

ITI “A. Monaco”

Frandina Ilaria

5°A Informatica

Anno Scolastico 2020/2021

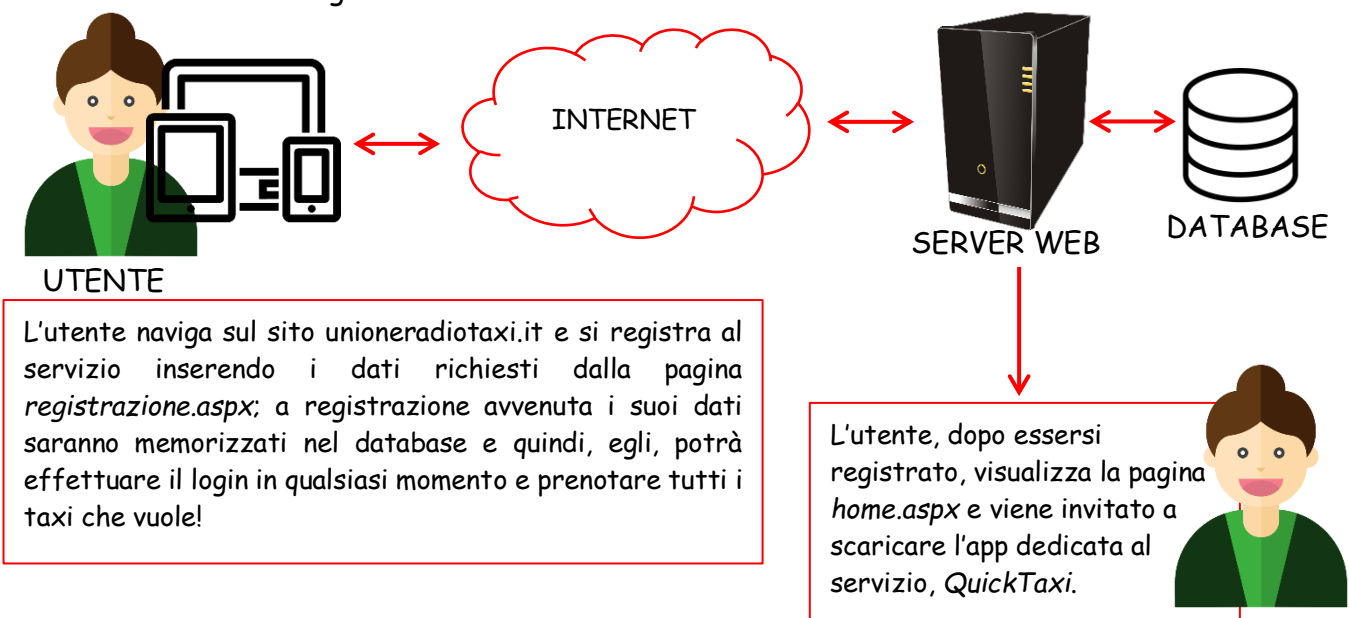
Elaborato Esame di Stato

Unione Radiotaxi

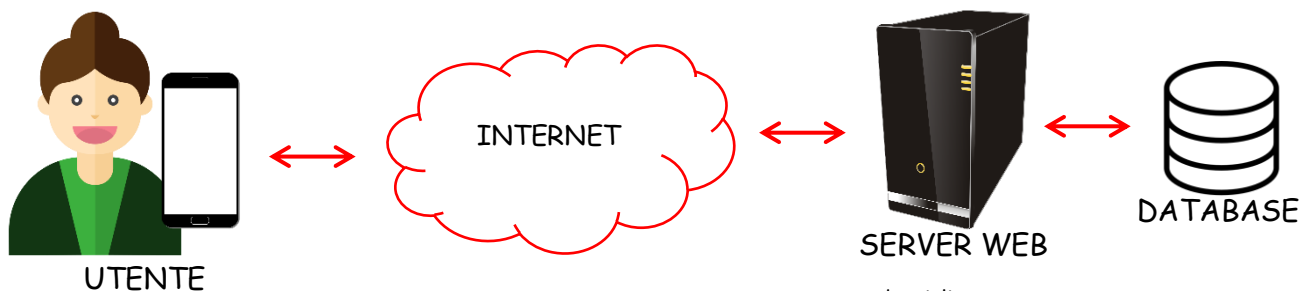
1) Analisi della realtà di interesse

Le entità coinvolte nell'infrastruttura richiesta saranno l'utente, che usufruirà del servizio di prenotazione taxi, il taxi (in particolare il taxista), che prenderà in carico le richieste che gli arriveranno e, sicuramente, il sistema informativo interno all'associazione Unione Radiotaxi, che vuole rendere disponibile il servizio e dovrà farsi carico della gestione dell'intera infrastruttura.

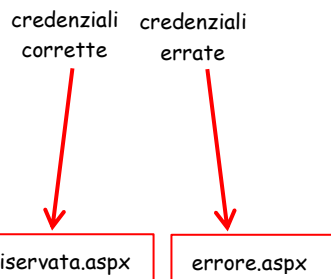
La prima interazione con il sistema verrà compiuta dall'utente che, volendo usufruire del servizio di prenotazione pubblicizzato sulla cartellonistica cittadina, navigherà sul sito web dell'associazione (unioneradiotaxi.it), dove grazie alla pagina `default.aspx` potrà visualizzare la risorsa `registrazione.aspx`, tramite un qualsiasi dispositivo, per registrarsi al servizio inserendo i propri dati, compresi gli estremi della sua carta di credito valida con cui effettuare i pagamenti. Dopo aver effettuato la registrazione, l'utente verrà reindirizzato sulla risorsa `home.aspx`; qui verrà invitato a scaricare l'app Android QuickTaxi sul proprio smartphone, alla quale accederà con le credenziali inserite nella fase di registrazione, che gli permetterà di prenotare un taxi attivando il servizio di geolocalizzazione.



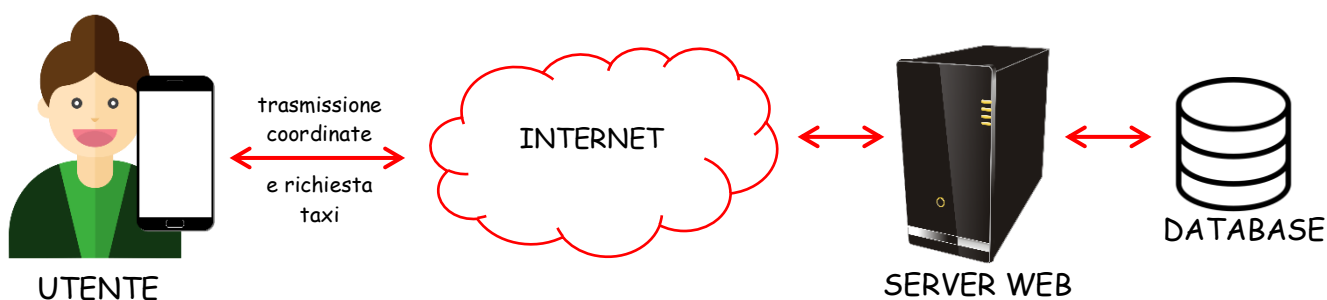
L'utente, dopo aver scaricato l'app QuickTaxi, sarà invitato ad accedere al servizio tramite le credenziali che ha inserito in fase di registrazione; sull'app sarà presente una WebView, perciò egli utilizzerà un'applicazione Android, ma sarà come se navigasse tramite un browser; l'accesso verrà effettuato tramite la risorsa `accesso.aspx`. Le credenziali inserite dall'utente saranno confrontate con quelle presenti nel Database: se verrà trovata una corrispondenza quest'ultime saranno valide e l'utente verrà reindirizzato sulla pagina `AreaRiservata.aspx`, dove potrà visionare le corse effettuate con i relativi dati, compreso l'importo che gli è stato addebitato, o prenotare un nuovo taxi. Altrimenti, se non dovesse risultare alcuna corrispondenza con le credenziali inserite, all'utente verrebbe presentata una pagina standard di errore (`errore.aspx`), in cui verrebbe invitato a accedere nuovamente. L'utente dopo il primo accesso potrà spostarsi tra le pagine che il sito Unioneradiotaxi.it mette a disposizione senza dover accedere di nuovo grazie al **meccanismo delle sessioni**.



L'utente, attraverso l'app Android QuickTaxi, naviga sulla risorsa `accesso.aspx` e inserisce le credenziali decise nella fase di registrazione. Se queste saranno presenti sul Database egli potrà visualizzare la pagina `AreaRiservata.aspx`, altrimenti visualizzerà una pagina di errore e sarà costretto ad effettuare nuovamente l'accesso.



Dopo aver effettuato correttamente l'accesso, trovatosi sulla pagina `AreaRiservata.aspx`, l'utente potrà visionare le corse effettuate, cliccando sull'apposito link che porta alla risorsa `CorseEffettuate.aspx`, o prenotare un nuovo taxi tramite la risorsa `PrenotaTaxi.aspx`. Per prenotare un nuovo taxi gli basterà attivare la geolocalizzazione e richiedere un taxi attraverso la pressione di un pulsante. Questo, dopo il calcolo delle coordinate scatenerà un PostBack che permetterà la trasmissione delle stesse al server. Sarà lo stesso server che si occuperà di trovare il taxi disponibile più vicino alla posizione dell'utente e di associare la prenotazione a quest'ultimo.

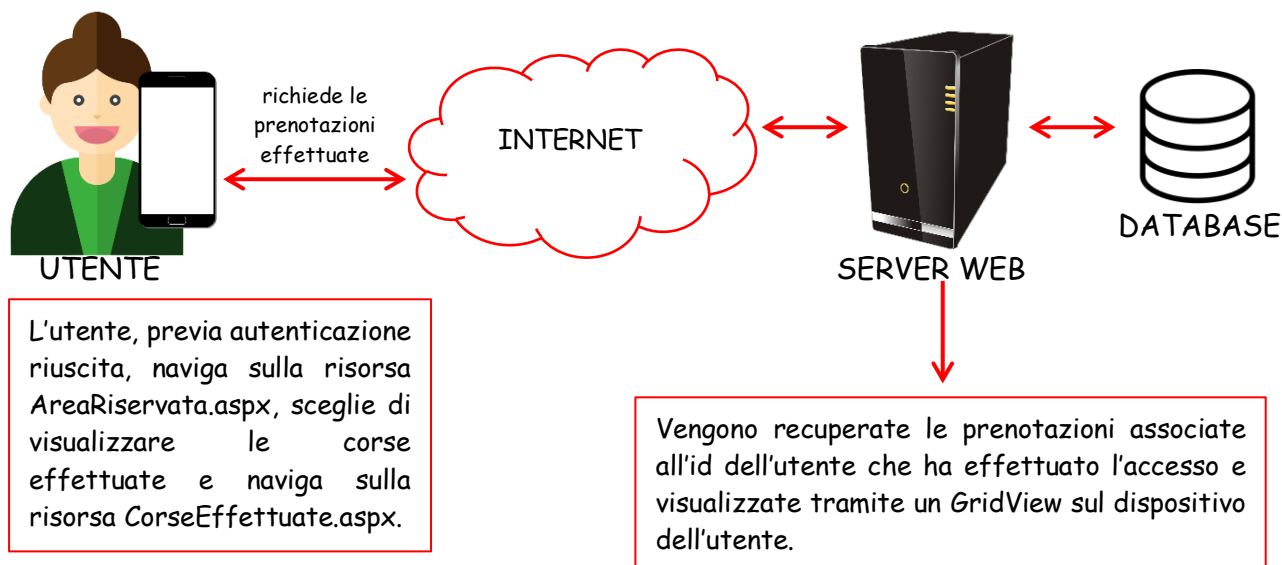


L'utente, previa autenticazione riuscita, naviga sulla risorsa `AreaRiservata.aspx`, sceglie di prenotare un nuovo taxi e visualizza la risorsa `PrenotaTaxi.aspx`. Dopo aver attivato la geolocalizzazione, tramite la pressione di un pulsante viene scatenato un PostBack che trasmette al server le coordinate correnti dell'utente.

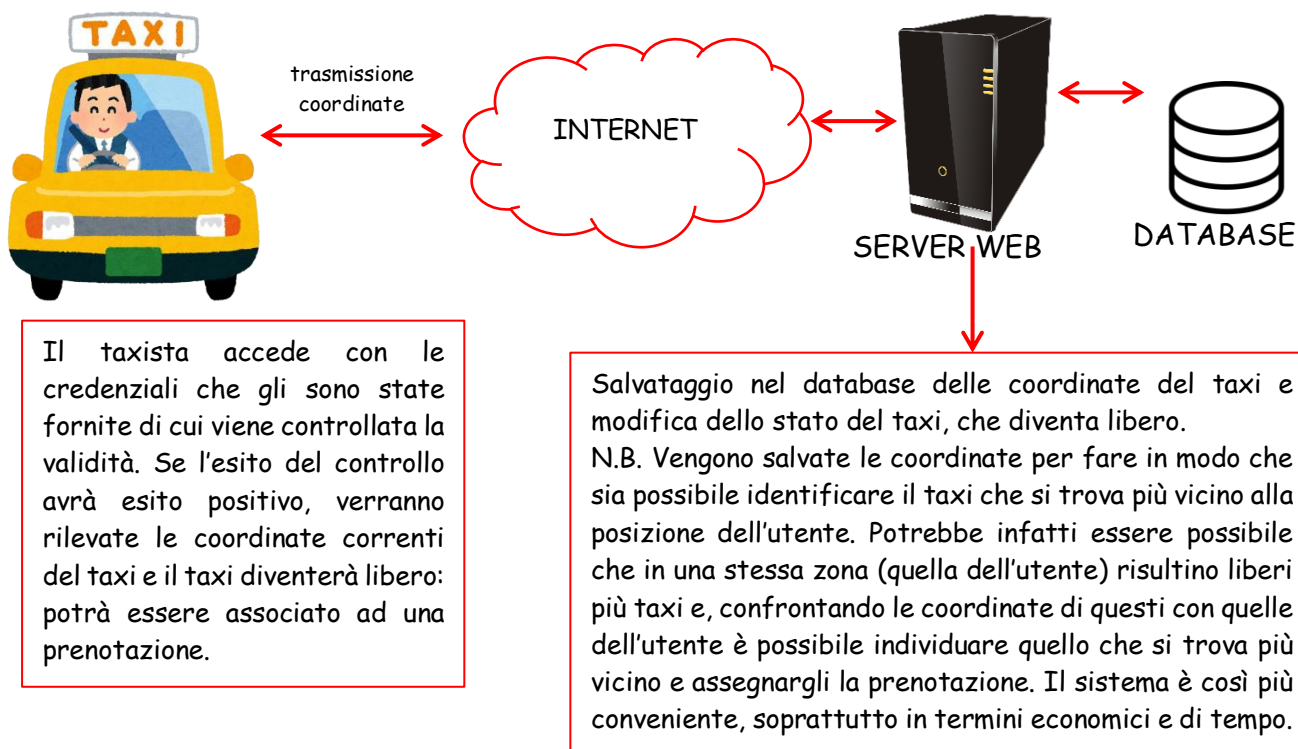
Il server controlla quali sono i taxi disponibili nella zona in cui si trova l'utente (calcolata automaticamente in base alle coordinate trasmesse) e, se ne trova qualcuno, assegna la prenotazione al taxi che, nella zona dell'utente si trova più vicino a quest'ultimo. Nel frattempo, il taxista sarà in attesa di una prenotazione sulla pagina `Prenotazioni.aspx`, in cui sarà presente un timer, che periodicamente controlla se c'è una nuova corsa da effettuare. Naturalmente, il taxista dovrà prima identificarsi tramite la risorsa `AutenticaTaxi.aspx`. Al suo accesso il taxi diventa libero e le sue coordinate vengono salvate (serve a stabilire il taxi più vicino all'utente). L'accesso è regolato grazie al meccanismo delle sessioni.



