliere se abilitare o meno il tracciamento dell'utente per un contenuto;
* **Nascondere i controlli:** l'utente può scegliere se nascondere o meno i controlli del *player*;
* ***China Delivery*:** l'utente può scegliere se abilitare o meno il *China Delivery*, ovvero la condivisione del contenuto sul mercato cinese;
* ***Context*:** l'utente può definire il contesto in cui verrà condiviso il contenuto, ad esempio un sito web o un'applicazione *mobile*.

Per alcuni tipi di contenuto inoltre, come audio e video, viene data la possibilità di impostare l'*autoplay*, la riproduzione in *loop* o l'eliminazione dell'audio del contenuto. Infine per i contenuti di tipo immagine vengono rese disponibili le seguenti funzionalità aggiuntive:

* **Ritaglio immagine:** l'utente può applicare un ritaglio all'immagine, scegliendo tra la modalità automatica, centrata, prodotto o manuale;
* **Filtri immagine:** l'utente può applicare dei filtri all'immagine, impostando manualmente i livelli di saturazione, luminosità, contrasto, nitidezza e sfocatura.

Una volte effettuate le eventuali modifiche è quindi possibile generare e salvare il codice di *embed* da poter copiare e utilizzare direttamente per condividere il contenuto.

## 2.2 Motivazioni

Il principale motivo che ha spinto l'azienda THRON verso la creazione di questo progetto di stage è stata la necessità di migliorare l'usabilità dell'attuale interfaccia grafica utilizzata all'interno della piattaforma THRON DAM per condividere i contenuti. Tale interfaccia ad oggi contiene infatti alcune funzionalità superflue, o il cui utilizzo da parte dell'utente risulta non molto chiaro, e non fornisce il supporto per altre funzionalità che dovrebbero invece essere integrate nella piattaforma. In Figura 2.1 è possibile vedere l'attuale GUI utilizzata all'interno della piattaforma THRON DAM.

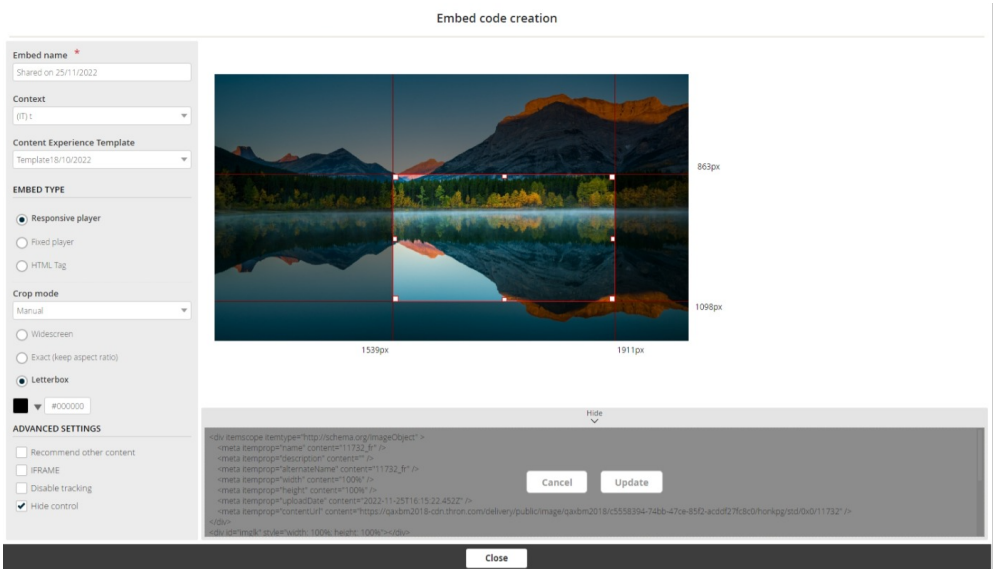


**Figura 2.1:** Schermata dell'attuale interfaccia grafica per la condivisione di contenuti

Da qui la necessità di effettuare un'analisi delle funzionalità principali per rivederne usabilità ed effettivo indice di utilizzo. A seguito di tale analisi, effettuata dal *team* Contenuti e dal *team* Design prima dell'inizio dell'attività di stage, sono state ridefinite le funzionalità e l'aspetto della GUI da sviluppare per questo progetto di stage. Di seguito vengono quindi elencate in Tabella 2.1 le principali differenze pianificate tra l'attuale GUI e quella da sviluppare durante l'attività di stage.

| **Funzionalità** | **Differenze dalla GUI attuale** | **Motivazioni** |
|---|---|---|
| Definizione nome condivisione | Funzionalità eliminata | Non utilizzata dagli utenti |
| Scelta dimensioni del *player* | Il *player* viene mantenuto sempre *responsive*, è stata quindi eliminata la possibilità di impostare delle dimensioni fisse | Non utilizzata dagli utenti |
| Condivisione tramite tag HTML | Funzionalità spostata in una GUI specifica per la condivisione tramite URL | Non pertinente, essendo la GUI utilizzata per condividere contenuti tramite codice di *embed* |
| Ritaglio in modalità prodotto | Funzionalità aggiunta | Già implementata lato *back-end* ma mai integrata nella GUI |
| Ritaglio manuale | Funzionalità ridefinita. Sono state eliminate le modalità *widescreen* e *exact*, mentre viene implicitamente mantenuta solo la modalità *letterbox* | Ritenuta non usabile. Le modalità *widescreen* e *exact* infatti creano confusione negli utenti, dato che non hanno il comportamento che ci si aspetterebbe basandosi sulla loro definizione. |
| Scelta del colore delle bande nere attorno ad un'immagine ritagliata | Funzionalità eliminata | Già implementata con la scelta del *template* grafico che è possibile applicare al *player* |
| Filtri immagine | Funzionalità aggiunta | Richiesta dagli utenti e già disponibile lato *back-end*, ma mai aggiunta alla GUI |

**Tabella 2.1:** Differenze rispetto all'attuale GUI per la condivisione di contenuti

In aggiunta alle motivazioni sopra elencate, la decisione di sviluppare la nuova GUI utilizzando il *web framework* Vue.js deriva dalla necessità di uniformare le varie componenti della piattaforma: ad oggi infatti la maggior parte delle componenti sviluppate lato *front-end* è stata nuovamente implementata utilizzando questo *framework*.

## 2.3 Obiettivi dello stage

In questa sezione vengono descritti gli obiettivi principali definiti per il progetto di stage.
La realizzazione di tali obiettivi è stata preceduta da una fase di formazione tecnologica, riguardo ai *framework* e alle librerie da utilizzare per lo sviluppo del progetto, e di studio della piattaforma THRON DAM.

### 2.3.1 Notazione

Si riporta la notazione scelta per classificare gli obiettivi pianificati nel piano di lavoro concordato con l'azienda.
Ogni obiettivo viene identificato da una sigla, per indicarne il grado di priorità, e da un numero sequenziale progressivo. Le sigle utilizzate per indicare la priorità assegnata all'obiettivo sono le seguenti:

- **OB:** obiettivo obbligatorio, vincolante in quanto requisito primario richiesto dal committente;
- **DE:** obiettivo desiderabile, non vincolante o strettamente necessario, ma il cui completamento fornisce riconoscibile valore aggiunto;
- **OP:** obiettivo opzionale, rappresentante valore aggiunto non strettamente competitivo.

### 2.3.2 Obiettivi prefissati

Seguono gli obiettivi definiti nel piano di lavoro concordato con l'azienda, ordinati secondo il livello di priorità assegnata.

***Obbligatori***

- **OB1:** Realizzazione di un'interfaccia che consenta la condivisione di un contenuto THRON di tipo immagine su molteplici canali tramite codice di *embed* utilizzando il *framework* Vue.js;
- **OB2:** Documentazione delle funzionalità implementate;
- **OB3:** Realizzazione di test di unità delle funzionalità implementate;

***Desiderabili***

- **DE1:** Implementazione della funzionalità di applicazione di filtri sulle condivisioni di contenuti di tipo immagine;
- **DE2:** Estensione della condivisione per gli altri tipi di contenuto THRON (video, audio, documenti, etc.);
- **DE3:** Implementazione della funzionalità per condividere un contenuto tramite URL;

***Opzionali***

- **OP1:** Implementazione della funzionalità per condividere un contenuto tramite pagina di *share*.

# Capitolo 3

# Analisi dei requisiti

*In questo capitolo viene descritta l'attività di analisi dei requisiti effettuata per il progetto di stage, durante la quale sono stati individuati i casi d'uso principali e le funzionalità da implementare.*

## 3.1 Introduzione

A seguito di un periodo di formazione tecnologica sul *framework* Vue.js e di introduzione al contesto della piattaforma THRON DAM, della durata circa di una settimana, è stata effettuata un'analisi approfondita dei requisiti necessari al corretto sviluppo del progetto.
I requisiti sono stati individuati mediante una stretta collaborazione con la tutor aziendale, prendendo come fonti sia la documentazione relativa all'attuale interfaccia per la gestione delle condivisioni di contenuti[11], sia l'elenco delle migliorie redatto dal *team* Contenuti precedentemente all'inizio dell'attività di stage descritto in Tabella 2.1.
Nelle sezioni seguenti vengono quindi riportati i casi d'uso e i requisiti individuati, classificati per priorità e obiettivi.

## 3.2 Casi d'uso

La seguente sezione descrive i casi d'uso principali individuati durante l'attività di analisi.
Ogni caso d'uso rappresenta uno scenario di utilizzo del prodotto effettuabile da un generico utente del sistema, denominato attore, per raggiungere un obiettivo significativo. Nel caso di questo progetto di stage è stato individuato un solo tipo di attore principale, ovvero un utente già autenticato: l'interfaccia per la condivisione di un contenuto è infatti raggiungibile solo previa autenticazione nel sistema, già gestita dalla piattaforma, e non vengono fatte ulteriori distinzioni per effettuare tutte le operazioni disponibili.
Viene inoltre mostrato nelle Figure 3.1, 3.2, 3.3, 3.4 e 3.5 il diagramma dei casi d'uso, ovvero una rappresentazione grafica in linguaggio Unified Modeling Language (UML), per i principali casi d'uso del sistema e per i casi d'uso più complessi. Per la creazione di tali diagrammi è stato utilizzato il *software* StarUML [12].

### 3.2.1 Notazione

Ogni caso d'uso viene identificato da una sigla, composta dalle lettere UC seguite da un numero intero progressivo, e da un titolo.
Al fine di facilitare la lettura, per ciascun caso d'uso del prodotto viene utilizzata la medesima struttura descrittiva:

* **Descrizione:** breve descrizione del caso d'uso;
* **Precondizione:** condizioni che sono identificate come vere prima del verificarsi degli eventi del caso d'uso;
* **Scenario principale:** rappresenta il flusso degli eventi del caso d'uso;
* **Post condizione:** condizioni che sono identificate come vere dopo il verificarsi degli eventi del caso d'uso;
* **Estensioni:** eventuali eccezioni che si possono verificare nello scenario principale.

Viene omesso l'attore primario in quanto uguale per tutti i casi d'uso, ovvero un generico utente già autenticato nel sistema.

### 3.2.2 UC0 - Ricerca condivisioni per ID contenuto

**Descrizione:** L'utente vuole cercare le condivisioni effettuate per un contenuto.

**Precondizione:** L'utente si trova nella pagina principale dell'applicativo.

**Scenario principale:** L'utente inserisce il codice ID del contenuto per cui desidera cercare le condivisioni.

**Post condizione:** L'utente visualizza l'elenco delle condivisioni presenti per il contenuto con l'ID inserito (UC1 3.2.5).

**Estensioni:**

* Il codice ID inserito non esiste (UC0.1 3.2.3);
* Non sono presenti condivisioni per il contenuto cercato (UC0.2 3.2.4);
* Si verifica un errore sconosciuto (UC7 3.2.21).

.

### 3.2.3 UC0.1 - ID contenuto errato

**Descrizione:** L'utente inserisce un ID per un contenuto che risulta inesistente.

**Precondizione:** L'utente ha effettuato la ricerca per un contenuto con un codice ID errato.

**Scenario principale:** L'utente viene notificato dell'errore e viene invitato ad effettuare nuovamente la ricerca con un codice ID valido.

