APPLICAZIONI WEB: LINGUAGGI E ARCHITETTURE

a.a. 2017/2018

Progetto di laboratorio

Modalità d'esame

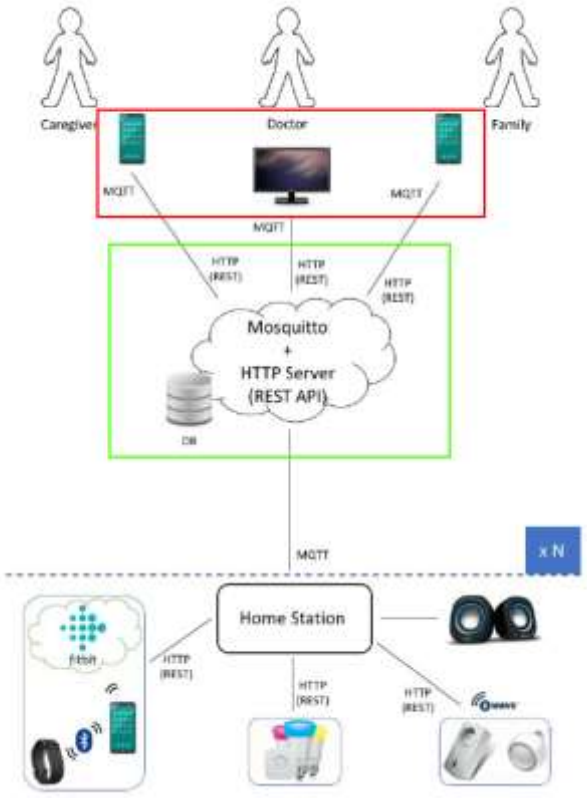
L'esame consta di due parti:

- un progetto di laboratorio, la cui realizzazione e consegna sono requisito indispensabile per poter accedere all'orale. Il progetto verrà valutato con un voto espresso in trentesimi.
- un'orale finalizzato a valutare la conoscenza acquisita relativamente agli argomenti oggetto del corso. L'orale dovrà essere sostenuto singolarmente da ogni studente.

Il voto finale è dato dalla media dei voti del progetto e dell'orale.

Modalità di implementazione del progetto

L'architettura di riferimento per il progetto di laboratorio è quella di *Healthwhere*, oggetto dello *"Smart City Challenge 2017/2018".* Il progetto di Applicazioni Web riguarda unicamente la parte evidenziata in rosso:



Per implementare il progetto di laboratorio lo studente può scegliere tra due diverse modalità:

- a specifiche *chiuse*, ovvero l'oggetto del presente documento, che consente di raggiungere un punteggio massimo di 27/30
- a specifiche *aperte*; in questo caso è il gruppo di lavoro che deve presentare un progetto e può optare per due percorsi
    - implementare l'intera architettura (fare riferimento al documento allegato); il progetto deve essere presentato e approvato dal docente di Applicazioni Web per quanto riguarda la parte evidenziata in rosso, dai docenti di Reti 2 per quanto riguarda invece il resto dell'architettura. Lo strato di API REST (parte in verde) può essere implementato in C#
    - implementare la parte in rosso presentando la proposta di progetto al solo docente di Applicazioni Web che la deve approvare.

Il punteggio massimo menzionato è relativo alla sola componente "progetto di laboratorio" che farà quindi media con il voto dell'orale.

Gruppo di lavoro
Il progetto può essere realizzato singolarmente o in gruppi di al massimo quattro studenti.

Specifiche chiuse
In questa modalità si assume che l'applicazione web dialoghi direttamente con il DBMS, per semplicità relazionale, senza la mediazione delle API REST (parte in verde). Lo studente può scegliere uno dei seguenti DBMS: SQL Server, PostgreSQL, MySQL, SQLite. L'accesso ai dati deve avvenire tramite linq2db.

Deve essere implementato lo schema della base dati in modo che si possa simulare la raccolta dei dati provenienti da più device (bracciali) Fitbit. Ogni device è identificato da un ID numerico univoco. Per ciascun device, i dati raccolti devono riguardare:

- battito cardiaco campionato ad ogni ora del giorno
- per ogni percorso fatto
    - giorno in cui si è svolto
    - a che ora è iniziato e terminato
    - tipologia del percorso (camminata/corsa)
    - km percorsi
    - numero di passi
    - dislivello in salita e discesa (espresso in piani[1])
- peso registrato dall'utente ad una certa data.