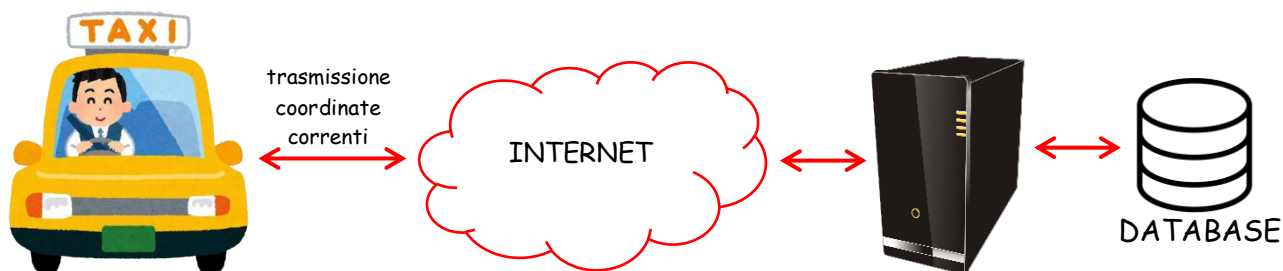
Come già detto, l'utente, dopo aver effettuato l'accesso, potrà decidere se prenotare un nuovo taxi o visionare le corse effettuate con tutte le informazioni collegate a queste, compreso il costo che gli è stato addebitato sulla carta fornita in fase di registrazione. Scegliendo di visualizzare le corse effettuate, l'utente verrà reindirizzato sulla risorsa *CorseEffettuate.aspx*. Qui verrà visualizzata una tabella contenente i dati di tutte le prenotazioni fatte.



Il taxista, invece, tramite il suo dispositivo mobile o tramite il dispositivo fornito dalla società, visitando il sito web *UnioneRadiotaxi.it*, potrà accedere al servizio tramite la risorsa *Autentica-Taxi.aspx* con le credenziali associate al suo taxi, comunicategli dall'associazione. Se le credenziali inserite risulteranno corrette, verranno rilevate le coordinate correnti del taxi e salvate nel database, naturalmente, dopo aver effettuato l'accesso il taxi risulterà ufficialmente libero e potrà essere associato ad una prenotazione.



Dopo l'autenticazione, il taxista potrà visualizzare la pagina Prenotazioni.aspx in cui attenderà una prenotazione; all'arrivo di quest'ultima egli potrà visualizzare la pagina PrenotazioneInCorso.aspx in cui sarà disponibile il percorso da compiere per arrivare alla posizione del cliente che ha prenotato il taxi (attraverso Google Maps). Appena il taxi inizierà a muoversi, l'utente visualizzerà in tempo reale, anche lui attraverso una mappa, il percorso compiuto dal taxi per poterne mantenere la tracciabilità; insieme a questa l'utente visualizzerà anche i minuti di attesa, aggiornati periodicamente, e il nome del taxi che egli ha prenotato. Queste informazioni saranno disponibili nella pagina tracciailmiotaxi.aspx, anch'essa presentata attraverso l'app QuickTaxi.

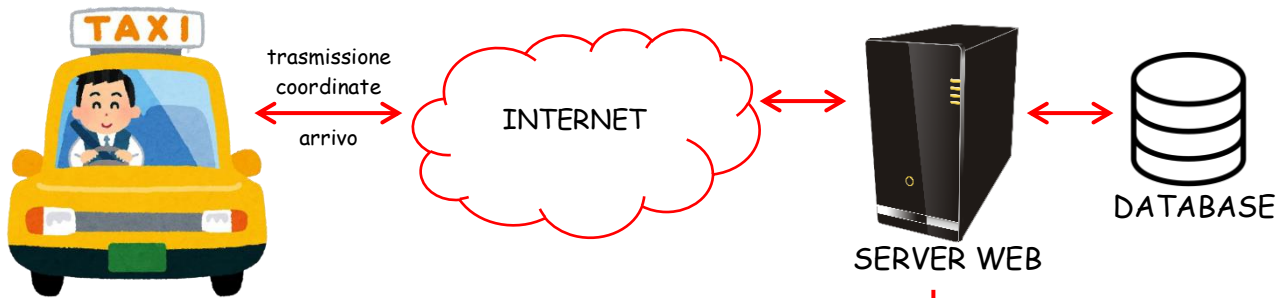


Dopo aver effettuato l'accesso il taxista visualizzerà la pagina Prenotazioni.aspx dove attenderà una prenotazione, appena gliene verrà assegnata una, potrà visualizzare la pagina PrenotazioneInCorso.aspx; qui potrà accedere ad una mappa, offerta da Google Maps, in cui sarà disponibile il percorso da compiere per arrivare nella posizione in cui si trova l'utente. Appena il taxi inizierà a muoversi, le sue coordinate verranno calcolate e trasmesse al server, che le salverà nel database, ogni trenta secondi. Questo meccanismo renderà possibile la visualizzazione dei movimenti del taxi in tempo reale per l'utente che lo aspetta, poiché sarà possibile recuperare la posizione del taxi e aggiornarne periodicamente la visualizzazione dei suoi movimenti sulla pagina TracciaIlMioTaxi.aspx, in cui sarà presente una mappa, offerta da Bings, che visualizza un marker in movimento (il taxi). Verrà utilizzato lo stesso meccanismo per i minuti di attesa, calcolati nella pagina prenotazioneincorso.aspx (tramite l'individuazione della velocità di movimento del taxi) trasmesse al server ogni 30 secondi.

Il taxista visualizza la pagina PrenotazioneInCorso.aspx e inizia a muoversi; trasmissione delle coordinate e dei minuti di attesa ogni 30 secondi.

L'utente visualizza i movimenti del taxi in tempo reale, i minuti di attesa aggiornati e il nome del taxi che gli è stato associato, attraverso l'app QuickTaxi che visualizzerà la pagina **TracciaIlMioTaxi.aspx**, in cui sarà presente una mappa offerta da Bings, aggiornata grazie al recupero delle coordinate dal database ogni 35 secondi.

Nel momento in cui il taxi raggiungerà l'utente, il taxista dovrà confermare la prenotazione (vengono salvati data e ora di partenza) attraverso un pulsante presente nella pagina PrenotazioneInCorso.aspx; a questo punto l'utente indicherà al taxista la sua destinazione. Arrivati a destinazione, il taxista non dovrà fare altro che registrare il termine della corsa tramite la pagina TerminaCorsa.aspx. In questo modo verranno calcolate le coordinate di arrivo e avverrà così un calcolo automatico dei km che permetterà l'addebito automatico dell'ammontare della corsa, calcolato grazie ad una tariffa generica per km percorso specificata dalla società Unione Radiotaxi, sulla carta che l'utente ha fornito al momento della registrazione. Se la destinazione dell'utente dovesse non rientrare nella zona associata al taxi, a quest'ultimo verrà addebitato un importo maggiore, variabile a seconda dei km percorsi fuori dalla zona associata al taxi.



Arrivati a destinazione, il taxista dovrà terminare la corsa; per farlo dovrà navigare sulla pagina `terminacorsa.aspx` e premere un pulsante che rileverà le coordinate correnti e le trasmetterà al server. Quest'ultime verranno salvate nel database e verrà automaticamente addebitato l'ammontare della corsa all'utente sulla carta che ha comunicato in fase di registrazione. L'importo verrà calcolato calcolando la distanza (in km) tra il punto di partenza e il punto di arrivo moltiplicata per la tariffa per km stabilita dall'associazione Unione Radiotaxi; se il punto di arrivo dovesse risultare fuori dalla zona associata al taxi verrà addebitato un importo maggiore all'utente.

calcolo dell'importo e
addebito sulla carta
dell'utente

L'utente potrà visualizzare le corse effettuate tramite la sua app. La risorsa che visualizzerà sarà `CorseEffettuate.aspx`, a cui può accedere dall'area riservata.

Metodo di funzionamento e gestione delle zone nello specifico

Nel mio sistema, ogni taxi opera in una zona, quindi, può accettare solo ed esclusivamente le richieste che provengono dalla sua zona. Se la destinazione dell'utente dovesse essere fuori dalla zona specifica del taxi, a quest'ultimo verrà addebitato un importo maggiore, variabile a seconda della distanza percorsa al di fuori dalla zona. Quando il taxista terminerà la corsa e il taxi risulterà libero questo potrà ricevere nuove prenotazioni, sempre appartenenti alla sua zona.

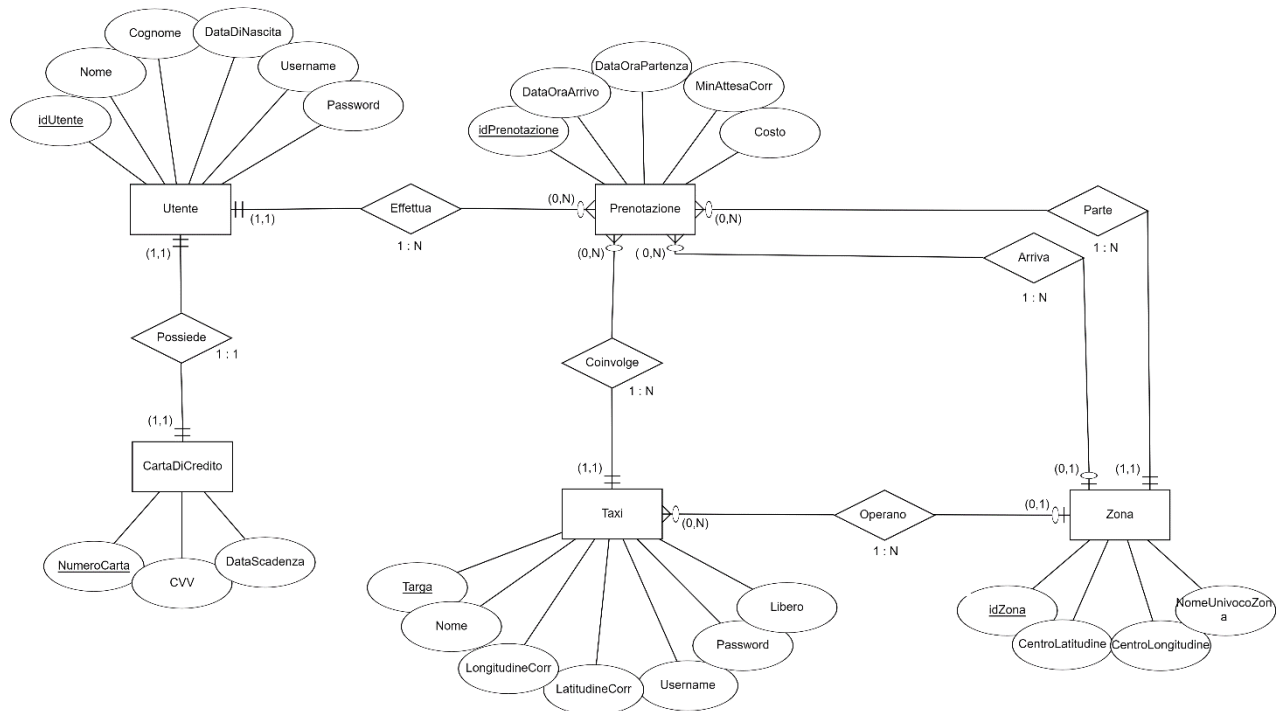
Ogni Zona è caratterizzata da un punto, formato da latitudine e longitudine; questo serve per determinare la zona di cui fanno parte le coordinate associate alle prenotazioni. Ogni centro di ogni zona dista dal centro da quella adiacente 10km, quindi ogni zona copre un cerchio di raggio 10 km.

Il calcolo dei minuti di attesa avviene nella pagina `PrenotazioneInCorso.aspx` tramite la rilevazione della velocità del taxi in m/s, la formula usata è $\text{Tempo} = \text{Spazio} / \text{Velocità}$.

Per rendere più automatico il sistema di prenotazione l'utente non dovrà far altro che premere un pulsante, non gli verrà chiesta neppure la sua destinazione in fase di prenotazione, questa dovrà essere esclusivamente comunicata al taxista che, arrivatoci, registrerà la fine della corsa e automaticamente anche la destinazione dell'utente. Grazie a questa organizzazione se l'utente nel corso del tragitto dovesse voler cambiare destinazione potrà farlo senza problemi.

