Studente: Vasiliu Rares

Matricola: 300795

Corso di Programmazione e Modellazione a Oggetti

Progetto per la sessione estiva 2020/2021

Docente: Saverio Delpriori

Specifica del software

Il software consiste in un'app realizzata utilizzando il framework open-source Xamarin.Forms, il quale consente di sviluppare applicazioni Android, iOS e Windows attraverso la condivisione di codice C# . L'app è stata sviluppata per essere disponibile su dispositivi Android e, vista la natura del framework utilizzato, potrà in futuro essere distribuita su altri sistemi operativi.

L'app permette di vedere una lista di luoghi, quali ad esempio ristoranti, musei e hotel, presenti nelle proprie vicinanze impostando, o meno, un raggio di ricerca. Cliccando su uno dei luoghi lo si potrà vedere sulla mappa, con la possibilità di aprire Google Maps per utilizzare altre funzionalità, come visualizzare le recensioni o informazioni aggiuntive.

Studio del problema

Analizzando il problema il punto più critico è l'ottenimento dei luoghi intorno all'utente. Per ottenere tali dati è stata utilizzata un'API di Google chiamata Places API. Tale API permette di cercare luoghi all'interno di un'area specificata tramite delle richieste di tipo "Nearby Search", restituendo un elenco di luoghi e le relative informazioni.

Limite dell'API

Le richieste Nearby Search presentano diversi limiti:

- Restituiscono un massimo di 60 luoghi divisi in 3 pagine da 20 luoghi ciascuno;
- Tramite il parametro "type" è possibile limitare i risultati ai luoghi che corrispondono al tipo specificato, ma la richiesta consente di specificare un solo tipo;
- Non è possibile impostare il raggio di ricerca e ordinare i risultati per distanza.

Al fronte di tali limiti si è deciso di puntare più sulla quantità di tipologie di luoghi che l'utente può cercare, dividendoli per categorie, come "intrattenimento", "cibo", "negozi", …, le quali conterranno le tipologie più idonee. Le tipologie supportate sono presenti nella documentazione dell'API.

Per gestire l'ordine in cui vengono visualizzati i risultati sono stati utilizzati i seguenti componenti:

- Uno slider, il quale rappresenta il raggio di ricerca, per ottenere i luoghi ordinati per importanza
- Uno switch per disabilitare lo slider e ottenere i luoghi ordinati per distanza

Sebbene l'applicazione necessiti di una schermata in più, per la selezione della tipologia, l'utente potrà beneficiare di una maggiore scelta dei luoghi e di una ricerca più omogenea.

Risposte dell'API

L'output delle richieste sono in formato JSON e per la loro deserializzazione è stato utilizzato un pacchetto chiamato Newtonsoft.Json.

Non tutti i luoghi presenti nella risposta contengono gli stessi campi. Alcuni luoghi non contengono un'immagine rappresentativa, mentre altri non contengono una valutazione. Se un luogo non contiene il primo campo è stato deciso di mostrare l'icona rispettiva alla tipologia. Per la valutazione, invece, è stato introdotto un flag per indicare la necessità della valutazione, per luoghi come ospedali e parcheggi, la valutazione è stata omessa mentre nel caso in cui luoghi come ristoranti e negozi non presentino il campo riguardante la valutazione vengono scartati.

Scelte architetturali

Design pattern principale

Per lo sviluppo dell'applicazione è stato utilizzato il pattern Model–View–ViewModel (MVVM), il pattern più diffuso per lo sviluppo di app cross-platform attraverso Xamarin, in quanto permette di separare facilmente i componenti visivi dalla logica e dai modelli di dati, rendendo il progetto facilmente mantenibile e le componenti facili da riutilizzare.

