Dipartimento di Ingegneria e Scienza dell'Informazione

Corso di Laurea in
Informatica

Elaborato finale

Base di dati del gestionale 'SanbàPolis' e sistema di video editing

Supervisore
Paolo Bouquet

Laureando
Alessandro Rizzo

Anno accademico 2022/2023

Indice

Sommario

Il progetto 'Sport Tech' nasce con l'idea di sviluppare un sistema software per sfruttare l'infrastruttura appositamente preparata in occasione dei Giochi Olimpici invernali Milano Cortina 2026. Per l'occasione infatti, negli impianti sportivi del Trentino Alto-Adige, nelle località di Val di Fiemme e Anterselva, verranno installati sofisticati e costosi sensori IoT per la raccolta dati e per il monitoraggio delle prestazioni sportive dei relativi atleti. Una volta terminata la manifestazione sportiva però, gli impianti ristrutturati avranno una serie di risorse hardware, che però senza un sistema software non potranno essere utilizzate. L'idea quindi del progetto è fornire un sistema completo per permettere al pubblico di sfruttare i nuovi strumenti installati.

Il primo step del progetto è quello di creare un prototipo di gestionale per il palazzo SanbàPolis dell'Opera Universitaria, il quale è destinato alle squadre di Pallavolo 'Trentino Volley' e Pallacanestro 'Aquila Basket'. Nell'impianto sono state installate 13 telecamere ed un sistema di sensori IoT (in via di sviluppo). Il principio di sviluppo è quindi lo stesso: un'infrastruttura adibita alla raccolta dati che necessita di un sistema di gestione. L'obiettivo del gruppo è quello di sviluppare un sistema software che permetta quindi di sfruttare questa infrastruttura.

Il gruppo ha iniziato a lavorare concretamente al progetto indicativamente da Marzo 2023, suddividendo il lavoro nel seguente modo: due studenti della magistrale di Human Computer Interaction si occupano della realizzazione dell'interfaccia del gestionale e della user experience. Io ed altri due studenti della triennale di informatica ci siamo occupati dello sviluppo del gestionale. La realizzazione del gestionale è stata divisa in tre macro aree, ognuna assegnata ad uno di noi tre studenti della triennale. Mattia Torregiani si è occupato dell'analisi dei requisiti e della stesura del documento SRS [6], dal quale poi io e Dario Tortorici ci siamo bastati per la realizzazione del gestionale. Dario ha sviluppato la business logic del software [7] mentre io mi sono occupato della gestione dei dati.

In questa tesi verrà trattato quindi nel dettaglio il mio lavoro nello sviluppo del gestionale. Il tutto verrà suddiviso in tre principali capitoli che analizzano: lo sviluppo del database del gestionale, lo sviluppo di un sistema di video-editing per i video registrati dalle telecamere del palazzetto, lo sviluppo di un servizio web per la gestione dei dati raccolti dai sensori IoT della palestra.

Il gestionale è stato sviluppato come applicativo web, programmato in linguaggio PHP lato server, Javascript lato client con chiaramente HTML e CSS per la formattazione delle pagine web. Per lo sviluppo dell'applicativo è stato messo a disposizione dall'università un server in cui è stato installato Apache HTTP Server come web server e MariaDB come DBMS per il gestionale ed InfluxDB come database per il salvataggio dei dati generati dai sensori IoT.

1 Introduzione

Nel 2026 si svolgeranno i Giochi Olimpici Invernali Milano Cortina, tra i vari luoghi in cui si terranno le gare, troviamo anche le località del Trentino Alto Adige Val di Fiemme e Anterselva. La regione Trentino Alto Adige ha investito diversi milioni per la ristrutturazione dei propri impianti sportivi adibiti ai Giochi Olimpici Invernali. Una volta terminata la competizione la regione presenterà delle sofisticate infrastrutture per praticare lo sci con relativa tecnologia per il monitoraggio delle prestazioni di chiunque sfrutti le piste. Affinché l'intera struttura risulti funzionante ed efficace, dopo la conclusione dei Giochi Olimpici Invernali, è necessario che gli impianti sportivi presentino un sistema software per l'intera gestione degli impianti.

La proposta del prof. Paolo Bouquet è di sviluppare con l'Università di Trento il sistema software per la gestione di questi impianti sportivi. Il sistema software in questione dovrebbe, oltre permettere

la raccolta e l'organizzazione dei dati dei vari sensori installati sulle piste, anche presentare un sistema di prenotazione online per i clienti che desiderano utilizzare gli impianti sportivi. Il sistema quindi dovrebbe offrire un tariffario dei vari servizi selezionabili, da chiunque desideri utilizzare le piste con poi relativa area riservata per consultare le proprie prestazioni registrate dal sistema. Lo sviluppo di un sistema del genere richiede molto tempo e lavoro, a partire dalla definizione specifica degli obiettivi passando dalla raccolta dei requisiti per poi arrivare allo sviluppo software.

Da questa idea del prof. Bouquet nasce il gruppo 'Sportech'. Il gruppo è composto da tre studenti della triennale di informatica: Rizzo Alessandro, Mattia Torregiani e Dario Tortorici. Oltre a questi sono presenti due studenti della magistrale di Human Computer Interaction: Sebastiano Rossi ed Eleonora Marchini. Il progetto viene supervisionato dal prof. Paolo Bouquet e Sabrina Vettorato. Parallelamente al gruppo supervisionato dal prof. Bouquet, lavora al progetto anche un gruppo supervisionato dal prof. Nicola Conci, che si occupa principalmente di sviluppare un sistema di intelligenza artificiale che analizzi la mole di dati raccolti dai sensori, compresi i video, per poi fornire un interpretazione.

L'obiettivo del gruppo è sviluppare l'idea presentata dal prof. Paolo Bouquet realizzando un sistema con caratteristiche molto simili a quelle degli impianti sciistici del Trentino Alto Adige ma in una realtà molto più piccola. La realtà in considerazione è quella del palazzetto dello sport 'SanbàPolis' dell'Opera Universitaria di Trento. La palestra viene utilizzata dalle squadre di Pallavolo 'Trentino Volley' e Pallacanestro 'Aquila Basket' per partite ed allenamenti. Nel palazzetto sono state installate 13 telecamere ed un sistema di rilevazione di posizione degli atleti in campo. Il primo step del progetto Sportech è quello di realizzare il sistema software per la gestione di questa infrastruttura.

Il progetto Sportech inizia con una prima riunione nel mese di Marzo in cui il prof. Paolo Bouquet illustra ai presenti l'idea generale del progetto e gli obiettivi da conseguire nel tempo. Seguiranno poi successive riunioni in cui verrà fatto un brainstorming delle idee sul progetto. Le prime riunioni del gruppo sono quindi servite ad organizzare le idee e a stabilire, a grandi linee, l'obiettivo iniziale da raggiungere da parte del gruppo. Inoltre, per stabilire cosa doveva fare il software è stata fatta una ricerca iniziale sullo stato dell'arte, per quindi verificare se esistessero già delle soluzioni che si adattavano al problema da risolvere. La maggior parte dei software in commercio, più o meno costosi, si concentrano sull'analisi dei dati e non quindi sulla raccolta. Non ci sono tuttora in commercio dei servizi che offrano un prodotto in grado di adattarsi a pieno all'impianto hardware di un centro sportivo.

In una visione molto ampia, il sistema gestionale del centro SanbaPolis dovrebbe comprendere:

- Gestione delle telecamere, permettendo di registrare allenamenti e partite oppure visualizzando l'evento in questione in diretta;
- Sistema di analisi dei dati raccolti, includendo nell'analisi anche i video;
- Sistema di editing dei video e sessioni di 'studio' dei dati raccolti;
- Gestione dell'organizzazione delle squadre, con i relativi giocatori, che usano il sistema;
- Sistema di prenotazione per l'utilizzo della palestra, con eventuale tariffario per i servizi selezionabili;
- Portale consultabile dai tifosi ed appassionati per seguire le squadre interessate.

Poiché la realizzazione di un sistema del genere, richiede una grande organizzazione ed una grande quantità di tempo, si è deciso di procedere a realizzare il sistema per gradi. La prima versione del sistema comprende:

- Registrazione video di partite ed allenamenti;
- Sistema di gestione delle sessioni di registrazione, con integrato un sistema di video editing;

- Sistema di prenotazione della palestra per le sessioni di registrazione;
- Sistema per la raccolta dei dati generati dai sensori di posizione.

La prima fase per la realizzazione di un software è la raccolta dei requisiti. Per modellare l'universo da trattare, definendo cosa dovesse fare il gestionale, sono stati realizzati inizialmente dei diagrammi UML dei casi d'uso. Sono stati definiti quindi gli attori coinvolti nell'utilizzo del sistema con relative funzionalità. Per comprendere al meglio le esigenze di allenatori e giocatori, gli studenti Sebastiano Rossi ed Eleonora Marchini hanno provveduto a raccogliere informazioni tramite interviste ad atleti ed allenatori che utilizzano la palestra SanbaPolis. L'intera raccolta dei requisiti è stata formalizzata dallo studente Mattia Torregiani nella propria tesi [6].

Successivamente alla raccolta dei requisiti, è iniziato lo sviluppo del gestionale. Il lavoro è stato suddiviso nel seguente modo tra i componenti del gruppo:

- Mattia Torregiani ha creato il documento di specifica dei requisiti;
- Dario Tortorici ha sviluppato la business logic del gestionale;
- Alessandro Rizzo si è occupato della gestione dati;
- Sebastiano Rossi ed Eleonora Marchini si sono occupati dell'interazione degli utenti con il sistema.

Il lavoro mio e di Dario Tortorici si è basato sull'analisi dei requisiti di Mattia Torregiani, il quale ha concluso il tirocinio per primo all'interno del gruppo.

Le tecnologie per lo sviluppo del gestionale sono state, scelte in comune accordo tra i membri del gruppo, sono:

- Linguaggio PHP per la programmazione lato server;
- HTML, CSS e Javascript per programmazione lato client;
- MariaDB come DBMS;
- Apache HTTP Server come server web;
- InfluxDB come database per il salvataggio dei dati sulla posizione, generati dai sensori IoT

Una volta definite le tecnologie, in accordo con il professore Paolo Bouquet è stata richiesta una macchina virtuale all'università per l'installazione di MariaDB, Apache ed InfluxDB. Il server quindi è stato utilizzato per lo sviluppo del gestionale.