Sempre col fine di rendere più automatico il servizio ho deciso di non coinvolgere un centralino che smista le prenotazioni ai vari taxi; sarà il taxista che tramite il suo telefono, o tramite un dispositivo mobile fornito dall'azienda, che potrà accedere a una parte del sito a lui dedicata con le credenziali associate al suo taxi e vedere se ci sono delle nuove corse da effettuare.

2) Modello concettuale



Regole di lettura

1) Associazione Utente - CartaDiCredito

1. Un utente deve possedere minimo 1 carta di credito e massimo 1 carta di credito; è necessario infatti che l'utente, in fase di registrazione inserisca la carta di credito, quindi il possesso della carta è una prerogativa per usufruire del servizio;
2. Una carta di credito deve essere associata minimo ad 1 utente e massimo ad 1 utente;

2) Associazione Utente - Prenotazione

1. Un utente può effettuare minimo 0 prenotazioni, ipotizzo la presenza di un utente appena iscritto che non ne ha ancora effettuate, e massimo N prenotazioni;
2. Una prenotazione deve essere effettuata minimo da 1 utente e massimo da 1 utente, non può esistere una prenotazione non associata ad un utente;

3) Associazione Prenotazione - Zona

1. Una prenotazione deve partire minimo da 1 zona e massimo da 1 zona, poiché se viene registrata una prenotazione è sicuro che quest'ultima abbia anche una zona di partenza;
2. Da una zona possono partire minimo 0 prenotazioni, poiché è possibile che la zona sia stata appena inserita, e massimo N prenotazioni;
3. Una prenotazione può terminare minimo in 0 zone, perché se la prenotazione è in corso la zona di arrivo non sarà ancora registrata, e massimo in 1 zona;
4. In una zona possono terminare minimo 0 prenotazioni, poiché è possibile che la zona sia stata appena inserita, e massimo N prenotazioni.

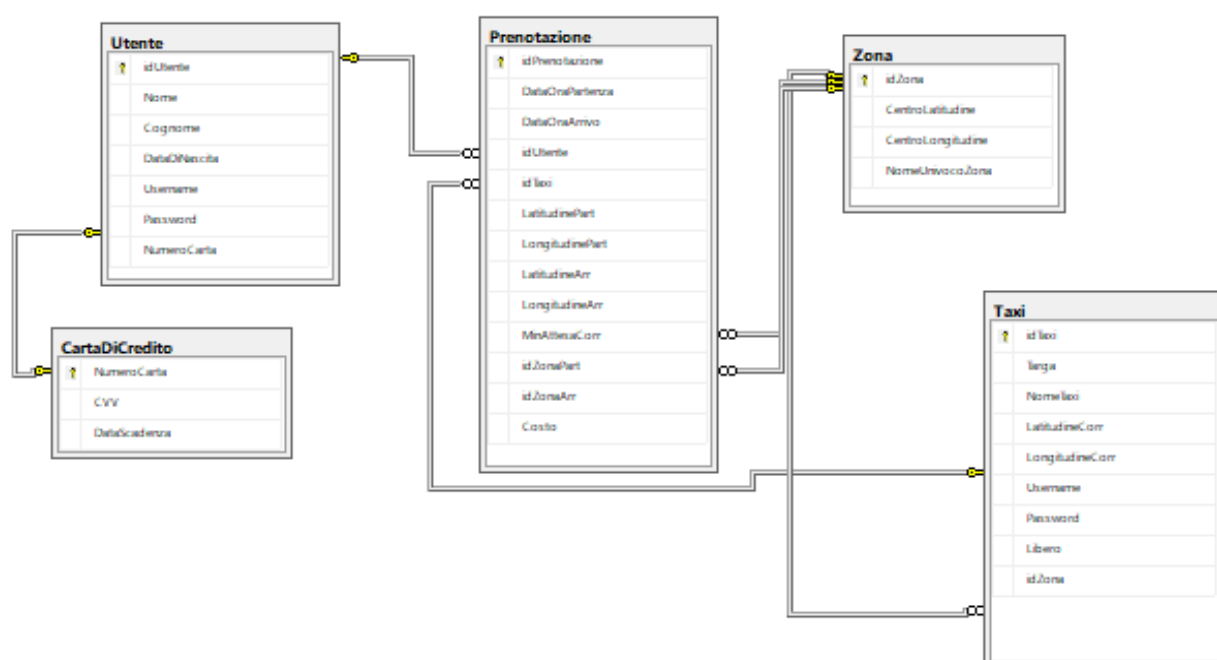
5) Associazione Zona - Taxi

1. In una zona possono operare minimo 0 taxi, ipotizzo la presenza di una zona appena inserita, e massimo N taxi;
2. Un taxi può operare minimo in 0 zone, ipotizzo la presenza di un taxi appena acquistato e non ancora associato ad una zona, e massimo in 1 zona;

6) Associazione Taxi - Prenotazione

1. Un taxi può essere coinvolto minimo in 0 prenotazioni, ipotizzo la presenza di un taxi che non ha ancora effettuato prenotazioni, e massimo in N prenotazioni;
2. Una prenotazione deve coinvolgere minimo 1 taxi e massimo 1 taxi, se una prenotazione è stata registrata deve essere stata per forza associata ad un taxista.

3) Modello logico - relazionale



UTENTE [idUtente(PK), Nome, Cognome, DataDiNascita, Username, Password, NumeroCarta(FK)]

CARTADICREDITO [NumeroCarta(PK), CVV, DataScadenza]

PRENOTAZIONE [idPrenotazione(PK), DataOraPartenza, DataOraArrivo, idUtente(FK), idTaxi(FK), LatitudineArr, LongitudineArr, LatitudinePart, LongitudinePart, MinAttesaCorr, Costo, idZonaArr(FK), idZonaPart(FK)]

ZONA [idZona(PK), CentroLatitudine, CentroLongitudine, NomeUnivocoZona]

TAXI [idTaxi(PK), Targa, NomeTaxi, LatitudineCorr, LongitudineCorr, Username, Password, Libero, idZona(FK)]

3) Modello fisico e la definizione della struttura delle tabelle

Tabella	Campo	Key	Tipo	Lung.	Note
UTENTE	idUtente	PK	smallint	16	Vincolo di chiave primaria, NOT NULL, UNIQUE
	Nome		NChar	30	NOT NULL
	Cognome		NChar	30	NOT NULL
	DataDiNascita		date	24	NOT NULL
	Username		NChar	30	NOT NULL, UNIQUE
	Password		NChar	30	NOT NULL
	NumeroCarta	FK	NChar	40	REFERENCES CartaDiCredito (NumeroCarta), ON CASCADE
CARTADICREDITO	NumeroCarta	PK	NChar	40	Vincolo di chiave primaria, NOT NULL, UNIQUE
	CVV		NChar	3	NOT NULL
	DataScadenza		NChar	5	NOT NULL
PRENOTAZIONE	idPrenotazione	PK	smallint	16	Vincolo di chiave primaria, NOT NULL, UNIQUE
	DataOraPartenza		datetime	32	NOT NULL
	DataOraArrivo		datetime	32	
	idUtente	FK	smallint	16	REFERENCES Utente (idUtente), ON CASCADE
	idTaxi	FK	smallint	16	REFERENCES Taxi (idTaxi), ON CASCADE
	LatitudinePart		float(25)	64	NOT NULL
	LongitudinePart		float(25)	64	NOT NULL
	LatitudineArr		float(25)	64	
	LongitudineArr		float(25)	64	
	MinAttesaCorr		tinyint	8	
	Costo		smallmoney	32	
	idZonaArr	FK	tinyint	8	REFERENCES Zona (idZona), SET NULL
	idZonaPart	FK	tinyint	8	REFERENCES Zona (idZona), SET NULL
ZONA	idZona	PK	tinyint	8	Vincolo di chiave primaria, NOT NULL, UNIQUE
	CentroLatitudine		float(25)	64	NOT NULL
	CentroLongitudine		float(25)	64	NOT NULL
	NomeUnivocoZona		NChar(20)	20	NOT NULL, UNIQUE
TAXI	idTaxi	PK	smallint	16	Vincolo di chiave primaria, NOT NULL, UNIQUE
	Targa		NChar	7	NOT NULL, UNIQUE
	NomeTaxi		NChar	30	NOT NULL, UNIQUE
	LatitudineCorr		float(25)	64	
	LongitudineCorr		float(25)	64	
	Username		NChar	30	NOT NULL, UNIQUE

	Password		NChar	30	NOT NULL
	Libero		bit	1	NOT NULL
	idZona	FK	tinyint	8	REFERENCES Zona (idZona), SET NULL

Definisco le tabelle con l'uso del DDL (Data Definition Language - Linguaggio di definizione dei dati), facente parte del linguaggio SQL, che permette di definire la struttura delle relazioni del database.

- 1) CREATE TABLE Utente(
idUtente SMALLINT PRIMARY KEY,
Nome NCHAR(30) NOT NULL,
Cognome NCHAR(30) NOT NULL,
DataDiNascita DATE NOT NULL,
Username NCHAR(30) NOT NULL,
Password NCHAR(30) NOT NULL,
NumeroCarta NCHAR(40) NOT NULL,
UNIQUE(Username, NumeroCarta),
FOREIGN KEY (NumeroCarta) REFERENCES CartaDiCredito (NumeroCarta) ON
DELETE CASCADE)

- 2) CREATE TABLE CartaDiCredito(
NumeroCarta NCHAR(40) PRIMARY KEY,
CVV NCHAR(3) NOT NULL,
DataScadenza NCHAR(5) NOT NULL)

- 3) CREATE TABLE Prenotazione(
idPrenotazione SMALLINT PRIMARY KEY,
DataOraPartenza DATETIME NOT NULL,
DataOraArrivo DATETIME,
idUtente SMALLINT NOT NULL,
idTaxi SMALLINT NOT NULL,
LatitudinePart FLOAT NOT NULL,
LongitudinePart FLOAT NOT NULL,
LatitudineArr FLOAT NOT NULL,
LongitudineArr FLOAT NOT NULL,
MinAttesaCorr TINYINT,
Costo SMALLMONEY,
idZonaPart TINYINT NOT NULL,
idZonaArr TINYINT,
FOREIGN KEY (idUtente) REFERENCES Utente (idUtente) ON DELETE CASCADE),
FOREIGN KEY (idTaxi) REFERENCES Taxi (idTaxi) ON DELETE CASCADE),
FOREIGN KEY (idZonaPart) REFERENCES Zona (idZona) ON DELETE SET NULL),

FOREIGN KEY (idZonaArr) REFERENCES Zona (idZona) ON DELETE SET NULL)

4) CREATE TABLE Zona(
idZona TINYINT PRIMARY KEY,
CentroLatitudine FLOAT(25) NOT NULL,
CentroLongitudine FLOAT(25) NOT NULL,
NomeUnivocoZona NCHAR(20) NOT NULL,
UNIQUE(NomeUnivocoZona))

5) CREATE TABLE Taxi(
idTaxi SMALLINT PRIMARY KEY,
Targa NCHAR(7) NOT NULL,
NomeTaxi NCHAR(30) NOT NULL,
LatitudineCorr FLOAT(25),
LongitudineCorr FLOAT(25),
Username NCHAR(30) NOT NULL,
Password NCHAR(30) NOT NULL,
Libero BIT NOT NULL,
idZona TINYINT,
UNIQUE(Targa, Username),
FOREIGN KEY (idZona) REFERENCES Zona (idZona) ON DELETE SET NULL)

3) Le query

- a) Visualizzare l'elenco dei taxi liberi che operano in una determinata zona della città

Per comodità ho inserito all'interno della tabella Taxi un apposito campo *Libero* di tipo bit che se è uguale a 1 indica che il taxi è libero, mentre se è uguale a 0 indica che il taxi è occupato in una prenotazione o fuori servizio, da cui è facile risalire ai taxi liberi in una zona.

```
SELECT Taxi.NomeTaxi
FROM Taxi INNER JOIN Zona ON Taxi.idZona=Zona.idZona
WHERE Taxi.Libero=1 AND Zona.idZona=1
```

Il risultato della query:

	NomeTaxi
1	TaxiStazione2
2	TaxiStazione3

I taxi liberi al momento sono due: TaxiStazione2 e TaxiStazione3.