**Post condizione:** L'utente è a conoscenza dell'errore e può effettuare nuovamente una ricerca.

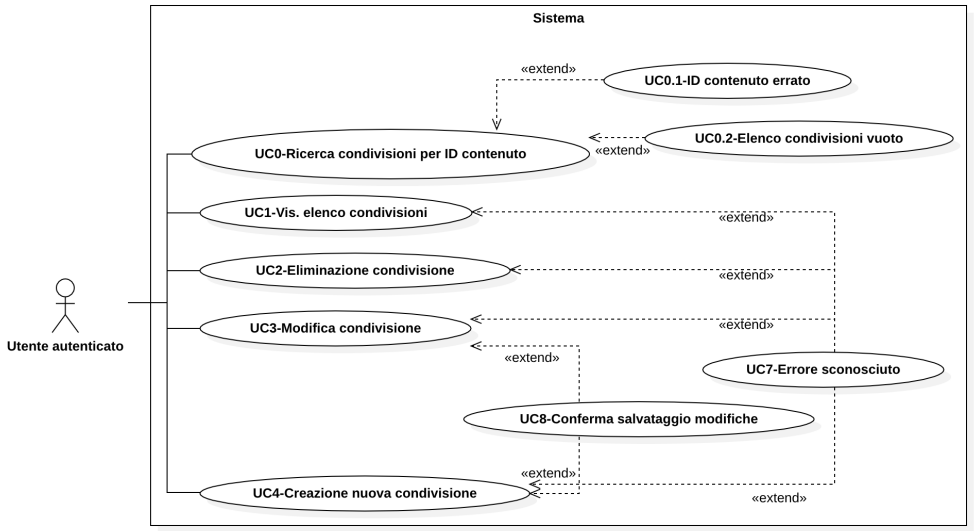


**Figura 3.1:** Diagramma dei casi d'uso principali del sistema

### 3.2.4 UC0.2 - Elenco condivisioni vuoto

**Descrizione:** L'utente ricerca un contenuto per cui non sono presenti condivisioni.

**Precondizione:** L'utente ha effettuato la ricerca per un contenuto con ID valido.

**Scenario principale:** L'utente viene informato che non risultano presenti condivisioni per il contenuto cercato.

**Post condizione:** L'utente è a conoscenza del fatto che non risultano presenti condivisioni.

### 3.2.5 UC1 - Visualizzazione elenco condivisioni

**Descrizione:** L'utente visualizza l'elenco delle condivisioni disponibili per un contenuto e le operazioni che è possibile effettuare per ogni condivisione presente.

**Precondizione:** L'utente ha effettuato la ricerca per un contenuto con ID valido e per cui è presente almeno una condivisione.

**Scenario principale:**

* L'utente visualizza gli ID delle condivisioni presenti;
* L'utente visualizza il nome delle condivisioni presenti;
* Per ogni elemento dell'elenco l'utente può eliminare la condivisione (UC2 3.2.6) o modificare la condivisione (UC3 3.2.7).

.

**Post condizione:** L'utente visualizza l'elenco delle condivisioni per il contenuto cercato.

**Estensioni:** Si verifica un errore sconosciuto (UC7 3.2.21).

### 3.2.6 UC2 - Eliminazione condivisione

**Descrizione:** L'utente vuole eliminare una condivisione.

**Precondizione:** L'utente visualizza correttamente l'elenco delle condivisioni e ha selezionato una condivisione da eliminare.

**Scenario principale:** L'utente seleziona l'eliminazione tra le azioni disponibili per una condivisione.

**Post condizione:** La condivisione selezionata viene eliminata e l'elenco delle condivisioni presenti viene aggiornato.

**Estensioni:** Si verifica un errore sconosciuto (UC7 3.2.21).

### 3.2.7 UC3 - Modifica condivisione

**Descrizione:** L'utente vuole modificare una condivisione.

**Precondizione:** L'utente visualizza correttamente l'elenco delle condivisioni e ha selezionato una condivisione da modificare.

**Scenario principale:**

* L'utente seleziona la modifica tra le azioni disponibili per una condivisione;
* L'utente visualizza l'anteprima del contenuto da condividere tramite il *player*;
* L'utente seleziona un nuovo *Context Experience Template*;
* L'utente modifica la modalità di ritaglio, se il contenuto è di tipo immagine (UC3.1 3.2.8);
* L'utente modifica le impostazione avanzate disponibili (UC3.2 3.2.11);
* L'utente modifica i filtri, se il contenuto è di tipo immagine (UC3.3 3.2.12);
* L'utente modifica il *Context* del contenuto;
* L'utente salva le modifiche e genera il codice di *embed*, l'URL oppure la pagina di *share*;
* L'utente torna a visualizzare l'elenco delle condivisioni.

.

**Post condizione:** La condivisione selezionata viene modificata e l'elenco delle condivisioni presenti viene aggiornato.

**Estensioni:**

* L'utente vuole tornare a visualizzare l'elenco delle condivisioni prima di aver salvato le modifiche effettuate (UC8 3.2.22);
* Si verifica un errore sconosciuto (UC7 3.2.21).

.

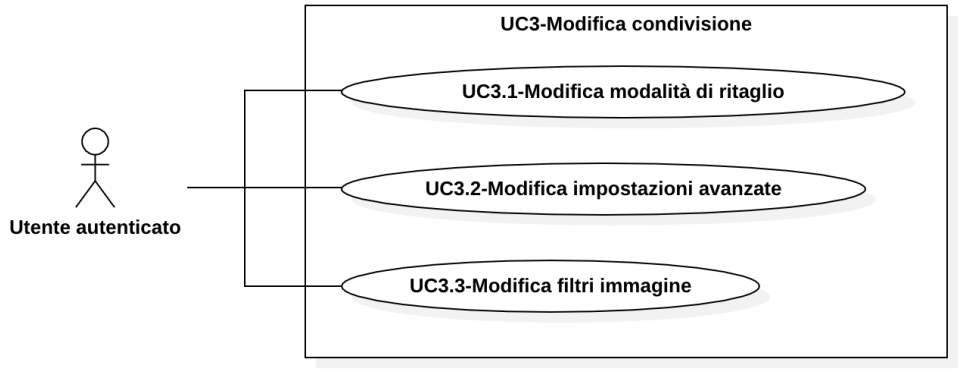


**Figura 3.2:** Diagramma del caso d'uso UC3-Modifica condivisione

### 3.2.8 UC3.1 - Modifica modalità di ritaglio

**Descrizione:** L'utente vuole modificare la modalità di ritaglio per un contenuto di tipo immagine.

**Precondizione:** L'utente sta modificando la condivisione selezionata e il contenuto è di tipo immagine.

**Scenario principale:**

* L'utente può modificare la modalità di ritaglio manuale (UC3.1.1 3.2.9);
* L'utente può modificare un'altra modalità di ritaglio (UC3.1.2 3.2.10);

.

**Post condizione:** Vengono applicate le modifiche alla modalità di ritaglio e l'anteprima è visibile nel *player*.

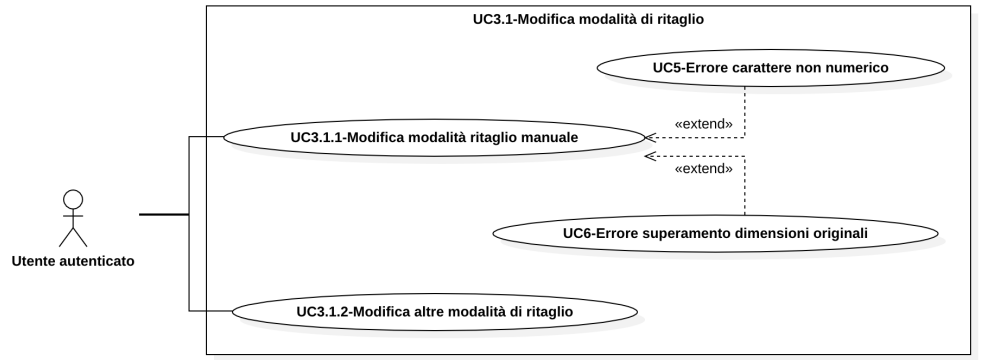


**Figura 3.3:** Diagramma del caso d'uso UC3.1-Modifica modalità di ritaglio

### 3.2.9 UC3.1.1 - Modifica modalità di ritaglio manuale

**Descrizione:** L'utente vuole modificare la modalità di ritaglio manuale per un contenuto di tipo immagine.

**Precondizione:** L'utente sta modificando la condivisione selezionata e il contenuto è di tipo immagine.

**Scenario principale:**

* L'utente annulla il ritaglio effettuato in precedenza e il ritaglio viene riportato ai valori di default per l'immagine;
* L'utente visualizza lo strumento di ritaglio manuale al posto del *player*;
* L'utente può modificare le dimensioni e la posizione del ritaglio utilizzando lo strumento di ritaglio;
* L'utente può modificare le dimensioni in pixel del ritaglio;
* L'utente può mantenere le proporzioni originali dell'immagine durante il ritaglio;
* L'utente conferma l'applicazione del ritaglio manuale;

.

**Post condizione:** Viene applicato il ritaglio manuale e l'anteprima del contenuto è visibile nel *player*.

**Estensioni:**

* L'utente inserisce un carattere non numerico durante l'impostazione delle dimensioni in pixel (UC5 3.2.19);
* L'utente supera le dimensioni originali dell'immagine durante l'impostazione delle dimensioni in pixel (UC6 3.2.20);

.

### 3.2.10 UC3.1.2 - Modifica altre modalità di ritaglio

**Descrizione:** L'utente vuole modificare la modalità di ritaglio non manuale per un contenuto di tipo immagine.

**Precondizione:** L'utente sta modificando la condivisione selezionata e il contenuto è di tipo immagine.

**Scenario principale:**

* L'utente può non applicare alcun ritaglio;
* L'utente può applicare un ritaglio automatico;
* L'utente può applicare un ritaglio centrato;
* L'utente può applicare un ritaglio *product*, che si focalizza sul prodotto principale;

.

**Post condizione:** Viene applicato il ritaglio e l'anteprima del contenuto è visibile nel *player*.

### 3.2.11 UC3.2 - Modifica impostazioni avanzate

**Descrizione:** L'utente vuole modificare le impostazione avanzate disponibili per il tipo di contenuto.

**Precondizione:** L'utente sta modificando la condivisione selezionata.

**Scenario principale:**

* L'utente modifica l'abilitazione al tracciamento utente;
* L'utente modifica l'abilitazione al *China Delivery*, se disponibile;
* L'utente modifica la visualizzazione dei controlli del *player*;
* L'utente modifica l'abilitazione dell'*autoplay*, se il contenuto è di tipo audio o video;
* L'utente modifica l'abilitazione della riproduzione in *loop*, se il contenuto è di tipo audio o video;
* L'utente modifica l'abilitazione dell'audio, se il contenuto è di tipo video;

.

**Post condizione:** Vengono applicate le modifiche alle impostazione avanzate.

### 3.2.12 UC3.3 - Modifica filtri immagine

**Descrizione:** L'utente vuole modificare i filtri immagine.

**Precondizione:** L'utente sta modificando la condivisione selezionata e il contenuto è di tipo immagine.

**Scenario principale:**

* L'utente può riportare i valori dei vari filtri ai valori di default;
* L'utente modifica il livello di contrasto;
* L'utente modifica il livello di nitidezza;
* L'utente modifica il livello di saturazione;
* L'utente modifica il livello di luminosità;
* L'utente modifica il livello di sfocatura;

.

**Post condizione:** Vengono applicate le modifiche ai filtri dell'immagine e l'anteprima è visibile nel *player*.

### 3.2.13 UC4 - Creazione nuova condivisione

**Descrizione:** L'utente vuole creare una nuova condivisione.

**Precondizione:** L'utente ha ricercato un contenuto con un ID valido.

**Scenario principale:**

* L'utente visualizza l'anteprima del contenuto da condividere tramite il *player*;
* L'utente seleziona un *Context Experience Template*;
* L'utente seleziona la modalità di ritaglio, se il contenuto è di tipo immagine (UC4.1 3.2.14);
* L'utente seleziona le impostazione avanzate disponibili (UC4.2 3.2.17);
* L'utente seleziona i filtri, se il contenuto è di tipo immagine (UC4.3 3.2.18);
* L'utente seleziona il *Context* del contenuto;
* L'utente salva le modifiche e genera il codice di *embed* l'URL oppure la pagina di *share*;
* L'utente torna a visualizzare l'elenco delle condivisioni.

.

**Post condizione:** Viene creata la nuova condivisione e l'elenco delle condivisioni presenti viene aggiornato.

**Estensioni:**

* L'utente vuole tornare a visualizzare l'elenco delle condivisioni prima di aver salvato le modifiche effettuate (UC8 3.2.22);
* Si verifica un errore sconosciuto (UC7 3.2.21).

.

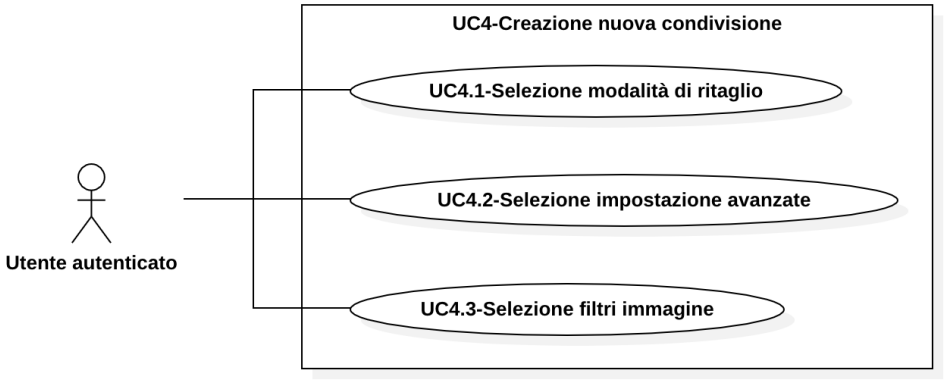


**Figura 3.4:** Diagramma del caso d'uso UC4-creazione nuova condivisione

### 3.2.14 UC4.1 - Selezione modalità di ritaglio

**Descrizione:** L'utente vuole selezionare una modalità di ritaglio per un contenuto di tipo immagine.

**Precondizione:** L'utente sta creando una nuova condivisione e il contenuto è di tipo immagine.

**Scenario principale:**

* L'utente può selezionare la modalità di ritaglio manuale (UC4.1.1 3.2.15);
* L'utente può selezionare un'altra modalità di ritaglio (UC4.1.2 3.2.16);

.

