

Software di gestione di una radio universitaria

17 Nov 2021

Introduzione	2
Requisiti Informali	3
Requisiti del Contesto (Requisiti)	6
Requisiti funzionali	6
Requisiti non funzionali	7
Tecnologie utilizzate	7
Requisiti del Software (Analisi)	8
Use Case Diagram	8
Progetto del Software	11
Database	11
Creazione dell'interfaccia grafica	11
Class Diagram	14
Activity Diagram	18
Sequence Diagram	19
Component Diagram	20
Deployment Diagram	21
Verifica e Validazione (Test)	21
Problematiche riscontrate	22

Metodi di collaudo usati	22
Bugs	25
Test funzionali: test case	25
Project Management	28
Metodologia Agile	28
Scrum	28
Scrum - Step 1: Calendario e cadenza	28
Scrum - Step 2: Product Backlog	29
Scrum - Step 3: Sprint Planning	32
Sprint 1	32
Sprint 1 - Backlog	33
Sprint 1 - Planning	34
Sprint 1 - Analisi	35
Sprint 1 - Review	38
Sprint 1 - Review: Retrospective	38
Sprint 2	38
Sprint 2 - Backlog	38
Sprint 2 - Planning	39
Sprint 2 - Analisi	39
Sprint 2 - Review	40
Sprint 2 - Review: Retrospective	40
Sprint 3	40
Sprint 3 - Backlog	40
Sprint 3 - Planning	41
Sprint 3 - Analysis	41
Sprint 3 - Review	42
Sprint 3 - Review: Retrospective	42
SCREENSHOT DEL SOFTWARE	43

Introduzione

Radio UniVersoMe è la radio ufficiale degli studenti dell'Università degli Studi di Messina.

La mission principale e' quella di insegnare agli studenti a stare dietro ad un microfono, cosa molto importante al giorno d'oggi in un mondo digitale fatto di podcast, video e conferenze. I contenuti della radio spaziano su vari campi come lo sport, la musica, notizie dal mondo e dall'Ateneo.

Va in onda dal Lunedì' al Venerdì' pubblicando dei contenuti musicali e divulgativi rivolti a studenti, professori universitari UniMe.

Attualmente va in onda su una piattaforma di streaming che consente di conservare le puntate live e di creare dei Programmi composti da Episodi.

L'ascolto di tali episodi e' possibile collegandosi alla piattaforma in questione tramite Browser o Applicazione mobile proprietaria della piattaforma.

L'applicazione e' facilmente accessibile sia agli utenti non registrati, ovvero i semplici ascoltatori, sia agli utenti che vogliono accedere per creare la propria lista dei preferiti.

Un'altra tipologia di utente e' lo speaker radiofonico che puo' aggiornare i contenuti multimediali memorizzati nel database.

Il progetto UniVersoMe vuole ampliarsi creando un software che si in grado di riprodurre questi contenuti multimediali ed offrire agli studenti un luogo di ascolto dove potersi incontrare, ascoltare la radio in compagnia col fine di creare opinioni e poter commentare tra colleghi cio' che accade nel nostro Ateneo, nella Provincia di Messina, ma anche nel mondo.

Il software è pensato per essere eseguito su un dispositivo dotato di schermo interattivo e degli altoparlanti che consentano di riprodurre la musica con una buona qualità audio. Il tutto piazzato in degli spazi comuni (come le sale studio, o il bar del dipartimento) dove gli studenti si riuniscono di frequente per scambiare idee e opinioni della vita quotidiana, sullo studio e sulle opportunità lavorative. Trattandosi di uno spazio condiviso, deve essere possibile spegnere la riproduzione per evitare confusione e disturbo durante le ore di studio.

Requisiti Informali

Si vuole realizzare un sistema software capace di riprodurre dei brani musicali o dei programmi radio pubblicati sulla piattaforma, in grado di dare all'utente un'interfaccia dalla quale potrà salvare i contenuti preferiti o ricercare i podcast di suo interesse.

Il sistema deve consentire agli studenti l'ascolto di contenuti multimediali realizzati dalla radio universitaria di Ateneo. Il sistema deve consentire agli utenti generici di prendere visione del catalogo delle puntate radio disponibili, suddivisi per programmi radiofonici, pubblicati sulla piattaforma di Spreaker. **Opzionale:** suddivisi in base alla partecipazione degli speaker che lo ha prodotto. In aggiunta a ciò, il sistema deve consentire agli studenti iscritti alla piattaforma di conservare in una playlist uno o più puntate. Lo studente deve avere la possibilità di sfogliare il catalogo dei programmi, visualizzando, per ogni programma,, una preview parziale contenente il titolo, l'immagine, e lo speaker.

Ogni utente ha la possibilità di registrarsi o di interagire con il player come utente non autenticato. In ogni caso deve poter cercare i programmi pubblicati sulla piattaforma, poterli riprodurre.

Esiste un'altra tipologia di utente che è lo Speaker di UniVersoMe, che oltre ai permessi degli altri utenti deve poter pubblicare, modificare i podcast.

Il programma si presenta come una raccolta di episodi, ognuno con un file di tipo .mp3, un titolo, una descrizione, e degli speaker.

Il software si presenterà dunque con un'interfaccia stile Media Player, con un menu inferiore che consente la navigazione tra le varie puntate pubblicate. Sempre sul menu superiore sarà possibile accedere alla sezione di registrazione o di accesso.

I contenuti (programmi, episodi, playlist musicali) già pubblicati risiedono sui server. Per questo motivo dobbiamo utilizzare le API di Spreaker per accedere a tali risorse. Utilizzando le API di Spreaker sarà possibile gestire il canale in base alla tipologia di utente. Quindi sarà possibile effettuare delle ricerche, come anche pubblicare ed aggiungere nuovi Podcast.

Obiettivi futuri saranno quelli di convertire quelli che ora sono dei Podcast, in Programmi Radio.

Il programma radio differisce dal podcast per la progressione dei contenuti. Mentre nel Podcast ci si focalizza su un particolare argomento da portare avanti arricchendolo con nuovi contenuti strettamente legati all'argomento e che vanno a creare un percorso di nozioni per trattare tutti i temi correlati, e che portano ad una conclusione.

Il programma radiofonico non ha una fine; esso ha un focus ben preciso su un determinato tema o ambito e gli argomenti che verranno trattati ad ogni puntata potrebbero essere completamente slegati da quelli precedenti per via dei fatti e che lo speaker vuole raccontare all'ascoltatore.

I contenuti (programmi, episodi, playlist musicali) già pubblicati risiedono sui server. Per questo motivo dobbiamo utilizzare le API di Spreaker per accedere a tali risorse. Utilizzando le API di Spreaker sarà possibile gestire il canale in base alla tipologia di utente. Quindi sarà possibile effettuare delle ricerche, come anche pubblicare ed aggiungere nuovi Podcast.

Server di appoggio

Per interagire con Spreaker e creare le tipologie di utente, e' necessario utilizzare un server che offre un database dove memorizzare i dati dell'utente, ed utilizza le API per interagire con Spreaker.

Ascoltatore non loggato:

Potra' ascoltare i contenuti, cercare degli argomenti in base ai tag o al titolo del podcast,
ascoltare la puntata live e navigare tra i vari episodi pubblicati

Ascoltatore loggato:

Tutto quello che puo' fare l'ascoltatore non loggato, ed in piu puo creare delle proprie playlist

Speaker:

Pubblica i contenuti in base al canale

Requisiti del Contesto (Requisiti)

Si definiscono i requisiti del contesto consultandosi con gli stakeholders, ovvero tutti quei soggetti interessati alla realizzazione del prodotto commerciale, in questo caso un software per la riproduzione di contenuti multimediali creati dalla redazione di UniVersoMe.

I requisiti del progetto emersi da un confronto iniziale con gli stakeholders sono:

Requisiti funzionali

Di seguito e' riportato l'elenco dei requisiti funzionali divisi per stake holder:

Utente anonimo:

- Visualizzare il catalogo dei programmi
- Visualizzare la schermata della live

- Visualizzare il catalogo degli avvisi universitari
- Visualizzare il catalogo dei contenuti musicali
- Ricerca di un contenuto multimediale.
- Avviare l'ascolto della LIVE
- Filtrare gli avvisi UniMe in base al dipartimento o Corso di Laurea.
- Filtrare il catalogo musicale in base al genere.
- Registrarsi al servizio
- Accedere alla pagina di login

Utente Ascoltatore (Listener):

- Autenticarsi al servizio
- Visualizzare il catalogo dei programmi
- Visualizzare la schermata della live
- Visualizzare il catalogo degli avvisi universitari
- Visualizzare il catalogo dei contenuti musicali
- Aggiungere un catalogo musicale tra i preferiti.

Utente Speaker

- Autenticarsi al servizio
- Visualizzare il catalogo dei programmi
- Visualizzare la schermata della live
- Visualizzare il catalogo degli avvisi universitari
- Visualizzare il catalogo dei contenuti musicali
- Aggiungere o modificare i programmi
- Aggiungere o modificare la fonte del live streaming
- Aggiungere o modificare il catalogo degli avvisi universitari
- Aggiungere o modificare il catalogo dei contenuti musicali.

Requisiti non funzionali

Di seguito e' riportato l'elenco dei requisiti non funzionali del sistema:

- Usabilità: l'interfaccia utente del sistema è stata implementata cercando di garantire la massima operabilità, un veloce apprendimento e una facile localizzazione dei comandi da utilizzare. Viene garantita inoltre un'interfaccia coerente in tutte le sezioni dell'applicazione.
- Sicurezza: l'applicazione gestisce informazioni sensibili, pertanto deve garantire un determinato livello di sicurezza per preservarle. È stata perciò implementata una procedura di autenticazione che permette di separare i diversi profili utente garantendo in questo modo diversi livelli di privilegi e di funzioni utilizzabili.