- b) Visualizzare il numero dei viaggi associati a ciascuna zona di partenza per studiare in quale parte della città è richiesta una intensificazione del servizio

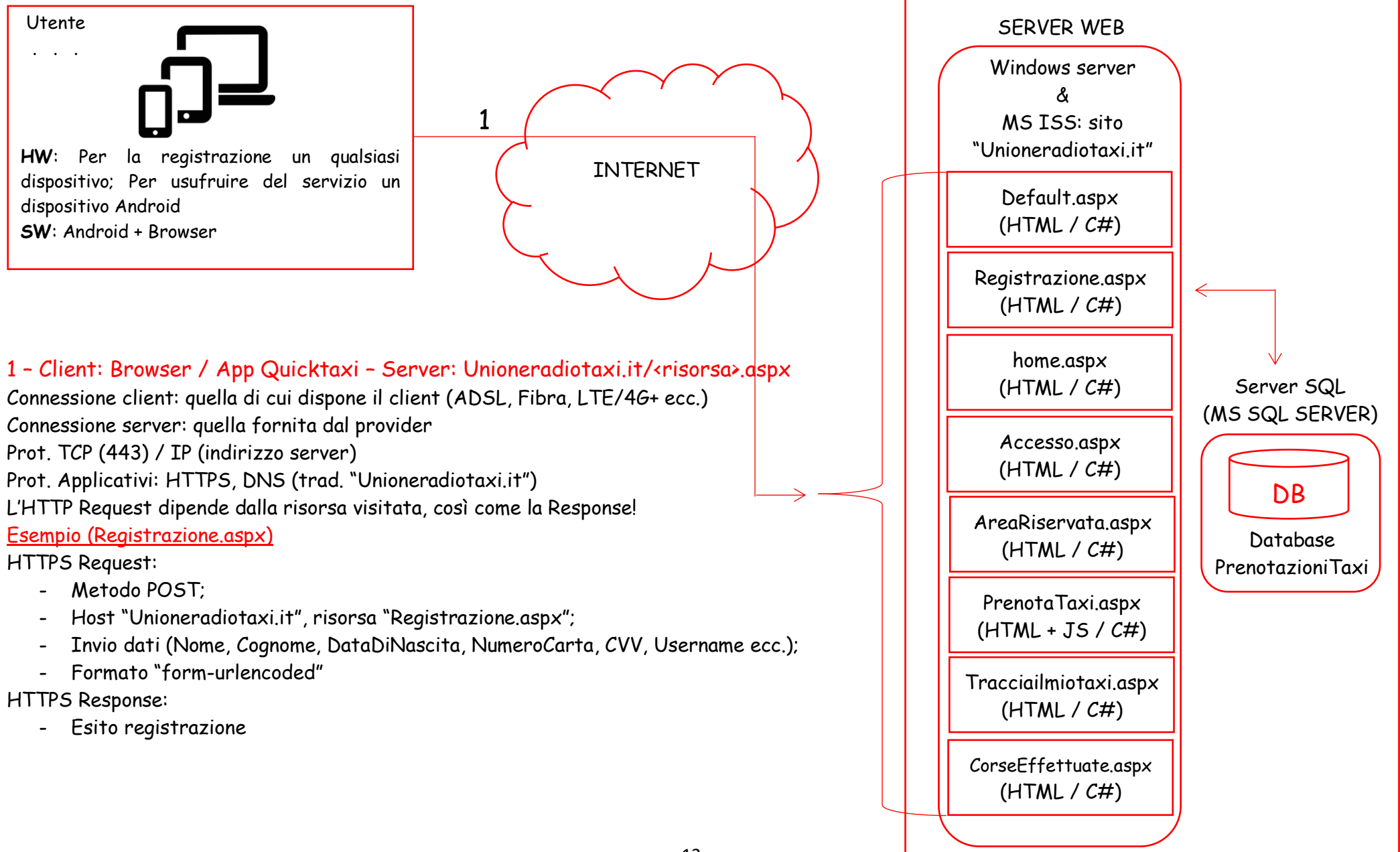
```
SELECT COUNT(idPrenotazione) AS NumViaggi, idZonaPart
FROM Prenotazione
GROUP BY idZonaPart
```

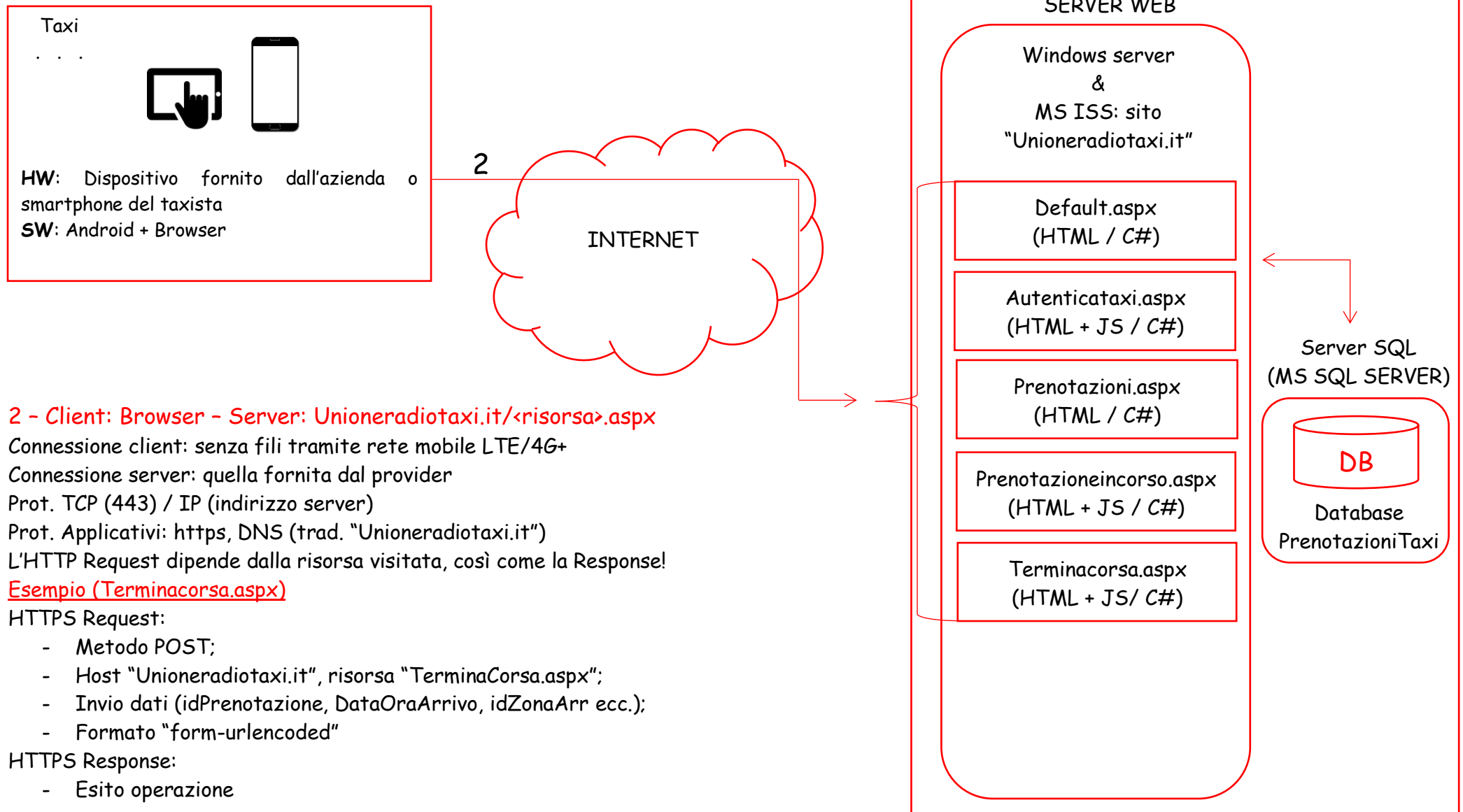
Il risultato della query:

	NumViaggi	idZonaPart
1	1	1
2	7	4

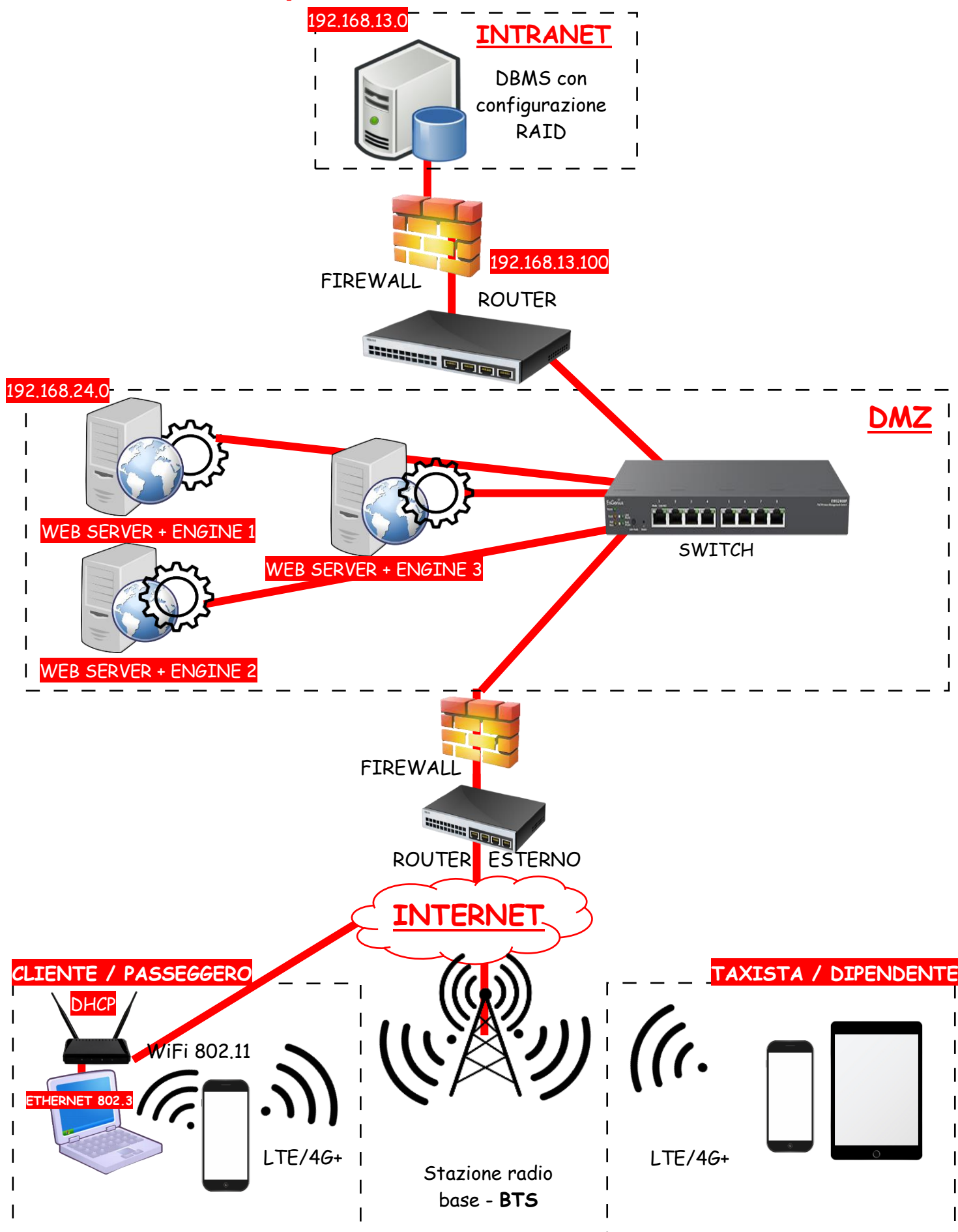
Dalla zona 1 è partita 1 prenotazione, mentre dalla zona 4 ne sono partite 7.

3) Descrizione e rappresentazione grafica dell'Infrastruttura tecnologica ed informatica di Rete





3) Schema di rete specifico



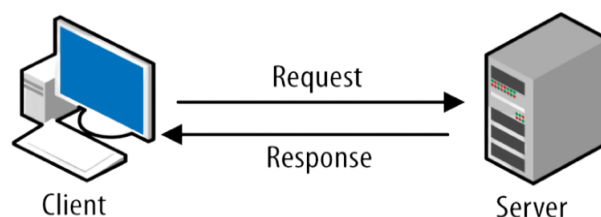
Dispositivi hardware e apparecchiature di rete coinvolti

I dispositivi coinvolti nella infrastruttura sopra schematizzata sono:

- a) l'**RDBMS** (Relation Database Management System, Sistema di gestione di base di dati relazionale), che si trova nella zona più protetta del sistema, proprio perché è necessario che i dati siano il più protetti possibile;
- b) i **router**, delle apparecchiature che si occupano di instradare i pacchetti basandosi sulle tabelle di routing;
- c) i **firewall** (letteralmente muro di fuoco), un sistema hardware-software che ha il compito di filtrare i pacchetti in arrivo e/o in uscita secondo delle apposite regole;
- d) una **DMZ** (Demilitarized Zone, Zona demilitarizzata), che permette di proteggere una rete nei punti in cui questa è a contatto con l'esterno, interponendosi tra la LAN (rete interna, intranet) e la WAN (rete esterna, internet). Nella mia infrastruttura ho inserito nella DMZ:
 - **Web Server**, che ha il compito di gestire il protocollo HTTP e la prima interfaccia col client;
 - **Script Engine**, che è il processo che si occupa della generazione delle pagine dinamiche, ha il compito di effettuare il rendering delle pagine (in ASP.NET il rendering consiste nella traduzione degli oggetti ASP.NET in codice HTML);
 - **Application Server**, che si occupa della logica applicativa che l'applicazione deve implementare.
- e) il **client**, su cui sarà presente un browser o l'app Android QuikTaxi (che si appoggia comunque ad un browser), che può connettersi tramite un dispositivo integrato modem - router con ADSL, Fibra o WiFi, oppure tramite connettività dati, fornita dal provider telefonico.

4) Tecniche di scambio dati fra client remoti e server tramite protocolli TCP e/o HTTP

Il protocollo utilizzato per permettere la trasmissione d'informazioni sul web sarà l'HTTPS (Hypertext Transfer Protocol), che lavora a livello applicativo (7) nella pila ISO/OSI e si appoggia al TLS. Generalmente un server che accetta richieste HTTPS resta in ascolto sulla porta 443 utilizzando il protocollo TCP/IP che lavora a livello trasporto (4) (il funzionamento dell'HTTPS è meglio spiegato nel punto 5). L'architettura è quindi quella tipica client - server.



5) Sicurezza delle Reti: Crittografia, Difesa Perimetrale e VPN

Gli attacchi in rete sono frequenti e c'è bisogno per questo che il sistema progettato sia sicuro. A questo scopo ho deciso di posizionare il DBMS il più lontano possibile dalla rete pubblica, poiché contiene i dati e quindi le informazioni sensibili. In questo modo, se un malintenzionato riuscisse a violare il primo **firewall**, accedendo ad un Web Server, non avrebbe libero accesso ai dati, che quindi risultano maggiormente isolati e protetti, perché per fare in modo che questo si verifichi dovrebbe violare anche il secondo firewall (è per questo motivo che si predilige la scelta di firewall di marche diverse, perché, essendo diverse le criticità potrebbero essere diverse e quindi l'infrastruttura risulta più sicura).