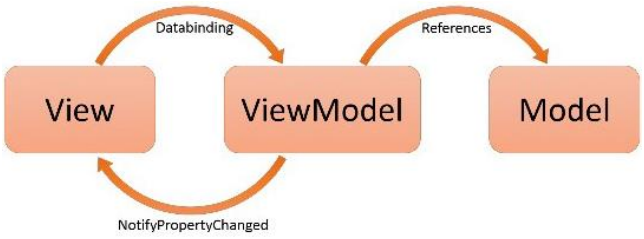
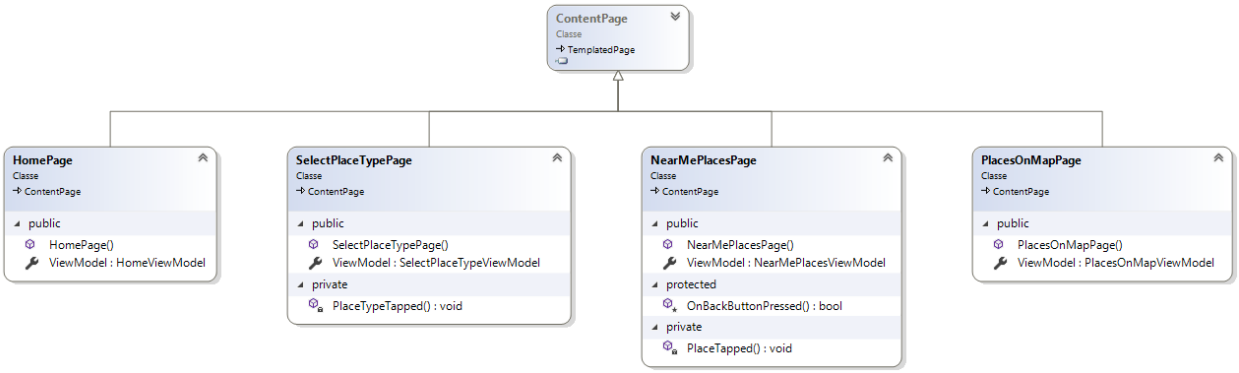
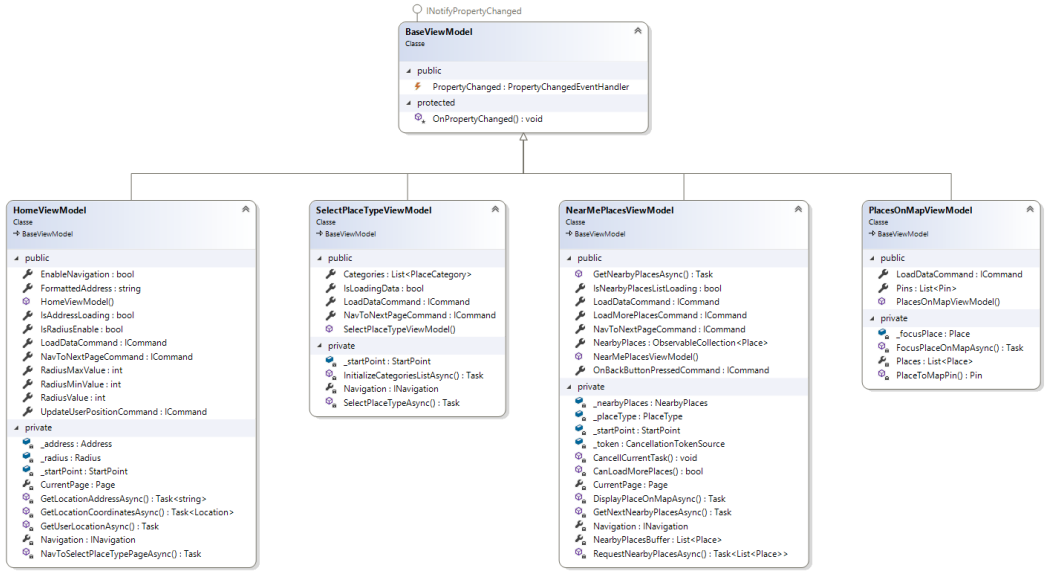