**Post condizione:** Viene applicato il ritaglio selezionato e l'anteprima è visibile nel *player*.

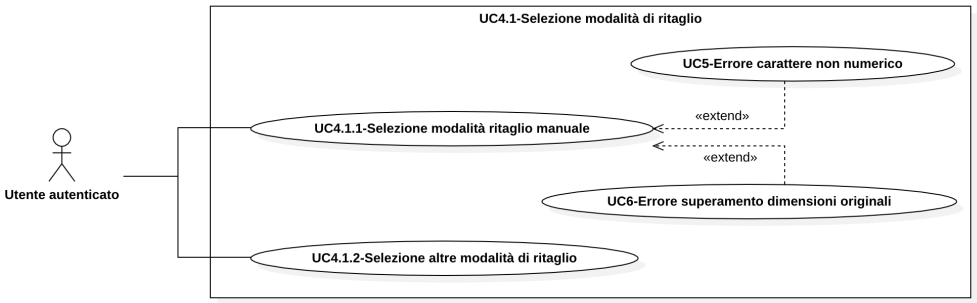


**Figura 3.5:** Diagramma del caso d'uso UC4.1-Selezione modalità di ritaglio

### 3.2.15 UC4.1.1 - Selezione modalità di ritaglio manuale

**Descrizione:** L'utente vuole selezionare la modalità di ritaglio manuale per un contenuto di tipo immagine.

**Precondizione:** L'utente sta creando una nuova condivisione e il contenuto è di tipo immagine.

**Scenario principale:**

* L'utente visualizza lo strumento di ritaglio manuale al posto del *player*;
* L'utente può impostare le dimensioni e la posizione del ritaglio utilizzando lo strumento di ritaglio;
* L'utente può impostare le dimensioni in pixel del ritaglio;
* L'utente può mantenere le proporzioni originali dell'immagine durante il ritaglio;
* L'utente conferma l'applicazione del ritaglio manuale;

.

**Post condizione:** Viene applicato il ritaglio manuale selezionato e l'anteprima è visibile nel *player*.

**Estensioni:**

* L'utente inserisce un carattere non numerico durante l'impostazione delle dimensioni in pixel (UC5 3.2.19);
* L'utente supera le dimensioni originali dell'immagine durante l'impostazione delle dimensioni in pixel (UC6 3.2.20);
* L'utente annulla il ritaglio effettuato e riporta i valori del ritaglio alle impostazioni di default per l'immagine.

.

### 3.2.16 UC4.1.2 - Selezione altre modalità di ritaglio

**Descrizione:** L'utente vuole selezionare una modalità di ritaglio non manuale per un contenuto di tipo immagine.

**Precondizione:** L'utente sta creando una nuova condivisione e il contenuto è di tipo immagine.

**Scenario principale:**

* L'utente può non applicare alcun ritaglio;
* L'utente può applicare un ritaglio automatico;
* L'utente può applicare un ritaglio centrato;
* L'utente può applicare un ritaglio *product*, che si focalizza sul prodotto principale;

.

**Post condizione:** Viene applicato il ritaglio selezionato e l'anteprima è visibile nel *player*.

### 3.2.17 UC4.2 - Selezione impostazioni avanzate

**Descrizione:** L'utente vuole selezionare le impostazione avanzate disponibili per il tipo di contenuto.

**Precondizione:** L'utente sta creando una nuova condivisione.

**Scenario principale:**

* L'utente sceglie se abilitare il tracciamento utente;
* L'utente sceglie se abilitare il *China Delivery*, se disponibile;
* L'utente sceglie se nascondere i controlli del *player*;
* L'utente sceglie se abilitare l'*autoplay*, se il contenuto è di tipo audio o video;

- L'utente sceglie se abilitare la riproduzione in *loop*, se il contenuto è di tipo audio o video;
- L'utente sceglie se disabilitare l'audio, se il contenuto è di tipo video;

.

**Post condizione:** Vengono applicate le impostazione avanzate.

### 3.2.18 UC4.3 - Selezione filtri immagine

**Descrizione:** L'utente vuole selezionare i filtri immagine.

**Precondizione:** L'utente sta creando una nuova condivisione e il contenuto è di tipo immagine.

**Scenario principale:**

- L'utente imposta il livello di contrasto;
- L'utente imposta il livello di nitidezza;
- L'utente imposta il livello di saturazione;
- L'utente imposta il livello di luminosità;
- L'utente imposta il livello di sfocatura;

.

**Post condizione:** Vengono applicati i filtri all'immagine e l'anteprima è visibile nel *player*.

**Estensioni:** L'utente riporta i valori dei vari filtri ai valori di default.

### 3.2.19 UC5- Errore carattere non numerico

**Descrizione:** L'utente inserisce un carattere non numerico durante l'impostazione delle dimensioni in pixel di un ritaglio manuale.

**Precondizione:** L'utente ha inserito un carattere non numerico durante l'impostazione delle dimensioni in pixel di un ritaglio manuale per un contenuto di tipo immagine.

**Scenario principale:** L'utente viene notificato dell'errore e il ritaglio non può essere applicato.

**Post condizione:** L'utente è al corrente dell'errore e deve inserire un carattere valido per effettuare il ritaglio correttamente.

### 3.2.20 UC6- Errore superamento dimensioni originali

**Descrizione:** L'utente vuole impostare dei valori che superano le dimensioni originali dell'immagine durante l'impostazione delle dimensioni in pixel di un ritaglio manuale.

**Precondizione:** L'utente ha inserito dei valori che superano le dimensioni originali

dell'immagine durante l'impostazione delle dimensioni in pixel di un ritaglio manuale per un contenuto di tipo immagine.

**Scenario principale:** L'utente viene notificato dell'errore e il ritaglio non può essere applicato.

**Post condizione:** L'utente è al corrente dell'errore e deve inserire delle dimensioni minori o uguali a quelle originali per effettuare il ritaglio correttamente.

### 3.2.21 UC7- Errore sconosciuto

**Descrizione:** Si verifica un errore sconosciuto durante lo svolgimento di un 'operazione.

**Precondizione:** L'utente ha provato ad effettuare un'operazione.

**Scenario principale:** L'utente viene notificato dell'errore.

**Post condizione:** L'utente è al corrente dell'errore.

### 3.2.22 UC8- Conferma salvataggio modifiche

**Descrizione:** L'utente vuole tornare a visualizzare l'elenco delle condivisioni senza aver prima salvato le modifiche effettuate per una condivisione.

**Precondizione:** L'utente sta creando una nuova condivisione o ha effettuato delle modifiche per una condivisione.

**Scenario principale:**

* Viene chiesto all'utente di confermare il salvataggio delle modifiche;
* L'utente conferma il salvataggio;
* Le modifiche vengono salvate;

.

**Post condizione:** Le modifiche sono state salvate e l'utente visualizza l'elenco delle condivisioni.

**Estensioni:** L'utente non conferma il salvataggio e le modifiche non vengono salvate.

## 3.3 Tracciamento dei requisiti

A seguito dell'identificazione dei principali casi d'uso dell'applicativo sono stati identificati e tracciati i requisiti per il progetto di stage.
Tali requisiti sono stati classificati nelle seguenti tipologie:

* **Funzionali:** descrivono le funzionalità che l'applicativo deve fornire all'utente, in termini di come il sistema reagisce a specifici input e di come si comporta in particolari situazioni;
* **Qualitativi:** descrivono gli standard da seguire e le metriche da soddisfare per garantire la qualità del sistema, come ad esempio la percentuale minima di *code coverage* per i test e le direttive di usabilità richieste dal committente;
* **Tecnologici:** riguardano i vincoli imposti dal committente riguardo alle tecnologie da utilizzare per l'implementazione del progetto.

È stata inoltre fatta un'ulteriore classificazione in base al livello di priorità assegnato a ciascun requisito:

* **Requisiti obbligatori:** requisiti la cui implementazione risulta di primaria importanza per il completamento dell'attività di stage;
* **Requisiti desiderabili:** requisiti non strettamente necessari, ma il cui soddisfacimento apporta riconoscibile valore aggiunto al progetto di stage;
* **Requisiti facoltativi:** requisiti opzionali, da soddisfare solo qualora vi siano tempo e risorse disponibili e solamente dopo aver completato i requisiti obbligatori e desiderabili.

Al termine dell'attività di analisi risultano identificati in tutto 71 requisiti, di cui 38 obbligatori, 30 desiderabili e 4 facoltativi.

### 3.3.1 Notazione

I requisiti, raccolti secondo la classificazione descritta nel paragrafo precedente nelle Tabelle 3.1, 3.2 e 3.3, vengono identificati mediante la seguente dicitura:

R{Priorità}{Tipologia}{Identificativo}

dove:

* **Priorità** indica il livello di priorità assegnato al requisito, ovvero obbligatorio (**1**), desiderabile (**2**) oppure facoltativo (**3**);
* **Tipologia** indica la tipologia di requisito, ovvero Funzionale (**F**), Tecnologico (**T**) oppure Qualitativo (**Q**);
* **Identificativo**, ovvero un numero intero progressivo.

Nelle tabelle vengono inoltre indicati, assieme ad una breve descrizione del requisito, i casi d'uso e le altre fonti da cui è stato tratto un requisito, al fine di facilitarne il tracciamento.

### 3.3.2 Requisiti funzionali

| **Requisito** | **Descrizione** | **Fonti** |
|---|---|---|
| R1F1 | L'utente deve poter ricercare le condivisioni per un contenuto tramite ID | UC0 |
| R2F2 | L'utente deve essere notificato se l'ID inserito risulta errato | UC0.1 |
| R2F3 | L'utente deve essere notificato se non sono presenti condivisioni per il contenuto cercato | UC0.2 |
| R1F4 | L'utente deve visualizzare l'elenco delle condivisioni per il contenuto cercato | UC1 |
| R1F5 | L'utente deve poter eliminare una condivisione | UC2 |
| R1F6 | L'utente deve poter visualizzare la schermata di modifica di una condivisione | UC3 |
| R1F7 | L'utente deve poter modificare il *Context Experience Template* | UC3 |
| R2F8 | L'utente deve poter vedere l'anteprima del *Context Experience Template* nel *player* | UC3,UC4 |
| R1F9 | L'utente deve poter modificare la modalità di ritaglio per un contenuto immagine | UC3.1 |
| R1F10 | L'utente deve poter modificare il ritaglio manuale con lo strumento di ritaglio | UC3.1.1 |
| R1F11 | L'utente deve poter modificare le dimensioni in pixel del ritaglio manuale | UC3.1.1 |
| R1F12 | L'utente deve poter mantenere le proporzioni originali nel ritaglio manuale | UC3.1.1, UC4.1.1 |
| R1F13 | L'utente deve poter confermare l'applicazione del ritaglio manuale | UC3.1.1, UC4.1.1 |
| R1F14 | L'utente deve poter annullare l'applicazione del ritaglio manuale | UC3.1.1, UC4.1.1 |
| R2F15 | L'utente deve poter vedere l'anteprima del ritaglio nel *player* | UC3.1, UC4.1.1 |
| R2F16 | Il sistema deve notificare l'utente con un messaggio di errore se immette un carattere non numerico impostando le dimensioni del ritaglio | UC5 |

| **Requisito** | **Descrizione** | **Fonti** |
|---|---|---|
| R2F17 | Il sistema deve notificare l'utente con un messaggio di errore se supera le dimensioni originali dell'immagine impostando le dimensioni del ritaglio | UC6 |
| R1F18 | L'utente deve poter non applicare alcun ritaglio ad un'immagine | UC3.1.2, UC4.1.2 |
| R1F19 | L'utente deve poter applicare un ritaglio automatico ad un'immagine | UC3.1.2, UC4.1.2 |
| R1F20 | L'utente deve poter applicare un ritaglio centrato ad un'immagine | UC3.1.2, UC4.1.2 |
| R1F21 | L'utente deve poter applicare un ritaglio *product* ad un'immagine | UC3.1.2, UC4.1.2 |
| R1F22 | L'utente deve poter abilitare il tracciamento utente | UC3.2, UC4.2 |
| R1F23 | L'utente deve poter disabilitare il tracciamento utente | UC3.2, UC4.2 |
| R1F24 | L'utente deve poter abilitare il *China Delivery* | UC3.2, UC4.2 |
| R1F25 | L'utente deve poter disabilitare il *China Delivery* | UC3.2, UC4.2 |
| R1F26 | L'utente deve poter mostrare i controlli del *player* | UC3.2, UC4.2 |
| R1F27 | L'utente deve poter nascondere i controlli del *player* | UC3.2, UC4.2 |
| R2F28 | L'utente deve poter abilitare l'*autoplay* per video e audio | UC3.2, UC4.2 |
| R2F29 | L'utente deve poter disabilitare l'*autoplay* per video e audio | UC3.2, UC4.2 |
| R2F30 | L'utente deve poter abilitare la riproduzione in *loop* per video e audio | UC3.2, UC4.2 |
| R2F31 | L'utente deve poter disabilitare la riproduzione in *loop* per video e audio | UC3.2, UC4.2 |
| R2F32 | L'utente deve poter abilitare l'audio per un video | UC3.2, UC4.2 |
| R2F33 | L'utente deve poter disabilitare l'audio per un video | UC3.2, UC4.2 |
| R2F34 | L'utente deve poter modificare il livello di contrasto di un'immagine | UC3.3 |

