Progetto d’esame: laboratorio di PISSIR

Anno accademico 2023/2024

DESCRIZIONE PROGETTO

Sistema di gestione di parcheggio con ricaricatori wireless mobili per auto elettriche: requisiti funzionali

Immaginiamo che esistano ricaricatori wireless mobili (MWbot)[^1] capaci autonomamente di spostarsi sotto le auto elettriche e ricaricarle per induzione. Ciò apre ad una gestione agile della ricarica in cui gli utenti possono lasciare l'auto parcheggiata e richiedere che venga caricata mentre fanno le loro commissioni. Quando la batteria dell'auto raggiunge la percentuale di carica richiesta dall'utente, l'MWbot può spostarsi per caricare altre auto. Si suppone che l'MWbot possa riconoscere il modello d'auto da ricaricare e in particolare la capacità in KW della batteria. Inoltre ogni posto auto è dotato di sensori per monitorarne l'occupazione.

Obiettivo del progetto è la progettazione e l'implementazione di un sistema di gestione di un parcheggio smart dotato di MWbot, per semplicità si può assumere la presenza di un singolo MWbot all'interno del parcheggio.

Gli utenti del sistema sono di tre tipi: utenti base che usufruiscono dei servizi di parcheggio e ricarica, utenti premium che possono in aggiunta prenotare preventivamente il posto, e l'amministratore che da remoto può monitorare i vari componenti del sistema e lo stato di occupazione del parcheggio.

Gli automobilisti tramite un'applicazione multi-piattaforma (smartphone o quadro strumenti digitale dell'auto) possono effettuare le seguenti operazioni:

- solo gli utenti premium, possono consultare la disponibilità di posti auto nel parcheggio ed eventualmente, se disponibile, prenotarne preventivamente uno fornendo le proprie credenziali e i dati della carta di credito, e indicando un determinato tempo di arrivo e durata permanenza previsti. A seguito di queste operazioni riceveranno l'identificativo del posto prenotato. Nel caso la prenotazione non venga cancellata e l'utente non arrivi entro il tempo di arrivo più un certo tempo di tolleranza (es. 15-30 minuti), la prenotazione viene automaticamente cancellata ed è addebitata una determinata somma per la mancata utilizzazione del posto. Per semplicità supponiamo che tutti gli utenti rispettino la durata dichiarata (con un margine simile a quello del tempo di arrivo) in modo da poter accettare più prenotazioni non sovrapposte temporalmente per uno stesso posto.
- Gli utenti base o premium, possono richiedere che una volta arrivati la loro auto venga ricaricata fino ad una desiderata percentuale di carica; in tal caso verranno informati del numero di auto da ricaricare prima della loro, o di una stima del tempo di attesa prima del completamento della ricarica e a quel punto potranno accettare o rifiutare la prestazione
- un utente che abbia richiesto ed accettato il servizio di ricarica, può richiedere di ricevere una notifica (nella realtà questa potrebbe essere inviata tramite messaggio whatsapp, SMS, mail, nel vostro sistema semplicemente inviando un messaggio tramite MQTT) quando la batteria raggiunge il livello di carica desiderato.

[^1]: prototipi simili esistono già: https://insideevs.it/news/592196/ziggy-colonnina-mobile-robot-parcheggi/ https://www.newsauto.it/guide/ricarica-wireless-auto-elettriche-2-2022-382355/

All'uscita dal parcheggio, all'automobilista è addebitato il costo per il tempo di sosta e il costo per l'eventuale tempo totale di ricarica.

L'utente amministratore può:

- monitorare lo stato di occupazione di ogni posto auto
- aggiornare il costo €/ora della sosta e €/Kw della ricarica
- visualizzare i pagamenti effettuati in un determinato intervallo di giorni/ore, eventualmente distinti tra quelli per la sosta e quelli per la ricarica, e tra le diverse tipologie di utenti base e premium

Il sistema deve:

- autenticare gli automobilisti tramite login e password o con Oauth2 ed eventualmente, per chi l'ha richiesto, indicare il posto prenotato
- tener traccia del numero di auto all'interno del parcheggio ed aggiornare l'occupazione corrente su un monitor all'ingresso
- gestire le prenotazioni degli automobilisti, la ricarica, le eventuali notifiche ed il pagamento dei servizi erogati
- gestire la coda di auto in attesa di ricarica: non è necessario gestire il movimento dell'MWbot all'interno del parcheggio, ma bisogna indicare all'MWbot quale sarà il successivo utente/posto/auto (a seconda di come si decide di identificarli) da ricaricare

Struttura del sistema (progettazione)

Il sistema dovrà essere composto da

- un “backend” che espone un'interfaccia di tipo REST e permette di inserire in un database i dati rilevanti sulle prenotazioni, soste, ricariche e pagamenti, ed inoltre permette di modificare o consultare tali dati. Il backend ha accesso esclusivo al DB (chiunque voglia aggiornare o leggere i dati deve chiederlo al backend).
- Una interfaccia utente (un'app, una web-app o una mobile app) che permetterà agli utenti di interagire con il backend (tramite le API-REST di quest'ultimo). Questa potrà essere una semplice interfaccia testuale oppure una interfaccia a finestre (per esempio eseguibile nel browser, quindi una web-app) o una mobile app (se avete seguito il corso di applicazioni mobili).
- Un gestore del sottosistema IoT: si tratta di un componente che può entrare in comunicazione con i sensori di occupazione posto, il monitor all'ingresso del parcheggio e l'MWbot. Il gestore dei sottosistema IoT dovrà interagire con gli altri sottosistemi tramite broker MQTT (per esempio per trasmettere le notifiche sull'occupazione/liberazione del posto auto, sullo stato dell'MWbot (libero, in movimento, in erogazione), o altro.