Diagramma delle classi

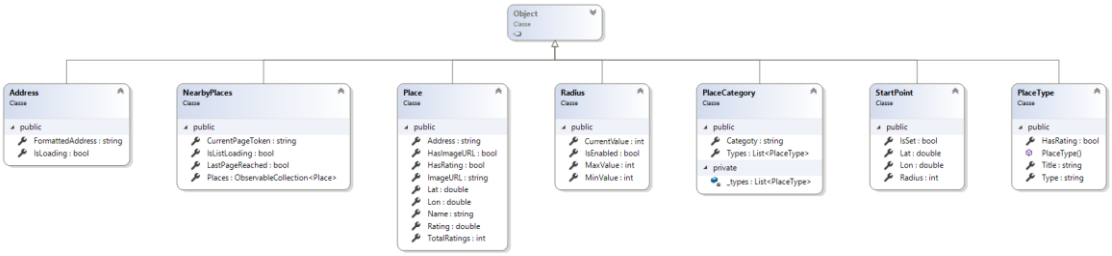
- **View:** le classi che si trovano nel livello visivo derivano dalla superclasse "ContentPage";



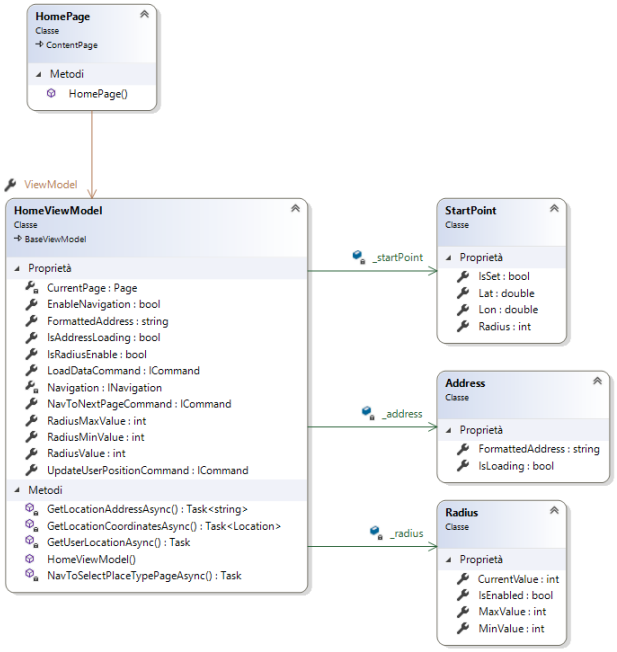
- **ViewModel:** le classi che si occupano della logica derivano dalla superclasse "BaseViewModel", la quale si occupa di avvisare il livello visivo dei cambiamenti dei dati;



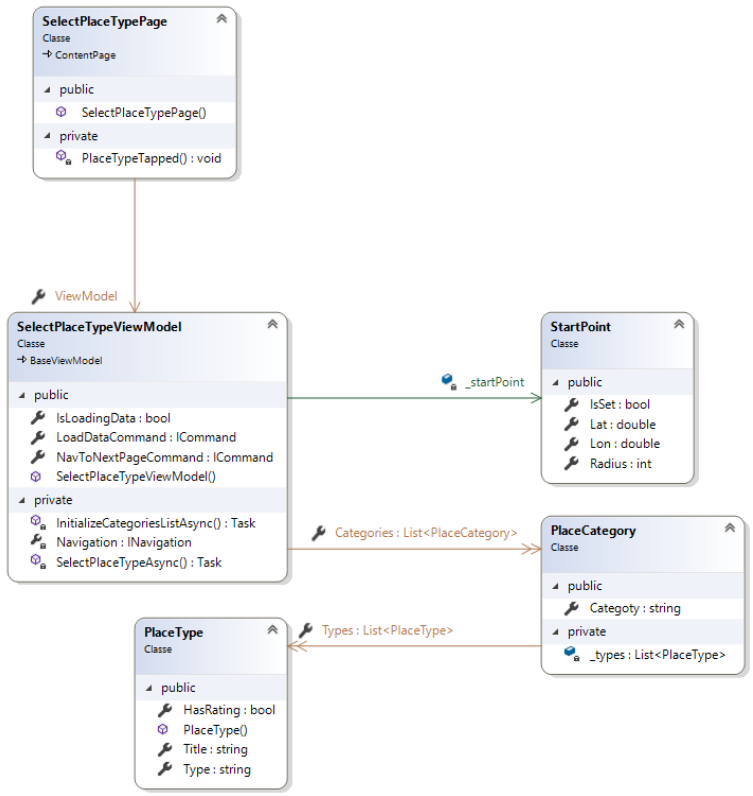
- **Model:** le classi che rappresentano i modelli derivano direttamente da Object;