I componenti più critici del sistema, i Web Server, direttamente al contatto con il mondo esterno, e quindi con internet, sono stati posti all'interno della **DMZ** (Demilitarized Zone). L'utilizzo di quest'ultima, quando vengono esposti servizi in rete, è fondamentale poiché se venisse utilizzata una semplice configurazione formata da un unico firewall, e questo venisse violato, il malintenzionato si troverebbe subito all'interno della rete LAN, magari su un server, e questo comporterebbe un grande rischio. Mentre, se questo dovesse verificarsi nonostante l'uso della DMZ e qualcuno riuscisse a violare un server posto nella zona demilitarizzata, la possibilità che l'attacco si propaghi all'interno della LAN è molto bassa. Per definire una DMZ è necessario stabilire un IP statico; questa permette di esporre su internet un solo computer (o altro dispositivo) a cui vengono inoltrate le richieste di connessione.

Oltre all'utilizzo dei firewall e della DMZ si potrebbe pensare di implementare anche un sistema **IDS** (Intrusion Detection System), che è un sistema intelligente che si occupa di analizzare il traffico di rete cercando di identificare delle intrusioni o degli accessi al sistema indesiderati in modo da poterli bloccare. Questi, inoltre, hanno anche la funzione di fornire all'amministratore di rete un'accurata analisi di ciò che si verifica all'interno di essa; in questo modo, è possibile capire se è necessario aumentare la sicurezza o meno.

Invece, per la protezione del traffico di dati effettivo, scambiato tra client e server, è necessario implementare un algoritmo crittografico. Quest'ultimo è di facile implementazione tramite l'utilizzo del protocollo HTTPS per lo scambio dei dati, così come ipotizzato nel punto 4.

Il protocollo **HTTPS** (Hypertext Transfer Protocol Secure) è un protocollo di livello applicativo (7) e generalmente utilizza la porta 443. Questo garantisce la comunicazione tramite il protocollo HTTP all'interno di una connessione criptata, garantita dal **TLS** (Transport Layer Security), una collezione di protocolli di sicurezza che lavora a livello di sessione, al di sopra di quello di trasporto. Il TLS è organizzato su due livelli: il primo è quello che comprende l'handshake protocol, nella quale si concordano i protocolli da utilizzare per la criptazione dei dati e tutti i parametri utili (il TLS è per questo un protocollo aperto); il secondo è quello che contiene il TLS Record Protocol che si occupa della trasmissione dei dati che vengono criptati con un algoritmo di crittografia simmetrica, per motivi di efficienza e di velocità, le cui chiavi vengono scambiate attraverso l'utilizzo di un algoritmo di crittografia asimmetrica, per aumentare la sicurezza ed evitare attacchi del tipo Man in the middle. Collegandoci ad un sito tramite l'HTTPS, inoltre, il server ci fornirà il proprio **certificato** identificandosi; questo ci garantisce che il server con cui stiamo colloquiando è sicuramente verificato. L'utilizzo dei certificati ci garantisce quindi anch'esso la sicurezza; un certificato è formato dal nome del server, dalla chiave pubblica del server

e dal nome della Certification Authority che lo ha firmato (è possibile anche non rivolgersi ad una CA per ottenere un certificato poiché è possibile firmarsi da soli il proprio, rendendolo self-signed ma non riconosciuto dai browser).

7) Configurazione di reti cablate e Wi-Fi

Le apparecchiature coinvolte nell'infrastruttura sono rappresentate nella pagina 14, così come i collegamenti, che avverranno in gran parte tramite dei cavi ethernet (standard 802.3).

La connessione in WiFi effettuata dall'utente seguirà invece lo standard 802.11 con una frequenza portante variabile (2,4GHz o 5GHz).

8) Modalità e caratteristiche dell'accesso a internet

Così come rappresentato nel grafico dell'infrastruttura, il cliente (l'utente), potrà accedere ad internet tramite un dispositivo integrato Modem che fa anche da router, firewall e access point tramite un collegamento con il cavo ethernet utilizzando lo standard **802.3** o tramite la connettività WiFi con l'uso dello standard **802.11**, con una velocità variabile da 2,4GHz a 5GHz; l'apparecchio farà anche da DHCP, gli indirizzi agli host verranno assegnati in automatico. La seconda alternativa riguarda anche il taxista, che così come l'utente, grazie al suo smartphone (o al dispositivo fornito dall'azienda) potrà avere accesso ad Internet tramite la connessione dati (3G, 4G, 5G), fornitagli dal suo provider telefonico e fruibile grazie alle antenne BTS (Base Transceiver Station, Stazione radio base), che serve i terminali mobili fornendogli la connessione comprendo una determinata area geografica coperta dalla cella radio.

9) Architetture Web e gestione di server Web

La configurazione di rete implementata prevede l'utilizzo di **tre tier e una server farm**. Questa scelta è scaturita dalla necessità di avere un'infrastruttura più scalabile (facilmente adattabile alle esigenze), maggiormente affidabile e con maggiori prestazioni. I tier sono per l'appunto tre, identificati da: Client, Web Server e Application Server, RDBMS. In questa architettura, così come rappresentato nello schema di rete, i componenti più critici e soggetti a maggiori accessi vengono duplicati, questi sono: Web Server e Script Engine. La duplicazione è applicabile ad ogni livello grazie all'introduzione di una server farm in configurazione RACS o RAPS. La configurazione RACS prevede due modalità di funzionamento: shared nothing, in cui gli host non condividono nulla, neppure i dati, che vengono anch'essi duplicati e shared disk in cui più server condividono gli stessi dischi, in questo caso non vi è la clonazione dei dati, ma solo degli host. La modalità shared nothing viene solitamente utilizzata nelle applicazioni CDN (Content Delivery Network) in cui non si ha bisogno di salvare alcun dato poiché sono applicazioni read-only, di sola lettura.

Nella duplicazione del web server, dello script engine e dell'application server è introdotta una configurazione RACS shared nothing.

Duplicando le componenti più critiche, web server e script engine, verrà migliorata anche la disponibilità, poiché se un server dovesse cadere, ci sarebbero gli altri due a gestire le richieste che arrivano. La duplicazione è inoltre molto utile perché consente di migliorare le prestazioni, il carico viene infatti distribuito dai load balancer in modo bilanciato. La gestione delle richieste tramite il load balancer ha però un limite: se le applicazioni non sono stateless, ed hanno bisogno di mantenere uno stato, e quindi delle variabili di sessione, c'è bisogno che le richieste arrivino sempre allo stesso server. Visto che la mia applicazione non è stateless, ma ha bisogno di memorizzare delle variabili di sessione, è necessario introdurre un load balancer "evoluto" che sia in grado di capire qual è il client da cui proviene la richiesta in modo da poter inoltrare quest'ultima sempre allo stesso server che per primo ha risposto. È possibile individuare più alternative a questo sistema, si potrebbe, infatti decidere di stanziare un server dedicato che gestisce le sessioni a cui arrivano tutte le request e a cui possono collegarsi tutti i web server, oppure si potrebbe pensare di memorizzare le informazioni di sessione su un DBMS server, questo garantirebbe il salvataggio dei dati anche a fronte della rottura di uno dei server web. Il problema del load balancing non è da sottovalutare, infatti per la trasmissione dei dati utilizziamo l'HTTPS e così facendo si presenta lo stesso problema. Il protocollo HTTPS, di fatti, si appoggia al protocollo TLS che per la crittazione dei dati utilizza la crittografia ibrida; è per questo necessario, anche in questo caso, che il client continui a dialogare sempre con lo stesso server, visto e considerato che è quest'ultimo che mantiene le informazioni sulla chiave usata nella trasmissione.

10) Continuità del servizio e fault-tolerance

La continuità del servizio è garantita in gran parte dalla presenza di più server web duplicati. Grazie a questi, infatti, se uno dovesse andare fuori uso, il servizio verrebbe comunque erogato senza problemi. Per garantire la fault-tolerance e rendere il sistema più affidabile l'alimentazione sarà ridondata, in modo da non subire gravi danni in caso di sbalzi di tensioni. Per aumentare la tolleranza ai guasti ho implementato anche l'utilizzo del sistema RAID, presente all'interno del server DBMS, con configurazione RAID 1. La configurazione RAID 1 riproduce una copia esatta di tutti i dati su due dischi rigidi separati (HDD interni al DBMS), garantendo una maggiore sicurezza dei dati (disco in mirror), se infatti uno dei due dischi dovesse rompersi, i dati non andrebbero persi perché memorizzati all'interno dell'altro. Inoltre, se un disco dovesse rompersi, una volta sostituito, sarebbe il RAID a ricomporlo, così com'era il precedente. Per aumentare questa sicurezza si potrebbe pensare di effettuare un backup dei dati presenti nel Database su un servizio di cloud fornito sul web.

11) I software utilizzati per lo sviluppo

Per lo sviluppo delle pagine web statiche, come quella di default o quella di errore, ho utilizzato del semplice linguaggio HTML utilizzando l'ambiente di sviluppo Microsoft Visual Studio 2019, nonostante fosse sufficiente l'uso di un semplice editor come Notepad++. Nelle pagine in cui era necessaria l'identificazione della posizione ho utilizzato degli script JavaScript, un linguaggio adibito alla programmazione lato client. L'applicazione QuickTaxi sarà invece sviluppata grazie all'ambiente di sviluppo Android Studio in linguaggio Java. Per la creazione delle pagine dinamiche, invece, mi sono servita dell'ambiente di sviluppo Microsoft Visual Studio utilizzando il linguaggio C# e gli oggetti forniti da ASP.NET. L'accesso al database viene gestito da ADO.NET, una piattaforma integrata all'interno del framework costituita da un insieme di classi che permettono di accedere al database, gestito dall'RDBMS, che attende le richieste di accesso ai dati un po' come fa il server con le request.

I dispositivi su cui installare l'applicazione dovranno necessariamente avere un sistema operativo Android, o l'app QuickTaxi non verrà supportata. Sottolineo, ancora una volta, che sull'app Android sarà presente una web view, quindi sarà come se l'utente navigasse tramite un browser sulle pagine web; vista la scelta è necessario che il sito sia **responsive** e che quindi sia in grado di adattarsi graficamente in modo automatico al dispositivo con il quale viene visualizzato.

Il dispositivo mobile del taxista potrà avere un SO a scelta, l'importante è che sia presente un browser atto alla navigazione. Lo stesso discorso vale per i dispositivi dell'utente usati per la registrazione, l'importante è che sia presente un browser. Sui server web sarà installato Windows Server 2019, mentre, sul server DBMS sarà installato Microsoft SQL Server 2014.

0) Implementazione delle pagine web e rappresentazione grafica delle pagine previste

Risorsa: **PrenotaTaxi.aspx**

È la pagina dedicata all'utente e gestisce la prenotazione di un taxi! Qui è presente lo script JavaScript che recupera le coordinate correnti del client; nelle pagine che lo richiedono verrà implementato lo stesso codice, utilizzando la stessa tecnica.

//Di seguito lo script (lato client) che recupera le coordinate correnti e le mostra nelle text-box apposite..

```
<script type="text/javascript">
    var lat = document.getElementById("txtLat");
    var long = document.getElementById("txtLong");

    function getLocation() { //Recupero la posizione
        if (navigator.geolocation)
            navigator.geolocation.getCurrentPosition(showPosition);
        else
            alert("La geolocalizzazione non è supportata da questo browser");
    }
}
```



```

        function showPosition(position) { //Visualizzo la posizione
            lat.value = position.coords.latitude;
            long.value = position.coords.longitude;
        }
    }</script>

//Di seguito il codice C#
using System.Data;
using System.Data.SqlClient;
using System.Device.Location; //Namespace necessario per l'utilizzo della classe geocoordinate

namespace PrenotazioneTaxi_Utente
{
    public partial class PrenotaTaxi : System.Web.UI.Page
    {
        SqlConnection CN = new SqlConnection("Server=(localdb)\\MSSQLLocalDB;Data-
        base=PrenotazioneTaxi; user id=sa;password=abacus");
        SqlCommand cmd;
        SqlDataReader DR;

        private int TaxiVicino(List<double> MiaLst) //Cerco il taxi più vicino all'utente
        {
            int PosIDMin = 0;
            for (int K = 3; K <= MiaLst.Count - 1; K += 2)
                if (MiaLst[PosIDMin + 1] > MiaLst[K])
                    PosIDMin = K - 1;
            return Convert.ToInt32(MiaLst[PosIDMin]);
        }

        private int IDZona(List<double> MiaLst) //Definisco la zona in cui si trova l'utente
        (zona di partenza della prenotazione)
        {
            int PosIDMin = 0;
            for (int K = 3; K <= MiaLst.Count - 1; K += 2)
                if (MiaLst[PosIDMin+1] > MiaLst[K])
                    PosIDMin = K-1;
            if (MiaLst[PosIDMin + 1] <= 10000)
                return Convert.ToInt32(MiaLst[PosIDMin]);
            else
                return -1;
        }

        private bool PrenAttiva() //Controllo se l'utente ha già una prenotazione attiva
        {
            bool Ris = false;
            cmd = new SqlCommand("SELECT idPrenotazione FROM Prenotazione WHERE LongitudineArr
            IS NULL AND idUtente=" + Session["idUtente"], CN);
            CN.Open();
            if (cmd.ExecuteScalar() != null)
            {
                plsPrenota.Visible = false;
                int idPren = int.Parse(cmd.ExecuteScalar().ToString());
                lblEsito.Text = "Hai già prenotato il tuo taxi!! La tua prenotazione è " + idPren ;
                hplTraccia.Visible = true;
                hplTraccia.NavigateUrl = "TracciaIlMioTaxi.aspx?idPrenotazione="+idPren;
                fld.Visible = false;
                Ris = true;
            }
        }
    }
}