2 Database del gestionale

Il gestionale necessita di un sistema di salvataggio dei dati, come nella maggior parte dei software con una certa complessità si è optato per un database. Il DBMS scelto è MariaDB, un database relazionale open source nato come fork di MySQL.

Di seguito verrà riportata la descrizione di ognuna delle fasi svolte per la realizzazione della base di dati. Il processo di progettazione di un database è composto da tre fasi: la progettazione concettuale, la progettazione logica con eventuale normalizzazione ed infine la creazione delle tabelle con la scrittura in codice DDL per il DBMS scelto.

Progettazione concettuale All'inizio del processo di creazione di un database, si affronta la progettazione concettuale. Il suo scopo è definire una rappresentazione corretta della realtà in questione. In questa prima fase, il ragionamento sulla realtà da modellare è molto astratto, infatti durante questo tipo di progettazione l'obbiettivo è delineare quali sono gli elementi della realtà che hanno rilevanza nella base di dati da progettare. Il lavoro ricavato dalla progettazione concettuale viene formalizzato in uno schema concettuale.

Progettazione logica Successivamente la progettazione concettuale, avendo lo schema concettuale come base di partenza, si inizia la progettazione logica. Poiché il risultato della progettazione concettuale è molto astratto, è necessario ricavare un modello efficiente che si adatti alle strutture del sistema di gestione che si intende utilizzare. La progettazione logica quindi consiste nell'adattare lo schema concettuale ad un modello logico scelto, ricavando quindi lo schema logico.

Creazione delle tabelle Il modello logico per la creazione dei database solitamente è il modello relazionale che quindi implica l'utilizzo di un DBMS relazionale. La fase finale quindi per la creazione di un database è formalizzare con il linguaggio del DBMS lo schema logico. Poiché il DBMS utilizzato in questo progetto è relazionale, l'ultima fase da compiere è definire in linguaggio DDL le tabelle delle varie relazioni specificate nello schema logico.

2.1 Progettazione concettuale

La prima fase della realizzazione di una base di dati è la progettazione concettuale. Il modello scelto per la realizzazione dello schema concettuale è il classico modello ER (Entity Relationship).

Di seguito verrà analizzato nel dettaglio il diagramma ER: inizialmente verranno descritte le entità del diagramma. Verranno successivamente trattate le varie associazioni tra entità suddividendo l'ER in tre 'sezioni', per rendere tutto il più comprensibile.

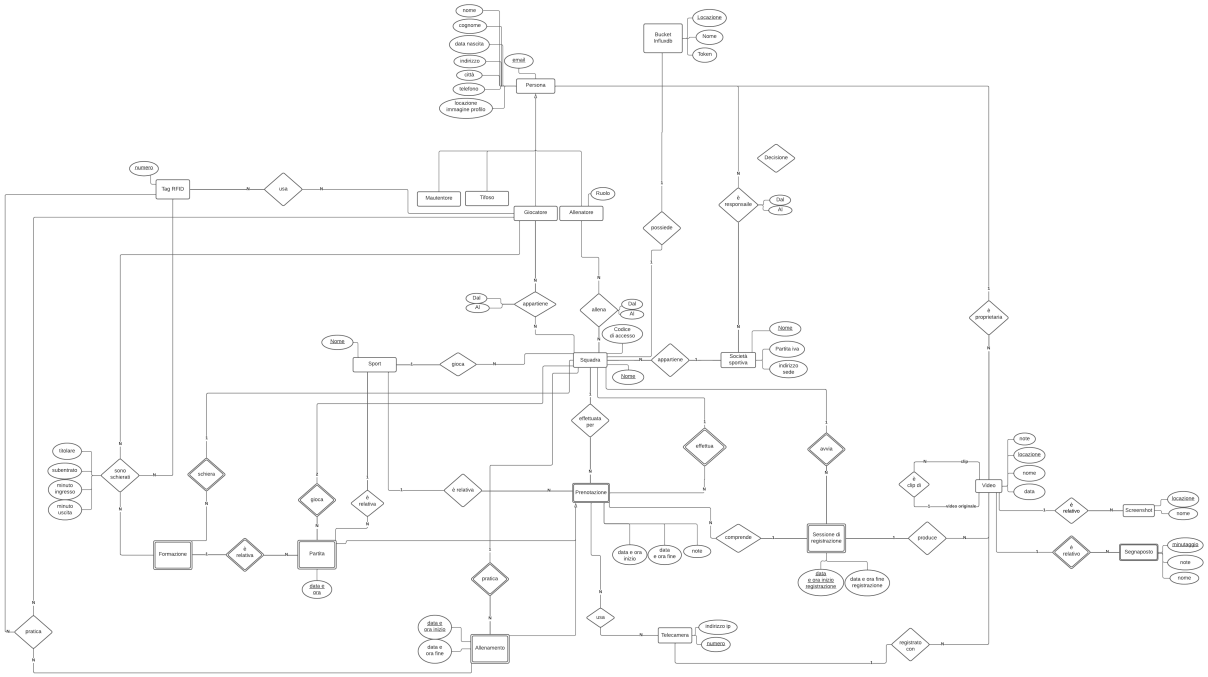

Figura 2.1: Diagramma ER della base di dati

2.1.1 Descrizione delle entità

Persona L'entità che descrive una qualsiasi persona registrata nel sistema, incorporando anche il concetto di utente. Gli attributi dell'entità sono dati anagrafici: nome, cognome, data di nascita, indirizzo, città, telefono, email; inoltre è salvata la locazione dell'immagine del profilo. La chiave primaria dell'entità è l'indirizzo email, che è un identificativo univoco. L'entità persona ha poi delle

entità figlie che definiscono i ruoli delle persone registrate all'interno del sistema: Manutentore, Tifoso, Allenatore, Giocatore.

Manutentore L'entità che rappresenta un amministratore del sistema, che ha i privilegi per l'intera gestione del sito.

Tifoso L'entità che rappresenta il tifoso, che ha la possibilità di registrarsi al sistema per seguire la propria o le proprie squadre preferite.

Allenatore L'entità che rappresenta l'allenatore o un componente dello staff dell'allenatore, discriminato dall'attributo 'tipo'.

Giocatore L'entità che rappresenta il giocatore di una qualsiasi squadra, relativa a un qualsiasi sport.
Le entità Manutentore, Tifoso, Giocatore non hanno attributi (escludendo la chiave primaria email; questo argomento verrà approfondito nella sezione dedicata allo schema logico); perché in questa fase del progetto ci si è limitati ad una prima visione in grado di dare l'organizzazione generale della base di dati. Successivamente quando verranno sviluppate funzionalità aggiuntive al sistema sarà comunque possibile aggiungere gli attributi necessari.

Tag Rfid Il sensore IoT che i giocatori, durante una partita o allenamento, indossano col fine di raccogliere dati relativi alla posizione e ai movimenti in campo. L'unico attributo è un numero identificativo che rappresenta il numero del tag. La numerazione non viene gestita dal sistema, si presume quindi che ogni tag venga etichettato con un apposito numero e successivamente inserito nel sistema.

Bucket InfluxDb Un'entità che descrive il concetto di bucket del database InfluxDb; un database che registra serie di dati temporali. L'argomento verrà approfondito nel terzo capitolo dedicato alla gestione dati generati dai tag RFID. Per il momento quello che interessa sapere è che ci sono gli attributi locazione, nome, token, squadra.

Società sportiva Questa entità rappresenta la società sportiva proprietaria di una o più squadre. Gli attributi di questa entità sono: nome, partita iva, indirizzo della sede. La chiave primaria è la partita iva, poiché identifica univocamente un soggetto che esercita un'attività.

Sport Un'entità con un singolo attributo, il quale è anche chiave primaria, che rappresenta i vari sport interessati dal sistema.

Squadra Una qualsiasi squadra che utilizza il sistema.

Prenotazione Il gestionale permette di prenotare il campo per sfruttare l'infrastruttura, è quindi necessario inserire l'entità prenotazione. L'entità è debole quindi è identificata tramite l'associazione con la squadra che ha effettuato la prenotazione e con la data e ora di inizio dell'evento relativo alla prenotazione. Una prenotazione può riguardare una partita o un allenamento, per questo 'prenotazione' ha due entità figlie: Partita e Allenamento. Gli attributi di questa entità sono: data e ora di inizio, data e ora di fine, note (si intende un testo per una descrizione dell'evento).

Allenamento Una delle entità figlie di prenotazione, rappresenta un allenamento di una qualsiasi squadra, di un qualsiasi sport. Ha come attributo la data e ora di inizio e la data e ora di fine.

Partita L'altra entità figlia di prenotazione, rappresenta una partita di un qualsiasi sport. La partita è identificata dalla relativa prenotazione e come attributo ha la data e ora dell'evento.

Formazione L'entità che rappresenta la formazione di una squadra schierata per una partita. Una formazione, essendo un'entità debole, è identificata della squadra che la schiera e dalla partita per cui è stata schierata.

Sessione di registrazione La registrazione video e la raccolta dei dati prodotti dai sensori IoT, di una partita o allenamento, viene identificata nel concetto di sessione di registrazione. Gli attributi di questa entità sono data e ora di inizio e fine registrazione. La sessione di registrazione è un'entità debole ed è quindi identificata dall'associazione con la squadra che avvia la sessione di registrazione e la data e ora di inizio registrazione.

Telecamere Rappresenta una delle telecamere installate nell'impianto. Gli attributi sono l'indirizzo IP ed il numero identificativo della telecamera. Si assume che ad ogni telecamera, esternamente dal sistema, venga assegnato un numero identificativo; la gestione della numerazione delle telecamere non è un problema della base di dati o del sistema.

Video L'entità che descrive un video, di un allenamento o partita, prodotto da una sessione di registrazione o da un processo di video editing. Gli attributi dell'entità sono: nome, note, data e locazione; quest'ultimo identifica univocamente il video.

Screenshot Il sistema di video editing prevede la possibilità di estrarre degli screenshot del video, quindi è stata creata un'apposita entità. Uno screenshot è identificato dalla sua locazione ed ha anche un attributo 'nome'.

Segnaposto Un segnaposto è una sorta di etichetta che poniamo, durante la riproduzione di un video, in un determinato minutaggio. Il segnaposto, oltre che il minutaggio, permette di aggiungere un nome ed una descrizione (tramite gli attributi 'nome' e 'nota'). Questo può essere utile per evidenziare momenti cruciali di un allemaneto o di una partita. L'entità è debole poiché necessita, oltre il minutaggio, anche l'associazione con il relativo video per essere identificata.