[1] Un piano equivale a tre metri di dislivello.

Un account Fitbit identifica una persona ed è associato ad un solo device per volta ma nel corso del tempo può essere stato associato a device differenti (in caso di sostituzione, guasto, ...).

L'applicazione web è strutturata in due aree accessibili previo login:

- area Admin, riservata agli utenti amministratori del servizio
- area Doctor, riservata al personale medico che può quindi monitorare i parametri dei propri pazienti (le persone monitorate attraverso i device Fitbit).

Ogni utente, indipendentemente al ruolo (Admin/Doctor), può registrarsi in autonomia indicando e-mail (utilizzata anche come nome utente) e password. Una volta loggato, in qualsiasi momento può completare/modificare il proprio profilo utente:

- e-mail
- cambio della password
- nome
- cognome.

Deve essere anche implementata la funzionalità di recupero password in caso di smarrimento.

Area Admin

Un utente con ruolo Admin può:

- vedere la lista degli utenti registrati (non la password) e attribuire loro un ruolo (Admin/Doctor)
- vedere la lista dei device registrati e gli estremi (e-mail) dell'account Fitbit a cui sono attualmente collegati
- associare uno o più account Fitbit ad un utente con ruolo Doctor (i pazienti del medico).

Le liste devono essere filtrabili e ordinabili. Qualora i dati siano molti, occorre implementare la tecnica di paginazione o scrolling virtuale.

Area Doctor

Un utente con ruolo Doctor deve poter vedere l'elenco degli account Fitbit a lui associati (email, nome, cognome).

Deve essere implementata una pagina "monitoraggio" fatta nel seguente modo:

- possibilità di filtrare l'account Fitbit di interesse (tra quelli a lui associati) e un range di date
- visualizzare, in forma tabellare e grafica, per ogni giorno nel range di date, i parametri raccolti (battito, km percorsi, numero di passi, durata delle attività, dislivello, peso).

Deve essere implementata una pagina "piani di allenamento" che permette al medico di:

- indicare un account Fitbit tra quelli a lui associati
- vedere la lista di piani di allenamento predisposti nel tempo per l'account selezionato

- inserire, cancellare o modificare un piano di allenamento.

Un piano di allenamento consiste in:

- un ID numerico univoco
- l'account Fitbit a cui è associato
- data di inserimento
- durata del piano in giorni
- peso target al termine del piano
- frequenza delle attività da svolgere (es. 3 volte a settimana)
- per ogni attività: numero minimo di km e di dislivello in salita, durata minima dell'attività.

Requisiti

- L'applicazione deve essere implementata in C# utilizzando ASP.NET Core MVC.
- Il codice deve essere ben scritto (NON procedurale!) e corredato di opportuni commenti dove necessario.
- Il docente deve essere messo in condizione di poter far girare l'applicazione in autonomia. Un'applicazione che non gira verrà penalizzata in termini di punteggio.
- Per quanto riguarda l'interfaccia grafica, lo studente è libero di scegliere l'eventuale libreria/framework da utilizzare.

Istruzioni per la consegna

- La consegna consiste in:
  - Codice sorgente dell'applicazione.
  - Se non si adotta l'approccio code-first, script SQL sufficiente a ricreare l'intero database ed eventuale setup di dati di base necessari per il primo avvio dell'applicazione
  - Relazione scritta contenente almeno i seguenti contenuti:
    - Istruzioni chiare per fare il setup iniziale della soluzione e permette quindi al docente di provarla
    - Illustrare scelte implementative ed eventuali note che lo studente ritiene significative
    - Descrizione del modello dei dati (non è obbligatorio un diagramma E-R)
    - Descrizione del modello ad oggetti.
- Una volta consegnato al docente, non è più possibile fare correzioni al codice. E' tuttavia possibile inviare eventuali note integrative purchè:
  - siano scritte (e-mail o documento integrativo)
  - siano un'eccezione.
- Il codice deve possibilmente essere esente da bug almeno per un utilizzo "standard" delle funzionalità richieste.