Tecnologie utilizzate

Per la realizzazione di questa applicazione sono stati utilizzati diversi framework e diversi strumenti software che hanno permesso una semplificazione del lavoro:

Application Server:

- Spring Boot Framework

Database Server:

- PostgreSQL

Linguaggio di programmazione:

- Java OpenJDK

Interfaccia grafica

- OpenJFX

Requisiti del Software (Analisi)

Use Case Diagram

I diagrammi dei casi d'uso riportati rappresentano la mappa riassuntiva dei casi d'uso e riportano tutti gli attori (stakeholder), i goal e le relazioni fra loro.

Gli utenti che interagiscono con il sistema vengono chiamati *stakeholders*.

Il software consentirà agli utenti di mostrare una lista dei podcast pubblicati, poter ascoltare gli episodi. Oltre ai podcast, gli altri contenuti audio sono le LIVE le uADS, ovvero Notizie riguardanti il nostro Ateneo.

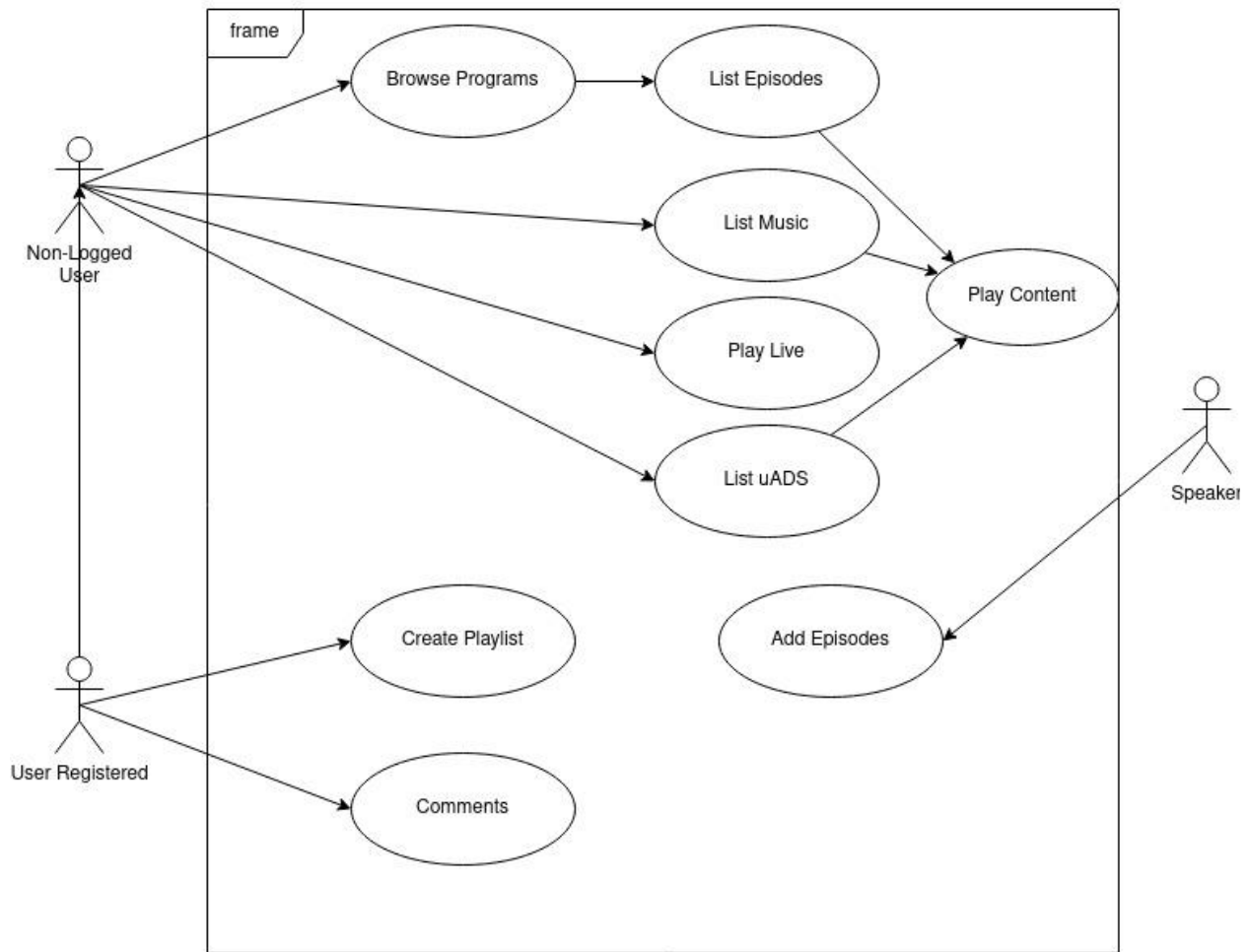
La Lista dei Podcast deve mostrare tutti i podcast pubblicati di Radio UniVersoMe e deve avere la possibilità di poterli aggiungere tra i preferiti se l'utente è registrato. Deve inoltre consentire l'aggiunta di Podcast da uno Speaker registrato.

Deve essere possibile ascoltare l'episodio, con la possibilità di metterlo in pausa, passare all'episodio successivo, inserirlo tra i preferiti.

La LIVE è una funzionalità che deve essere sempre in risalto, ovvero quando è presente una puntata live, l'utente deve accorgersene.

Le uADS sono delle notizie sul proprio dipartimento che possono interessare lo studente ed informarlo sulle opportunità. Avrà una ripetitività fissata a ogni ora.

Il canale musica è dedicato solo all'ascolto delle nuove uscite musicali degli artisti consigliati da Radio UVM.



Attore principale: Utente Anonimo

Visualizzare il catalogo dei contenuti multimediali informativi e di intrattenimento.

Passi:

Dalla dashboard iniziale e' possibile selezionare i programmi con le relative puntate, la live, i cataloghi della musica consigliata da UniVersoMe suddivisa in playlist in base al genere, gli avvisi universitari.

L'utente, per fruire del contenuto seleziona il programma ed il sistema visualizza l'elenco delle puntate disponibili per quel programma, ordinati per data.

Regole di utilizzo del servizio

Registrazione al servizio

1. Seleziona dal link in alto a destra la funzionalità 'Accedi
2. Il sistema visualizza un modulo per accedere o registrarsi
3. L'utente compila il modulo per la registrazione in tutti i campi e poi preme sul tasto di conferma.

Progetto del Software

Database

Il database dell'applicazione e' gestito da PostgreSQL. E' stato creato uno schema 'radiouvm_schema' al cui interno risiedono le tabelle:

- episode
- music
- playlist
- program
- university_ads
- user

Tali tabelle sono riempite con dei record in fase di creazione ed avvio del server.

La chiusura del server causa la perdita di tutti i records.

Creazione dell'interfaccia grafica

Iniziamo la progettazione partendo dal Client che verra' installato su dispositivi come Raspberry Pi e poi distribuiti per l'Ateneo. Questo Client si interfaccera' con il Server per la raccolta e registrazione degli utenti.

Welcome Screen

E' la schermata che accompagna l'apertura di un'applicazione. Spesso il solo scopo e' quello di migliorare esteticamente l'applicazione. Tuttavia potrebbe essere utilizzato insieme ad un animazione per il caricamento delle funzionalita' di base del software e mostrare lo stato di completamento dell'inizializzazione.

Home Page

La HomePage deve essere completa ed in grado di far comprendere all'utilizzatore tutte le funzionalita' e contenuti che potra ascoltare.

L'idea e' quella di avere una schermata divisa in tre blocchi:

1. Il primo blocco sara' il menu dei contenuti, con annesso il Logo della Radio. Consentira' di navigare ed aggiornare il blocco 2 o l'intera schermata, in base al contenuto scelto. Ad esempio Podcast, mostrera' i podcast pubblicati, Musica, mostrera' i canali musicali piu richiesti creati ad hoc da Radio UVM in base al genere musicale. LIVE saltera' alla schermata piena del programma live in onda nel momento con i dettagli degli Speaker e della Regia.
2. Sara' il rettangolo piu' grande posizionato a sinistra. Conterra' la lista dei contenuti audio disponibili. In base al riconoscimento della tipologia di utente, sara' possibile aggiungere dei contenuti oppure semplicemente ascoltarli.
3. Rettangolo verticale con i dettagli del contenuto audio scelto. Conterra' i pulsanti di play e pausa, e per passare al contenuto successivo. Per gli utenti registrati, si avra' la possibilita di aggiungerlo ai contenuti preferiti.

PROGRAMMI

L'utente utilizza questa sezione per visualizzare i podcast pubblicati dagli utenti Speaker di Radio UniVersoMe.

Una volta selezionata la sezione, questa mostrera' i contenuti audio e dara' la possibilita' all'utente di ascoltarli.

Se l'utente e' autenticato come Speaker, oltre a poter ascoltare le puntate, e' possibile caricarne altre tramite l'apposito pulsante, eliminare le puntate che ha precedentemente caricato o modificarne la descrizione.

LIVE

La schermata della puntata Live, visualizza i controlli per l'ascolto, come il pulsante Play, Pausa, Riavvolgi di 15 secondi, e Dal Vivo. Mostrera' i dettagli della puntata in onda, come il Titolo, una descrizione, il nome dello speaker, un'immagine o un video nel caso in cui si vuole mostrare qualcosa all'ascoltatore, o direttamente il video in diretta della puntata con gli speaker.