2.1.2 Descrizione delle associazioni

Di seguito verranno analizzate le varie associazioni tra le entità. Poiché il diagramma ER è molto esteso, suddividiamo il diagramma in tre 'sotto sezioni': Gestione delle persone, Gestione degli eventi, Gestione dei video. Verrà riportata, per ognuna di queste sezioni, una porzione di diagramma ER. Per una maggiore leggibilità, in alcune porzioni, sono stati rimossi gli attributi meno significativi poiché lo scopo in questo momento è risaltare appunto le associazioni.

Gestione delle persone

Persona-Società Sportiva Per ogni società sportiva, viene nominato un responsabile, di conseguenza troviamo un'apposita associazione. La cardinalità dell'associazione è $n : n$, poiché è possibile che una persona sia responsabile di più società sportive o che una società abbia avuto più responsabili nel tempo; tutto questo in intervalli temporali diversi, per questo motivo l'associazione presenta due attributi: data inizio, data fine che indicano, per l'appunto, le date di inizio e fine in cui un persona ha ricoperto il ruolo di responsabile per una società. L'attributo 'data fine', rimane vuoto finché una persona è il responsabile in carica della società.

Giocatore-Tag RFID Durante un evento, che sia partita o allenamento ad un giocatore viene associato un tag RFID. La cardinalità dell'associazione è $n : n$ perché un giocatore probabilmente non utilizzerà sempre lo stesso tag ad ogni evento e lo stesso tag più essere, ad eventi diversi, essere usato da diversi giocatori.

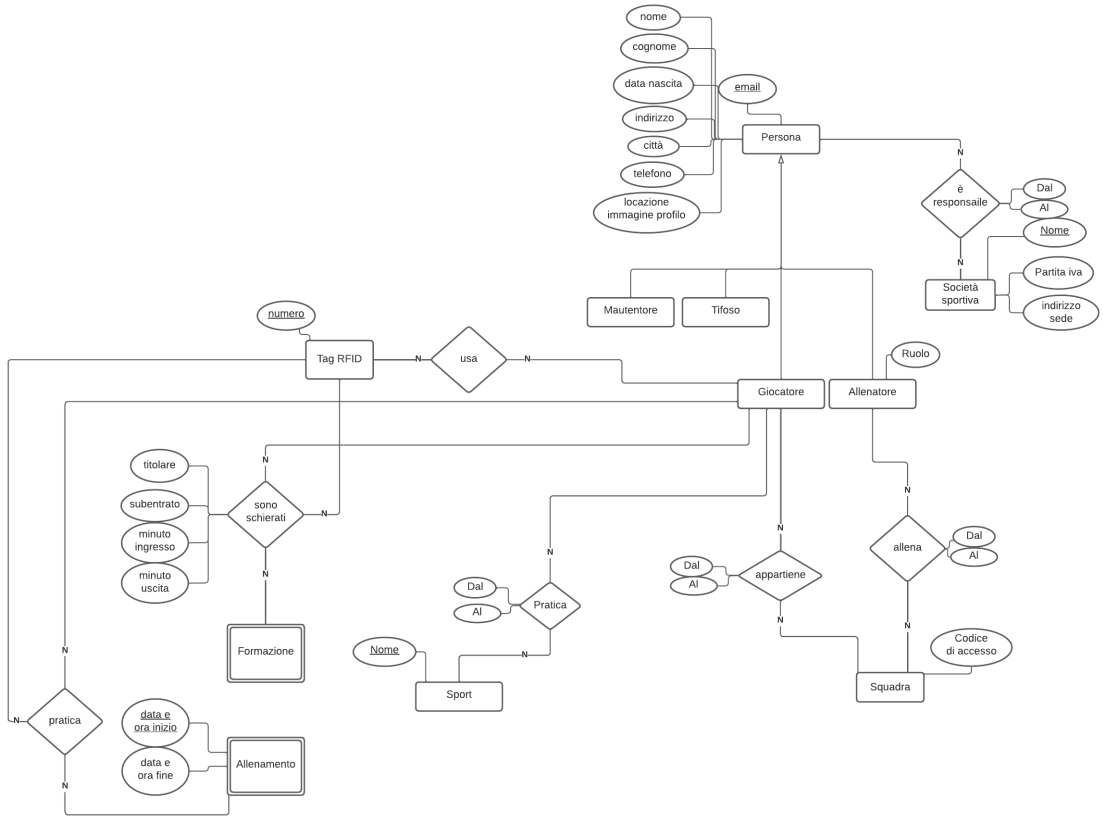

Figura 2.2: Porzione di ER che gestisce la modellazione delle Persone

Giocatore-Formazione Quando si svolge una partita, vengono schierati in campo dei giocatori secondo una formazione. Viene quindi creata l'associazione $n : n$ tra giocatori e formazione poiché in ogni formazione vengono schierati più giocatori ed ogni giocatore sarà presente durante le varie formazioni delle varie partite. L'associazione presenta i seguenti attributi: titolare (per sapere se il giocatore è stato schierato titolare), subentrato (per sapere se il giocatore è subentrato a partita in corso), minutaggio di ingresso e minutaggio di uscita.

Tag RFID-Formazione Nell'associazione tra giocatore e formazione, troviamo anche la partecipazione dell'entità Tag RFID. Durante una partita i giocatori, per il monitoraggio della prestazione indosseranno un sensore IoT. Di partita in partita, il sensore associato ad ogni giocatore non sarà necessariamente lo stesso, di conseguenza ad ogni formazione schierata, viene associato ad ogni giocatore, il tag utilizzato per quella partita.

Giocatore-Allenamento I giocatori praticano regolarmente degli allenamenti, quindi viene creata un'associazione tra lo specifico allenamento ed i giocatori che ne prendono parte. La cardinalità è $n : n$ poiché ad un allenamento prendono parte più giocatori, mentre un giocatore pratica più allenamenti.

Tag RFID-Allenamento Lo stesso concetto che sta alla base dell'associazione
Tag RFID-Formazione lo possiamo applicare a Tag RFID-Allenamento: durante ogni allenamento un giocatore indosserà un Tag per il monitoraggio della prestazione. Poiché tra i vari allenamenti il tag usato da ogni giocatore non è necessariamente lo stesso, ci serviamo di questa associazione per registrare il particolare sensore utilizzato dal giocatore nello specifico allenamento.

Giocatore-Squadra L'associazione che definisce a quale squadra appartiene un giocatore; troviamo due attribui (dal, al) che indicano l'intervallo di tempo in cui un giocatore appartiene ad una determinata squadra. La cardinalità è $n : n$ poiché, nonostante un giocatore appartenga ad una sola

squadra alla volta, viene conservato lo storico delle associazioni tra squadra e giocatore con relativo intervallo di tempo. Ad una squadra invece ovviamente, è composta da più giocatori.

Allenatore-Squadra Ogni squadra è allenata da un allenatore, ma come per il discorso applicato ai giocatori, viene conservato lo storico dei vari allenatori di una squadra, per questo la cardinalità è $n : n$; anche qui per salvare lo storico ci serviamo di due attributi dell'associazione 'data inizio', 'data fine'. Come una squadra può avere nel tempo più allenatori, un allenatore può aver allenato più squadre nel tempo.

Gestione degli eventi

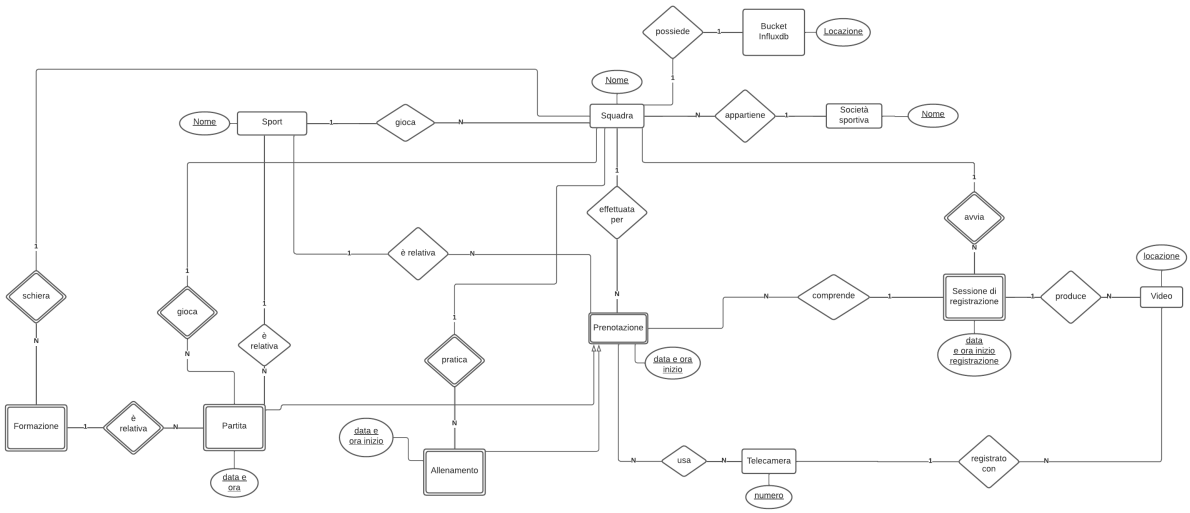

Figura 2.3: Porzione di ER che gestisce la modellazione degli Eventi

Squadra-Sport Una squadra pratica un determinato sport, per questo inseriamo l'associazione $1 : n$ tra le due entità. Lo stesso sport chiaramente viene praticato da più squadre.

Squadra-Prenotazione Una prenotazione viene effettuata per una determinata squadra, a sua volta la stessa squadra potrà effettuare varie prenotazioni. La cardinalità è quindi $1 : n$.

Squadra-Società sportiva Ogni squadra appartiene ad una società sportiva, mentre una società può possedere più squadre. La cardinalità è quindi $1 : n$.

Squadra-Sessione di registrazione Il sistema permette di sfruttare l'infrastruttura hardware per il monitoraggio delle prestazioni degli atleti; per questo motivo associamo ad ogni sessione di registrazione, la squadra che l'ha avviata. Poiché una squadra gioca più partite o pratica più allenamenti, avvierà più sessioni di registrazione. Ogni sessione invece è associata solamente ad una squadra. Cardinalità quindi $1 : n$.