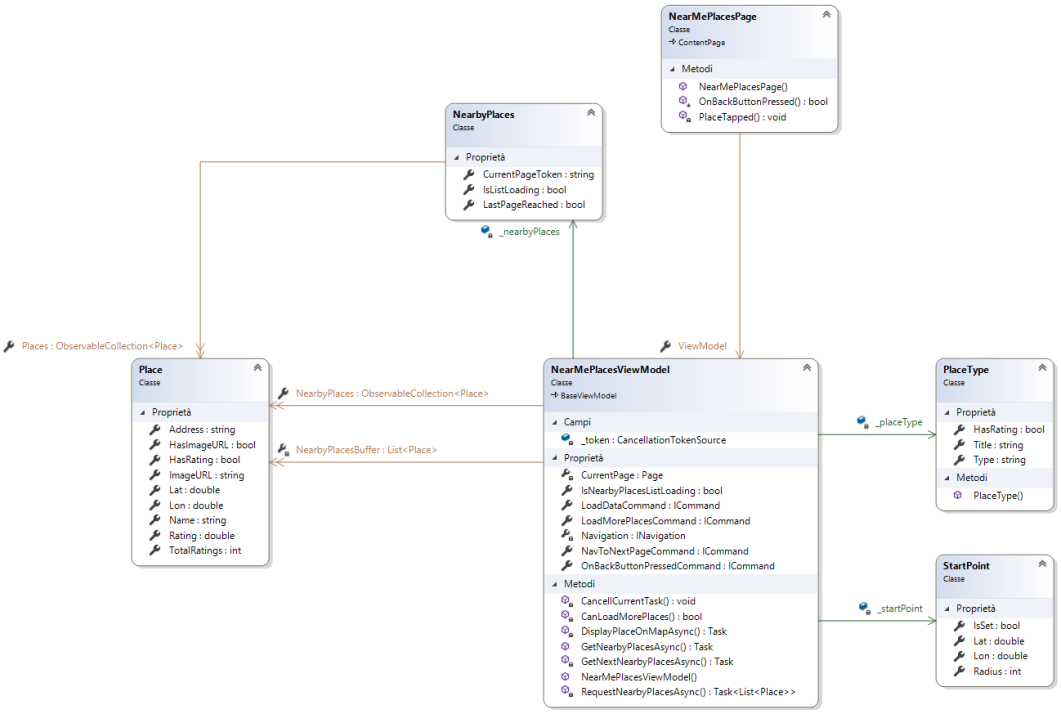
- **Home MVVM:**



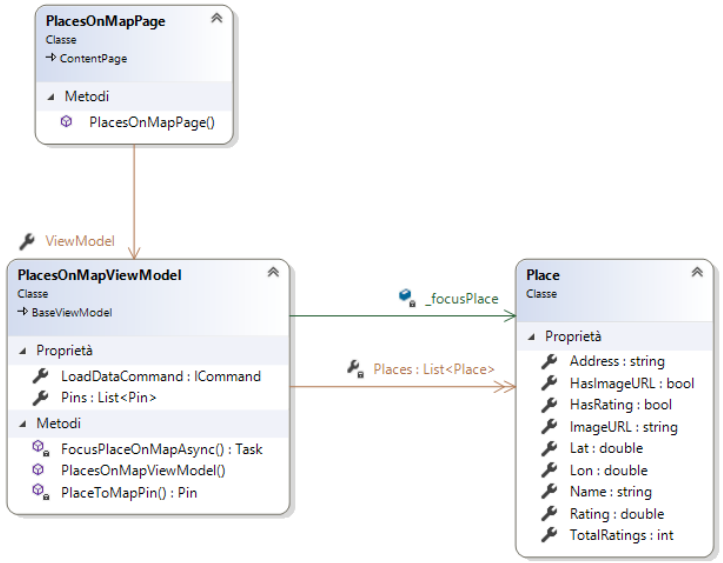
- **SelectPlaceType MVVM:**



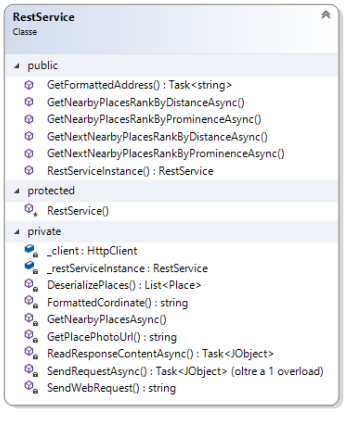
- **NearMePlaces MVVM:**



- **PlacesOnMap MVVM:**



- **RestService:** classe che si occupa di interagire con le API. Tale classe utilizza il pattern creazionale Singleton in quanto una sola istanza è necessaria per effettuare le richieste alle varie API utilizzate. L'operazione RestServiceInstance() permette l'accesso all'istanza.



Documentazione sull'utilizzo

Per eseguire il progetto è necessaria avere una Google API Key per l'utilizzo delle API e della mappa. Inoltre è necessaria una licenza per il pacchetto Syncfusion, il quale fornisce il componente grafico SfListView, utilizzato per la realizzazione di liste orizzontali e liste dotate di un miglior controllo.

La Google API Key e la licenza vanno inseriti nella classe "Secrets", presente nella cartella "Data".

Inoltre Google API Key va inserita anche nel AndroidManifest, presente nella cartella "Properties", nel parametro value del tag:

```
<meta-data android:name="com.google.android.geo.API_KEY" android:value="" />
```

Use Cases

Use Case: Avvio pagina Home

ID: UC1

Actor: A1 user, A2 servizio API

Preconditions:

- Deve essere presente una connessione internet
- Deve essere abilitata e attiva la geolocalizzazione

Basic course of events:

1. L'app è appena stata avviata da A1
2. Il sistema ottiene le coordinate geografiche, relative alla posizione corrente di A1
3. Il sistema chiede a A2 di convertire le coordinate geografiche in indirizzo
4. L'indirizzo in cui si trova l'utente viene mostrato nella pagina
5. A1 imposta la distanza di ricerca
6. A1 preme il bottone per la ricerca

Postconditions:

- Viene salvata la posizione attuale dell'utente sotto forma di coordinate geografiche
- Viene salvata la distanza di ricerca