L'utente utilizza questa sezione per ascoltare la diretta radio quando disponibile.

Dal punto di vista della User UI, il box si colorerà di un Rosso acceso pulsante che indicherà la presenza di un programma LIVE.

L'utente che cliccherà sul box, verrà riportato su una schermata dove ci saranno tutte le info sulla LIVE, come gli argomenti che verranno trattati, gli speaker.

UNIVERSITY ADS

Con l'University ADS si vuole abbattere la barriera della disinformazione studentesca nei confronti delle opportunità dell'ateneo. Ogni dispositivo verrà configurato in base al polo dove risiede ma sarà possibile ascoltare l'avviso di altri dipartimenti.

L'utente che visiterà la sezione, vedrà in primo piano, il podcast dedicato al proprio dipartimento, ed avrà la possibilità di ascoltare quelli degli altri dipartimenti o corsi di laurea.

MUSICA

L'utente utilizza questa sezione per visualizzare i programmi musicali, ognuno che tratterà un genere musicale diverso. Le funzioni sono simili a quelle dei podcast, e ci sarà uno speaker responsabile della pubblicazione dei vari programmi. E' possibile effettuare delle operazioni come creazione, modifica ed eliminazione del programma musicale.

Class Diagram

Programma

Un programma radiofonico o trasmissione radiofonica è un qualsiasi programma trasmesso a mezzo delle [onde radio](#) e ricevibile con degli apparecchi ricevitori. Può essere un programma [giornalistico](#), [musicale](#), [culturale](#) o di puro intrattenimento. Viene inserito nel palinsesto settimanale ed e' composto da piu' episodi. Ogni programma ha degli Speaker, un orario in cui viene trasmesso. La produzione di un programma radiofonico e' delegata al [produttore radiofonico](#).

Di un programma si vuole registrare:

1. Id
2. Titolo
3. Descrizione
4. Speaker

Episodio

Un programma radiofonico puo' durare una o piu' stagioni, dunque e' necessario scomporlo in episodi. In ogni episodio e' possibile che siano presenti degli ospiti.

Di un programma si vuole registrare:

5. Id
6. Titolo
7. Descrizione
8. Guest
9. Type

Utenti

I ruoli degli utenti che potranno interagire con il programma sono:

- Anonimo (non-loggato)

- Ascoltatore
- Speaker

Ci interessa registrare sul Database le seguenti informazioni

1. Username
2. Full Name
3. Email
4. Password
5. Role

Playlist

L'utente registrato ha la possibilità di creare una Playlist dei contenuti preferiti.

Live

La live sarà un canale di riproduzione che trasmetterà sempre dei contenuti, che questi siano episodi di programmi o semplicemente delle playlist musicali.

University ADS

Gli avvisi universitari sono dei podcast creati volontariamente dai radiofonici o studenti che collaborano con UniVersoMe con l'obiettivo di informare gli studenti sulle opportunità che offre il nostro ateneo in termini di Tirocini, Bandi, scadenze. Ogni avviso è suddiviso in base al dipartimento.

Di un avviso universitario si vuole registrare:

1. Id
2. Titolo
3. Descrizione
4. Dipartimento

Musica

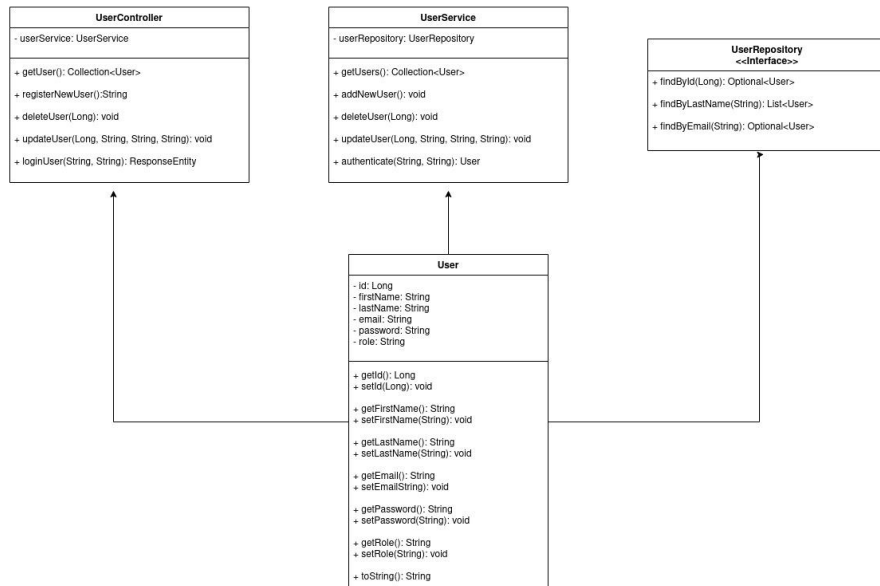
Raccolte di musica suddivise per generi musicali consigliate dai radiofonici di Radio UVM.

Di un avviso universitario si vuole registrare:

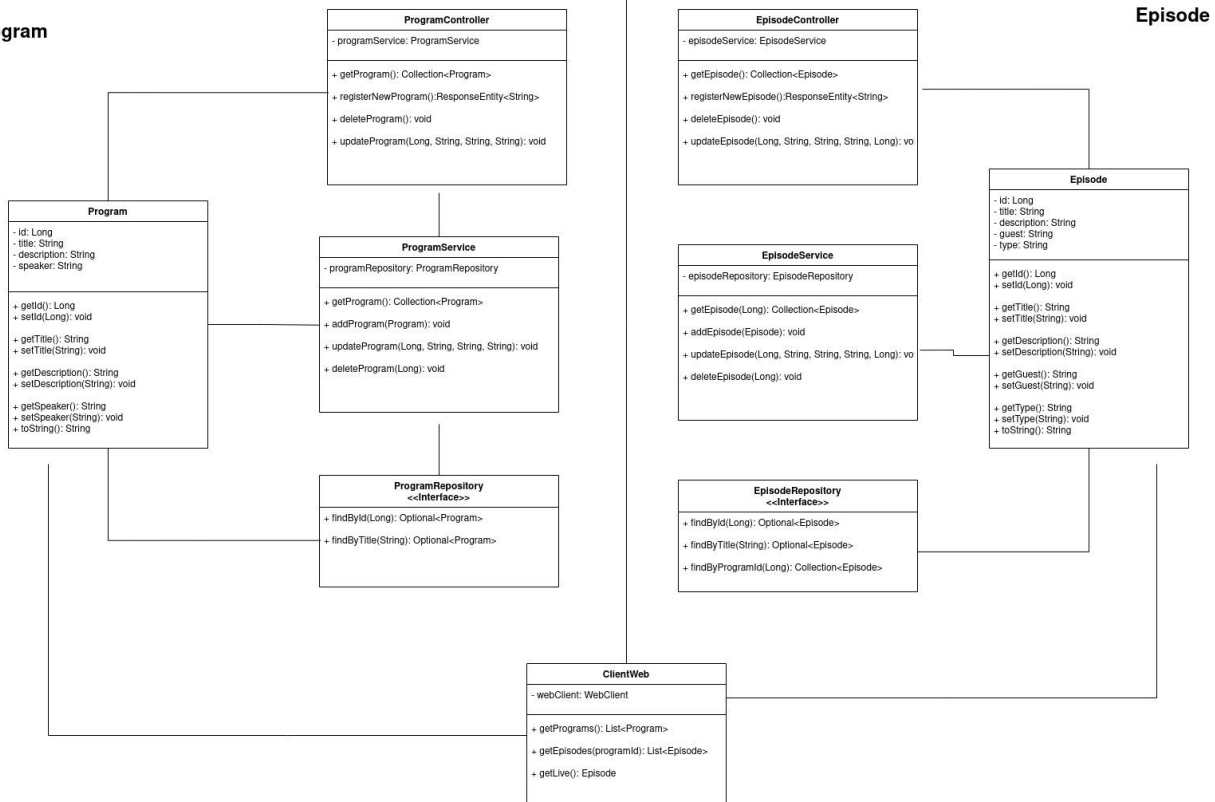
1. Id
2. Titolo
3. Descrizione
4. Genere