```

```

    }
    CN.Close();
    return Ris;
}

protected void Page_Load(object sender, EventArgs e)
{
    hplTraccia.Visible = false;
    if (Session["idUtente"] == null)
        Response.Redirect("Errore.aspx");
    else
    {
        if (!IsPostBack)
        {
            cmd = new SqlCommand("SELECT (Nome + ' ' + Cognome) AS Nominativo FROM Utente
WHERE idUtente=" + Session["idUtente"], CN);
            CN.Open();
            string Nominativo = cmd.ExecuteScalar().ToString().Trim();
            CN.Close();
            if (!PrenAttiva())
                lblUtente.Text = "Benvenuto nella tua area personale " + Nominativo.ToUp-
per() + ", qui potrai prenotare un taxi; ricorda di attivare il GPS!";
            else
                lblUtente.Text = "Benvenuto nella tua area personale " + Nominativo.ToUp-
per() + " hai già una prenotazione attiva, traccia il tuo taxi con il link qui sotto!";
        }
    }
}

protected void plsPrenota_Click(object sender, EventArgs e)
{
    if (txtLat.Text.Length>0 && txtLong.Text.Length>0)
    {
        List<double> DistanzeInM = new List<double>();
        double Latitudine=double.Parse(txtLat.Text);
        double Longitudine = Convert.ToDouble(txtLong.Text);
        GeoCoordinate geo = new GeoCoordinate(Latitudine, Longitudine);
        cmd= new SqlCommand("SELECT * FROM Zona", CN);
        CN.Open();
        DR = cmd.ExecuteReader();
        GeoCoordinate PuntoZona;
        if(DR.HasRows)
        {
            while (DR.Read())
            {
                DistanzeInM.Add(Convert.ToDouble(DR["idZona"]));
                PuntoZona = new GeoCoordinate(Convert.ToDouble(DR["CentroLatitudine"].To-
String()), Convert.ToDouble(DR["CentroLongitudine"].ToString()));
                DistanzeInM.Add(geo.GetDistanceTo(PuntoZona));
            }
        }
        CN.Close();
        int idZona = IDZona(DistanzeInM); //idZona contiene l'id della zona in cui si trova l'utente
        if (idZona != -1)
        {
            cmd = new SqlCommand("SELECT idTaxi, LatitudineCorr, LongitudineCorr FROM Taxi
WHERE Libero=1 AND idZona="+idZona, CN); //Cerco i taxi liberi nella zona dell'utente
            CN.Open();
            DR = cmd.ExecuteReader();
            GeoCoordinate geotaxi;

```

```

DistanzeInM.Clear();
if (DR.HasRows)
{
    while (DR.Read())
    {
        geotaxi = new GeoCoordinate(Convert.ToDouble(DR["LatitudineCorr"].ToString()),
Convert.ToDouble(DR["LongitudineCorr"].ToString()));
        DistanzeInM.Add(Convert.ToDouble(DR["idTaxi"]));
        DistanzeInM.Add(geo.GetDistanceTo(geotaxi));
    }
    int IdTaxiVicino = TaxiVicino(DistanzeInM); //Salvo l'id del taxi più vicino alla posizione
dell'utente a cui assegnerò la prenotazione
    CN.Close();
    cmd = new SqlCommand("INSERT INTO Prenotazione (idUser, idTaxi, LatitudinePart,
LongitudinePart, idZonaPart) VALUES (" + Session["idUser"] + ", " + IdTaxiVicino + ",
@Lat, @Long, " + idZona + "); SELECT @@Identity; UPDATE Taxi SET Libero=0 WHERE idTaxi=" +
IdTaxiVicino, CN); //Assegno la prenotazione al taxi e lo rendo occupato
    cmd.Parameters.Add("@Lat", SqlDbType.Float);
    cmd.Parameters["@Lat"].Value = Latitudine;
    cmd.Parameters.Add("@Long", SqlDbType.Float);
    cmd.Parameters["@Long"].Value = Longitudine;
    CN.Open();
    int idPren = Convert.ToInt32(cmd.ExecuteScalar()); //Salvo l'id della prenotazione appena
fatta per rendere possibile il tracciamento
    CN.Close();
    if (idPren >= 1)
    {
        hplTraccia.NavigateUrl = "TracciaIlMioTaxi.aspx?idPrenotazione=" + idPren;
        lblEsito.Text = "HAI PRENOTATO CORRETTAMENTE IL TUO TAXI! SEGUILO CLICCANDO IL
LINK QUI SOTTO!";
        hplTraccia.Visible = true;
    }
    else
        lblEsito.Text = "OPS.. Qualcosa è andato storto.. Riprova tra qualche minuto!";
}
else
    lblEsito.Text = "Non sono disponibili taxi al momento; risultano tutti occupati!!
Riprova tra poco";
}
else
    lblEsito.Text = "La posizione in cui ti trovi è in una zona non coperta dal servi-
zio; prova a spostarti!!";
}
else
    lblEsito.Text = "IMPOSSIBILE PRENOTARE IL TAXI; PRIMA DEVI ATTIVARE LA GEOLOCALIZZA-
ZIONE!";
}
}
}

```

Risorsa: **Prenotazioni.aspx**

È una pagina dedicata al taxista, che dopo aver effettuato l'accesso verrà reindirizzato qui dove potrà vedere se gli è stata assegnata una prenotazione. La prenotazione gli verrà assegnata dopo aver effettuato l'accesso, momento in cui le sue coordinate correnti vengono salvate e il taxi diventa libero (il recupero delle coordinate avviene nella pagina autenticataxi.aspx grazie allo script utilizzato anche nella pagina PrenotaTaxi.aspx).

All'interno di questa pagina ho inserito un timer, che ogni 20 secondi scatena un postback. Allo scattare di quest'ultimo viene controllato se al taxista è stata assegnata qualche prenotazione, se gliene è stata assegnata una compariranno i dati relativa a quest'ultima sulla pagina e un collegamento, che gli permette di andare sulla pagina PrenotazioneInCorso.aspx e visualizzare il percorso da compiere per arrivare dal suo cliente.

Per inserire il timer ho dovuto utilizzare un'estensione di ajax. Il timer si trova all'interno di uno ScriptManager e di un UpdatePanel, grazie a questi due elementi la pagina non verrà ricaricata tutta; verranno aggiornati solo gli oggetti che si trovano al loro interno. Nel mio caso:

```
<asp:ScriptManager ID="ScrMng"
    runat="server" />
    <asp:UpdatePanel ID="UpdatePanel"
        runat="server">
        <ContentTemplate>
        <asp:Timer ID="tmrPrenotazione" runat="server" Interval="20000" On-
Tick="tmrPrenotazione_Tick"></asp:Timer>
        <asp:Label ID="lblPrenotazione" runat="server" ></asp:Label><br /><br />
        <asp:HyperLink ID="hplPercorso" Text="VISUALIZZA IL PERCORSO DA COM-
PIERE" runat="server"></asp:HyperLink>
        </ContentTemplate>
    </asp:UpdatePanel>
```

//Di seguito il codice C# della pagina

```
SqlConnection CN = new SqlConnection("Server=(localdb)\\MSSQLLocalDB;Database=Prenota-
zioneTaxi; user id=sa;password=abacus");
SqlCommand CMD;
```

```
protected void Page_Load(object sender, EventArgs e)
```

```
{
    if (Session["idTaxi"] == null)
        Response.Redirect("Errore.aspx");
    else
    {
        if (!IsPostBack)
        {
            CMD = new SqlCommand("SELECT NomeTaxi FROM Taxi WHERE idTaxi=" + Ses-
sion["idTaxi"], CN);
            CN.Open();
            string NomeTaxi = CMD.ExecuteScalar().ToString().Trim();
            lblUtente.Text = "Ciao, benvenuto nella pagina dedicata al taxi che guidi:" +
NomeTaxi.ToUpper() + ", qui potrai attendere nuove prenotazioni; appena te ne arriverà
una verrai reindirizzato in una nuova pagina dove potrai visualizzare il percorso da
compiere per arrivare dal tuo cliente!";
            CN.Close();
            hplPercorso.Visible = false;
        }
    }
}
```

```
protected void tmrPrenotazione_Tick(object sender, EventArgs e)
```

```
{
    CMD = new SqlCommand("SELECT idPrenotazione, Cognome, Nome, LatitudinePart, Longi-
tudinePart FROM Prenotazione INNER JOIN Utente ON Prenotazione.idUtente=Utente.idUtente
WHERE LatitudineArr IS NULL AND idTaxi=" + Session["idTaxi"], CN);
```

```

CN.Open();
SqlDataReader DR = CMD.ExecuteReader();
string Query = "";
if (DR.Read())
{
    lblPrenotazione.Text = "Ti è stata affidata la prenotazione numero " +
DR["idPrenotazione"].ToString() + "\n" + "Il tuo passeggero è: " + DR["Cognome"].To-
String().Trim().ToUpper() + " " + DR["Nome"].ToString().Trim().ToUpper() + "\n" + "Per
visualizzare il percorso da compiere clicca sul link qui sotto!";
    hplPercorso.NavigateUrl = "PrenotazioneInCorso.aspx?Prenotazione=" +
DR["idPrenotazione"];
    hplPercorso.Visible = true;
}
else
    lblPrenotazione.Text = "Non ti è stata affidata nessuna prenotazione!!";
CN.Close();
}

```

Risorsa: **PrenotazioneInCorso.aspx**

È una pagina dedicata al taxista, a cui potrà accedere tramite un link presente nella pagina Prenotazioni.aspx. Qui sarà presente un timer, che, scatenando un postback ogni trenta secondi, trasmetterà al server le coordinate correnti del tassista e i minuti di attesa calcolati con la formula $\text{Tempo} = \text{Spazio} / \text{Velocità}$. Le coordinate e la velocità verranno calcolate grazie ad uno script JavaScript che riporto di seguito. All'interno di questa pagina sarà, inoltre, presente un link che permetterà al taxista di visualizzare il percorso da compiere per arrivare nella posizione dell'utente; questo è stato realizzato con l'uso di un api di Google. Il timer che si occuperà di scatenare il postback è uguale a quello presente nella pagina Prenotazioni.aspx.

//Di seguito lo script (lato client) che recupera le coordinate correnti e la velocità corrente espressa in m/s e li mostra nelle apposite textbox..

```

<script type="text/javascript">
    var lat = document.getElementById("txtLat");
    var long = document.getElementById("txtLong");
    var temp = document.getElementById("TEMP");
    var myVar = setInterval(Timer, 1000);
    function getLocation() {
        if (navigator.geolocation)
            navigator.geolocation.getCurrentPosition(showPosition);
        else
            alert("La geolocalizzazione non è supportata da questo browser");
    }

    function showPosition(position) {
        lat.value = position.coords.latitude;
        long.value = position.coords.longitude;
        temp.value = position.coords.speed; //Recupero velocità
    }
    function Timer() {
        getLocation();
    }
</script>

```

//Il codice C#

```

SqlConnection CN = new SqlConnection("Server=(localdb)\\MSSQLLocalDB;Database=Prenota-
zioneTaxi; user id=sa;password=abacus");

```

```

SqlCommand CMD;
string idPrenotazione = "";
double LongitudineCliente = 0;
double LatitudineCliente = 0;

protected void Page_Load(object sender, EventArgs e)
{
    idPrenotazione = Request.QueryString["idPrenotazione"];
    if (Session["idTaxi"] == null && idPrenotazione == "")
        Response.Redirect("Errore.aspx");
    else
    {
        hplTermina.Visible = false;
        plsConferma.Visible = false;
        string NomeTaxi = "";
        string CognomeUtente = "";
        string NomeUtente = "";
        CMD = new SqlCommand("SELECT NomeTaxi, LatitudinePart, LongitudinePart, DataOraPartenza, DataOraArrivo, Cognome, Nome FROM Prenotazione INNER JOIN Utente ON Prenotazione.idUtente=Utente.idUtente INNER JOIN Taxi ON Prenotazione.idTaxi=Taxi.idTaxi WHERE idPrenotazione=" + idPrenotazione, CN);
        CN.Open();
        SqlDataReader DR = CMD.ExecuteReader();
        if (DR.HasRows)
        {
            while (DR.Read())
            {
                LongitudineCliente = Convert.ToDouble(DR["LongitudinePart"].ToString()).Replace(".", ",");
                LatitudineCliente = Convert.ToDouble(DR["LatitudinePart"].ToString()).Replace(".", ",");
                NomeTaxi = DR["NomeTaxi"].ToString().Trim();
                CognomeUtente = DR["Cognome"].ToString().Trim();
                NomeUtente = DR["Nome"].ToString().Trim();
                if (DR["DataOraPartenza"] != null) //Verifico se la prenotazione è stata già confermata..
                    plsConferma.Visible = true;
                else
                {
                    if (DR["DataOraArrivo"] != null) //Verifico se la prenotazione è stata già terminata..
                    {
                        hplTermina.NavigateUrl = "TerminaCorsa.aspx?idPrenotazione=" + idPrenotazione;
                        hplTermina.Visible = true;
                        lblPren.Text = "Questa prenotazione è stata confermata il " + Convert.ToDateTime(DR["DataOraPartenza"]).ToString() + ". Quando arrivi a destinazione puoi terminarla cliccando sull'apposito link!";
                        miop.Visible = false;
                    }
                }
                else
                    Response.Redirect("Prenotazioni.aspx");
            }
        }
        CN.Close();
        double LatitudineCorr = Convert.ToDouble(txtLat.Text);
        double LongitudineCorr = Convert.ToDouble(txtLong.Text);
        lblUtente.Text = "Ciao, benvenuto nella pagina dedicata al taxi che guidi:" + NomeTaxi.ToUpper() + ", qui potrai attendere nuove prenotazioni; appena te ne arriverà una

```



```

potrai visualizzare il percorso da compiere per arrivare dal tuo cliente cliccandos sul
link qui in basso!";
lblPrenotazione.Text = "TI E' STATA ASSEGNATA LA PRENOTAZIONE NUMERO " + idPrenotazione
+ " IL TUO CLIENTE E' " + CognomeUtente + " " + NomeUtente;
miop.NavigateUrl = "https://www.google.cl/maps/?saddr=" + LatitudineCorr.ToString().Re-
place(",", ".") + "," + LongitudineCorr.ToString().Replace(",", ".") + "&daddr=" + La-
titudineCliente.ToString().Replace(",", ".") + "," + LongitudineCliente.ToString().Re-
place(",", ".") + "&directionsmode=driving"; //Percorso Google Maps
tmrPrenotazione.Enabled = true;
    }
}
}