Squadra-Partita Le squadre giocano più partite; ogni partita invece, sebbene sia giocata da due squadre, è associata solamente alla squadra di casa. Questo perché il sistema, ad oggi, ha mappate solamente le squadre che utilizzato il palazzetto SanbàPolis. Per questo motivo abbiamo una cardinalità $1 : n$

Squadra-Allenamento Allo stesso modo delle partite, una squadra pratica diversi allenamenti. Ogni allenamento è associato invece ad una sola squadra, per cui cardinalità $1 : n$.

Squadra-Bucket InfluxDb Come accennato precedentemente, il Bucket InfluxDb è un concetto che verrà approfondito nel terzo capitolo. Per il momento basta sapere che ogni ha squadra ha associato il proprio Bucket del Database InfluxDb.

Squadra-Formazione Ad ogni partita, una squadra schiera una formazione diversa quindi andiamo a creare questa associazione $1 : n$. Una formazione viene schierata da una sola squadra, mentre la stessa squadra andrà a schierare più formazioni: una per ogni partita.

Prenotazione-Sport La prenotazione è relativa ad evento di un determinato sport; per questo motivo aggiungiamo un'associazione $1 : n$. Uno sport chiaramente è associato a varie prenotazioni.

Prenotazione-Sessione di registrazione Per poter registrare un evento (partita o allenamento) è necessario aver prenotato il campo. Solitamente per ogni prenotazione, dovremmo avere una sessione di registrazione, tuttavia non si esclude la possibilità di interrompere e riprendere la registrazione, andando quindi a creare più sessioni di registrazione per una prenotazione. Per questo motivo creiamo un'associazione $1 : n$.

Prenotazione-Telecamera Il palazzetto SanbàPolis, ad oggi è dotato di 13 telecamere; tuttavia per una prenotazione è necessario sapere quante, delle 13 telecamere, sono state messe a disposizione. Questo perché più telecamere vengono utilizzate, più video verranno prodotti e di conseguenza più stoarge verrà occupato. Quindi, è verosimile che in certi momenti non venga consentito o semplicemente non è nell'interesse della squadra utilizzare un elevato numero di telecamere per un determinato evento. La cardinalità è $n : n$ perché la stessa telecamera verrà utilizzata in più prenotazioni, mentre una prenotazione avrà più telecamere abilitate per la registrazione.

Partita-Sport Ogni partita giocata è relativa ad un singolo sport, mentre dello stesso sport vengono giocate più partite. Abbiamo quindi un'associazione $1 : n$.

Partita-Formazione Il concetto di formazione esiste nel momento in cui viene giocata una partita, di conseguenza c'è un'associazione che lega la formazione in questione alla relativa partita. La cardinalità è $1 : 1$ perché esiste una sola formazione per ogni partita.

Gestione dei Video

Video-Persona Ogni video presente nel sistema ha un collegamento alla persona proprietaria del video, che possiamo riconoscere come autore. Nella maggior parte dei casi è probabile che la figura dell'allenatore risulti proprietaria del video, poiché è colui che supervisiona l'allenamento e che quindi avvia[1] o comunque che ha interesse fruire, della sessione di registrazione. Il proprietario di un video, tuttavia, è stato identificato nell'Entità Persona (non Allenatore quindi), poiché il sistema permette di usare il sistema di video editing anche al giocatore, dove potrà quindi creare i propri video manipolando le registrazioni.[2] La cardinalità è di $1 : n$ poiché un video ha un solo proprietario, ma una persona può essere proprietaria di più video.

Video-Sessione di registrazione La sessione di registrazione ovviamente produce dei video, quindi troviamo un'associazione $1 : n$. Ogni video appartiene ad una sola sessione di registrazione, mentre ogni sessione produce più video. Una sessione produce tanti video quante sono le telecamere attive come minimo; non si esclude un numero superiore in caso di interruzione e ripresa delle registrazioni.

[1] Nello sviluppo finale del gestionale, è altamente probabile che venga sviluppato un sistema di avvio automatico della sessione, in base agli orari specificati.

[2] L'argomento verrà approfondito nel secondo capitolo.

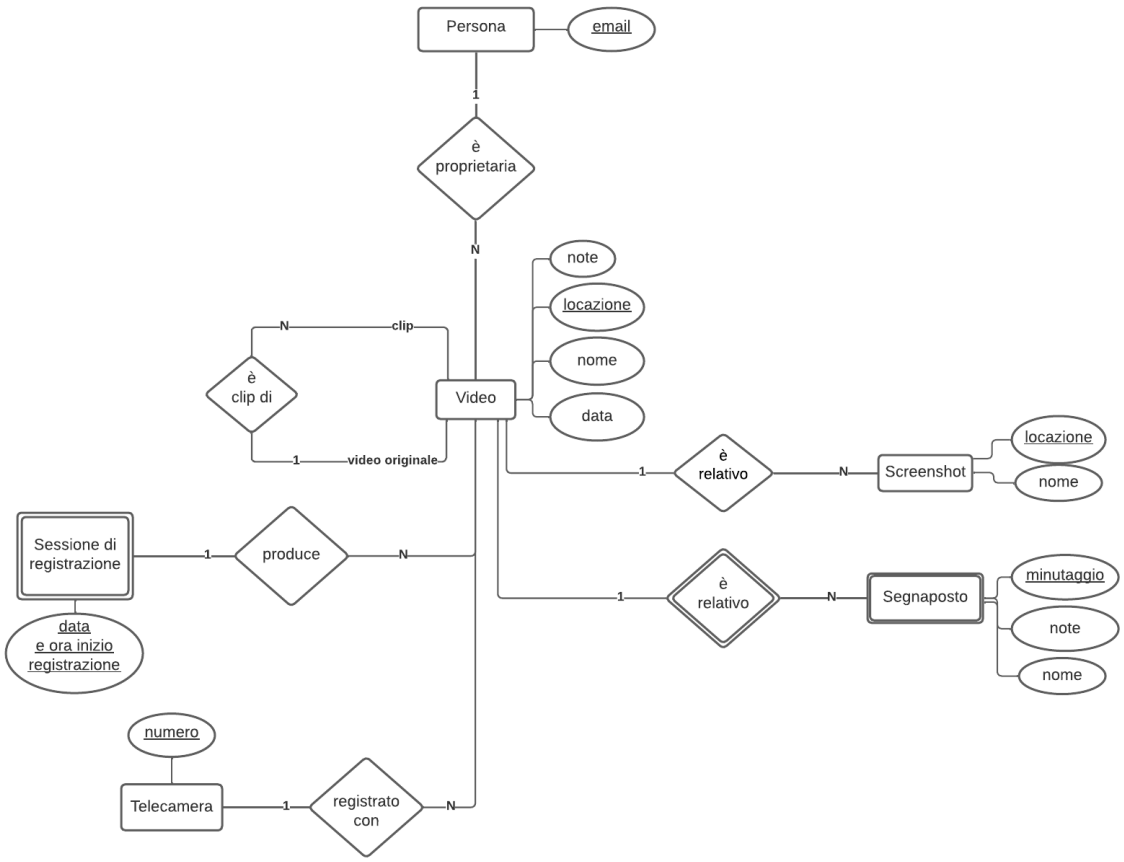

Figura 2.4: Porzione di ER che gestisce la modellazione dei Video

Video-Telecamera Ogni video, è generato dalla registrazione con una determinata telecamera, di conseguenza troviamo un'associazione $1 : n$. Una telecamera chiaramente potrà, in varie sessioni di registrazione, registrare più video.

Video-Screenshot Come accennato precedentemente, il sistema permette di estrarre degli screenshot da un video quindi è necessario avere un'associazione tra uno screen ed il video da cui è stato estratto. Possiamo estrarre più screenshot da un video, mentre uno screenshot è relativo ad un unico video. Per questo motivo, troviamo una cardinalità $1 : n$.

Video-Segnaposto Per ogni video è possibile salvare dei segnaposto, quindi anche qui come per gli screenshot, abbiamo un'associazione $1 : n$. Per un video si possono salvare diversi segnaposto, ma un segnaposto è relativo ad un solo video.

Ruolo di Video e Clip Per l'entità video, possiamo definire un ruolo: il ruolo di video originale ed il ruolo di clip. Durante l'editing video è possibile estrarre delle clip da una registrazione; la clip non è altro che un nuovo video. Distinguiamo quindi tramite un'associazione quelli che sono dei video originali e quelli che invece sono delle clip, estratte tramite il processo di video editing.

2.2 Progettazione Logica

Una volta terminata la progettazione concettuale, da cui abbiamo prodotto il diagramma ER, passiamo alla progettazione logica. La progettazione logica consiste nel manipolare il risultato della progettazione concettuale, per ottenere uno schema logico. Poiché il database utilizzato, MariaDB, è un database relazionale andremmo a sviluppare una progettazione logica relazionale che darà come risultato uno schema logico relazionale.

In questo processo, tutte le entità verranno trasformate il relazioni, mentre in base alle necessità le associazioni sono state trasformate come chiavi esterne o a loro volta delle associazioni. Di seguito troviamo lo schema logico.

2.2.1 Schema logico

- **Persona** (email, password, nome, cognome, data nascita, città, indirizzo, telefono, locazione immagine profilo, verificato, data registrazione)
- **Allenatore** (email, squadra, tipo, privilegi telecamere)
- **Tipo-Allenatore** (nome)
- **Tifoso** (email)
- **Giocatore** (email)
- **Manutentore** (email)
- **Tag RFID** (id)
- **Video** (locazione, nome, nota, autore, sessione, telecamera)
- **Clip-Video** (locazione video originale, locazione clip)
- **Screenshot** (locazione, video, nome, note)
- **Segnaposto** (minutaggio, video, nome, note)
- **Squadra** (nome, società, sport, codice di accesso)
- **Bucket-InfluxDb**(locazione, nome, token, squadra)
- **Società Sportiva** (nome, indirizzo, partita iva, responsabile)
- **Sport**(nome)
- **Allenatore-Squadra** (allenatore, squadra, data inizio, data fine)
- **Giocatore-Squadra** (giocatore, squadra, data inizio, data fine)
- **Formazione** (squadra, partita)
- **Partita** (squadra di casa, data e ora inizio, data e ora fine, sport, prenotazione)
- **Allenamento** (squadra, data e ora inizio, data e ora fine, prenotazione)
- **Prenotazione** (autore, data-ora inizio, data-ora fine, sport, squadra, sessione, allenamento, partita)
- **Sessione** di registrazione (autore, data-ora inizio, data-ora fine, prenotazione, calendar event)
- **Calendar Events** (id, groupId, allDay, title, start, end, daysOfWeek, startTime, endTime, startRecur, endRecur, url, interactive, className, editable, startEditable, durationEditable, resourceEditable, resourceId, resourceIds, display, overlap, color, backgroundColor, borderColor, textColor)
- **Formazioni-Giocatori** (formazione, giocatore, titolare, subentrato, minuto ingresso, minuto uscita, tag giocatore)
- **Allenamenti-Giocatori** (allenamento, giocatore, tag giocatore)
- **Inviti-Allenatori** (email)

- **Inviti-Giocatori** (email)
- **Telecamera** (numero, indirizzo ipv4, indirizzo ipv6)
- **Telecamere-Prenotazioni** (prenotazione, telecamera)

Andremo ora a spiegare nel dettaglio le relazioni che necessitano di un approfondimento, poiché molti concetti sono già stati trattati nella definizione delle entità, di seguito verranno descritte le differenze e novità rispetto alle informazioni che si possono ricavare dalla progettazione concettuale.