Radio UVM Server

User



Program

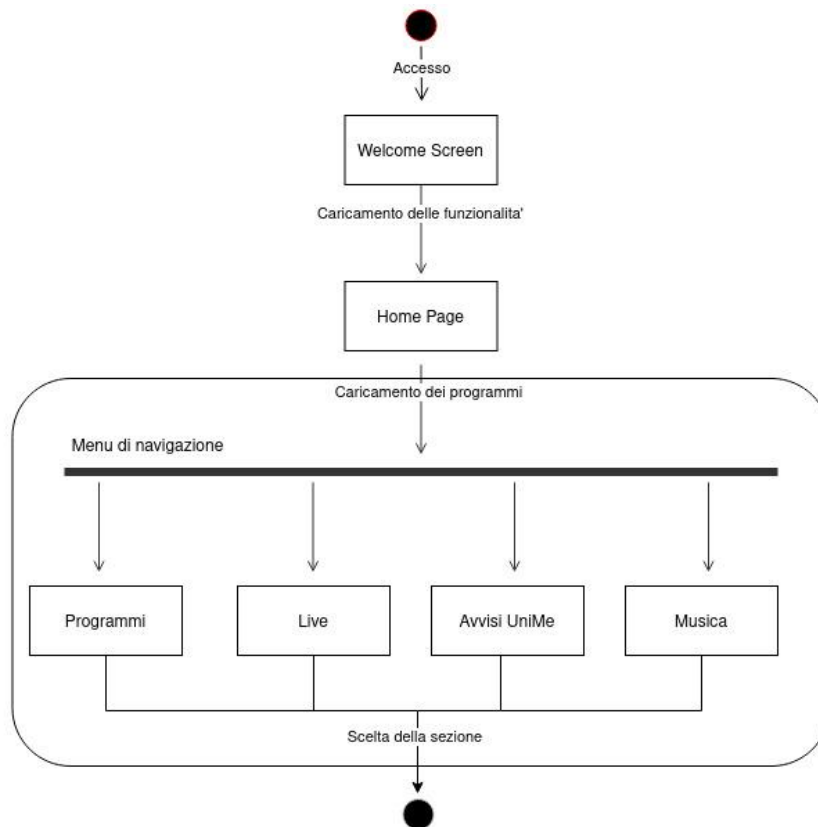


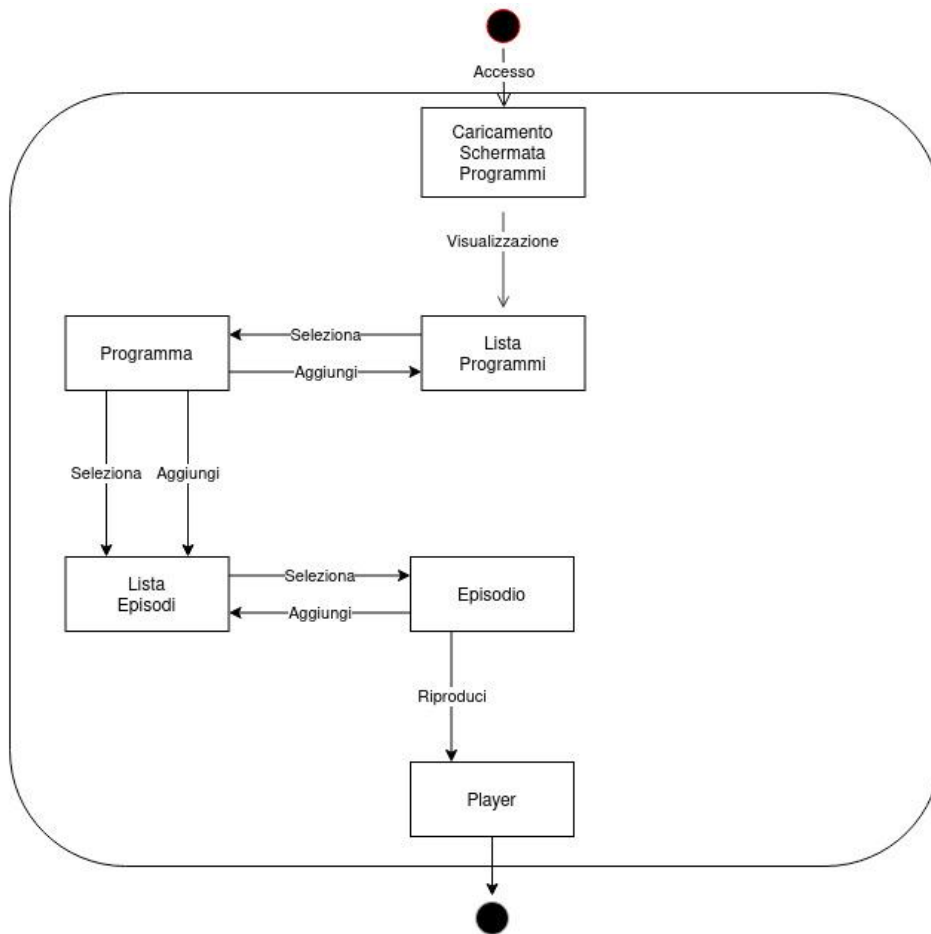
Activity Diagram

I diagrammi di attività descrivono la sequenza delle attività che intervengono nel flusso di un programma. Essi mostrano i passi (chiamati propriamente attività), i punti decisionali e i rami (cioè le relazioni tra le attività) che intervengono nel flusso di un programma e descrivono le attività necessarie per il completamento dei casi d'uso (all'interno di un singolo caso d'uso vi possono essere diversi percorsi possibili).

Rispetto ai diagrammi di sequenza (proposti in seguito) permettono di comprendere meglio gli algoritmi usati essendo molto simili a 'flow charts'.

Di seguito vengono riportati i diagrammi di attività relativi ai casi d'uso di maggiore rilevanza





Sequence Diagram

I diagrammi di sequenza servono a specificare il funzionamento a basso livello della sequenza di operazioni tra le classi.

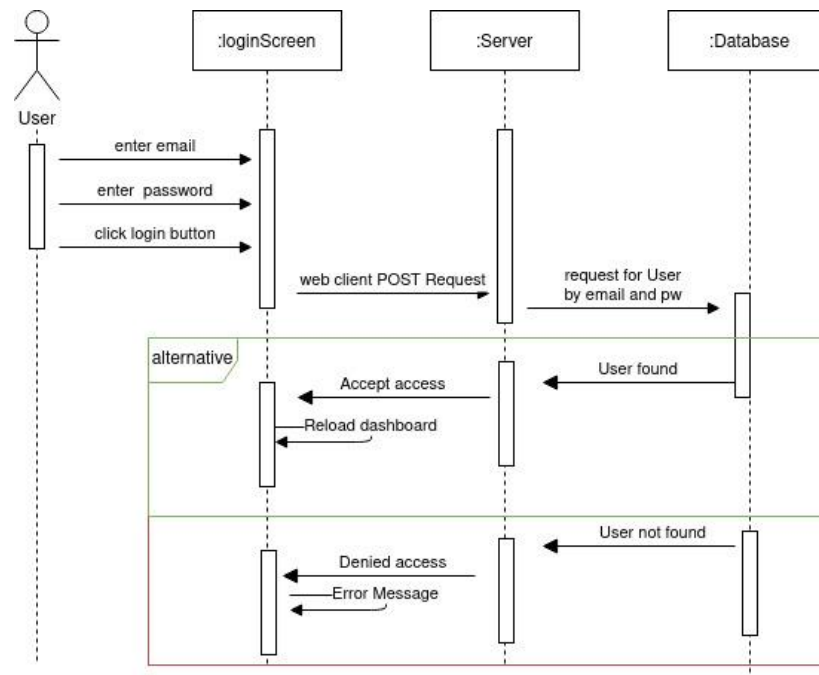
Sono molto importanti soprattutto se qualche processo ha durata più lunga rispetto ad altri, o riveste un ruolo più importante.

Essi documentano tipicamente il comportamento di un singolo scenario ed includono:

- un certo numero di oggetti

- i messaggi scambiati tra essi durante l'esecuzione del caso d'uso.

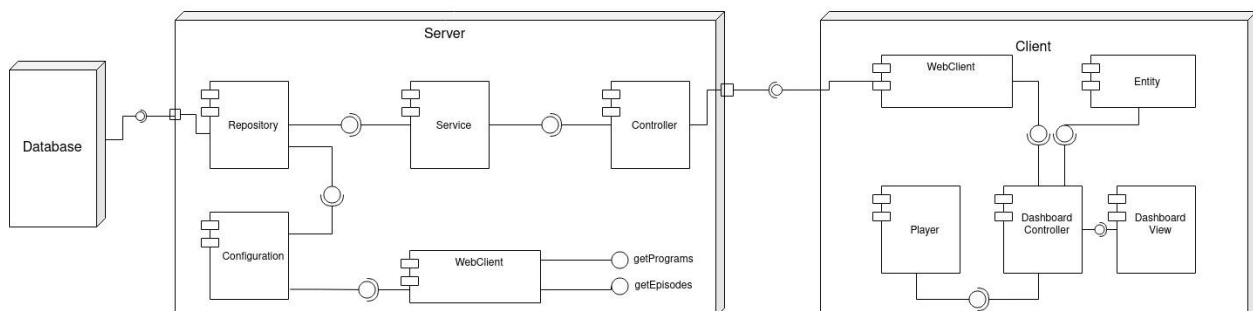
Di seguito sono rappresentati il diagramma di sequenza relativo al login:



Component Diagram

Il Component Diagram si focalizza nel rappresentare gli elementi principali del sistema e come si relazionano tra di loro. Se unito con il Deployment Diagram è possibile descrivere come i vari elementi interagiscono tra di loro.

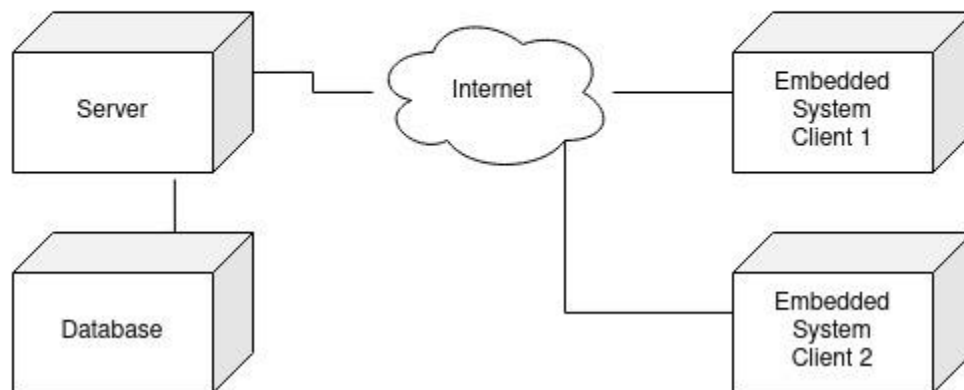
Component Diagram



Deployment Diagram

Il Deployment Diagram è un diagramma di tipo statico per descrivere un sistema in termini di risorse hardware, dette *nod*i. Spesso si utilizza un diagramma che mostra come le componenti software siano distribuite rispetto alle risorse hardware disponibili sul sistema; questo diagramma è costruito unendo il **Component Diagram** e il **Deployment Diagram**.