| **Requisito** | **Descrizione** | **Fonti** |
|---|---|---|
| R2F35 | L'utente deve poter modificare il livello di nitidezza di un'immagine | UC3.3 |
| R2F36 | L'utente deve poter modificare il livello di saturazione di un'immagine | UC3.3 |
| R2F37 | L'utente deve poter modificare il livello di luminosità di un'immagine | UC3.3 |
| R2F38 | L'utente deve poter modificare il livello di sfocatura di un'immagine | UC3.3 |
| R2F39 | L'utente deve poter vedere l'anteprima dei filtri immagine nel *player* | UC3.3, UC4.3 |
| R1F40 | L'utente deve poter modificare il *Context* | UC3 |
| R1F41 | L'utente deve poter salvare le modifiche su una condivisione | UC3, UC4 |
| R1F42 | L'utente deve poter creare una nuova condivisione | UC4 |
| R1F43 | L'utente deve poter selezionare un *Context Experience Template* | UC4 |
| R1F44 | L'utente deve poter selezionare una modalità di ritaglio per un contenuto immagine | UC4.1 |
| R1F45 | L'utente deve poter selezionare il ritaglio manuale con lo strumento di ritaglio | UC4.1.1 |
| R1F46 | L'utente deve poter selezionare le dimensioni in pixel del ritaglio manuale | UC4.1.1 |
| R2F47 | L'utente deve poter selezionare il livello di contrasto di un'immagine | UC4.3 |
| R2F48 | L'utente deve poter selezionare il livello di nitidezza di un'immagine | UC4.3 |
| R2F49 | L'utente deve poter selezionare il livello di saturazione di un'immagine | UC4.3 |
| R2F50 | L'utente deve poter selezionare il livello di luminosità di un'immagine | UC4.3 |
| R2F51 | L'utente deve poter selezionare il livello di sfocatura di un'immagine | UC4.3 |
| R2F52 | L'utente deve poter resettare i valori dei filtri immagine | UC3.3,UC4.3 |
| R1F53 | L'utente deve poter selezionare un *Context* | UC4 |

| Requisito | Descrizione | Fonti |
|---|---|---|
| R1F54 | L'utente deve poter generare il codice di *embed* per una condivisione | UC3,UC4 |
| R2F55 | Il sistema deve chiedere conferma all'utente se salvare o meno le modifiche prima di chiudere la finestra di creazione/modifica di una condivisone | UC8 |
| R2F56 | L'utente deve poter scegliere se salvare o meno le modifiche | UC8 |
| R2F57 | Il sistema deve notificare l'utente con un messaggio di errore se si verifica un errore sconosciuto | UC7 |
| R2F58 | L'utente deve poter condividere un contenuto tramite URL | UC3, UC4 |
| R3F59 | L'utente deve poter condividere un contenuto tramite pagina di *share* | UC3, UC4 |
| R1F60 | L'utente deve poter copiare il codice di *embed* o l'URL generato | UC3, UC4 |

**Tabella 3.1:** Tabella del tracciamento dei requisiti funzionali

### 3.3.3 Requisiti qualitativi

| Requisito | Descrizione | Fonti |
|---|---|---|
| R1Q1 | Il versionamento del codice sorgente deve essere gestito tramite *Git* | Azienda |
| R1Q2 | I test di unità devono coprire almeno l'80% del codice prodotto | Azienda |
| R1Q3 | Le funzionalità implementate devono essere opportunamente documentate | Azienda |
| R3Q4 | I *crash* improvvisi dell'applicativo devono essere tracciati e inviati al *back-end* | Azienda |
| R2Q5 | L'applicativo deve essere accessibile anche per gli utenti con difficoltà visive | Azienda |

**Tabella 3.2:** Tabella del tracciamento dei requisiti qualitativi

### 3.3.4 Requisiti tecnologici

| Requisito | Descrizione | Fonti |
|---|---|---|
| R1T1 | L'applicativo deve essere sviluppato utilizzando il *framework* Vue.js v. 3 | Azienda |
| R1T2 | I componenti devono essere scritti utilizzando le *Composition API* di Vue 3 | Azienda |
| R1T3 | La grafica dell'interfaccia deve seguire il *design system* THRON | Azienda |
| R2T4 | I componenti di base (ad esempio bottoni e input) devono essere forniti dalla libreria THRON *Components* v. 4.0.0 | Azienda |
| R1T5 | I componenti devono interfacciarsi ai servizi THRON utilizzando la libreria THRON *Models* v. 4.9.0 | Azienda |
| R3T6 | Devono essere utilizzate le funzioni di *utility* della libreria THRON *Helpers* v. 1.1.0 | Azienda |

**Tabella 3.3:** Tabella del tracciamento dei requisiti tecnologici

## 3.4 Analisi delle tecnologie di sviluppo

Una volta identificati i casi d'uso e i requisiti principali del progetto, è stato ritenuto opportuno svolgere un'ulteriore analisi riguardo alle librerie di supporto da utilizzare durante lo sviluppo al fine di rispettare i vincoli richiesti dall'azienda.

In particolare sono state svolte delle ricerche con lo scopo di scegliere la tecnologia migliore per poter effettuare il ritaglio manuale di un'immagine. Sono stati presi in considerazione tre differenti componenti grafici, *Vue Cropper.js* [13], *Vue Advanced Cropper* [14] e *Cropper.js* [15], per cui sono state considerate le seguenti caratteristiche:

1. Funzionalità per gestire le coordinate del ritaglio;
2. Funzionalità per mantenere una proporzione fissa durante il ritaglio;
3. Facilità di utilizzo;
4. Numero di funzionalità superflue;
5. Numero di dipendenze esterne;
6. Grado di utilizzo da parte di sviluppatori su npm;
7. Qualità della documentazione.

Dato che tutti i componenti considerati fornivano le funzionalità fondamentali richieste per lo sviluppo dello strumento di ritaglio manuale, la scelta infine è ricaduta sul primo componente, *Vue Cropper.js*, per i seguenti motivi:

* **Semplice e leggero:** il componente presenta poche funzionalità superflue rispetto soprattutto a *Vue Advanced Cropper*, e ha un basso numero di dipendenze esterne;
* **Ottimizzato per Vue 3**: il componente è una versione di *Cropper.js* ottimizzata però per Vue 3;
* **Molto utilizzato e ben documentato:** il componente ha un alto numero di *download* settimanali su npm, un basso numero di *issues* e presenta una documentazione aggiornata e ben redatta.

# Capitolo 4

# Progettazione e codifica

*Il seguente capitolo ha lo scopo di descrivere le attività di progettazione e codifica dell'applicativo. Verranno descritte le tecnologie utilizzate per il progetto, le scelte di progettazione e i componenti sviluppati, e infine le problematiche che sono state riscontrate durante lo sviluppo del progetto.*

## 4.1 Tecnologie di sviluppo

Di seguito viene fornita una panoramica delle tecnologie utilizzate per lo sviluppo del progetto di stage.

### 4.1.1 Vue.js

Vue.js[16] è un *framework* JavaScript utilizzato per la creazione di interfacce utente, basato sul *rendering* dichiarativo e sulla composizione per componenti, come i *framework* simili React e Angular. Con il termine *rendering* dichiarativo si intende la capacità del *framework* di estendere lo standard HTML per poter rappresentare dichiarativamente dati nel Document Object Model (DOM) utilizzando una semplice sintassi di *template*. Questo approccio permette di collegare dati e DOM rendendo l'applicazione reattiva: il DOM infatti viene aggiornato automaticamente appena viene cambiato uno stato JavaScript.
Vue.js implementa il *pattern* Model-View-ViewModel (MVVM) [17], che a differenza del *pattern* Model-View-Controller (MVC) permette di definire nel *ViewModel* il comportamento dell'applicazione a *runtime*. La struttura di questo *design pattern* viene illustrata in maniera più approfondita nella Sezione 4.2.1.
Per il progetto di stage si è scelto di utilizzare la versione 3 del *framework* e le *Composition API* offerte da Vue, grazie alle quali è possibile utilizzare le variabili e i metodi definiti in uno *script* di *setup* direttamente nel *template* HTML. Questo tipo di approccio permette maggiore libertà e flessibilità per poter riutilizzare e organizzare la logica dei componenti.

### 4.1.2 Altre tecnologie e librerie di supporto

* **HTML5:** linguaggio di *markup* che definisce le proprietà e i contenuti di una pagina web. Rispetto allo standard HTML consente di utilizzare *tag* più semantici e di memorizzare in locale molte informazioni utili per ottimizzare l'utilizzo di applicazioni web [18];
* **Sass:** estensione del linguaggio CSS che permette di utilizzare variabili, definire funzioni e organizzare lo stile dell'applicazione in più file [19];
* **JavaScript:** linguaggio di programmazione lato *client* utilizzato per rendere interattive le pagine web. Il *framework* Vue.js si basa su questo linguaggio [20];
* **Node.js (versione 16):** *runtime* JavaScript costruito sul motore JavaScript V8 di Google Chrome [21];
* **Vite.js:** *build tool* utilizzato per inizializzare applicazioni Vue garantendo velocità ed efficienza grazie alla separazione tra *Dependency Modules* e *Application Modules* [22];
* **Vitest:** *framework* per l'implementazione di test di unità. Rispetto a Jest, *framework* solitamente utilizzato in ambiente Vue, risulta ottimizzato per il *build tool* Vite, utilizzato per questo progetto [23];
* **coverage c8:** strumento per calcolare la copertura del codice, compatibile con Vitest [24];
* **Vue I18n:** libreria Vue utilizzata per tradurre i contenuti in altre lingue [25];
* **Vue Router:** componente Vue per il *routing* e la navigazione delle viste dell'applicazione [26];
* **Vue Cropper.js:** componente grafico per il ritaglio manuale di un'immagine, ottimizzato per Vue.js [13];
* **THRON Models (v. 4.9.0):** libreria THRON per interagire con i servizi aziendali forniti dal *back-end* [27];
* **THRON Components (v. 4.0.0):** libreria THRON per utilizzare i componenti comuni all'interno delle applicazioni THRON, come ad esempio bottoni e input [28];
* **THRON Helpers (v. 1.1.0):** libreria THRON contenente metodi di utilità [29];

## 4.2 Progettazione

### 4.2.1 Architettura di Vue 3

Le unità fondamentali che compongono un'applicazione Vue sono i componenti. Ciascun componente viene creato in un unico file, chiamato Single-File Component (SFC), che contiene i metodi e le logiche JavaScript, il *template* HTML e gli stili.
Seguendo la struttura del *design pattern* MVVM, simile al più famoso *pattern* MVC, all'interno di un componente la *View* è rappresentata dal *template* HTML e dagli stili definiti in Sass, mentre il *ViewModel* è il *core* del componente stesso, all'interno del quale sono contenuti sia i dati che i metodi che definiscono il comportamento di questi ultimi.
In Figura 4.1 viene rappresentata la struttura MVVM utilizzata dal *framework*. I

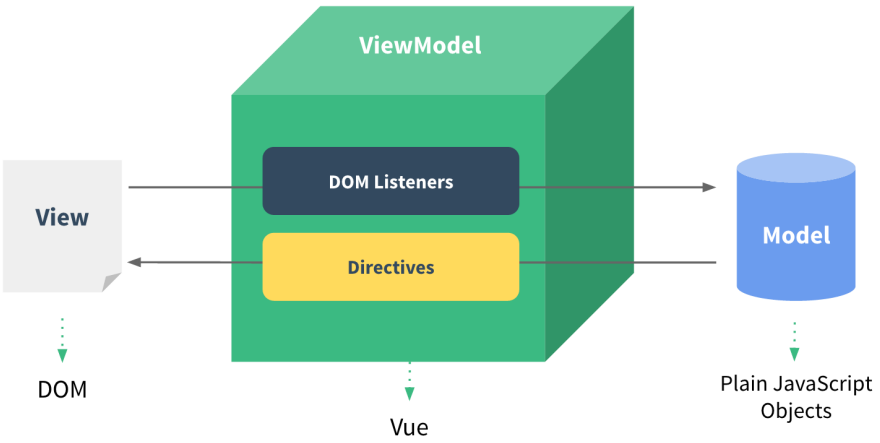


**Figura 4.1:** Struttura del pattern MVVM

dati sono collegati alla *View* grazie al *data-binding*, ovvero la possibilità di associare un elemento HTML con una variabile o un oggetto JavaScript per assicurarne il cambiamento agendo solo sulla variabile o sull'oggetto. Il *data-binding* può essere effettuato in Vue anche tramite l'uso di **direttive**, ossia particolari attributi HTML che permettono di definire il comportamento a livello di presentazione.
I componenti, organizzati gerarchicamente, possono "passarsi" dati, chiamati anche *props*, ed utilizzare servizi, che possono essere iniettati direttamente in un componente come dipendenze, rendendo quindi il codice modulare, riutilizzabile ed efficiente.