Persona La relazione Persona presenta gli stessi attributi dell'entità 'Persona', tuttavia troviamo alcuni nuovi attributi:

- 'password': la password dell'utente, nel database non verrà salvata in chiaro ma verrà garantita la segretezza salvandone il digest calcolato tramite una funzione di hash;
- 'verificato': attributo booleano che viene impostato a *true*, una volta verificato l'account tramite il diffuso metodo della verifica tramite email;
- 'data-registrazione': la data di quando un utente effettua la registrazione

Tifoso, Giocatore, Manutentore, Allenatore La specializzazione delle quattro classi figlie di 'Persona' è stata tradotta in un'associazione 1 : 1 tra persona e l'entità figlia. Troviamo quindi nelle relazioni l'attributo email che funge da chiave esterna, ed anche chiave primaria, a Persona. A differenza delle altre tre, alla figura dell'allenatore aggiungiamo un attributo intero 'privilegi telecamere' per appunto attribuire un livello di accesso alle telecamere.

Tipo Allenatore Semplicemente specifica qual'è il ruolo dell'allenatore (allenatore, vice allenatore...).

Clip-Video L'associazione tra clip e video viene tradotta con una relazione, in cui salvo una coppia con due chiavi esterne alla relazione video. Un attributo indica la locazione della clip, mentre l'altro attributo specifica il relativo video originale, da cui la clip è stata estratta.

Allenatore-Squadra L'associazione tra Allenatore e Squadra, viene convertita in una relazione, dove vengono associati l'allenatore con la relativa squadra allenata; specificando l'intervallo di tempo. 'Allenatore' e 'Squadra' sono due chiavi esterne alle rispettive relazioni. La chiave primaria è data dalla tripla (allenatore, squadra, data inizio). L'attributo data fine può non contenere valori, nel caso in cui il rapporto tra allenatore e squadra non sia terminato.

Giocatore-Squadra Lo stesso concetto applicato ad 'Allenatore-Squadra' lo possiamo applicare anche per questa relazione: convertiamo l'associazione dell'ER in una relazione. Abbiamo due chiavi esterne, alle relative relazioni che sono 'Giocatore' ed 'Allenatore'; le quali, insieme a 'data inizio' formano la chiave primaria. L'attributo 'data fine' assumerà un valore solamente quando il giocatore cesserà di essere un membro della squadra in questione.

Calendar Events e Prenotazione Per la gestione del calendario, è stato utilizzato un framework JavaScript che necessita nel database una specifica tabella. Andiamo quindi a creare la specifica relazione, che concettualmente non è altro che una prenotazione. Per questo motivo la relazione 'Prenotazione' ha una chiave esterna, l'attributo 'calendar event', alla relazione 'Calendar Events'. Non era possibile fare il contrario, ovvero mettere una chiave esterna a 'Calendar Events' verso 'Prenotazione', poiché la tabella richiesta dal Framework necessita di specifici attributi. La chiave primaria della relazione 'Prenotazione', come specificato per la relativa entità nell'ER, è la coppia ('autore', 'data e ora inizio'), mentre per la relazione 'Calendar Events' la chiave primaria è data dall'attributo 'title'. L'univocità del valore title sarà garantita dal sistema PHP.

Formazione-Giocatori L'associazione che associa i giocatori schierati in una formazione è stata tradotta nello schema logico con una relazione. La relazione comprende gli attributi dell'associazione tra le due entità in questione del diagramma ER, inoltra ha le due chiavi esterne alle rispettive relazioni 'Formazione' e 'Giocatore'. Notiamo l'attributo 'tag giocatore' che è una chiave esterna alla relazione Tag RFID che indica appunto il tag indossato dal giocatore durante la partita. La chiave primaria è data dalla coppia (formazione, giocatore), le due chiavi esterne per l'appunto.

Allenamenti-Giocatori Come per la relazione precedente, l'associazione tra un allenamento ed i rispettivi giocatori che ne prendono parte è stata convertita in un'altra relazione. Abbiamo come attributi, le due chiavi esterne 'allenamento' e 'giocatore' alla rispettive relazioni le quali, assieme, fungono da chiave primaria. Abbiamo inoltre l'attributo tag giocatore che non è altro che una chiave esterna alla relazione Tag RFID; indica il tag indossato dal giocatore durante l'allenamento.

Inviti Allenatori e Inviti Giocatori Per la registrazione di giocatori ed allenatori è necessario ricevere un invito via email. Sono state quindi pensate due relazione atte a contenere la lista di email a cui il manutentore, nel caso dell'allenatore, o l'allenatore nel caso del giocatore inviare l'invito per la registrazione. Da notare che, poiché l'utente non è ancora stato registrato, questa relazione non può conterete la chiave esterna a 'Persona' o 'Allenatore' o 'Giocatore'.

Telecamera Oltre al numero identificativo, alla relazione 'Telecamera' sono stati forniti due attributi per specificare l'indirizzo IP in base alla versione. Ad oggi le telecamere dovrebbero lavorare in IPv4, tuttavia preventivamente è stato aggiunto l'attributo per lavorare in IPv6.

Telecamere-Prenotazioni L'associazione tra l'entità 'Telecamera' e l'entità 'Prenotazione' è tradotta in una relazione contenente una coppia (prenotazione, telecamere) la quale è anche chiave primaria, che contiene le due chiavi esterne alle rispettive relazioni.

2.3 Tabelle del Database

L'ultima cosa fare per la realizzazione del database è quella scrivere il codice SQL in DDL per creare effettivamente la base di dati in MariaDB. Il codice per la creazione del database lo si può trovare nella repository del progetto su GitHub[5]. Di seguito troviamo lo schema delle tabelle del database.

- persone (id, email, nome, cognome, data_nascita, citta, indirizzo, telefono, digest_password, locazione_immagine_profilo, verificato, data_ora_registrazione)
- tipi_allenatori (nome_tipo)
- allenatori (email, tipo, privilegi_cam)
- tifosi (email)
- giocatori (email)
- manutentori (email)
- tag_rfid (id)
- societa_sportive (partita_iva, nome, indirizzo, responsabile, codice)
- sport (nome_sport)
- squadre (id, nome, societa, sport, codice)
- allenatori_squadre (id, email_allenatore, id_squadra, data_inizio, data_fine)
- giocatori_squadre (id, email_giocatore, id_squadra, data_inizio, data_fine)