Deployment Diagram



Verifica e Validazione (Test)

In questa sezione si descrivono i procedimenti, le strategie e le metodologie usate per organizzare, pianificare, eseguire e gestire il testing del sistema software.

Gli obiettivi che si vogliono raggiungere con il collaudo del sistema sono:

1. Verificare che si siano implementate tutte le funzionalità dichiarate nella specifica dei requisiti.
2. Verificare che il software soddisfi alcuni requisiti di qualità

Problematiche riscontrate

Database:

- Warning nel drop della tabella:
utilizzare spring.jpa.hibernate.ddl-auto=update nel file application.properties
- Abilitazione Debug:
<https://docs.spring.io/spring-boot/docs/current/reference/html/features.html#features.logging.console-output>
- Recupero Password utente postgres:
<https://stackoverflow.com/questions/10845998/i-forgot-the-password-i-entered-during-postgres-installation>
- [Postgres DDL error: 'syntax error at or near "user"' \[duplicate\]](#)

Gestore delle dipendenze

- Problema risorse, file fxm (FXMLoader):
<https://stackoverflow.com/questions/35861606/getresource-is-returning-null-with-gradle-project>

Metodi di collaudo usati

Le tecniche utilizzate per collaudare il sistema sono principalmente 2:

1. Test di unità, che ci hanno consentito di verificare il corretto funzionamento delle funzioni di interfacciamento con il database
2. Test funzionali, che basandosi esclusivamente sulle specifiche, ci hanno permesso di verificare il corretto funzionamento del sistema e la robustezza dello stesso

I test di unità sono stati implementati man mano che venivano dichiarate le funzioni da testare, mentre i test funzionali sono stati implementati al termine della fase di sviluppo.

I test di unita' sono stati eseguiti manualmente accertandosi del corretto funzionamento del componente:

Test del Server - Entita'

Avvio del Server	OK
Configurazione del Database PostgreSQL - OK	OK
Creazione della tabella User - OK	OK
Creazione della classe POJO User - OK	OK
Creazione della tabella Program - OK	OK
Creazione della classe POJO Program - OK	OK
Creazione della tabella Episode - OK	OK
Creazione della classe POJO Episode- OK	OK
Creazione della tabella UAds - OK	OK
Creazione della classe POJO UAds - OK	OK
Creazione della tabella Music - OK	OK
Creazione della classe POJO Music - OK	OK
Creazione della tabella Playlist - OK	OK
Creazione della classe POJO Playlist - OK	OK

Test del Server - Interazione con il Database

Recupero dati User per Id	OK
Recupero dati User per LastName	OK
Recupero dati User per Email	OK

Recupero dati Program per Id	OK
Recupero dati Program per Title	OK
Recupero dati Episode per Id	OK
Recupero dati Episode per Title	OK
Recupero dati Episode per ProgramId	OK
Recupero dati UAds per Id	OK
Recupero dati UAds per Title	OK
Recupero dati UAds per Department	OK
Recupero dati Music per Id	OK
Recupero dati Music per Title	OK
Recupero dati Music per Genre	OK
Recupero dati Playlist per Id	NOT TESTED
Recupero dati Playlist per Title	NOT TESTED
Recupero dati Playlist per UserId	NOT TESTED

Test del Server - Operazioni CRUD

Test del percorso per le API User	OK
Test per il recupero della lista degli User	OK
Test per la creazione di un nuovo User	OK
Test per la modifica di uno User	OK
Test per la cancellazione di uno User	OK
Test per il login di uno User	OK

Test del percorso per le API Program	OK
Test per il recupero della lista dei Program	OK
Test per la reazione di un nuovo Program	OK
Test per la modifica di un Program	OK
Test per la cancellazione di un Program	OK

Test del Server - Inizializzazione

Test delle operazioni CRUD per Radio UVM Server

Test delle operazioni CRUD per Radio UVM Client - Music

- Visualizzazione del pulsante per aggiungere un elemento (utente Speaker);
- Visualizzazione della schermata per aggiungere le informazioni sul nuovo elemento.
- Log della richiesta POST - OK
- Verifica della chiusura e ricarica della sezione Musica

Test della Dashboard

- Test di interazione con il blocco visuale del brano - OK
- Test di esecuzione del brano - OK

Bugs

- Il player resta attivo anche quando viene cambiata la schermata.

Project Management

Per lo sviluppo del progetto, essendo un software i cui requisiti possono variare nel tempo a seconda dell'esigenza del cliente, si è scelto l'approccio della metodologia Agile. In particolare ci si è affidati allo strumento per lo sviluppo di prodotti software chiamato Scrum, largamente utilizzato in questi contesti.

Metodologia Agile

È un insieme di metodologie incrementalì ed iterative. Tramite la metodologia Agile, nel project management è possibile focalizzarsi sulla produttività, sulla soddisfazione del cliente e sul miglioramento continuo del progetto.

Benchè sia spesso utilizzata nei progetti, resta pur sempre una filosofia, che viene utilizzata in un contesto pratico attraverso vari strumenti.

Scrum

È un framework agile per gestire lo sviluppo di software complessi. Attraverso Scrum, è possibile utilizzare una varietà di processi e tecniche per creare un prodotto.

Scrum - Step 1: Calendario e cadenza

Periodo di lavoro: 7 Settimane (3 sprint)

Suddividere il lavoro del team in iterazioni (**Sprint**).

Iterazioni

Definire un lasso di tempo preciso: 1 settimana fino ad un massimo di 4 settimane per sprint.

In questo caso utilizziamo un lasso di tempo di 2 settimane (10 giorni lavorativi).

Incrementi

Lavorare al prodotto finale pezzo per pezzo per ogni funzionalità, *in base al loro ordine di priorità*.

Ruoli

Product Owner (PO): è il responsabile della pianificazione dello sviluppo delle funzionalità. Decide cosa implementare per prima.

Scrum Master (SM): si occupa di sorvegliare il processo durante la sua esecuzione. Lavora a stretto contatto con il PO ed il DevTeam

DevTeam: si occupano di come le funzionalità devono essere implementate.

Prodotto:

Scomporre il prodotto finale in sottoparti per facilitare l'implementazione. Ogni parte deve essere realizzata dal team in modo indipendente dalle altre.

User stories

PO, SM e Dev Team realizzano delle storie per raccontare le funzionalità da svolgere in modo amichevole. Tuttavia, il PO è l'unico che può assegnare la priorità alle User Story.

Strutture della user story: Titolo, Descrizione, Acceptance Criteria per definire se una storia è completa o no.

Scrum - Step 2: Product Backlog

E' l'insieme delle funzionalità che devono essere sviluppate. Vengono ordinate in una struttura a pila in base alla loro priorità. In cima alla pila, ci sono gli elementi che devono essere implementati per primi. In basso ci sono le funzionalità ed i macrocomponenti.

Per ridimensionare gli elementi che hanno dimensioni elevate, viene utilizzato un processo denominato grooming, ovvero “raffinamento”.

Finito lo sviluppo di un elemento, e quindi funzionante, privo di bug e pronto per essere messo in produzione, si dice che il pezzo è **Done Done**.

E' necessario creare una ToDo List per il Product Backlog (PB). Ogni componente di questa pila ordinata viene chiamato Product Backlog Items (PBI).

Per lavorare in modo più semplice ad ogni componente della pila, il PBI può essere scomposto in **Tasks**. Una sequenza di task, porta a conclusione lo sviluppo di un componente.

In altri casi, i Task potrebbero corrispondere ad attività come: implementazione della persistenza dei dati (vedi [Persistenza \(Informatica\)](#)), definizione dello schema del Database ecc.

Product Backlog del progetto

Il progetto in questione richiede un quantitativo importante di requisiti funzionali in quanto, ai fini dello sviluppo del software e' necessario:




















































- Valutare quale dispositivo hardware sia in grado di ospitare il software, valutando anche la complessità, dispendio energetico, risorse necessarie per l'esecuzione fluida del software.
- Scegliere uno schermo touch-screen con cui poter interagire con l'applicazione
- Valutare quale architettura e' migliore e si adatta meglio ai requisiti. Capire se si tratta di un micro servizio o e' meglio utilizzare un architettura three-tier composta da un'Interfaccia utente, un server e un database. Si ricorda che il three-tier e' un modello di architettura software e allo stesso tempo uno schema di progettazione software (design pattern)

- Scegliere il linguaggio di sviluppo piu' appropriato
- Scegliere un framework che velocizzi lo sviluppo dell'applicativo
- Adattare il formato dati in risposta dal server all'oggetto utilizzato dal linguaggio
- Creare un server centrale per recuperare i dati.
- Memorizzare il dato recuperato sul database.
- Gestione informazioni sull'utente
- Gestione informazioni sui programmi
- Gestione informazioni sugli episodi
- Gestione informazioni sugli avvisi universitari
- Gestione informazioni sul catalogo musicale
- Creare un player per l'ascolto del contenuto multimediale
- Creare un pulsante per ascoltare il contenuto
- Creare un pulsante per mettere in pausa il contenuto
- Creare una lista dei programmi pubblicati
- Creare, per ogni programma pubblicato, una lista degli episodi contenuti in quel programma.
- Permettere la riproduzione dell'episodio tramite un player.
- Creare un player per la riproduzione di streaming di contenuti trasmessi in diretta live.
- Creare una lista degli avvisi universitari

- Permettere la riproduzione dell'avviso universitario tramite un player.
- Creare una lista delle raccolte musicali
- Permettere la riproduzione della raccolta tramite un player.
- Gli altoparlanti del dispositivo devono avere una buona qualità audio.