### 4.2.2 Architettura del progetto di stage

Per lo sviluppo del progetto di stage, prendendo come riferimento le linee guida fornite dalla documentazione ufficiale di Vue [16] e gli standard aziendali, è stata progettata un'architettura avente i seguenti moduli principali:

- **Components:** raccoglie tutti i moduli per i componenti riutilizzabili all'interno del sistema, che possono quindi essere importati all'interno di altri componenti;

* **Views:** raccoglie tutti i moduli per le viste del sistema, definite ciascuna in un componente separato;
* **Helpers:** raccoglie i file JavaScript che contengono logiche riutilizzabili all'interno del sistema, suddivisi per ambito di utilizzo;

I moduli **Router** e **I18n** servono a raccogliere invece i servizi di *routing* e di traduzione globali nel sistema.
In Figura 4.2 viene mostrata la struttura ad alto livello dei principali moduli del sistema.
Ogni sotto-modulo di *Views* e *Components* contiene:

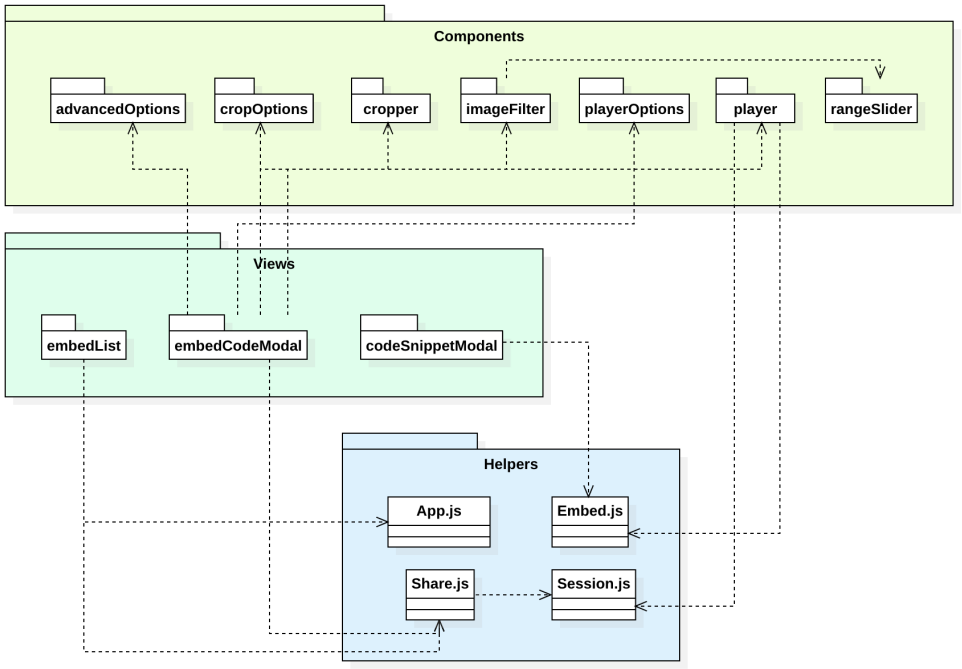


**Figura 4.2:** Moduli ad alto livello del sistema

* Il SFC con i dati, i metodi e il *template* del componente;
* Il file Sass con gli stili da applicare al *template*;
* Il file I18n con le traduzioni utilizzate nel componente;

Questa organizzazione comune ai moduli permette di avere coerenza e uniformità all'interno dell'applicazione, e di modularizzare quanto più possibile il codice per agevolare migliorie e modifiche future.

## 4.3 Codifica

Di seguito vengono descritti i servizi e i componenti sviluppati per il progetto di stage, seguendo le specifiche poste in fase di progettazione architetturale.

### 4.3.1 *Helpers*

Il modulo *Helpers* raccoglie i file JavaScript di utilità suddivisi per ambito di utilizzo, i cui metodi vengono utilizzati da più componenti all'interno del sistema.

#### App

Il file *App.js* contiene alcuni metodi per ricavare quali funzionalità riguardanti la condivisione dei contenuti siano disponibili per un determinato codice servizio, come ad esempio l'abilitazione del *China Delivery*.

#### Session

Il file *Session.js* contiene alcuni metodi riguardanti la sessione utente, come ad esempio il *getter* del *token* di sessione, necessario per effettuare le chiamate alle API dei servizi aziendali.

#### Share

Il file *Share.js* contiene i metodi per gestire le condivisioni di un contenuto, interagendo con i servizi aziendali tramite la libreria THRON *Models*[27]. I metodi definiti servono ad esempio per creare, eliminare o modificare una condivisione, oppure per incapsulare in maniera più usabile all'interno dell'applicazione i dati del *back-end* riguardo le proprietà di un contenuto.

#### Embed

Il file *Embed.js* contiene dei metodi per la realizzazione del codice di *embed*, da utilizzare anche per visualizzare l'anteprima del contenuto nel *player*. Si è scelto di mantenere questi metodi separati dal file *Share.js* per mantenere il codice più modulare una volta implementate anche le altre modalità di condivisione (URL e pagina di *share*), che necessitano di metodi e dati differenti.

### 4.3.2 *Components*

Il modulo *Components* raccoglie tutti i componenti che possono essere riutilizzati all'interno del sistema, anche in previsione di un loro utilizzo futuro per implementare diverse modalità di condivisione, ovvero tramite URL o pagina di *share*. Alcuni componenti utilizzano anche i componenti base, come ad esempio input e bottoni, forniti dalla libreria THRON *Components*[28].

#### PlayerOptions

Il componente *PlayerOptions* si occupa di gestire la selezione del *Context Experience Template* da applicare al *player* THRON. Il *template* viene selezionato dall'utente tra quelli disponibili per il codice servizio tramite un menù a discesa e l'anteprima viene automaticamente visualizzata nel *player* (Figura 4.3).

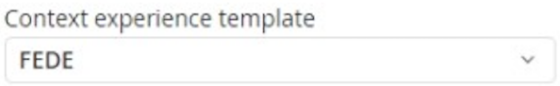


**Figura 4.3:** Menù a discesa per la selezione del *Context Experience Template*

#### CropOptions

Il componente *CropOptions* si occupa di gestire la selezione della modalità di ritaglio di un contenuto di tipo immagine. L'utente può scegliere tra le modalità *No crop* (nessun ritaglio), *Auto* (automatico), *Centered* (centrato), *Product* (prodotto) oppure *Manual* (manuale) cliccando sull'icona corrispondente (Figura 4.4).
Per la modalità manuale vengono visualizzate le dimensioni del ritaglio in pixel, le quali si possono impostare tramite input oppure utilizzando lo strumento di ritaglio (*Cropper* 4.3.2) *drag and drop*.
L'utente può anche mantenere fisse le proporzioni originali dell'immagine durante il ritaglio cliccando sull'icona .
Il ritaglio manuale deve essere inoltre applicato, cliccando sul bottone *Apply*, per poter vedere il risultato finale nel *player*. Nel caso l'utente voglia annullare il ritaglio manuale effettuato, deve cliccare sul bottone *Reset*.

#### Cropper

Il componente *Cropper* utilizza il componente grafico *Vue Cropper.js* per effettuare il ritaglio manuale di un'immagine *drag and drop*. Comunica con il componente *CropOptions* (4.3.2) per mantenere aggiornate le dimensioni del ritaglio nei due input numerici (Figura 4.4).

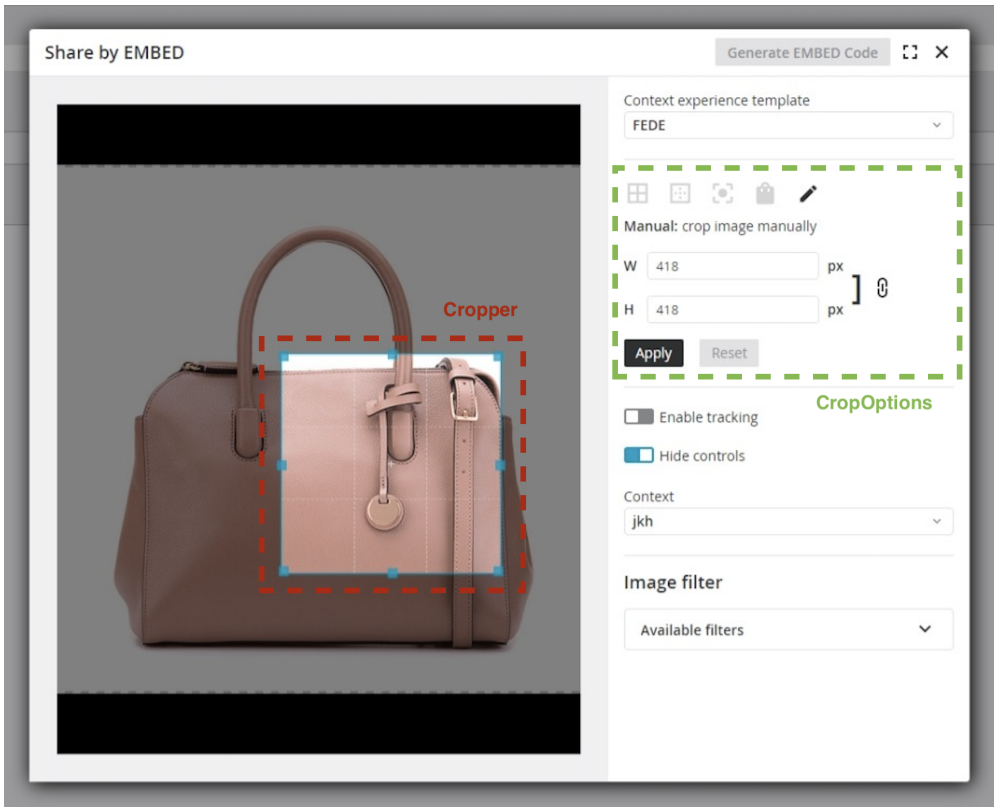


**Figura 4.4:** Schermata in cui sono evidenziati i componenti per la selezione della modalità di ritaglio e lo strumento di ritaglio manuale per un'immagine

### AdvancedOptions

Il componente *AdvancedOptions* si occupa di gestire le impostazioni avanzate disponibili per un contenuto e la scelta del contesto di condivisione del contenuto (*Context*), selezionabile tra quelli disponibili tramite menù a discesa (Figura 4.5).
Le impostazioni avanzate diverse dal *Context* possono essere abilitate o disabilitate utilizzando un bottone *on/off*.

### RangeSlider

Il componente *RangeSlider* viene utilizzato per creare un *range slider*, ovvero un particolare input che permette all'utente di selezionare un valore tra un minimo e un massimo spostando un cursore grafico. Si è scelto di non utilizzare l'alternativa già presente in HTML5, ovvero un input di tipo *range*, perché non risultava sufficientemente personalizzabile, dal punto di vista grafico, e preciso per quanto riguarda il valore numerico selezionato con il cursore.

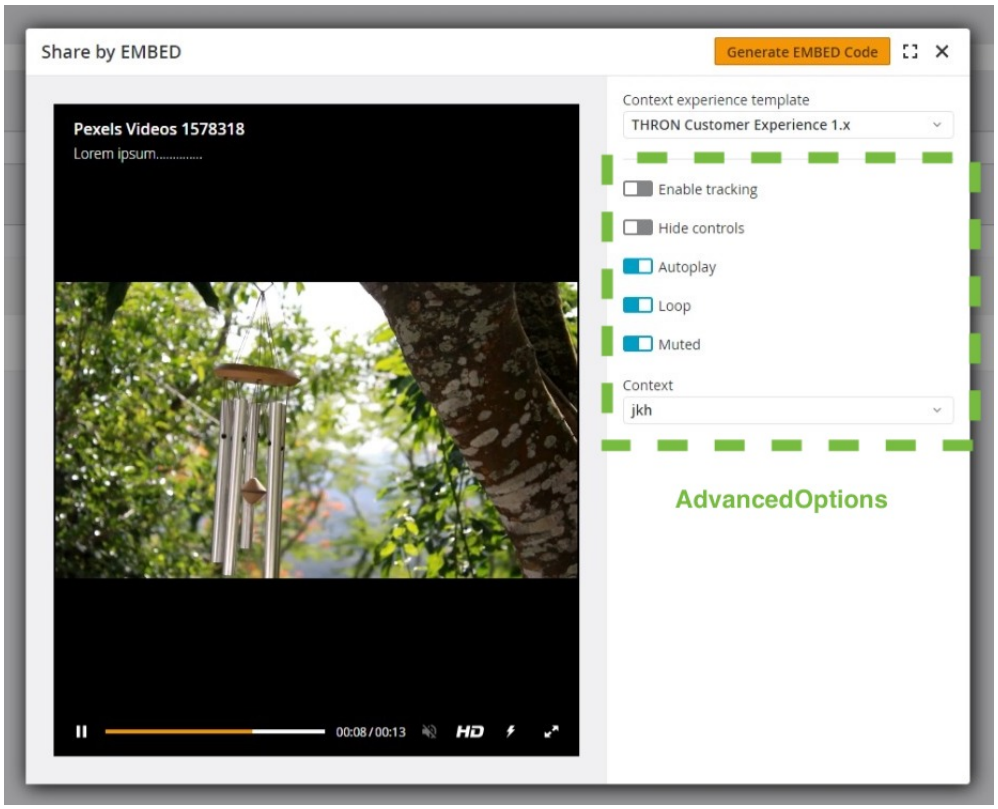


**Figura 4.5:** Schermata in cui viene evidenziato il componente per la gestione delle impostazioni avanzate per un video

### ImageFilter

Il componente *ImageFilter* si occupa di gestire l'impostazione dei filtri per un contenuto di tipo immagine. Utilizza il componente *RangeSlider* (4.3.2) per dare la possibilità all'utente di selezionare i livelli di contrasto, saturazione, luminosità, nitidezza e sfocatura. Viene inoltre data la possibilità di ripristinare i livelli ai valori di default tramite un bottone *Reset* (Figura 4.6). L'intervallo di valori selezionabili per ogni proprietà è stato ricavato da una documentazione non ufficiale THRON: questa documentazione prevedeva un *range* da -200 a 200, che è stato tradotto in un range da 0 a 100 per questioni di usabilità utente.
I vari *sliders* sono contenuti in un menù a scomparsa in modo da non costringere l'utente a fare lo *scroll* della modale inutilmente, qualora non sia interessato ad applicare filtri all'immagine.
Le modifiche ai valori dei filtri vengono visualizzate in tempo reale nel *player*.