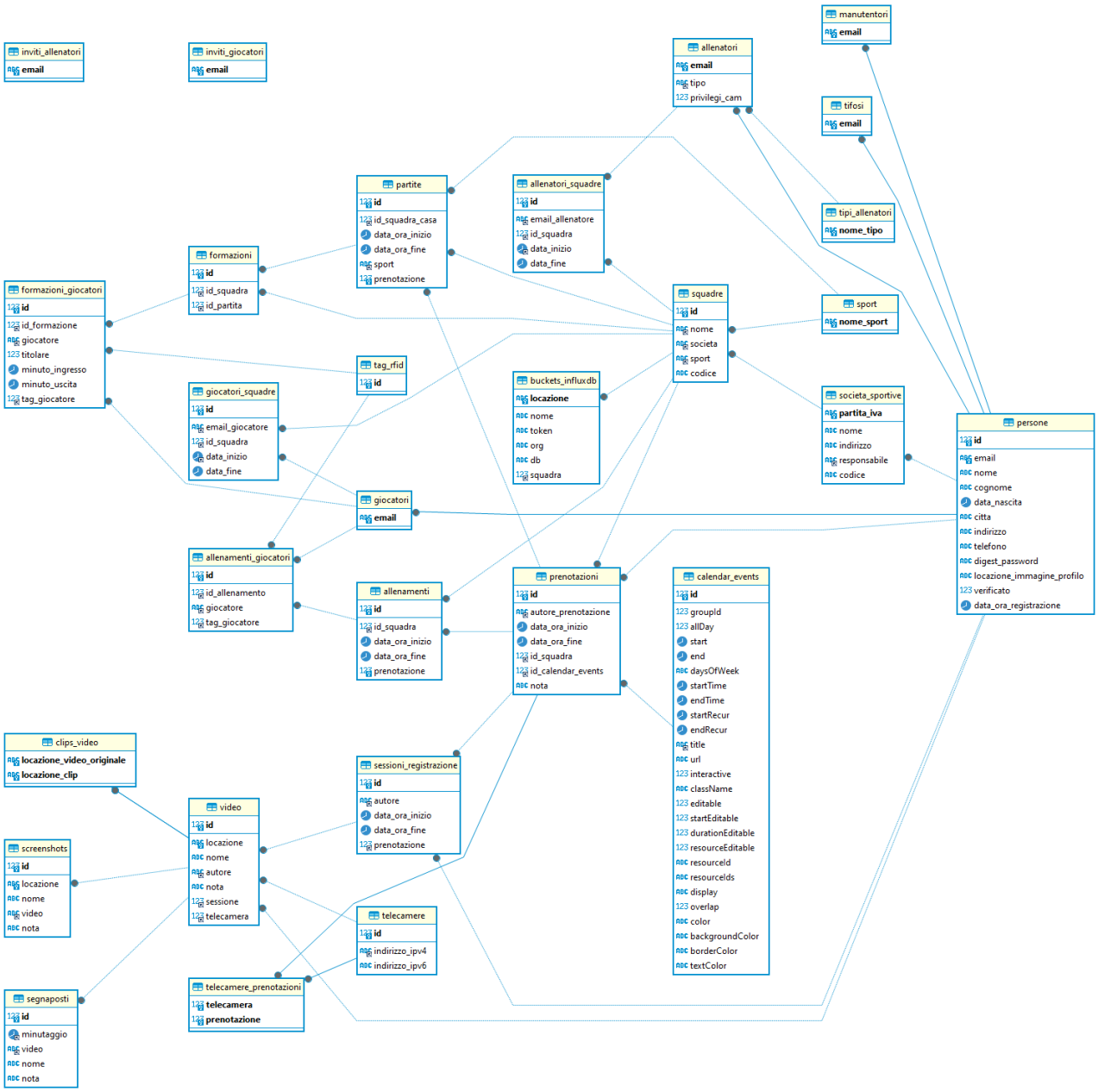

Figura 2.5: Rappresentazione grafica dello schema logico del Database

- calendar_events (id, groupId, allDay, start, end, daysOfWeek, startTime, endTime, startRecur, endRecur, title, url, interactive, className, editable, startEditable, durationEditable, resourceEditable, resourceId, resourceIds, display, overlap, color, backgroundColor, borderColor, textColor)
- prenotazioni (id, autore_prenotazione, data_ora_inizio, data_ora_fine, id_squadra, id_calendar_events, nota)
- sessioni_registrazione (id, autore, data_ora_inizio, data_ora_fine, prenotazione)
- telecamere (id, indirizzo_ipv4, indirizzo_ipv6)
- video (id, locazione, nome, autore, nota, sessione, telecamera)
- clips_video (locazione_video_originale, locazione_clip)
- screenshots (id, locazione, nome, video, nota)

- segnaposti (id, minutaggio, video, nome, nota)
- partite (id, id_squadra_casa, data_ora_inizio, data_ora_fine, sport, prenotazione)
- allenamenti (id, id_squadra, data_ora_inizio, data_ora_fine, prenotazione)
- formazioni (id, id_squadra, id_partita)
- formazioni_giocatori (id, id_formazione, giocatore, titolare, minuto_ingresso, minuto_uscita, tag_giocatore)
- allenamenti_giocatori (id, id_allenamento, giocatore, tag_giocatore)
- inviti_allenatori (email)
- inviti_giocatori (email)
- telecamere_prenotazioni (telecamera, prenotazione)
- buckets_influxdb (locazione, nome, token, org, db, squadra)

Rispetto allo schema logico, sono state effettuate delle modifiche per il database. Tutte le relazioni che avevano chiavi primarie multi attributo alle quali facevano riferimento altre relazioni, sono state convertite in tabelle con un nuovo attributo id (valore intero, numero auto incrementale) come chiave primaria. Questo per non avere chiavi esterne multi attributo. Ovviamente gli attributi che dovevano comporre la chiave multi valore, sono stati impostati come univoci.

In certi casi, nonostante la chiave primaria fosse composta da un solo attributo, si è optato per inserire comunque un id, vincolando la precedente chiave come univoca. Questo perché in alcuni casi, nello sviluppo del sito web, è risultato più utile ricercare una tpla di una tabella tramite numero piuttosto che una stringa molto lunga. Di seguito troviamo la lista dei vincoli.

- V1(persone): (email) Univoco
- V2(squadre): (nome, societa) Univoco
- V3(allenatori_squadre): (email_allenatore, id_squadra, data_inizio) Univoco
- V4(giocatori_squadre): (email_giocatore, id_squadra, data_inizio) Univoco
- V5(calendar_events): (title, start) Univoco
- V6(prenotazioni): (autore_prenotazione, data_ora_inizio, data_ora_fine) Univoco
- V7(sessioni_registrazione): (autore, data_ora_inizio) Univoco
- V8(telecamere): (indirizzo_ipv4, indirizzo_ipv6) Univoco
- V9(video): (locazione) Univoco
- V10(screenshots): (locazione) Univoco
- V11(segnaposti): (minutaggio, video) Univoco
- V12(partite): (id_squadra_casa, data_ora_inizio) Univoco
- V13(allenamenti): (id_squadra, data_ora_inizio) Univoco
- V14(formazioni): (id_squadra, id_partita) Univoco
- V15(formazioni_giocatori): (id_formazione, giocatore) Univoco
- V16(allenamenti_giocatori): (id_allenamento, giocatore) Univoco

2.4 Il database sfruttato dalla business logic del gestionale

Le principali funzioni del gestionale, che necessitano della base di dati, ad oggi sono: la gestione delle prenotazioni per le sessioni di registrazione e l'editing video. Vediamo quindi di seguito come il database viene sfruttato dalla business logic del gestionale.

2.4.1 Autenticazione

Per sfruttare le funzionalità del gestionale è necessario essere autenticati ed avere i permessi necessari; tutti i dati anagrafici delle persone registrate nel gestionale, si trovano nella tabella 'Persone'. Nel momento in cui, un utente si autentica, il sistema verifica le credenziali confrontandole con quelle salvate nella tabella 'Persone'. Una volta autenticato, l'utente viene distinto in base alle tabelle che definiscono i ruoli: Allenatore, Giocatore, Manutentore, Tifoso. In base alla categoria di utenti a cui appartiene il sistema permetterà all'utente di accedere alla funzionalità di cui ha i permessi.

2.4.2 Gestione delle prenotazioni

La gestione delle prenotazioni avviene tramite un calendario; per ogni evento mostrato sul calendario è necessario sapere quali sono le informazioni che lo distinguono come per esempio la data, l'ora, la squadra coinvolta, il tipo di evento ecc. Queste varie informazioni vengono ricavate da più tabelle.

- La tabella 'Prenotazioni' specifica la squadra interessata, l'autore della prenotazione e data e ora dell'evento;
- Per distinguere il tipo di evento è necessario consultare le relative tabelle, 'Allenamenti' o 'Partite', e verificare in quale di queste è specificata la prenotazione in questione;
- Per sapere quante telecamere sono associate alla prenotazione ci serviamo della tabella 'Telecamere-Prenotazioni';
- Infine la tabella 'Sessioni-Registrazione' c'è il riferimento alla relativa prenotazione

Ovviamente la base di dati contiene tutte le informazioni che riguardano l'organizzazione delle varie squadre con i relativi atleti ed allenatori.

2.4.3 Gestione dei video

Le sessioni di registrazione producono dei video che poi è possibile consultare e modificare tramite le funzionalità dell'editing video. La sezione di editing necessita di sapere diverse cose ad esempio quali sono gli screenshot ed i segnaposto associati al video in questione, oppure quali sono le clip estratte dal video. Per ogni sessione ci possono essere più video dato che è possibile riprendere l'evento in questione da inquadrature diverse, con le diverse telecamere. Per accedere a queste informazioni ci appoggiamo alle varie associazioni descritte nella sezione 2.1.2.

2.5 Considerazione sull'efficienza delle query

Fino a questo momento, il sistema è stato sviluppato concentrandosi sulle funzionalità di base, dove l'accesso al database avveniva mediante query semplici; ad esempio 'SELECT * FROM Persone' e 'SELECT * FROM Squadre'. I dati attualmente richiesti comprendono esclusivamente quelli necessari per l'autenticazione dell'utente e le informazioni generali per la visualizzazione delle varie pagine. Il DBMS quindi non subisce un eccessivo carico di richieste.

Nelle espansioni future, quando la mole di dati memorizzata aumenterà in modo esponenziale con frequenti accessi al database, risulta interessante valutare l'opportunità di implementare indici sulle

tabelle maggiormente soggette a richieste. Gli indici migliorano i tempi di ricerca dati in un database, di conseguenza miglioreranno l'efficienza nell'utilizzo dell'applicazione.

3 Sistema di Video-Editing

3.1 Panoramica del sistema

L'applicazione implementa un sistema di elaborazione video, consentendo una manipolazione efficiente delle registrazioni video relative ad allenamenti o competizioni.

Lo sviluppo del sistema di video editing è stata parte integrante del tirocinio su cui si fonda la presente tesi. L'attività di editing video implica la generazione di nuovi file, come ad esempio clip video e screenshot. Pertanto, l'organizzazione dei nuovi file è una parte interessata della gestione dati del sistema, e di conseguenza, il sistema di video editing è stato sviluppato simultaneamente con il processo di organizzazione di tali nuovi file.

La realizzazione del sistema di elaborazione video è stata effettuata mediante l'utilizzo del linguaggio di programmazione PHP, sfruttando le API messe a disposizione dal software FFMpeg.

Il sistema di video editing nello specifico supporta:

- estrazione di uno screenshot del video;
- estrazione di una clip del video;
- posizionamento di un segnaposto nel video.

La schermata principale presenta il web player, con una serie di pulsanti con le varie funzioni. Sul lato destro troviamo tutti i video che appartengono alla sezione di registrazione selezionata. Sotto al web player abbiamo la lista dei vari segnaposti relativi al video in riproduzione.
Il sistema di video editing, essendo una parte del sito web, è stato sviluppato in linguaggio PHP e JavaScript. Il software utilizzato per manipoliate i video è FFmpeg [1]; nello specifico è stata utilizzata una libreria PHP reperibile nella repository GitHub [2]. Il back-end del servizio di video editing è stato sviluppato dal laureando Rizzo Alessandro, tuttavia l'integrazione con il gestionale, allineando il codice HTML ed i CSS con il gestionale, è avvenuta per mano dello studente Dario Torotorici.

3.1.1 Screenshot

Durante la riproduzione del video, è possibile estrarre il frame relativo al minutaggio corrente premendo il pulsante 'Screen'. Sotto alla lista dei segnaposto, apparirà la lista di tutti gli screenshot. Per ogni screenshot è possibile modificarne il nome e la descrizione, entrando nella sezione dei dettagli dello screen; oltre a questo è chiaramente possibile eliminare l'immagine.

3.1.2 Segnaposto

La gestione dei segnaposti è molti simile a quella degli screen: se si preme il pulsante 'Aggiungi segnaposto' se ne potrà aggiungere uno nuovo, si aprirà quindi una schermata che chiederà di inserire i dettagli: nome e descrizione. Nel caso in cui si provi ad inserire un segnaposto, dove ce ne è già presente un altro chiaramente il sito darà un messaggio di errore. Ogni segnaposto poi, può venire modificato (nome e descrizione) oltre che essere eliminato.