Radio UniVersoMe Team  

Backlog Analytics [+ New Work Item](#) [View as Board](#) [Column Options](#) ...

 	Order	Work Item Type	Title	State	Effort	Value Area	Iteration Path
	1	Product Backlog Item	 Valutazione del dispositivo embedded	 New		Business	Radio UniVersoMe
	2	Product Backlog Item	 Scelta dello schermo touch-screen	 New		Business	Radio UniVersoMe
	3	Product Backlog Item	 Creazione delle entità lato Client	 New		Business	Radio UniVersoMe\Sprint 3
	4	Product Backlog Item	 Definizione dei ruoli lato Client	 New		Business	Radio UniVersoMe\Sprint 3
	5	Product Backlog Item	 Creazione del WebClient	 New		Business	Radio UniVersoMe\Sprint 3
	6	Product Backlog Item	 Ottimizzazione del codice	 New		Business	Radio UniVersoMe\Sprint 3
	7	Product Backlog Item	 Test di unità	 New		Business	Radio UniVersoMe\Sprint 3
	8	Product Backlog Item	 Scelta delle tecnologie per l'applicazione Client	 New		Business	Radio UniVersoMe\Sprint 2
	9	Product Backlog Item	 Configurazione iniziale di Spring Boot + OpenJFX	 New		Business	Radio UniVersoMe\Sprint 2
	10	Product Backlog Item	 Comprensione e pulizia del codice	 New		Business	Radio UniVersoMe\Sprint 2
	11	Product Backlog Item	 Creazione della finestra iniziale	 New		Business	Radio UniVersoMe\Sprint 2
	12	Product Backlog Item	 Creazione della Home	 New		Business	Radio UniVersoMe\Sprint 2
	13	Product Backlog Item	 Creazione dell'Header	 New		Business	Radio UniVersoMe\Sprint 2
	14	Product Backlog Item	 Creazione del Body	 New		Business	Radio UniVersoMe\Sprint 2
	15	Product Backlog Item	 Schermata di Login	 New		Business	Radio UniVersoMe\Sprint 2
	16	Product Backlog Item	 Schermata di Registrazione	 New		Business	Radio UniVersoMe\Sprint 2
	17	Product Backlog Item	 Creazione di un Media Player	 New		Business	Radio UniVersoMe\Sprint 2
	18	Product Backlog Item	 Creare un Media Player per le Dirette	 New		Business	Radio UniVersoMe\Sprint 2
	19	Product Backlog Item	 Creazione del Footer	 New		Business	Radio UniVersoMe\Sprint 2
	20	Product Backlog Item	 Creazione dei metodi CRUD	 Committed	6	Architectural	Radio UniVersoMe\Sprint 1
	21	Product Backlog Item	 Creazione dell'interfaccia al database	 Committed	2	Architectural	Radio UniVersoMe\Sprint 1
	22	Product Backlog Item	 Creazione dell'autenticazione User	 Committed	3	Business	Radio UniVersoMe\Sprint 1
	23	Product Backlog Item	 Test delle operazioni CRUD	 Committed	2	Business	Radio UniVersoMe\Sprint 1
	24	Product Backlog Item	 Scelta altoparlanti	 Approved	1	Business	Radio UniVersoMe\Sprint 1

Scrum - Step 3: Sprint Planning

E' una riunione dove si decide quali saranno i prossimi passi per lo sviluppo del prodotto.

Di solito dura 2 ore per ogni 5 giorni di sprint, dunque nel nostro caso durerà 4 ore.

Il PO presenta la lista delle cose da fare e le divide in base ai membri del DevTeam. E' una riunione di fondamentale importanza in quanto si decidono quante cose implementare in quello sprint ed è legato alla velocità di lavorazione del team.

Il PO presenta un numero n di user stories prese dal product backlog, le discute con l'SM e il DevTeam, dunque ognuno assegna un peso a tale storia.

Una volta assegnato un peso la storia viene aggiunta allo sprint backlog.

Lo sprint backlog avrà un punteggio totale dato dalla somma dei pesi di tutte le storie che lo compongono.

Sprint 1

Al fine di rendere più chiaro lo sviluppo dell'applicazione si è deciso di caricare i dati e di modificare lo stato dei task solo dopo aver esposto le modifiche durante la riunione di fine Sprint. Questo ha reso possibile una visione più chiara delle implementazioni a tutto il team prima che esse potessero intaccare la struttura definitiva. Di seguito verranno mostrate le singole User Story divise tra gli Sprint con un'accurata descrizione del loro funzionamento, dei requisiti che soddisfano e della motivazione del punteggio assegnato. Verranno inoltre descritti gli "acceptance criteria", ovvero tutti quei criteri da soddisfare al fine di ritenere una User Story completa.

Sprint 1 - Backlog

Radio UniVersoMe Team ☆ 🔍 No iteration dates Set dates !

Taskboard **Backlog** Capacity Analytics + New Work Item Column Options ... Sprint 1 🔍 🔗 ⚙️ 🔗

	Order	Work Item Type	Title	State	Effort	Assigned To
+	1	Product Backlog Item	Scelta dell'architettura di sistema	Done	1	GIANLUCA CARBONE
	2	Product Backlog Item	Scelta del linguaggio di programmazione	Done	1	GIANLUCA CARBONE
	3	Product Backlog Item	Scelta del gestore delle dipendenze	Done	1	GIANLUCA CARBONE
	4	Product Backlog Item	Scelta del framework	Done	3	GIANLUCA CARBONE
	5	Product Backlog Item	Scelta del formato per lo scambio di dati	Done	3	GIANLUCA CARBONE
	6	Product Backlog Item	Creazione del Database	Done	1	GIANLUCA CARBONE
	7	Product Backlog Item	Creazione di un server centrale	Done	1	GIANLUCA CARBONE
	8	Product Backlog Item	Creazione delle entita'	Done	5	GIANLUCA CARBONE
	9	Product Backlog Item	Creazione dei metodi CRUD	Committed	6	GIANLUCA CARBONE
	10	Product Backlog Item	Creazione dell'interfaccia al database	Committed	2	GIANLUCA CARBONE
	11	Product Backlog Item	Creazione di un WebClient per le API Speaker	Done	3	GIANLUCA CARBONE
	12	Product Backlog Item	Creazione dell'autenticazione User	Committed	3	GIANLUCA CARBONE
	13	Product Backlog Item	Test delle operazioni CRUD	Committed	2	GIANLUCA CARBONE
	14	Product Backlog Item	Scelta altoparlanti	Approved	1	GIANLUCA CARBONE

Sprint 1 - Planning

Nella prima fase dello Sprint le storie sono state aggiunte allo Sprint Backlog e suddivise in task.

Le storie da completare sono 2 e sono state suddivise in 8 task.

Creazione del database

La prima story non ha nessun task in quanto la "creazione del database" consiste nella definizione delle tabelle con gli attributi e le funzioni base necessarie (Insert, Update, Delete). Le tabelle implementate sono:

- User: questa tabella è dedicata al salvataggio dei dati di autenticazione di tutti gli utenti suddivisi in tipologie "ANON" o "LISTENER" o "SPEAKER"

- Program: questa tabella e' dedicata al salvataggio dei dati derivanti dai programmi pubblicati sulla piattaforma Spreaker.
- Episode: questa tabella contiene tutti gli episodi legati ai vari programmi pubblicati su Spreaker
- UAds: questa tabella contiene tutti i contenuti rivolti agli avvisi universitari
- Music: questa tabella contiene tutti i cataloghi musicali.

Lo **Story Point** è di 4 su 10, in quanto viene previsto un ampliamento futuro e costante della struttura del database, che dovrà avere una base solida ed intoccabile.

Acceptance criteria:

1. Nomi delle tabelle sensati e univoci;
2. Connessione sicura e stabile con il db;
3. Funzioni complete quali: create, update, delete;
4. Gestione degli exception ed errori vari.

Configurazione del server

In questa **User Story** sono previsti **5 task** relativi a:

- configurazione iniziale del server per e test della pagina iniziale
- configurazione della connessione tra server e database
- configurazione dei log
- creazione delle entita'
- creazione degli endpoint e controller per le operazioni CRUD sulle entita'.

Lo Story Point è di 3 su 10, in quanto non viene richiesta la creazione di ogni pagina all'interno dell'applicazione ma solo di quelle iniziali per guidare il secondo sprint in modo fluido.

Acceptance criteria:

1. Gli endpoint devono essere chiari e devono rispettare le operazioni richieste
2. Ogni entita' deve essere correttamente mappata tra il formato JSON e POJO

3. E' necessario gestire i codice di errore delle chiamate HTTP

Sprint 1 - Analisi

Dal momento che si prevede una durata del progetto pari circa ad un mese, si prevede uno Sprint ogni 2 settimane, con degli orari di lavoro di 5 ore giornaliere dal lunedì al venerdì.

Di seguito riporto lo sprint backlog relativo al primo sprint con le funzionalità incluse nelle varie voci.

Ogni giorno verrà effettuato un Daily Standup Meeting della durata di 15 minuti per allinearsi con il team.

In questo primo Sprint si sceglie di sviluppare prima il server ed il database, le interfacce con il database ed le interfacce con il client.

Il Team di sviluppo ha scelto il linguaggio con cui lavorare ed il relativo framework ed il gestore delle dipendenze che sono:

- Linguaggio: Java 16 (OpenJDK 16)
- Framework: Spring Boot 2.5.4
- Gestore delle dipendenze: Gradle 7.1.1

Il Database in uso sarà PostgreSQL.