### Player

Il componente *Player* incapsula il *player* THRON per visualizzare il contenuto da condividere, aggiornandolo con le impostazioni selezionate dall'utente in tempo reale (Figura 4.7).
Questo componente utilizza i metodi contenuti nel file *Embed.js* 4.3.1 ricevere i valori del-

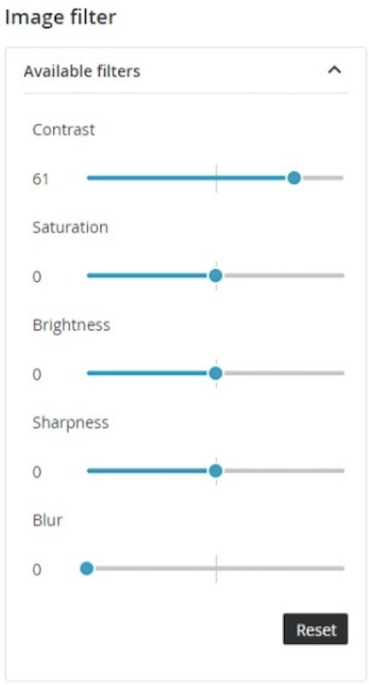


**Figura 4.6:** Componente per l'impostazione dei filtri in un contenuto di tipo immagine

le proprietà impostate dall'utente e riportarli nel formato richiesto dalla documentazione del *player* [11].

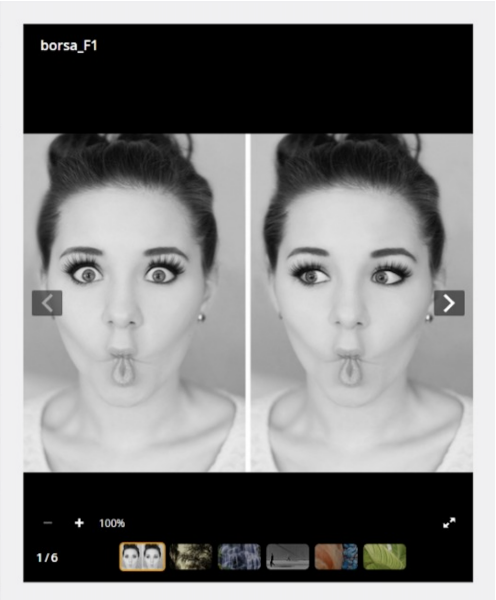


**Figura 4.7:** *Player* THRON per una *playlist* di immagini

### 4.3.3 *Views*

Il modulo *Views* raccoglie le viste dell'applicazione, create dalla composizione dei componenti descritti nelle precedenti sezioni. L'applicazione prevede anche una finestra di *login* e la possibilità di effettuare il *logout*, implementata dalla tutor aziendale prima dell'inizio dello stage.

#### EmbedList

*EmbedList* è la vista principale dell'applicazione, da cui l'utente può cercare, tramite la barra di ricerca, e visualizzare in una tabella tutte le condivisioni per un determinato contenuto. Tramite questa vista è inoltre possibile creare una nuova condivisione, cliccando sul bottone *Add Share*, oppure scegliere se eliminare o modificare una condivisione già presente nella tabella con l'elenco delle condivisioni (Figura 4.8).
Quando l'utente vuole modificare o creare una nuova condivisione viene aperta una finestra modale utilizzando un metodo *factory* implementato nella libreria THRON *Models*.
Questa vista utilizza i metodi contenuti nel file *Share.js* 4.3.1 per creare, aggiornare o eliminare una condivisione quando l'API *promise* viene soddisfatta da *EmbedCodeModal* 4.3.3


**Figura 4.8:** Vista principale dell'applicazione

#### EmbedCodeModal

La vista *EmbedCodeModal* rappresenta una finestra modale per la creazione o la modifica di una condivisione. Viene utilizzata un'unica vista per effettuare le due diverse operazioni, che vengono differenziate tramite una *props* iniettata da *EmbedList* al momento della scelta dell'operazione da parte dell'utente (Figura 4.9). All'interno di questa vista vengono utilizzati i componenti presenti nel modulo *Components*. Questa finestra modale può essere anche visualizzata in modalità *full screen*, cliccando sull'icona ⛶.
In caso di creazione di una nuova condivisione, il bottone per generare il codice di *embed* viene abilitato solo dopo aver impostato almeno una proprietà tra quelle disponibili, nel qual caso viene aperta un'ulteriore modale contenente lo *snippet* di codice da poter copiare (*CodeSnippetModal* 4.3.3).
Nel caso l'utente provi a chiudere la modale senza aver salvato le modifiche e generato il codice, viene mostrato un *alert* per chiedere la conferma o meno del salvataggio delle modifiche prima di ritornare alla vista principale (Figura 4.10).

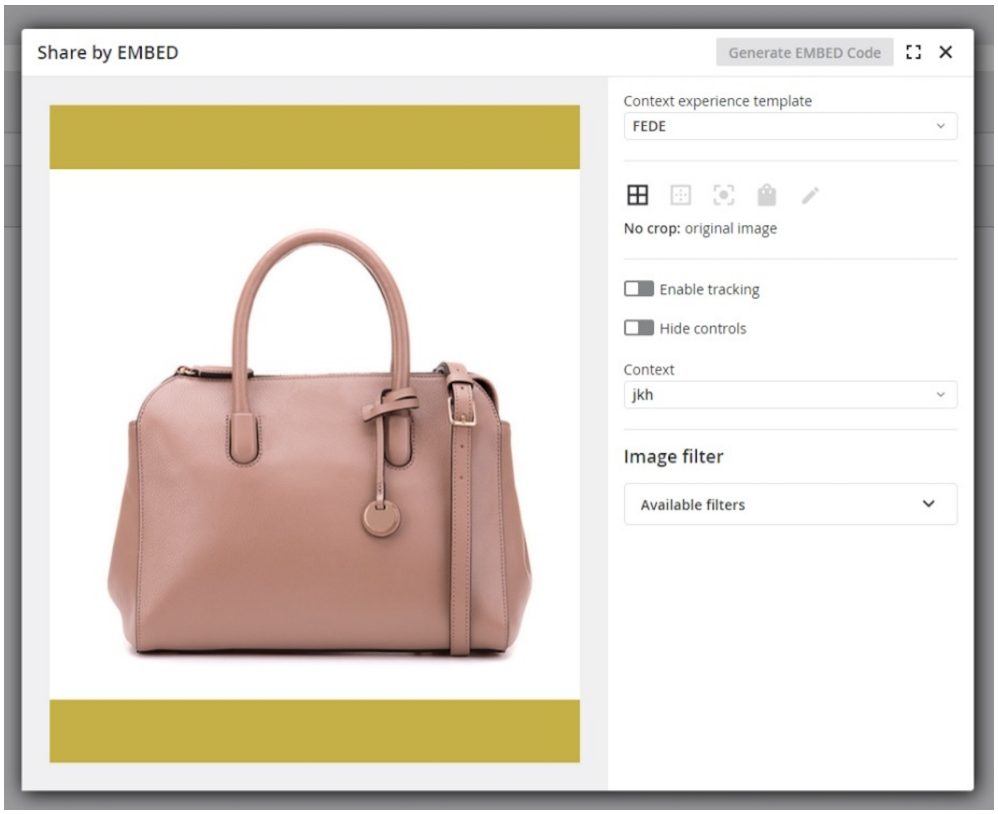


**Figura 4.9:** Modale per la creazione/modifica di una condivisione

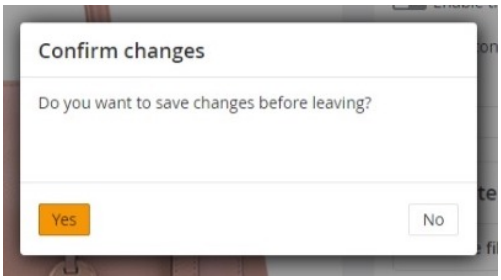


**Figura 4.10:** *Alert* per richiedere la conferma da parte dell'utente sul salvataggio delle modifiche effettuate prima di chiudere la modale

### CodeSnippetModal

La vista *CodeSnippetModal* rappresenta la finestra modale in cui viene mostrato il codice di *embed* da poter copiare per condividere il contenuto su diversi canali (Figura 4.11). Il codice generato può essere copiato in toto, utilizzando il bottone *Copy Embed Code*, oppure parzialmente selezionando con il cursore del mouse le righe da copiare.

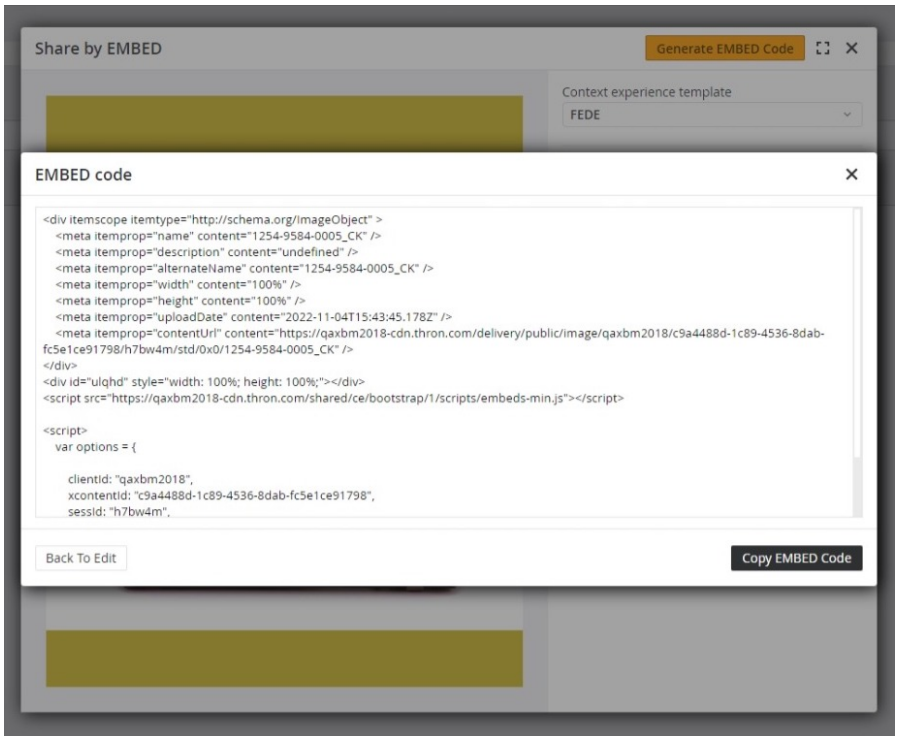


**Figura 4.11:** Modale con il codice di *embed* generato

## 4.4 Problematiche riscontrate

### 4.4.1 Token Protection

Durante l'implementazione della funzionalità di ritaglio manuale di un'immagine è stato riscontrato un problema riguardante la *Token Protection* dei dati di un contenuto. I dati di un contenuto di tipo immagine, ovvero l'URL sorgente e i valori delle dimensioni originali, venivano resi inaccessibili allo scadere di un *timeout* sul *token* della sessione utente, a causa del quale l'immagine spariva se il ritaglio manuale veniva effettuato dopo 4 minuti a partire dall'avvio dell'applicazione.
Tale problema inizialmente non era stato considerato, poiché la *Token Protection* viene attivata solo in alcuni codici servizio, tra cui quello utilizzato per il *deploy* dell'applicazione ma non quello utilizzato per lo sviluppo.
Il modo in cui è stato possibile risolvere questa problematica è stato quello di effettuare nuovamente la chiamata API per richiedere i dati del contenuto e rimpiazzare l'attributo *src* richiesto da *Vue Cropper* per ricaricare l'immagine non appena veniva segnalata la scadenza del *token*.
Nonostante tale soluzione risulti poco efficiente, su consiglio della tutor aziendale si è scelto di non impiegare troppe energie nel cercare soluzioni alternative dato che la *Token Protection* viene disabilitata negli altri codici servizio di *deploy*.