3.1.3 Clip

Per l'estrazione di una clip c'è un'apposita schermata (figura 3.2), dove vanno specificati minutaggio di inizio e fine della clip; ci sono due appositi pulsanti per 'prendere' il minutaggio direttamente dal minutaggio corrente della barra di progressione del web player, senza inserire i valori 'a mano'. Una volta estratta, la clip, apparirà nell'elenco dei video della sessione di registrazione.

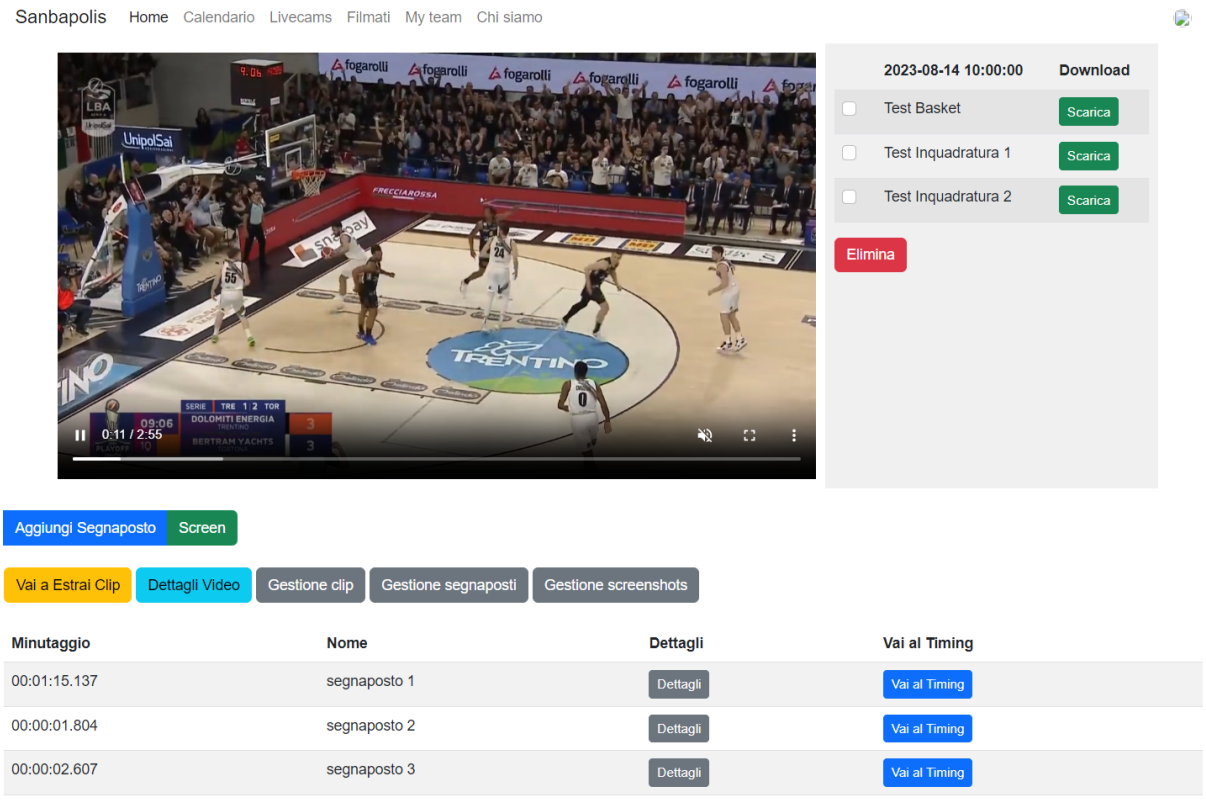

Figura 3.1: Schermata della pagina di video editing

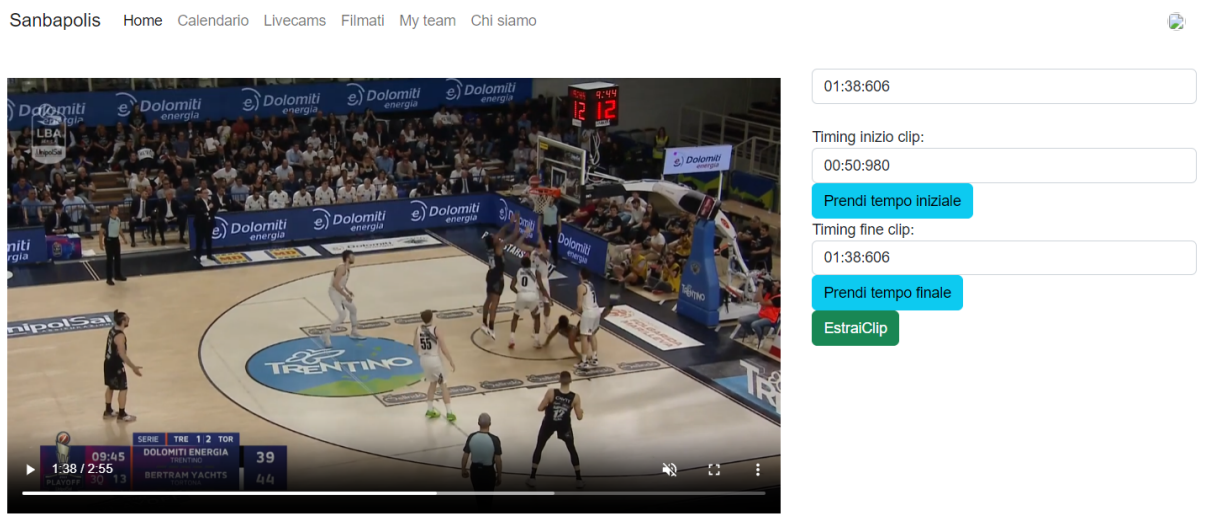

Figura 3.2: Pagina di estrazione delle clip

3.2 Storage dei video

Il gestionale ad oggi è ancora in fase di sviluppo, le telecamere del SanbàPolis non sono ancora fruibili dal sito di conseguenza non è possibile produrre delle registrazioni. Per testare il sistema di video editing sono stati utilizzati dei video di pochi minuti, con quindi un peso limitato. I video e gli screenshot per il momento si trovano nel server in cui il gestionale è in esecuzione, tuttavia lo spazio di memoria della macchina virtuale in cui è installato il server non è sufficiente a contenere probabilmente

nemmeno una sessione di registrazione; si dovrà quindi optare per una soluzione di storage esterna.

4 Gestione dei dati generati dai sensori IoT

La palestra SanbàPolis ha installato un sistema di rilevazione di dati tramite sensori IoT per il monitoraggio delle prestazioni degli atleti. I dati che vengono raccolti attualmente sono le coordinate della posizione, con una frequenza pari a frazioni di secondo. Non si esclude la raccolta di altri tipi di dati in futuro. I dati generati da questi sensori verranno immagazzinati in un database InfluxDb [4]. InfluxDb è un database per salvare serie di dati temporali, come quelle trattate in questo caso.

4.1 InfluxDb

InfluxDB è una base di dati pensata per la raccolta di serie temporali.

4.1.1 Organizzazione dei dati

- **Bucket**: Uno spazio, con un nome, in cui vengono salvate serie di dati temporali. Le varie entry di ogni bucket sono definiti 'Points';
- **Measurement**: Una misurazione, quindi un gruppo organizzato di dati;
- **Tags**: Una coppia chiave-valore che solitamente usata per salvare metadati della misurazione;
- **Fields**: I campi sono i valori che vengono effettivamente rilevati dalla misurazione;
- **Timestamp**: Poiché stiamo lavorando con serie temporali, ad ogni Point di un bucket è associato un timestamp; la precisione può essere di *s, ms, ns* (*ns* è il valore di default).

Il database, per la raccolta dei dati relativi alle posizioni dei giocatori, è organizzata nel seguente modo:

- per ogni squadra viene creato un apposito bucket;
- viene scelto un nome per ogni misurazione;
- vengono specificati i seguenti tag:
  - numero del tag RFID;
  - numero di sessione di registrazione;
- i campi specificati sono le coordinate della posizione (x, y, z);
- il timestamp in cui la posizione è stata rilevata.

Il bucket per ogni squadra viene creato manualmente da un amministratore collegandosi alla InfluxDb UI all'indirizzo istar.disi.unitn.it:8086. L'accesso ai dati InfluxDB di una squadra è permesso al relativo allenatore, per questo è stata creata l'entità e l'associazione nel database del gestionale.

L'entità 'Bucket InfluxDb' del diagramma ER prevedeva i seguenti attributi: locazione, nome, token. La locazione non è altro che l'indirizzo alla base di dati, abbiamo poi il nome del bucket, infine il token di accesso. Il token di accesso è una stringa che viene generata per gestire i livelli di accesso ad ogni bucket di InfluxDb. Ogni bucket può avere uno o più token, allo specifico token, vengono assegnati determinati permessi di lettura o scrittura. L'amministratore, quando andrà a creare un nuovo bucket, che assocerà ad una determinata squadra, andrà a creare anche il relativo token con permessi di lettura e scrittura. Ogni squadra avrà quindi associato il proprio bucket con il relativo token di accesso.

4.2 Servizio web

Il servizio web, di tipo REST, è stato sviluppato anch'esso in PHP ed integrato con il gestionale. Lo scopo del servizio web è quello di permettere ad un futuro software di analisi dati di accedere e manipolare i dati raccolti, salvati in InfluxDb. Il servizio quindi supporta le quattro operazioni CRUD (create, read, update, e delete). Il servizio web è raggiungibile all'indirizzo http://istar.disi.unitn.it/tagrfid/; dove troviamo una pagina HTML con le istruzioni per l'utilizzo.

Le quattro interfacce che rispettano le operazioni CRUD, le troviamo all'indirizzo:
http://istar.disi.unitn.it/tagrfid/api/

Metodo	**End Point**	**Servizio**
POST	create.php	Inserimento
GET	read.php	Lettura
POST	query.php	Modifiche e Lettura
DELETE	delete.php	Eliminazione

4.2.1 Autenticazione

Per utilizzare una qualsiasi delle API è necessario autenticarsi. L'autenticazione avviene tramite il metodo 'Basic access authentication': nella richiesta HTTP al web service, vanno quindi specificate le credenziali email e password. L'utente autorizzato ad accedere è l'allenatore della squadra. Il sistema, in caso di credenziali corrette, verificherà che l'utente autenticato sia un allenatore, attualmente impegnato ad allenare una squadra. In caso di errore verrà inviato un messaggio in formato JSON, altrimenti si procederà con l'operazione.