```

```

protected void plsConferma_Click(object sender, EventArgs e)
{
    CMD = new SqlCommand("UPDATE Prenotazione SET DataOraPartenza=@DataOraPartenza WHERE
idPrenotazione=" + idPrenotazione, CN);
    CMD.Parameters.Add("@DataOraPartenza", SqlDbType.DateTime);
    CMD.Parameters["@DataOraPartenza"].Value = DateTime.Now;
    CN.Open();
    int MOD=CMD.ExecuteNonQuery();
    CN.Close();
    if (MOD == 1)
    {
        lblPren.Text = "La prenotazione è stata confermata; il pagamento avverrà automati-
camente quando questa verrà chiusa!";
        hplTermina.NavigateUrl = "TerminaCorsa.aspx?idPrenotazione=" + idPrenotazione;
        hplTermina.Visible = true;
        plsConferma.Visible = false;
        miop.Visible = false;
        tmrPrenotazione.Enabled = false;
    }
    else
        lblPren.Text = "OPS.. C'è qualche problema..";
}

```

```

protected void tmrPrenotazione_Tick(object sender, EventArgs e)
{
    double LatitudineCorr = Convert.ToDouble(txtLat.Text);
    double LongitudineCorr = Convert.ToDouble(txtLong.Text);
    int Velocita =int.Parse(TEMP.Text);
    GeoCoordinate CoordCliente = new GeoCoordinate(LatitudineCliente, LongitudineCliente);
    GeoCoordinate CoordCorr = new GeoCoordinate(LatitudineCorr, LongitudineCorr);
    double Spazio =Math.Truncate(CoordCorr.GetDistanceTo(CoordCliente));
    double MinAttesa =Math.Truncate((Spazio/Velocita)/60);
    CMD = new SqlCommand("UPDATE Taxi SET LatitudineCorr=@Lat, LongitudineCorr=@Long
WHERE idTaxi=" + Session["idTaxi"] + "; UPDATE Prenotazione SET MinAttesaCorr=@min WHERE
idPrenotazione=" + idPrenotazione, CN);
    CMD.Parameters.Add("@Lat", SqlDbType.Float);
    CMD.Parameters["@Lat"].Value = LatitudineCorr;
    CMD.Parameters.Add("@Long", SqlDbType.Float);
    CMD.Parameters["@Long"].Value = LongitudineCorr;
    CMD.Parameters.Add("@min", SqlDbType.TinyInt);
    CMD.Parameters["@min"].Value =Convert.ToSByte(MinAttesa);
    CN.Open();
    int Mod = CMD.ExecuteNonQuery();
    CN.Close();
    if (Mod>=1)
        lblMin.Text = "MINUTI ATTESA:" + MinAttesa.ToString();
}

```

Risorsa: **TracciaIlMioTaxi.aspx**

È una pagina dedicata all'utente. In questa pagina egli visualizzerà una mappa in cui sarà presente un marker che permetterà di visualizzare il taxi in movimento. Per la realizzazione della mappa ho utilizzato Bing Maps, un servizio di mappe virtuali offerto dal motore di ricerca Bing di Microsoft. Per usufruire del servizio ho creato un account sviluppatore sulla piattaforma Bing Maps e richiesto una chiave per il mio sito. Di seguito riporto il codice JavaScript utilizzato per la generazione della mappa, contenente il marker che indica il taxi la cui posizione è aggiornata ogni 25 secondi grazie ad un timer e al recupero delle coordinate correnti del taxi nel database ogni 30 secondi, grazie all'uso di un timer identico a quello presente nella pagina Prenotazioni.aspx che genera un postback.

//Lo script JavaScript (lato client)

```
<script type='text/javascript'>
    function GetMap() {
        var Lat = document.getElementById("txtLat");
        var Long = document.getElementById("txtLong");
        var myVar = setInterval(Timer, 25000);
        var map = new Microsoft.Maps.Map('#myMap', {
            credentials: 'AvSIS_iy0lmyDPiCP2ddMy-
iLv02n_XK8M8HnatNk2Fp6CTnUX096Y8zSs208NM-p',
            center: new Microsoft.Maps.Location(Lat.value, Long.value)
        });
        var center = map.getCenter();
        var pin = new Microsoft.Maps.Pushpin(center, {
            title: 'Posizione corrente',
            text: 'TAXI',
            color: 'red'
        });
        map.entities.push(pin);
    }

    function Timer() {
        GetMap();
    }
</script>
<script type='text/javascript' src='http://www.bing.com/api/maps/mapcontrol?call-
back=GetMap' async defer></script>
```

//Il codice C#

```
SqlConnection CN = new SqlConnection("Server=(localdb)\\MSSQLLocalDB;Database=Prenotazione-
Taxi; user id=sa;password=abacus");
SqlCommand cmd;
SqlDataReader DR;
int idPrenotazione = 0;

protected void Page_Load(object sender, EventArgs e)
{
    idPrenotazione = Convert.ToInt32(Request.QueryString["idPrenotazione"]);
    if (Session["idUtente"] == null && idPrenotazione <= 0)
        Response.Redirect("Errore.aspx");
    else
    {
        if (!IsPostBack)
        {
            cmd = new SqlCommand("SELECT (Nome + ' ' + Cognome) AS Nominativo FROM Utente WHERE
idUtente=" + Session["idUtente"], CN);
```

```

        CN.Open();
        string Nominativo = cmd.ExecuteScalar().ToString().Trim();
        CN.Close();
        lblUtente.Text = "Benvenuto nella tua area personale " + Nominativo.ToUpper() + ",
qui potrai prenotare seguire il taxi che hai prenotato (" + idPrenotazione + ")!";
    }
}

protected void tmrPrenotazione_Tick(object sender, EventArgs e)
{
    string NomeTaxi = "";
    double Lat = 0;
    double Long = 0;
    int Min = 0;
    cmd = new SqlCommand("SELECT NomeTaxi, LatitudineCorr, LongitudineCorr, MinAttesaCorr FROM
Prenotazione INNER JOIN Taxi ON Prenotazione.idTaxi=Taxi.idTaxi WHERE idPrenotazione=" +
idPrenotazione, CN);
    CN.Open();
    DR = cmd.ExecuteReader();
    if (DR.HasRows)
        while (DR.Read())
        {
            NomeTaxi = DR["NomeTaxi"].ToString().Trim();
            txtLat.Text = DR["LatitudineCorr"].ToString().Replace(",", ".");
            txtLong.Text = DR["LongitudineCorr"].ToString().Replace(",", ".");
            Min = int.Parse(DR["MinAttesaCorr"].ToString());
            txtMin.Text = Min.ToString();
        }
    else
        lblRis.Text = "Qualcosa non va, i dati non sono disponibili. Contatta l'associazione
Unione RadioTaxi!";
    CN.Close();
}

```

La rappresentazione grafica delle pagine dedicate al cliente

Risorsa: default.aspx

BENVENUTO NEL SITO UNIONE RADIOTAXI!



Sei un utente?

[REGISTRATI](#)

Se hai già un account

[ACCEDI](#)

Sei un taxista?

Accedi con le credenziali associate al tuo taxi!

[ACCEDI](#)

Risorsa: Registrazione.aspx

REGISTRATI AL SERVIZIO DI RADIOTAXI!



Nome:

Cognome:

Data di nascita:

INSERISCI I DATI DELLA TUA CARTA DI CREDITO

Numero di carta:

CVV:

Scadenza:

INSERISCI USERNAME E PASSWORD

Username:

Password:

[REGISTRATI](#)

©2021 Ilaria Frandina

Risorsa: Accesso.aspx

ACCEDI AL SERVIZIO DI PRENOTAZIONE!

Username:

Password:

[ACCEDI](#)

Risorsa: AreaRiservata.aspx

UNIONE RADIOTAXI



Ciao ILARIA FRANDINA benvenuto nella tua area riservata

[VISUALIZZA LE CORSE EFFETTUATE](#)

[PRENOTA UN TAXI](#)

Risorsa: home.aspx

Grazie per esserti registrato al servizio di prenotazione. Per iniziare ad usufruire del servizio scarica l'app QuickTaxi!

[SCARICA l'app Quicktaxi](#)

Risorsa: PrenotaTaxi.aspx

PRENOTA IL TUO TAXI!

Benvenuto nella tua area personale ILARIA FRANDINA, qui potrai prenotare un taxi; ricorda di attivare il GPS!

COORDINATE CORRENTI

[GEOLOCALIZZAMI](#)

[PRENOTA IL TUO TAXI](#)

Risorsa: CorseEffettuate.aspx

UNIONE RADIOTAXI

Ciao ILARIA FRANDINA qui trovi tutte le corse effettuate e i relativi costi che ti sono stati addebitati sulla carta che hai fornito al momento della registrazione!

ID	Taxi	Data&Ora partenza	Data&Ora arrivo	Costo
2	TaxiPiazza3	29/05/2021 23:07:33	29/05/2021 23:07:39	10,0000
4	TaxiStazione1	28/05/2021 22:15:11	28/05/2021 22:15:16	12,0000
106	TaxiPiazza2	30/05/2021 07:42:29	30/05/2021 07:42:41	9,0000
107	TaxiPiazza2	30/05/2021 08:22:35	30/05/2021 08:22:48	5,0000
110	TaxiPiazza2			12,0000

**Risorsa: PrenotaTaxi.aspx - Prenotazione
avvenuta correttamente**

PRENOTA IL TUO TAXI!

Benvenuto nella tua area personale ILARIA FRANDINA, qui potrai prenotare un taxi; ricorda di attivare il GPS!

COORDINATE CORRENTI


HAI PRENOTATO CORRETTAMENTE IL TUO TAXI! SEGUILO
CLICCANDO IL LINK QUI SOTTO!

[TRACCIA IL MIO TAXI](#)

Se l'utente si trova in una zona non coperta non potrà prenotare nessun taxi e verrà invitato a spostarsi, mentre, se i taxi sono tutti occupati verrà invitato a riprovare più tardi!

**Risorsa: TracciaIMioTaxi.aspx
UNIONE RADIO TAXI**

SEGUI IL TUO TAXI



Benvenuto nella tua area personale ILARIA FRANDINA, qui potrai prenotare seguire il taxi che hai prenotato (111)!

COORDINATE DEL TAXI

MINUTI DI ATTESA

La rappresentazione grafica delle pagine dedicate al taxista

Risorsa: default.aspx

BENVENUTO NEL SITO UNIONE RADIOTAXI!



<p>Sei un utente?</p> <p><input type="button" value="REGISTRATI"/></p> <p>Se hai già un account</p> <p><input type="button" value="ACCEDI"/></p>	<p>Sei un taxista?</p> <p>Accedi con le credenziali associate al tuo taxi!</p> <p><input type="button" value="ACCEDI"/></p>
--------------------------------------------------------------------------------------------------------------------------------------------------	-----------------------------------------------------------------------------------------------------------------------------

Risorsa: AutenticaTaxi.aspx

ACCEDI CON LE CREDENZIALI ASSOCIATE AL TUO TAXI

Potrai visualizzare le prenotazioni che affidate al tuo taxi e il percorso da compiere per arrivare dove si trova il tuo cliente!

Username:

Password:

Risorsa: Prenotazioni.aspx -
nessuna prenotazione assegnata

UNIONE RADIOTAXI

Ciao, benvenuto nella pagina dedicata al taxi che guidi: TAXISTAZIONE1, qui potrai attendere nuove prenotazioni; appena te ne arriverà una potrai visualizzare il percorso da compiere per arrivare dal tuo cliente!

Non ti è stata affidata nessuna prenotazione!!

Risorsa: Prenotazioni.aspx -
prenotazione assegnata

UNIONE RADIOTAXI

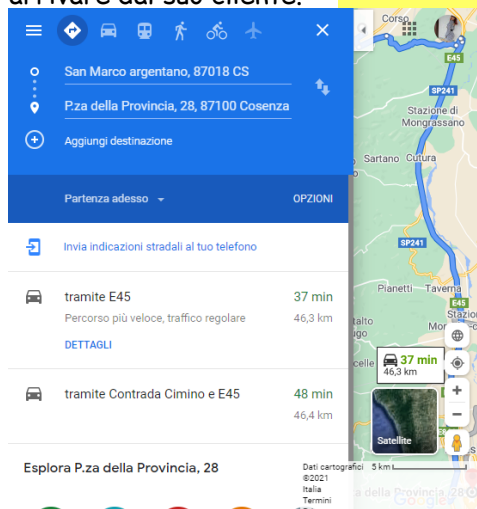
Ciao, benvenuto nella pagina dedicata al taxi che guidi: TAXISTAZIONE1, qui potrai attendere nuove prenotazioni; appena te ne arriverà una potrai visualizzare il percorso da compiere per arrivare dal tuo cliente!

Ti è stata affidata la prenotazione numero 112 Il tuo passeggero è: FRANDINA ILARIA Per visualizzare il percorso da compiere clicca sul link qui sotto!

[VISUALIZZA IL PERCORSO DA COMPIERE](#)

Risorsa: PrenotazioneInCorso.aspx

Collegamento a Google Maps con mappa dinamica; il percorso è quello che il taxi deve compiere per arrivare dal suo cliente.



UNIONE RADIO TAXI

PRENOTAZIONE IN CORSO

TAXI

Ciao, benvenuto nella pagina dedicata al taxi che guidi: TAXISTAZIONE1, qui potrai attendere nuove prenotazioni; appena te ne arriverà una potrai visualizzare il percorso da compiere per arrivare dal tuo cliente cliccando sul link qui in basso!

TI E' STATA ASSEGNATA LA PRENOTAZIONE NUMERO 112 IL TUO CLIENTE E' Frandina Ilaria

MINUTI ATTESA: 49

COORDINATE CORRENTI

39.554337499999995
16.1636648

VELOCITA' CORRENTE (m/sec)

[VISUALIZZA IL PERCORSO PER ARRIVARE DAL CLIENTE](#)

N.B. La velocità non è indicata poiché è una simulazione; la velocità per calcolare i minuti di attesa è stata inserita da codice ed è uguale a 40km/h.