### 4.4.2 Aggiornamento del *player*

Il *player*, racchiuso in un *iframe* HTML, non veniva inizialmente aggiornato quando veniva cambiato il *template* grafico. Tale problema, che si è rivelato più ostico del previsto, era causato dal fatto che il *player*, una volta inizializzato, mantiene in memoria i dati del *template* che non possono quindi essere aggiornati.
Il problema è stato quindi risolto distruggendo e creando nuovamente l'*iframe*, e quindi anche il *player*, ogni volta che viene cambiato il *template*.

# Capitolo 5

# Attività di verifica

*In questo capitolo vengono descritte le attività di verifica svolte al fine di garantire una buona qualità del prodotto. In particolare vengono illustrati i test implementati e vengono descritte le migliorie future del prodotto proposte a seguito di un confronto con il team di lavoro.*

## 5.1 Test di unità

Durante lo svolgimento dell'attività di codifica del progetto di stage sono stati effettuati dei test di unità sui componenti implementati per verificarne il corretto funzionamento. In particolare, i test sono stati definiti e implementati al termine di ciascuno dei seguenti incrementi del prodotto:

- **I incremento:** realizzazione delle funzionalità obbligatorie per effettuare la condivisione di un contenuto di tipo immagine;
- **II incremento:** realizzazione della funzionalità di impostazione dei filtri per un contenuto di tipo immagine;
- **III incremento:** estensione delle funzionalità per supportare le altre tipologie di contenuto.

Sono stati testati i componenti *ImageFilter*, *AdvancedOptions*, *Cropper*, *CropOptions* e *PlayerOptions*. Il funzionamento del componente *RangeSlider* è stato testato tramite il componente padre *ImageFilter*, mentre il componente *Player* è stato escluso dai test poiché incapsula semplicemente il *player* THRON, il cui funzionamento è stato dato per corretto.
Sono stati implementati anche dei test per la vista principale *EmbedList*, al fine di controllare il corretto funzionamento delle chiamate API effettuate per creare, eliminare o aggiornare una condivisione. Per verificare il funzionamento delle chiamate con parametri noti sono stati creati degli oggetti *mock*, ovvero degli oggetti fittizi, da confrontare con gli oggetti ritornati dalle chiamate.
I test di unità sono stati effettuati utilizzando il *framework* Vitest[23], mentre la copertura del codice è stata calcolata utilizzando come *provider* *c8*, i cui risultati sono illustrati in Figura 5.1.

% Coverage report from c8

| File | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s |
|---|---|---|---|---|---|
| All files | 90.61 | 86.66 | 61.76 | 90.61 | |
| components/advancedOptions | 93.87 | 95.65 | 80 | 93.87 | |
| AdvancedOptions.test.js | 100 | 100 | 100 | 100 | |
| AdvancedOptions.vue | 88.67 | 92.85 | 80 | 88.67 | 78-92,128-130 |
| advancedOptionsMocks.js | 100 | 100 | 100 | 100 | |
| i18n.js | 100 | 100 | 100 | 100 | |
| components/cropOptions | 93.84 | 83.92 | 85.71 | 93.84 | |
| CropOptions.test.js | 100 | 100 | 100 | 100 | |
| CropOptions.vue | 88.64 | 79.54 | 85.71 | 88.64 | 51-53,58-65,74-81,131,181-182,193-196,198-210,273-275 |
| cropOptionsMocks.js | 100 | 100 | 100 | 100 | |
| i18n.js | 100 | 100 | 100 | 100 | |
| components/cropper | 83.24 | 100 | 33.33 | 83.24 | |
| Cropper.test.js | 100 | 100 | 100 | 100 | |
| Cropper.vue | 79.19 | 100 | 33.33 | 79.19 | 55,57,69-74,76-81,83-87,89-92,112-119 |
| components/imageFilter | 93.19 | 94.11 | 54.54 | 93.19 | |
| ImageFilter.vue | 89.84 | 88.88 | 54.54 | 89.84 | 36-38,42-44,58-60,74-76,106-108,126-127,133-135 |
| i18n.js | 100 | 100 | 100 | 100 | |
| imageFilter.test.js | 100 | 100 | 100 | 100 | |
| components/playerOptions | 100 | 100 | 100 | 100 | |
| PlayerOptions.test.js | 100 | 100 | 100 | 100 | |
| PlayerOptions.vue | 100 | 100 | 100 | 100 | |
| i18n.js | 100 | 100 | 100 | 100 | |
| playerOptionsMocks.js | 100 | 100 | 100 | 100 | |
| views/embedList | 75.76 | 37.5 | 33.33 | 75.76 | |
| EmbedList.test.js | 100 | 100 | 100 | 100 | |
| EmbedList.vue | 69.65 | 37.5 | 33.33 | 69.65 | 56-67,82-87,91-95,97-101,107-115,118-131,134-146,152-155,160-162 |
| i18n.js | 100 | 100 | 100 | 100 | |

**Figura 5.1:** Percentuale di codice coperto dai test di unità

Sia *Branch Coverage*, ovvero la percentuale di copertura dei rami di codice, che *Lines Coverage*, ovvero la percentuale di copertura delle linee di codice, risultano essere superiori alla soglia di accettabilità dell'80% posta come vincolo obbligatorio dal committente.

## 5.2 Collaudo

Dopo aver implementato e superato i test di unità definiti, l'applicazione è stata collaudata nei seguenti *web browser*:

* Google Chrome v. 103
* Firefox v.101
* Safari v.14
* Opera v.88
* Edge v.102

Sono stati effettuati quindi i test di accettazione, o collaudo, per accettarsi che il prodotto rispettasse i requisiti richiesti e non presentasse malfunzionamenti o anomalie particolari.
Il collaudo è stato effettuato dalla tutor aziendale e dagli altri membri del *team* Contenuti, su dispositivi con sistemi operativi sia Mac che Windows.

## 5.3 Migliorie future

Al termine dell'attività di *testing* e collaudo dell'applicazione, si è svolto un incontro con i membri del *team* Contenuti per discutere di eventuali migliorie future da applicare al prodotto per aumentarne la qualità e l'efficienza.
Durante l'incontro sono state proposte dalla laureanda le seguenti migliorie per il prodotto:

* **Invertire il funzionamento della proprietà *Disable tracking*:** al momento l'utente può scegliere se disabilitare il tracciamento utente. Tuttavia risulterebbe più intuitivo poter **abilitare** il tracciamento, dato che di default un contenuto non viene tracciato dal sistema;
* **Fornire dei filtri predefiniti per un'immagine:** potrebbe risultare utile fornire all'utente non solo la possibilità di scegliere singolarmente i livelli di contrasto, saturazione, etc... ma anche una serie di *setup* predefiniti con dei livelli preimpostati, come avviene in molteplici editor fotografici.

# Capitolo 6

# Conclusioni

*Il capitolo conclusivo di questo documento ha lo scopo di presentare l'insieme degli obiettivi raggiunti e delle conoscenze acquisite al termine dell'attività di stage. Viene infine presentata una personale valutazione dell'esperienza di stage.*

## 6.1 Obiettivi raggiunti

Il progetto di stage svolto ha avuto come obiettivo primario la creazione di un'interfaccia grafica per consentire la condivisione di contenuti THRON su molteplici canali tramite codice di *embed*, utilizzando il *framework* Vue.js.
Nel presente documento di tesi sono state descritte le principali fasi di sviluppo che hanno portato alla realizzazione di tale progetto, partendo dall'attività di analisi dei requisiti. In questa fase iniziale dello sviluppo sono stati individuati i principali casi d'uso e requisiti del prodotto, poi classificati in base al loro livello di priorità.
Una volta individuati i requisiti, sono state analizzate le tecnologie e gli strumenti di sviluppo da utilizzare per il progetto ed è stata progettata l'architettura del sistema. Sono stati poi implementati i vari componenti dell'applicazione, basandosi sull'architettura precedentemente progettata.
Nel presente documento sono state anche riportate e descritte le principali problematiche riscontrate durante l'attività di codifica; le difficoltà incontrate nella gestione della *Token Protection* e per aggiornare correttamente il *player* THRON non sono state tuttavia particolarmente onerose sia per tempo che per risorse spese.
Infine sono state riportate le attività di verifica e di validazione svolte dalla laureanda al fine di verificare il corretto funzionamento del prodotto finale. Per soddisfare tale obiettivo sono stati implementati dei test di unità ed è stato effettuato un collaudo su diversi *web browser* anche dagli altri membri del *team* di lavoro.
Sono state inoltre discusse con il *team* alcune migliorie che potrebbero essere applicate al prodotto al fine di migliorarne ulteriormente l'usabilità e l'efficienza.

Vengono ora riassunti i requisiti che risultano soddisfatti al termine delle attività di sviluppo del progetto.
La Tabella 6.1 descrive quali obiettivi, a partire da quelli definiti in fase di pianificazione del lavoro, sono stati effettivamente rispettati. Per la notazione utilizzata per identificare gli obiettivi si rimanda alla sezione 2.3.1.

| **Obiettivo** | **Descrizione** | **Stato** |
|---|---|---|
| OB1 | Realizzazione di un'interfaccia in Vue.js per la condivisione di un contenuto di tipo immagine tramite codice di *embed* | Completato |
| OB2 | Documentazione delle funzionalità implementate | Completato |
| OB3 | Realizzazione di test di unità delle funzionalità implementate | Completato |
| DE1 | Implementazione della funzionalità di applicazione di filtri sulle condivisioni di contenuti di tipo immagine | Completato |
| DE2 | Estensione della condivisione per gli altri tipi di contenuto THRON (video, audio, documenti, etc.) | Completato |
| DE3 | Implementazione della funzionalità per condividere un contenuto tramite URL | Non completato |
| OP1 | Implementazione della funzionalità per condividere un contenuto tramite pagina di *share* | Non completato |

**Tabella 6.1:** Tabella di resoconto degli obiettivi completati

Risultano quindi completati tutti gli obiettivi obbligatori e la maggior parte degli obiettivi desiderabili, mentre l'unico obiettivo opzionale non è stato soddisfatto. In accordo con la *tutor* aziendale infatti si è scelto di utilizzare le ore riservate all'implementazione delle altre due modalità di condivisione, ovvero tramite URL e pagina di *share*, per migliorare la qualità del codice e rendere più modulari, efficienti e mantenibili i componenti sviluppati per le funzionalità più importanti.
Prendendo infatti come riferimento le Tabelle per il tracciamento dei requisiti 3.1, 3.2 e 3.3, redatte in fase di analisi dei requisiti, risultano soddisfatti tutti e 38 i requisiti obbligatori, 29 requisiti desiderabili su 30 e 1 requisito opzionale su 4, per un totale di 68 requisiti soddisfatti su 71 requisiti totali definiti.

## 6.2 Consuntivo orario finale

In questa sezione viene riportato il consuntivo orario finale del progetto, al fine di evidenziare e motivare alcune variazioni rispetto al piano di lavoro redatto inizialmente con l'azienda.

**1. Introduzione al contesto del prodotto THRON**
**Obiettivi:** studio della piattaforma THRON DAM e in particolare analisi del comportamento dell'attuale interfaccia per la condivisione dei contenuti.
**Durata prevista:** 16 ore.
**Durata effettiva:** 16 ore.
**Motivazioni variazione:** -.

**2. Formazione tecnologica**
**Obiettivi:** formazione sull'utilizzo del *framework* Vue.js e sulle altre tecnologie e strumenti da utilizzare per il progetto.
**Durata prevista:** 40 ore.
**Durata effettiva:** 48 ore.
**Motivazioni variazione:** lo studio del *framework* Vue.js ha richiesto 1 giorno lavorativo in più per approfondire alcune caratteristiche, come ad esempio come utilizzare *watchers* e *computed properties* per osservare i cambiamenti di stato delle variabili reattive.

**3. Analisi dei requisiti e casi d'uso**
**Obiettivi:** analisi dei requisiti e individuazione dei casi d'uso.
**Durata prevista:** 8 ore.
**Durata effettiva:** 24 ore.
**Motivazioni variazione:** sono stati impiegati 2 giorni lavorativi in più per l'analisi dei requisiti, in particolare per identificare i requisiti relativi alla funzionalità di ritaglio manuale di un'immagine e per scegliere il componente grafico più adatto come strumento di ritaglio.

**4. Realizzazione funzionalità per la creazione di *embed* per immagini**
**Obiettivi:** progettazione e implementazione delle funzionalità obbligatorie per contenuti di tipo immagine.
**Durata prevista:** 80 ore.
**Durata effettiva:** 80 ore.
**Motivazioni variazione:** -.

**5. Realizzazione *mocks* per test di unità**
**Obiettivi:** implementazione degli oggetti *mock* per testare le chiamate API dell'applicazione.
**Durata prevista:** attività non prevista.
**Durata effettiva:** 4 ore.
**Motivazioni variazione:** inizialmente non erano previsti dei test per le chiamate API effettuate dai componenti.