4.2.2 File CSV dei dati generati

I dati prodotti dai sensori verranno inizialmente salvati in file CSV nel server locale della palestra, successivamente verranno inviati al server del gestionale. Questo perché la rete della struttura non è attualmente sufficientemente performante; per evitare quindi congestione della banda, si attueranno delle politiche in cui i dati verranno inviati solo durante specifici slot temporali (verosimilmente di notte, quando c'è meno traffico). Allo stato attuale il sistema di rilevazione dati non è funzionante quindi in questa versione iniziale, il servizio web è stato implementato lavorando con dei CSV di esempio, nell'ipotesi che siano già presenti nel server del gestionale.

4.2.3 Create

L'inserimento di nuovi dati verrà effettuato con una richiesta POST a
istar.disi.unitn.it/tagrfid/api/create.php. Nella richiesta HTTP andranno specificati:

- Credenziali d'accesso via Basic access authentication
  - email;
  - password;
- Come multipart/form-data:
  - numero di sessione di registrazione;
  - numero di misurazione;
  - nome del file CSV;
- Come parametri della richiesta:
  - numero di sessione;
  - precisione del timing

Nel caso in cui, uno dei campi precedenti sia mancante dalla richiesta, il sistema risponderà con un messaggio di errore altrimenti con un messaggio di successo, tutto in formato JSON.

Il nome del file CSV da specificare è il nome del file su cui sono stati salvati i dati raccolti dai sensori. Il file in questa fase di sviluppo, si trova nel filesystem del web server su cui il servizio è in esecuzione; in futuro quando il sistema di raccolta dati sarà funzionante, la cosa andrà sicuramente cambiata.

4.2.4 Read

InfluxDb permette di effettuare delle richieste con due linguaggi: il classico linguaggio SQL ed il proprio linguaggio Flux[3]. L'interfaccia read.php, permette di effettuare richieste in SQL; per le chieste in Flux, riandiamo alla sottosezione Update 4.2.5.

La lettura dei dati si effettua inviando una richiesta GET all'indirizzo
istar.disi.unitn.it/tagrfid/api/read.php.
Oltre alle credenziali, vanno specificate nella richiesta HTTP, come parametri:

- numero di sessione di registrazione;
- la query SQL.
- *e.g.* .../api/read.php?session=1&query=SELECT%20*%20FROM%20home

La risposta viene fornita in formato JSON; la risposta comprende: un valore booleano che indica o meno il successo della richiesta e il risultato anch'esso restituito come stringa, formattato in JSON.

```
{"success": true,
 "results":
        {\"results\":[{\"statement_id\":0,\"series\":
        [{\"name\":\"home\",\"columns\":
        [\"time\",\"co\",\"hum\",\"room\",\"temp\"],\"values\":
        [[\"1970-01-01T00:00:01.641024Z\",0,35.9,\"Kitchen\",21]]}]}]}\n"
}
```

4.2.5 Update

Flux è un linguaggio più articolato e completo rispetto ad SQL; permette oltre che la semplice lettura dei dati anche manipolazioni complesse. Tramite l'interfaccia query.php è possibile interrogare il database con delle query in Flux. Una richiesta all'API update, si effettua all'indirizzo istar.disi.unitn.it/tagrfid/api/query.php specificando:

- credenziali di accesso via Basic access authentication;
- numero di sessione come parametro della richiesta;
- il codice in linguaggio Flux nel body della richiesta, in formato plain/text.

La risposta alla richiesta, come per la read, viene fornita in JSON con un booleano che indica il successo dell'operazione ed in caso di query che legge dei dati, ci saranno i valori dal campo 'results'. I dati 'all'interno' di 'results', sono formattati in CSV.

```
{
    "success": true,
    "results": ",result,table,_start,_stop,
    _time,_value,_field,_measurement,tag_id\r\n,
    _result,0,2020-01-01T08:00:00Z,2024-01-01T20:00:01Z,2023-08-04T00:16:47Z,
    -11.99449869227957,x_kf,coordinates,100\r\n,"
}
```

4.2.6 Delete

L'eliminazione di dati si effettua tramite la richiesta, con metodo DELETE,
all'indirizzo istar.disi.unitn.it/tagrfid/api/delete.php.

Nella richiesta HTTP, vanno specificati:

- le credenziali di accesso via Basic access authentication;
- numero di sessione come parametro della richiesta;

- nel body della richiesta, come application/json, i parametri per l'eliminazione scritti appunto in formato JSON.

Vediamo un esempio di codice JSON che specifica i parametri dell'eliminazione:

```
{
    "start": "2022-01-19T00:16:01.059Z",
    "stop": "2024-01-19T00:16:01.059Z",
    "predicate": "_measurement=\"coordinates\""
}
```

Deve essere specificato il range temporale ed un predicato che filtra i dati che si desidera eliminare.

5 Conclusioni

5.1 Stato dell'applicazione

La prima fase di sviluppo del gestionale termina con la fine dei rispettivi tirocini degli studenti Mattia Torregiani, Dario Tortorici ed il mio. L'applicazione ha implementato le funzionalità descritte nell'introduzione; allo stato attuale è quindi possibile:

- prenotare la palestra per un determinato evento, consultando l'apposito calendario online;
- consultare la propria area personale, per visualizzare i propri video sfruttando se necessario il sistema di video editing;
- salvare in modo permanente i dati di posizione dei giocatori di una determinata squadra durante una sessione di registrazione.

Il particolare risultato, prodotto dal tirocinio su cui si basta questa tesi, è stato sviluppare un sistema per la memorizzazione e gestione dei dati del gestionale suddividendo l'organizzazione in due basi di dati:

- la base di dati relazionale per salvare tutte le informazioni per il funzionamento del gestionale;
- la base di dati per serie temporali per il salvataggio dei dati sulla posizione degli atleti;

Inoltre, durante questo tirocinio è stato sviluppato un sistema di di video editing con delle semplici ma chiari strumenti per modificare le registrazioni degli eventi (partite o allenamenti). In particolare il sistema di video editing permette il taglio dei video, estrazione di un frame salvandolo come immagine ed infine è possibile sfruttare un sistema di segnaposti per salvare uno specifico minutaggio all'interno della registrazione.

Il sito web ad oggi non è ancora consultabile al di fuori della rete universitaria per ragioni di sicurezza, poiché il tutto è ancora in via di sviluppo. Altri tirocinanti proseguiranno poi nella realizzazione del sistema software.

5.2 Considerazioni personali

Il tirocinio svolto su questo progetto, mi ha consentito di mettere in pratica concetti studiati durante i corsi universitari e soprattutto ha permesso di sfruttare contemporaneamente nozioni affrontate in argomenti differenti. Oltre alle competenze tecniche, è stata un'esperienza che mi ha permesso di lavorare in gruppo e quindi ho imparato a lavorare ed interfacciarmi con altre persone. Oltre ad applicare concetti conosciuti, ho dovuto interfacciarmi per la prima volta con un database non relazionale e di conseguenza nel mio bagaglio professionale aggiungo il prodotto InfluxDB.

5.3 Sviluppi futuri

5.3.1 Sviluppi nel breve termine

Il lavoro svolto fino a questo momento consisteva in una prima versione con funzionalità di base. Sicuramente il gestionale Sanbàpolis necessiterà di altro lavoro che verosimilmente porteranno avanti altri laureandi. Il database del gestionale, in base alle necessità, potrà sicuramente essere ampliato. L'intero sistema di gestione dei dati di posizione degli atleti dovrà essere ultimato una volta che il sistema IoT sarà funzionante. Ricordiamo che ad oggi il servizio web per la gestione di questi dati è stato sviluppato partendo da un file CSV di esempio contenente una serie di dati; l'intero servizio web dovrà essere adattato al formato del file finale, che verosimilmente doverebbe essere appunto in CSV. C'è inoltre da considerare che ad oggi l'invio dei sulla posizione generati dai sensori IoT potranno venire inviati solamente dopo che la sessione di registrazione sarà terminata; questo perché la banda della connessione Internet del Sanbàpolis non è ottimale. In futuro, se si dovesse migliorare la portata della connessione, si potrebbe implementare un sistema che invii in tempo reale i dati da salvare direttamente ad InfluxDb.

5.3.2 Sviluppi nel medio-lungo periodo

Gli aggiornamenti appena menzionati sono interventi che possono venire svolti nel breve periodo, per la realizzazione delle funzionalità di base del gestionale. Lavorando per gradi tuttavia il gestionale Sanbàpolis ha una grande potenzialità di espansione. Fino ad ora l'intera sezione del sito web dedicata a tifosi ed appassionati è stata solamente menzionata ma è un intero settore che può essere sviluppato. Il sistema prevede di raccogliere video e dati su partite o allenamenti ma per ora il sistema non presenta un sistema di analisi dati; il lavoro del gruppo supervisionato dal prof. Nicola Conci verrà, in futuro, integrato con i dati raccolti dal sistema. Gli utilizzatori dell'applicazione web quindi potranno, in una versione futura, consultare il materiale raccolto dalle sessioni di registrazione, con degli strumenti di data analisi che aiutino meglio a studiare le prestazioni delle squadre.

5.3.3 Estensioni future

L'intero progetto dello sviluppo del gestionale della palestra Sanbàpolis, nasce come prototipo per il sistema di gestione delle piste da sci in Trentino. In un futuro prossimo quindi, il frutto del lavoro per il palazzetto Sanbàpolis potrà venire applicato alle infrastrutture degli impianti sciistici. Il gestionale inoltre è in via di sviluppo per il palazzetto Sanbàpolis di Trento, tuttavia lo stesso sistema potrebbe venire adattato ad altre palestre con attrezzatura simile per il monitoraggio delle prestazioni di atleti.

Bibliografia

[1] Ffmpeg. https://ffmpeg.org/.

[2] Ffmpeg php. https://github.com/PHP-FFMpeg/PHP-FFMpeg.

[3] Flux. https://docs.influxdata.com/flux/.

[4] Influxdb. https://www.influxdata.com/.

[5] Sanbàpolis github. https://github.com/SportTech-UniTN/Sanbapolis.

[6] Mattia Torregiani. *SVILUPPO DI UN SISTEMA DI DATA ANALYSIS PER UN IMPIANTO SPORTIVO*. Università di Trento, 2023.

[7] Dario Tortorici. *PIATTAFORMA WEB PER PROGETTO SANBAPOLIS*. Università di Trento, 2023.