Quando il taxista arriverà dal suo cliente confermerà la prenotazione, successivamente a destinazione cliccherà sul link "TERMINA LA CORSA" per registrarne il termine.

Risorsa: TerminaCorsa.aspx

Ciao TAXISTAZIONE1, termina la corsa numero 112 subito!

PRENOTAZIONE CHIUSA! IL PAGAMENTO AVVERRÀ IN AUTOMATICO

Dopo aver terminato la corsa, se il taxista ha finito il suo orario di lavoro, potrà disconnettersi dal servizio!

N.B i soldi verranno scalati automaticamente dalla carta dell'utente.

■ L'accessibilità in rete

Quando si offrono dei servizi sul web è necessario che questi siano accessibili a tutti. L'accessibilità in rete è nata dalla necessità di poter rendere fruibile un sito web, o un qualsivoglia sistema informatico, anche a coloro che sono affetti da disabilità temporanee e non, che ne possono compromettere l'utilizzo.

Per garantire l'accessibilità c'è bisogno di avere molti accorgimenti; parlando per esempio di un sito web, una migliore accessibilità potrebbe essere raggiunta grazie all'utilizzo di caratteri più grandi o di colori non fastidiosi per la vista. Il sito web deve inoltre essere strutturato utilizzando un layout chiaro e di facile intuizione che permetta all'utente un facile utilizzo, lo stesso deve essere garantito anche per il linguaggio utilizzato che non deve essere complesso e non deve presentare terminologie di gergo o acronimi inspiegabili. Abbiamo inoltre detto che l'accessibilità si propone di scavalcare i limiti delle disabilità che abbiamo classificato in temporanee e non; perché una disabilità può essere anche la mancanza di un mouse, come può esserlo l'impossibilità di muovere gli arti permanentemente; e se un sito web accessibile deve rimuovere questi ostacoli cosa può fare? Per ovviare alla mancanza del mouse, potrebbe essere sufficiente rendere fruibile il servizio attraverso l'uso della tastiera, mentre per ovviare alla disabilità fisica legata al movimento degli arti potrebbe bastare l'utilizzo di un sintetizzatore vocale. Detta così, gli accorgimenti da mettere in atto per rendere più accessibile un sito web sembrano poco onerosi di tempo e semplici da attuare, ma allora perché è così difficile rendere totalmente accessibile un sito web? A parer mio, è complesso perché è impossibile far fronte ad ogni disabilità, che sia temporanea o che non lo sia.

La WAI (Web Accessibility Initiative), parte del W3C (World Wide Web Consortium), a proposito di accessibilità, ha prodotto tre documenti per rendere accessibile il web, fra questi vi è il WCAG (Web Content Accessibility Guidelines), che contiene le linee guida per rendere una pagina web accessibile. Esistono diverse versioni della WCAG, quella attualmente utilizzata è la 2.1 che contiene 13 linee guida e 78 criteri.

È molto difficile rendere una pagina web accessibile in tutto e per tutto e proprio per questo il W3C ha deciso di introdurre 3 livelli di conformità che ci permettono di stabilire la qualità della nostra pagina:

- A: conformità minima;
- AA: conformità media;
- AAA: conformità massima.

Gli accorgimenti da avere per rispettare le linee guida della WCAG sono molti, e io ho provato ad applicarli in una delle pagine web del mio progetto, che successivamente ho validato con i tool offerti da WAVE e W3C.

La pagina che ho validato e che ho cercato di rendere il più accessibile possibile è stata quella dedicata alla registrazione.

UNIONE RADIOTAXI

REGISTRATI AL SERVIZIO DI RADIOTAXI!



Nome:

Cognome:

Data di nascita: 

INSERISCI I DATI DELLA TUA CARTA DI CREDITO

Numero di carta:







CVV:

Scadenza:



Il codice sottoposto ai tool di validazione è presente nelle pagine successive. Questo è stato sottoposto ai tool offerti dal W3C e anche a quello di WAVE; il riscontro ottenuto è stato questo:

Summary
Details
Reference
Structure
Contrast

 0 Errors	 0 Contrast Errors
 0 Alerts	 10 Features
 25 Structural Elements	 0 ARIA

Document checking completed. No errors or warnings to show.

Used the HTML parser. Externally specified character encoding was utf-8.
Total execution time 117 milliseconds.

Congratulazioni! Nessun errore trovato.

Questo documento è valido rispetto alla specifica [CSS versione 3 + SVG](#) !

```

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml" lang="it">
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
  <title>Registrazione</title>
</head>
<body style="background-color:#ffff66;">
  <form id="form1">
    <div style="text-align:center;">
      <header><h1>UNIONE RADIOTAXI</h1></header>
      <main id="Principale">
        <section><h2>REGISTRATI AL SERVIZIO DI RADIOTAXI!</h2>
          
        <table style="margin-left: auto; margin-right: auto; text-align:left;">
          <tr>
            <th><label for="txtNome">Nome:</label></th>
            <th> <input type="text" id="txtNome"></th>
          </tr>
          <tr>
            <th><label for="txtCognome">Cognome:</label></th>
            <th> <input type="text" id="txtCognome"></th>
          </tr>
          <tr>
            <th><label for="txtData">Data di nascita:</label></th>
            <th> <input type="date" id="txtData"></th>
          </tr>
          <tr>
            <th colspan="2"> <hr />INSERISCI I DATI DELLA TUA CARTA DI CREDITO</th>
          </tr>
          <tr>
            <th><label for="txtNum">Numero di carta:</label></th>
            <th> <input type="text" id="txtNum" MinLength="16" MaxLength="16"></th>
          </tr>
          <tr>
            <th><label for="txtCVV">CVV:</label></th>
            <th> <input type="text" id="txtCVV" ></th>
          </tr>
          <tr>
            <th><label for="txtScad">Scadenza:</label></th>
            <th> <input type="text" id="txtScad" ></th>
          </tr>
          <tr>
            <th colspan="2"> <hr />INSERISCI USERNAME E PASSWORD </th>
          </tr>
          <tr>
            <th><label for="txtUser">Username:</label></th>
            <th> <input type="text" id="txtUser"></th>
          </tr>
        </table>
      </main>
    </div>
  </form>

```

```

        <tr>
            <th><label for="txtPSW">Password:</label></th>
            <th> <input type="password" id="txtPSW"></th>
        </tr>
<tr style="text-align:center;">
<th colspan="2" ><br /> <input type="submit" id="plsVAI" value="REGISTRATI"></th>
</tr>
</table>
<br><br>
</section>
</main>
    <footer>
        <small>©2021 Ilaria Frandina</small>
    </footer>
    <p>
<a href="https://jigsaw.w3.org/css-validator/check/referer">
    
</a>
    </p>
</div>
</form>
</body>
</html>

```

■ L'analisi dei costi e dei benefici

SPESE		RISORSE	
Costi fissi inizio attività		Finanziamenti interni	
Costi commerciali e legali	5.000,00	Socio1	150.000,00
Costi promozione e marketing	3.000,00	Socio2	150.000,00
Creazione sito web	5.000,00	Socio3	150.000,00
Acquisto fabbricato	200.000,00	Totale	450.000,00
Acquisto mobile ed attrezzature d'ufficio	5.000,00	Finanziamenti esterni	
Acquisti software	3.000,00	Banca1	120.000,00
Rilascio licenza taxi	100.000,00	Banca2	106.000,00
Acquisto parco macchine	150.000,00	Totale	226.000,00
Totale	471.000,00		
Media di costi fissi annuali			
Costi per il personale	150.000,00		
Premi assicurativi e bollo auto	10.000,00		
Costi di manutenzione ordinaria	5.000,00		
Interessi passivi	2.000,00		
Utenze (telefono, luce, connettività ecc.)	2.000,00		
Totale	169.000,00		
Media costi variabili annuali			
Carburante	30.000,00		
Materiale di consumo	1.000,00		
Costi imprevisti	5.000,00		
Totale	36.000,00		
TOTALE SPESE	676.000,00	TOTALE RISORSE	676.000,00

Per concretizzare la mia idea di impresa, legata a questo progetto, ho deciso di redigere l'analisi dei costi e dei benefici.

L'analisi dei costi e dei benefici (ACB) è una tecnica di valutazione utilizzata per prevedere gli effetti di un progetto, verificando se con la sua realizzazione la società ottiene un beneficio o un costo netto. Questo, è importante sia per gli attori interni all'impresa, ma, in particolare, per quelli esterni come banche e finanziatori, i quali possono facilmente valutare la solvibilità dell'impresa.

L'analisi sopra rappresentata è divisa in due sezioni, spese e risorse.

Le spese riguardano tutti i costi che l'impresa deve sostenere, esse sono suddivise in:

- costi fissi di inizio attività, che rappresentano tutti i costi sostenuti per l'avvio dell'impresa e sono pluriennali, poiché non esauriscono la propria utilità nell'anno in cui vengono sostenuti ma hanno utilità nel lungo termine, avremo sicuramente tra questi i costi di impianto, di ricerca e sviluppo, acquisto parco auto, fabbricato, software e attrezzature varie, per un totale di 471.000,00 euro.
- media di costi fissi annuali, che comprendono tutti i costi annuali necessari allo svolgimento dell'attività di impresa, fissi perché vengono sostenuti indipendentemente dall'andamento economico dell'impresa; tra questi ricordiamo i costi assicurativi, le utenze, manutenzioni, costi del personale ecc. per un totale di 169.000,00 euro.

- media di costi variabili annuali, che prevedono tutti quei costi che variano al variare dell'andamento economico dell'impresa, tra cui il carburante, materiale di consumo ed eventuali spese impreviste, per un totale di 36.000,00 euro.

Nella sezione risorse, invece, ho specificato come e da chi l'attività di impresa verrà finanziata. Ho previsto due tipi di finanziamenti, quelli interni e quelli esterni.


I finanziamenti interni comprendono i fondi stanziati da tre soci ipotetici per un valore di 150.000,00 euro a testa; per un totale di 450.000,00 euro. Mentre, i finanziamenti esterni rappresentano i prestiti contratti con soggetti esterni all'impresa, nel caso specifico con due banche per un valore complessivo di 226.000,00 euro.

Tirando le somme per avviare questa attività, ho ipotizzato una spesa totale di 676.000,00 euro.














Per rappresentare anche le tempistiche del progetto di impresa e pianificare al meglio ogni cosa, ho realizzato due diagrammi di Gantt. Per farli ho utilizzato ProjectLibre, un software di gestione progettuale, che permette di pianificare al meglio l'idea di progetto, suddividendo quest'ultimo in più sottobiattivi. In automatico, questo software, genera anche la WBS del progetto, che riassume le fasi di quest'ultimo e offre una panoramica di queste.

Di seguito riporto i diagrammi di Gantt realizzati.

■ Il diagramma di Gantt della realizzazione del sito web

		Nome	Durata	Data di Avvio	Data di chiusura	Predecessori	Nome risorsa
1		Avvio	63 giorni?	11/05/21 9.00	06/08/21 9.00		
2		Sistema software	16 giorni?	11/05/21 9.00	02/06/21 9.00		
3		Documentazione requisiti	6 giorni?	11/05/21 9.00	19/05/21 9.00		Programmatore1
4		Documentazione sull'architettura	10 giorni?	19/05/21 9.00	02/06/21 9.00	3	Programmatore2
5		backed	7 giorni?	02/06/21 9.00	11/06/21 9.00		
6		Progettazione DB	7 giorni?	02/06/21 9.00	11/06/21 9.00	4	Programmatore1
7		front end	15 giorni?	11/06/21 9.00	02/07/21 9.00		
8		Template sito	15 giorni?	11/06/21 9.00	02/07/21 9.00	6	Programmatore2
9		Pagine Web	25 giorni?	02/07/21 9.00	06/08/21 9.00		
10		Pagine statiche	5 giorni?	02/07/21 9.00	09/07/21 9.00	8	Programmatore2
11		Pagine dinamiche	20 giorni?	09/07/21 9.00	06/08/21 9.00	10	Programmatore1

■ Il diagramma di Gantt della realizzazione dell'impresa

		Nome	Durata	Data di Avvio	Data di chiusura	Predecessori	Nome risorsa
1		Avvio	65 giorni?	07/05/21 8.00	05/08/21 17.00		
2		Analisi di mercato	18 giorni?	07/05/21 8.00	01/06/21 17.00		
3		Tipologie di finanziamento	8 giorni?	07/05/21 8.00	18/05/21 17.00		Socio1
4		Licenza e aspetti legali	10 giorni?	19/05/21 8.00	01/06/21 17.00	3	Socio2
5		Progettazione	15 giorni?	02/06/21 8.00	22/06/21 17.00		
6		Individuazione personale	15 giorni?	02/06/21 8.00	22/06/21 17.00	4	Socio1;Socio3
7		Individuazione fabbricato con parcheggio auto	10 giorni?	02/06/21 8.00	15/06/21 17.00	4	Socio2
8		Progettazione parco auto	10 giorni?	02/06/21 8.00	15/06/21 17.00	4	Socio1;Socio2
9		Marketing e sito web	15 giorni?	02/06/21 8.00	22/06/21 17.00	4	Socio3
10		Rilascio licenza	10 giorni?	02/06/21 8.00	15/06/21 17.00	4	Socio2;Socio3
11		Realizzazione	30 giorni?	16/06/21 8.00	27/07/21 17.00		
12		Acquisizione fabbricato con contratti	20 giorni?	16/06/21 8.00	13/07/21 17.00	7	Socio2;MaterialeUfficio;Disp...
13		Sito web	30 giorni?	16/06/21 8.00	27/07/21 17.00	10	Socio1
14		Acquisizione Parco auto	20 giorni?	16/06/21 8.00	13/07/21 17.00	8	Auto;Socio2
15		Chiusura progetto	17 giorni?	14/07/21 8.00	05/08/21 17.00		
16		Collaudo	7 giorni?	28/07/21 8.00	05/08/21 17.00	13	Socio3
17		Pratiche amministrative	15 giorni?	14/07/21 8.00	03/08/21 17.00	14	Socio2