Per un'implementazione rapida del server, il team si affida alle configurazioni di default di Spring Boot. Tuttavia, nel caso in cui fossero necessarie delle modifiche, queste verranno aggiunte nelle configurazioni del framework.

Creazione del database:

Viene creato un database vuoto con cui testare la connessione tramite le configurazioni di Spring.

Il Team fa uso della libreria Spring Boot Starter Jpa Data per interfacciarsi con il database.

Dunque si passa alla configurazione del file `application.properties` i settaggi sulla connessione.

Una volta effettuato il test sulla connessione, si procede con la creazione delle interfacce al database e le REST API per il client per quanto riguarda i profili Utente da registrare.

Definizione dell'Utente:

Si procede quindi con la creazione dell'oggetto `User`, definizione degli attributi e dei metodi `getter` e `setter` per ognuno, nonché del metodo `toString` per la rappresentazione dei dati in forma testuale.

Tramite il framework, mappiamo l'utente nel database con la corrispettiva tabella, creata dinamicamente.

Interfaccia dell'Utente al Database

Successivamente si passa alla creazione dell'Interfaccia al database per il recupero dei record memorizzati mediante opportune funzioni, che verranno chiamate dalle REST API una volta implementate.

Si crea dunque la funzione per il recupero degli utenti in base all'identificativo o ad altri attributi affini.

Servizio di trasferimento dal database alle REST API dell'utente:

I record ottenuti dall'interfaccia verranno manipolati dalla componente che gestisce il trasferimento dei dati da database alle REST API. Le funzionalità del server qui definite serviranno per filtrare i dati passati da e verso il database.

Creazione delle REST API per il client tramite il Controller:

Tramite il controller creiamo le 4 operazioni CRUD per gestire i dati sul database.

- Recupero
- Creazione

- Cancellazione
- Modifica

Popolamento (o immissione dei dati) nel database:

Creiamo dei record fittizi per immettere dei dati all'interno del database e testare le REST API e l'interfaccia al database.

Le procedure eseguite per l'utente vengono replicate ed adattate alle restanti entita':

- Programmi
- Episodi
- Avvisi Universitari
- Musica

Le prime due settimane sono state utilizzate dal DevTeam per comprendere il framework ed le varie componenti, utili per implementare il server e le interfacce.

Sprint 1 - Review

Le user Story di questo sprint sono state contrassegnate come complete, in quanto gli Acceptance criteria di entrambe risultano soddisfatti. Al cliente non e' stato mostrato alcun prodotto in quanto non e' ancora presentabile attraverso un'interfaccia grafica intuitiva.

Sprint 1 - Review: Retrospective

Grazie a questo primo sprint sono stati individuati i punti di forza del team e come la collaborazione possa portare a vantaggi implementativi. Non avendo dati sufficienti per un'analisi dei bug non ci sono cambiamenti al tipo di lavoro pianificato, ma rimangono noti a tutto il team dei controlli futuri di testing ovvero:

- Le funzioni di gestione del database dovranno mantenere una connessione stabile anche se richiamate da Classi;
- Gli endpoint devono essere testati per capirne il comportamento in caso di richieste con mancanza di dati.

Sprint 2

Sprint 2 - Backlog

Radio UniVersoMe Team ☆ 🔊 No iteration dates Set dates !

Taskboard **Backlog** Capacity Analytics + New Work Item Column Options ... Sprint 2 🔍 🔧 🔗

Order	Work Item Type	Title	State	Effort	Assigned To
1	Product Backlog Item	Scelta delle tecnologie per l'applicazione Client	Done	1	GIANLUCA CARBONE
2	Product Backlog Item	Configurazione iniziale di Spring Boot + OpenJFX	Done	8	GIANLUCA CARBONE
3	Product Backlog Item	Comprensione e pulizia del codice	Done	8	GIANLUCA CARBONE
+ 4	Product Backlog Item	Creazione della finestra iniziale	Committed	8	GIANLUCA CARBONE
	Task	Design del Welcome Screen	Done		
	Task	Creazione del Welcome Screen	Done		
	Task	Visualizzazione del Welcome Screen	Done		
	Task	Creazione Spinner di caricamento iniziale	Done		
	Task	Inserimento del Logo di Radio UVM	In Progress		
5	Product Backlog Item	Creazione della Home	Done	6	GIANLUCA CARBONE
	Task	Creazione del Controller component	Done		
6	Product Backlog Item	Creazione dell'Header	Done	3	GIANLUCA CARBONE
	Task	Data e Orario	Done		
	Task	Logo Radio UVM	Done		
	Task	Menu impostazioni	Done		
	Task	Link di accesso	Done		
7	Product Backlog Item	Creazione del Body	Done	3	GIANLUCA CARBONE
	Task	Titolo descrittivo del contenuto	Done		
	Task	Pulsante di interazione	Done		
	Task	Programmi in formato Card	Done		
	Task	Episodi in formato lista	Done		
	Task	University Ads in formato lista	Done		

Sprint 2 - Planning

Conclusa la creazione del server, il team si decide di concentrarsi sulla parte estetica del progetto. Durante lo sprint ogni membro del team si dedicherà alla definizione delle schermate, la loro composizione

Sprint 2 - Analisi

Per la creazione dell'interfaccia grafica, trattandosi di un'applicazione Java, la scelta era tra il framework Swing, ormai obsoleto, ed l'application client JavaFX che si è posto come obiettivo rimpiazzare appunto Swing con una creazione delle interfacce grafiche tramite la potenza del linguaggio di markup FXML.

Prima di poter visualizzare un'interfaccia grafica funzionante ed interattiva, il DevTeam ha dovuto dividersi i compiti riguardanti il backend dell'interfaccia ed il frontend.

Dunque parte del Team si è occupato del front-end dell'applicativo, interfacciandosi con il linguaggio FXML e un tool di supporto chiamato SceneBuilder per una visualizzazione immediata delle componenti che vengono utilizzate, quali pannelli, pulsanti, etichette e il posizionamento di questi

Mentre la restante parte del Team si è occupato dell'integrazione del framework Spring Boot in un progetto JavaFX, per riuscire ad utilizzare le componenti già testate e offerte dal framework. Infatti, nel primo sprint, si è considerata la creazione di REST API messe a disposizione dal Server, il che richiede che ci sia una componente Client che possa richiedere i dati al server.

In tale ottica è stato necessario utilizzare il WebClient della libreria Spring Boot Starter WebFlux, per interagire con il server.

Completata l'integrazione del framework, il team che si dedica al backend dell'applicativo, si dedica alla creazione delle componenti che andranno a gestire gli eventi che accadranno nell'interfaccia grafica. Tali componenti vengono chiamati Controller, e ognuno di loro è associato ad una componente grafica.

Creazione della Home Page

Il team del front-end presenta la bozza della pagina principale composta da un'intestazione contenente il logo ed un link per accedere all'area dell'utente.

Il contenuto principale e' dato dai programmi radiofonici presentati all'interno di riquadri contenenti il logo che presenta il programma ed il titolo del programma.

Nella parte inferiore, invece abbiamo il menu di navigazione. Interagendo con i pulsanti e' possibile spostarsi tra le schermate dei programmi, della live, degli avvisi e dei cataloghi musicali.

La bozza viene approvata dal DevTeam e dal Product Manager ed e' pronta per essere presentata al cliente per ricevere ulteriori feedback.

Creazione del player

Il player si presenta principalmente con una foto degli Speaker che conducono la puntata radio, e con in basso la barra di completamento del contenuto multimediale. Inoltre sono disponibili i pulsanti Play / Pause e avanti e indietro di 10 sec.

Sprint 2 - Review

Al termine del secondo Sprint, e' possibile far vedere al cliente parte del prodotto finale, che consiste nell'interfaccia grafica del Client, adattata per visualizzare i contenuti richiesti.

Il Cliente approva la demo e si procede con lo Sprint 3.

Sprint 2 - Review: Retrospective

Sprint 3

Sprint 3 - Backlog

Radio UniVersoMe Team
☆
👤

No iteration dates
[Set dates](#)
! ...

Taskboard **Backlog** Capacity Analytics
+ New Work Item ...
Sprint 3

	Order	Work Item Type	Title	State	Effort
	1	Product Backlog Item	Creazione delle entita' lato Client	● Done	3
		Task	Creazione del modello User	● Done	
		Task	Creazione del modello Program	● Done	
		Task	Creazione del modello Episode	● Done	
		Task	Creazione del modello University Ads	● Done	
		Task	Creazione del modello Music	● Done	
	2	Product Backlog Item	Definizione dei ruoli lato Client	● Done	1
	3	Product Backlog Item	Creazione del WebClient	● Done	5
		Task	Richiesta di Login	● Done	
		Task	Richiesta registrazione	● Done	
		Task	Richiesta lista programmi	● Done	
		Task	Richiesta lista Music	● Done	
		Task	Richiesta aggiunta Music	● Done	
		Task	Richiesta modifica music	● Done	
	4	Product Backlog Item	Ottimizzazione del codice	● Committed	3
+	5	Product Backlog Item	Test di unita'	● Committed	4

Sprint 3 - Planning

Avendo completato la creazione dell'interfaccia grafica nello Sprint 2, si passa al recupero dei dati dal server, decidendo di creare un client con lo Spring WebFlux per accedere alle REST API esposte dal server. Dopodiché, si andranno a gestire le varie azioni e interazioni tra l'utente e la dashboard, migliorando la tolleranza agli errori, prevedendo la gestione delle eccezioni e proseguendo con la gestione della fase di login o registrazione.