**6. Implementazione test unità**
**Obiettivi:** definizione e implementazione dei test di unità per le componenti sviluppate.
**Durata prevista:** 32 ore.
**Durata effettiva:** 24 ore.
**Motivazioni variazione:** grazie alla semplicità di utilizzo del *framework* Vite-

st e alla creazione dei *mocks* è stato possibile risparmiare 1 giorno lavorativo per l'implementazione dei test di unità.

**7. Aggiunta funzionalità filtri sulla condivisione di immagini**
**Obiettivi:** implementazione della funzionalità di applicazione di filtri per contenuti di tipo immagine.
**Durata prevista:** 24 ore.
**Durata effettiva:** 40 ore.
**Motivazioni variazione:** l'implementazione dei filtri da applicare ad un contenuto di tipo immagine ha richiesto 2 giorni lavorativi in più a causa della creazione non preventivata del componente *custom RangeSlider* e della necessità di dover trasporre i valori massimi e minimi definiti inizialmente per ogni filtro in valori più intuitivi per l'utente finale.

**8. Supporto condivisione di contenuti di altre tipologie**
**Obiettivi:** aggiunta del supporto per la condivisione di altri tipi di contenuto oltre alle immagini.
**Durata prevista:** 32 ore.
**Durata effettiva:** 32 ore.
**Motivazioni variazione:** -.

**9. Implementazione condivisione tramite URL**
**Obiettivi:** implementazione della condivisione di contenuti tramite URL invece che solo tramite codice di *embed*.
**Durata prevista:** 16 ore.
**Durata effettiva:** 0 ore.
**Motivazioni variazione:** modalità di condivisione non implementata.

**10. Implementazione condivisione tramite pagine di *share***
**Obiettivi:** implementazione della condivisione di contenuti tramite pagine di *share* in aggiunta alle altre due modalità di condivisione (codice di *embed* e URL).
**Durata prevista:** 16 ore.
**Durata effettiva:** 0 ore.
**Motivazioni variazione:** modalità di condivisione non implementata.

**11. Recupero ore**
**Obiettivi:** ore aggiuntive da utilizzare per *bug fixing* e imprevisti di percorso.
**Durata prevista:** 8 ore.
**Durata effettiva:** 16 ore.
**Motivazioni variazione:** è stato impiegato 1 giorno lavorativo in più per migliorare il codice e fare *bug fixing* al termine delle attività di *testing* sui componenti implementati.

**12. Stesura documentazione**
**Obiettivi:** stesura della documentazione necessaria per il progetto.
**Durata prevista:** 24 ore.
**Durata effettiva:** 28 ore.
**Motivazioni variazione:** la stesura della documentazione ha richiesto mezza giornata lavorativa in più, per via della redazione di un documento non formale aggiuntivo riguardo alle scelte architetturali effettuate in fase di progettazione.

**13. Preparazione presentazione interna del lavoro svolto**
**Obiettivi:** preparazione della presentazione del lavoro svolto da condividere con i vari *team* aziendali.

**Durata prevista:** 24 ore.
**Durata effettiva:** 8 ore.
**Motivazioni variazione:** la creazione della presentazione del lavoro svolto ha richiesto 2 giorni lavorativi in meno grazie alla buona stesura della documentazione.

## 6.3 Conoscenze e capacità acquisite

Il progetto di stage svolto mi ha sicuramente permesso di acquisire e migliorare molte conoscenze e abilità, in aggiunta a quelle acquisite durante il percorso di studi universitari.
In primis è stato molto istruttivo e stimolante studiare il funzionamento e le caratteristiche di una piattaforma di Digital Asset Management, di cui non conoscevo nulla prima di iniziare l'attività di stage. Ritengo molto formativo anche l'aver dovuto dedurre, su consiglio della tutor aziendale, alcune funzionalità e parte della struttura dell'interfaccia da sviluppare basandomi non tanto sulla documentazione interna ma ispezionando l'attuale GUI utilizzata per la condivisione di contenuti. Questo approccio non convenzionale mi ha permesso di capire più velocemente il comportamento del sistema e soprattutto come venivano strutturati i parametri e le risposte delle chiamate ai servizi aziendali.
Anche il fatto di imparare ad utilizzare un *web framework* come Vue.js è stata un'attività a mio parere altamente formativa. *Framework* come questo vengono infatti ormai utilizzati per lo sviluppo della maggior parte delle applicazioni web e la loro conoscenza viene sempre più spesso richiesta dalle aziende del settore nella ricerca di nuovo personale di lavoro.
L'attività di formazione tecnologica mi ha inoltre permesso di approfondire ulteriormente le conoscenze, precedentemente acquisite nel corso di Tecnologie Web, riguardo ai linguaggi web e in particolare di JavaScript, linguaggio che non ero riuscita ad utilizzare in maniera così sostanziosa durante il progetto previsto per il superamento del corso.
L'abilità principale acquisita grazie all'attività di stage svolta è stata tuttavia la capacità di lavorare efficientemente all'interno di un *team* di sviluppo. Nonostante infatti durante il percorso di studi ci siano state molteplici occasioni di lavorare in gruppo per lo sviluppo di un progetto, la scarsa o in alcuni casi inesistente collaborazione con gli altri membri del gruppo non mi aveva ancora permesso di apprezzare i vantaggi del lavoro in *team*.
Anche l'apprendimento della metodologia *agile* Scrum è stata altamente istruttiva e mi ha permesso di mettere in pratica le conoscenze acquisite durante il corso di Ingegneria del Software.

## 6.4 Valutazione personale dell'esperienza

Ritengo l'esperienza di stage svolta molto istruttiva e stimolante, sia da un punto di vista formativo che da un punto di vista umano. Imparare nuovi *framework* e metodi di lavoro è stato per me molto interessante, soprattutto perché mi ha permesso di mettere in pratica molte conoscenze acquisite durante il percorso di studi.
Lavorare all'interno di un *team* molto giovane e dinamico, in cui ci si confronta quotidianamente e in maniera costruttiva per trovare soluzioni alternative e innovative ai problemi che inevitabilmente fanno parte di una realtà aziendale, è stato molto incoraggiante e motivante, nonostante la mia diffidenza iniziale dovuta a insoddisfacenti lavori di gruppo svolti durante il percorso di studi.
In particolare il supporto, l'esperienza e la professionalità della tutor aziendale mi hanno permesso di sviluppare un prodotto di buona qualità e che soddisfacesse la maggior parte dei requisiti richiesti senza troppi intoppi o difficoltà, lasciandomi comunque larga autonomia nelle principali scelte progettuali e di sviluppo durante tutta la durata dell'attività di stage.
L'unico aspetto che non ha interamente soddisfatto le mie aspettative sull'esperienza di stage è stato lo scarso interesse dell'azienda riguardo la definizione dei test per verificare il funzionamento dell'applicazione. Nel contesto aziendale infatti spesso non vengono effettuati test e documentazioni adeguate; mi sono quindi dovuta documentare in completa autonomia sulle tecniche e gli strumenti per effettuare i test di unità.
Nonostante questo aspetto, valuto in maniera estremamente positiva l'esperienza di stage svolta e la ritengo una delle attività più utili e formative all'interno del mio percorso di studi universitari.

# Acronimi e abbreviazioni

**DAM** Digital Asset Management. 1, 50

**DOM** Document Object Model. 27, 50

**GUI** Graphical User Interface. 4

**MVC** Model-View-Controller. 27, 50

**MVVM** Model-View-ViewModel. 27, 29, 50

**SFC** Single-File Component. 29, 30, 49

**UML** Unified Modeling Language. 8, 50

# Glossario

**Angular** *framework* sviluppato da Google che utilizza il linguaggio TypeScript per la creazione di interfacce utente. 27, 50

**Codice di *embed*** frammento di codice HTML da inserire in un documento, ad esempio una pagina web, per incorporare e leggere un contenuto multimediale, caricando il contenuto direttamente dalla fonte tramite un *player*. ii, v, 4–7, 11, 15, 24, 31, 36–38, 43, 44, 50

**DAM** un *Digital Asset Management* (DAM) è un sistema integrato per la gestione strategica centralizzata dei contenuti. Si tratta quindi di un software che consente di creare, organizzare e distribuire contenuti su differenti canali, come ad esempio siti web e applicazioni, da un unico punto, al fine di aumentare l'efficacia della comunicazione.. 47, 49

**DOM** standard W3C di rappresentazione di documenti strutturati come modello orientato agli oggetti. 27, 49

**MVC** *design pattern* architetturale in cui la *View*, l'interfaccia utente, e il *Model*, ovvero la logica del sistema, comunicano tra loro attraverso un *Controller*. 29, 49

**MVVM** *design pattern* architetturale in cui la *View*, l'interfaccia utente, e il *Model*, ovvero la logica del sistema, sono collegati dal *ViewModel*, che rappresenta tutte le informazioni e i comportamenti della corrispondente *View*. 29, 49

**npm** gestore di pacchetti predefinito per l'ambiente di runtime JavaScript Node.js. 25, 26, 50

**React** *framework* JavaScript per la creazione di interfacce utente basate su componenti. Utilizza il *pattern* MVC. 27, 50

**UML** in ingegneria del software *UML, Unified Modeling Language* (ing. linguaggio di modellazione unificato) è un linguaggio di modellazione e specifica basato sul paradigma object-oriented. L'*UML* svolge un'importantissima funzione di "lingua franca" nella comunità della progettazione e programmazione a oggetti. Gran parte della letteratura di settore usa tale linguaggio per descrivere soluzioni analitiche e progettuali in modo sintetico e comprensibile a un vasto pubblico. 49

# Bibliografia

## Riferimenti bibliografici

[30] Ian Sommerville. *Software Engineering, 9th ed.* Pearson, 2010.

## Siti web consultati

[1] *Sito THRON S.p.A.* URL: https://www.thron.com/it/ (cit. a p. 1).

[2] *Manifesto per lo Sviluppo Agile del Software.* URL: http://agilemanifesto.org/iso/it/manifesto.html (cit. a p. 1).

[3] *Sito ufficiale Scrum.org.* URL: https://www.scrum.org (cit. a p. 1).

[4] *Guida Scrum.* URL: https://scrumguides.org/scrum-guide.html (cit. a p. 2).

[5] *Sito ufficiale Jira Software.* URL: https://www.atlassian.com/it/software/jira (cit. a p. 3).

[6] *Sito ufficiale Microsoft 365.* URL: https://www.microsoft.com/it-it/microsoft-365?rtc=1 (cit. a p. 3).

[7] *Sito ufficiale Amazon Web Services.* URL: https://aws.amazon.com/it/ (cit. a p. 3).

[8] *Sito ufficiale Fork.* URL: https://git-fork.com (cit. a p. 3).

[9] *Sito ufficiale Confluence.* URL: https://www.atlassian.com/it/software/confluence (cit. a p. 3).

[10] *Sito ufficiale Figma.* URL: https://www.figma.com (cit. a p. 3).

[11] *Documentazione Context Experience Player THRON.* URL: http://hub-cdn.thron.com/shared/ce/assets/1.1.38.000/doc/public/index.html (cit. alle pp. 8, 35).

[12] *Sito ufficiale StarUML.* URL: https://staruml.io (cit. a p. 8).

[13] *Documentazione Vue cropper.js.* URL: https://github.com/Agontuk/vue-cropperjs (cit. alle pp. 25, 28).

[14] *Documentazione Vue Advanced cropper.* URL: https://advanced-cropper.github.io/vue-advanced-cropper (cit. a p. 25).

[15] *Documentazione Cropper.js*. URL: https://fengyuanchen.github.io/cropperjs (cit. a p. 25).

[16] *Documentazione ufficiale Vue.js*. URL: https://vuejs.org/guide/introduction.html (cit. alle pp. 27, 29).

[17] *Guida al pattern MVVM in Vue.js*. URL: https://012.vuejs.org/guide/index.html (cit. a p. 27).

[18] *Documentazione W3C di HTML5*. URL: https://dev.w3.org/html5/spec-LC (cit. a p. 28).

[19] *Documentazione ufficiale Sass*. URL: https://sass-lang.com/documentation (cit. a p. 28).

[20] *Documentazione JavaScript*. URL: https://www.w3schools.com/js (cit. a p. 28).

[21] *Documentazione ufficiale Node.js*. URL: https://nodejs.org/it (cit. a p. 28).

[22] *Documentazione ufficiale Vite.js*. URL: https://vitejs.dev (cit. a p. 28).

[23] *Documentazione ufficiale Vitest*. URL: https://vitest.dev (cit. alle pp. 28, 40).

[24] *Documentazione c8*. URL: https://github.com/bcoe/c8 (cit. a p. 28).

[25] *Documentazione Vue I18n*. URL: https://vue-i18n.intlify.dev (cit. a p. 28).

[26] *Documentazione Vue Router*. URL: https://router.vuejs.org (cit. a p. 28).

[27] *Documentazione THRON Models*. URL: https://hub-cdn.thron.com/shared/lib/common/models/4.9.0/docs (cit. alle pp. 28, 31).

[28] *Documentazione THRON Components*. URL: https://examples-cdn.thron.com/shared/lib/common/vuecomponents/2.9.0/docs/getting-started (cit. alle pp. 28, 32).

[29] *Documentazione THRON Helpers*. URL: https://hub-cdn.thron.com/shared/lib/common/helpers/1.1.0/docs (cit. a p. 28).