COMPONENTI ACCESSORI (facoltativi)

Il sistema potrebbe inglobare anche alcuni componenti accessori: seguono alcuni esempi.

- Un sistema esterno di autenticazione / autorizzazione (es. keycloak) da utilizzare per permettere l'accesso ai soli automobilisti autorizzati (previa autenticazione);

- un sistema esterno di esecuzione delle transazioni per il pagamento dei servizi usufruiti dall'utente;
- supponendo l'uso di molteplici MWbot all'interno di un singolo parcheggio e che gli stessi necessitino periodicamente di essere ricaricati, un sistema di ottimizzazione che permetta di ricaricare i MWbot garantendo, per quanto possibile, la fruizione continua del servizio di ricarica auto.
- Alla ricezione della notifica di raggiungimento del livello di carica desiderata, gli utenti possono eventualmente richiedere un'ulteriore ricarica fino ad un altro livello. In tal caso la nuova richiesta verrà inserita in fondo all'eventuale coda di richieste di ricarica e nuovamente l'utente sarà informato della sua lunghezza, o del tempo complessivo d'attesa stimato per il completamento

SENSORI E ATTUATORI

i sensori e gli attuatori potranno essere reali o emulati, in base alla disponibilità di strumenti fisici. In ogni caso per poter effettuare più facilmente i test sul sottosistema IoT e i test d'integrazione occorrerà predisporre delle piccole applicazioni in grado di fornire misure fittizie (che siano rappresentative di scenari realistici, e che potrebbero essere memorizzate e quindi lette da un file: ciò sarebbe molto utile sia per lo svolgimento dei test che in occasione della demo da mostrare all'esame) ma anche di recepire i comandi per gli attuatori e visualizzarli in formato testo o grafico (una possibilità è usare l'emulatore delle Philips Hue assegnando a ciascuna lampadina emulata un significato, come rappresentante di un attuatore fisico, e in base al comando ricevuto accenderla, spegnerla o farle cambiare colore).

FASI DEL LAVORO

Specifica

in base ai requisiti elencati sopra, eventualmente dettagliati grazie a interviste che i gruppi potranno organizzare con il committente (i docenti del corso), occorrerà definire con sufficiente dettaglio il dominio e i casi d'uso. Per evitare ambiguità si dovranno usare schemi chiari preferibilmente utilizzando i diagrammi UML (diagramma delle classi del dominio e diagramma dei casi d'uso, corredati di brevi commenti testuali complementari ai diagrammi che spiegano in cosa consiste il caso d'uso; a vostra discrezione potrete usare altri tipi di diagrammi per specificare meglio l'articolazione in termini di successione di passi dei diversi casi d'uso).

Progettazione

in questa fase si dovrà scegliere il tipo di architettura (che dovrà essere basata su microservizi), si dovranno individuare i sottosistemi, le loro interfacce e le interazioni. Inoltre all'interno di ogni gruppo si dovranno definire dei sottogruppi a cui attribuire la responsabilità della progettazione di 1 o 2 sottosistemi (AI GRUPPI PIU' NUMEROSI SARA' RICHIESTO DI REALIZZARE UNA DELLE FUNZIONALITA' ACCESSORIE). Per ciascun sottosistema dovrà essere preparato un documento che definisca in modo sufficientemente preciso le classi che si dovranno implementare nella fase successiva: per evitare ambiguità la documentazione dovrà contenere schemi chiari, possibilmente utilizzando diagrammi UML. In particolare dovrete schematizzare il diagramma delle classi (questa volta sarà il diagramma delle classi che dovranno essere implementate) e dei package, e i diagrammi di sequenza per descrivere le interazioni tra i componenti (sia all'interno di uno stesso microservizio, sia tra microservizi diversi). In questa fase si dovranno definire sia gli end-point delle API REST, con gli eventuali parametri

e il formato delle informazioni scambiate, sia i topic MQTT con il formato dei messaggi inviati sui diversi topic

Implementazione

L'implementazione delle classi definite in fase di progettazione può essere essere fatta in Java, ma potete usare altri linguaggi a voi più congeniali (il supporto da parte dei docenti potrebbe non essere ugualmente proficuo se si scelgono altri linguaggi). I diversi microservizi possono essere implementati con linguaggi diversi (ciascun sottogruppo può implementare nel linguaggio preferito il proprio microservizio: l'importante è che le interfacce siano rispettate).

Test

Lo sviluppo dovrà prevedere anche il test dei componenti e dei sottosistemi in isolamento. Quando il gruppo avrà completato tutti i sottosistemi dovrà svolgere anche test di integrazione. I test dovranno coprire un numero adeguato di casistiche tipiche, e saranno anche la base per la dimostrazione di funzionamento che dovrà essere illustrata durante la parte dell'esame relativa al progetto.

Demo

durante l'esame sulla parte pratica dovrà essere preparata una demo che permetta di mostrare tutte le funzionalità implementate e che sia esempio significativo di alcuni dei test più rilevanti realizzati prima della consegna.