Sprint 3 - Analysis

Nell'ultimo Sprint, il DevTeam si e' occupato della creazione di un Client di richieste al server per recuperare le informazioni da visualizzare nell'interfaccia grafica. Ha utilizzato la componente WebClient di Spring per effettuare delle richieste HTTP e recuperare i dati in formato JSON.

Dunque ha dovuto mappare tali dati in classi Java tramite la libreria Jackson.

E' stato utilizzato il WebClient per la fase di registrazione, di login, di recupero dei programmi, episodi, diretta live, UAds e Musica.

Inoltre, per la tipologia di Utente SPEAKER e' stato utilizzato per aggiungere nuovi contenuti o modificarne i dettagli.

Infine, ha apportato delle migliorie al codice, riuscendo a separare le componenti dal controller dell'interfaccia grafica e creando un package unico per il recupero delle informazioni dal server.

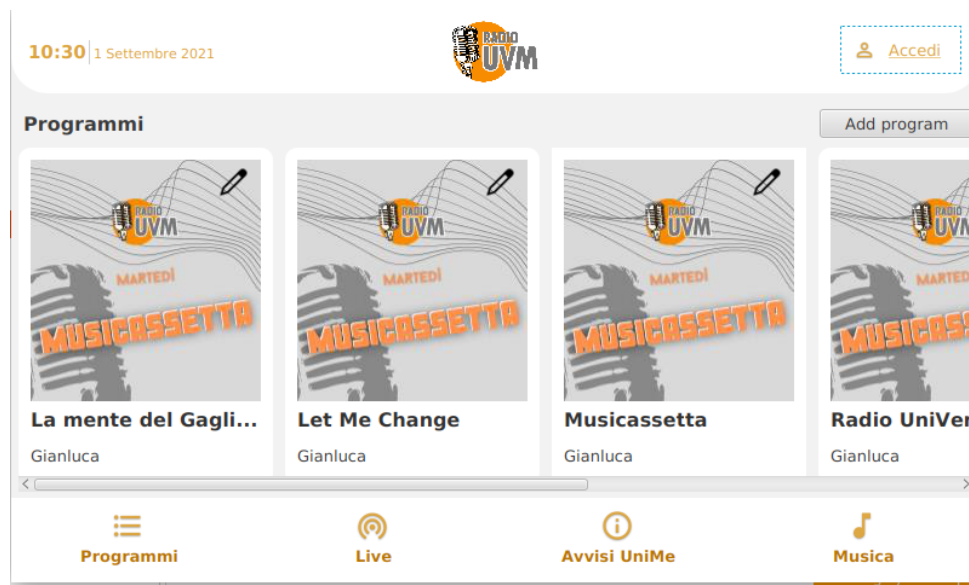
Sprint 3 - Review

Il prodotto e' a buon punto per essere rilasciato al cliente, ma devono essere ancora implementate delle migliorie a livello estetico.

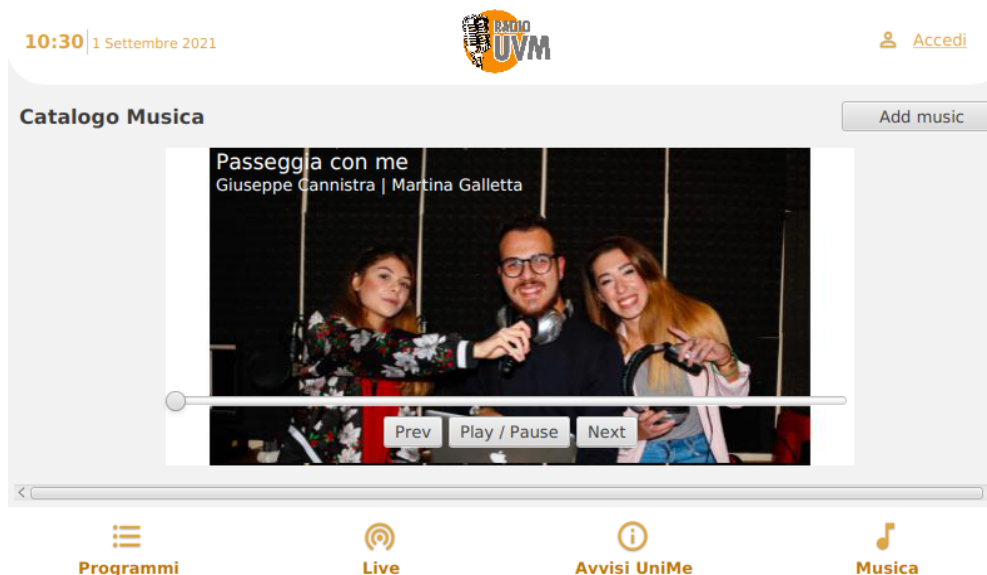
Sprint 3 - Review: Retrospective

Lo SCRUM Master rileva che e' necessario apportare delle modifiche estetiche, soprattutto nella parte relativa al player.

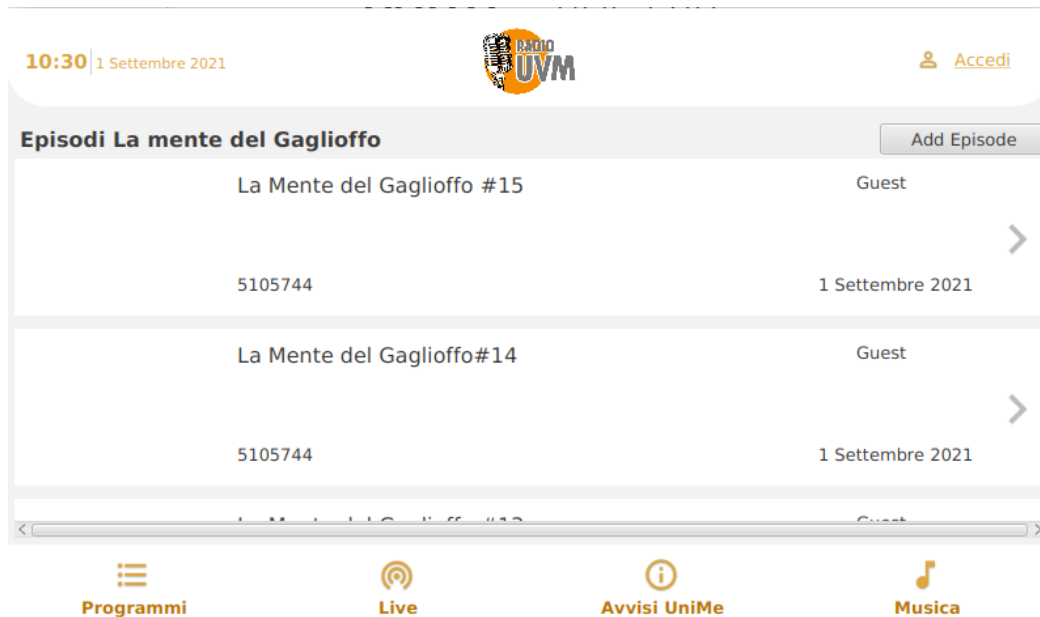
SCREENSHOT DEL SOFTWARE



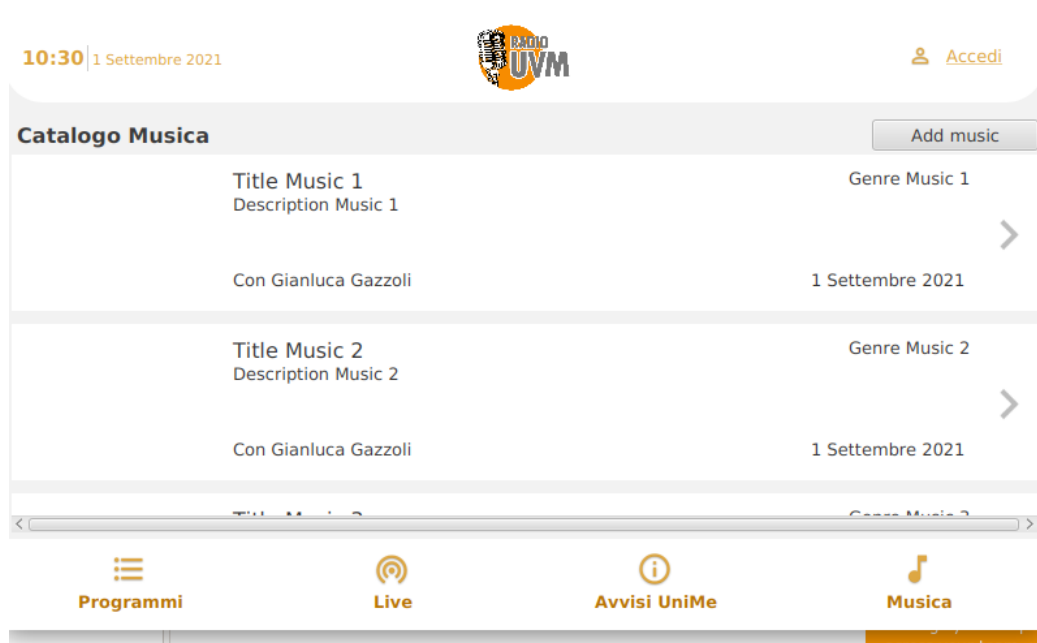
Home Page



Player live



Episodi



Schermata catalogo musica

10:30 | 1 Settembre 2021

Catalogo M

Musica

Title

Description

Aggiungi

dd music

usic 1

>

2021

usic 2

>

2021

Programma

Schermata form per aggiungere musica

UNIVERS ME
TESTATA MULTIFORME DEGLI STUDENTI UNIME

Email

Password

Registrati

Accedi

Accedi con

Facebook

Schermata di accesso/registrazione