- L'indirizzo in cui si trova l'utente viene mostrato nella pagina
- Viene aperta la pagina per la selezione della tipologia di luoghi da cercare

Alternative paths:

1. Se 2. o 3. falliscono viene visualizzata una finestra di dialogo di avviso a A1
2. A1 disabilita la distanza di ricerca

Use Case: Navigazione verso la pagina SelectPlaceType

ID: UC2

Actor: A1 user

Preconditions:

- A1 raggiunge la pagina tramite il bottone di ricerca della pagina Home

Basic course of events:

1. Viene inizializzata la lista con le categorie di luoghi
2. A1 seleziona una tipologia di luoghi dalla lista di categorie

Postconditions:

- Viene salvata la tipologia di luoghi da cercare
- Viene aperta la pagina contenente la lista di luoghi vicini a A1

Use Case: Navigazione verso la pagina NearMePlaces
ID: UC3
Actor: A1 user, A2 servizio API
Preconditions: • A1 raggiunge la pagina dopo aver selezionato un tipo dalla pagina SelectPlaceType • Deve essere presente una connessione internet
Basic course of events: 1. Il sistema chiede a A2 di cercare i luoghi nelle vicinanze di A1 2. La lista di luoghi viene visualizzata nella pagina 3. A1 scorre tutta la lista 4. Il sistema carica, se ci sono, nuovi luoghi nella lista 5. A1 seleziona un luogo
Postconditions: • I luoghi vicini a A1 vengono salvati e mostrati nella pagina • Viene aperta la pagina contenente la mappa con i pin relativi ai luoghi vicini a A1
Alternative paths: 1. Se 1. o 4. falliscono viene visualizzata una finestra di dialogo di avviso a A1

Use Case: Navigazione verso la pagina PlacesOnMap
ID: UC4
Actor: A1 user
Preconditions: • A1 raggiunge la pagina dopo aver selezionato un tipo dalla pagina NearMePlaces • Google Play Services e Google Maps devono essere installati nel dispositivo
Basic course of events: 1. Il sistema carica la mappa con i pin relativi ai luoghi vicini a A1 2. A1 naviga nella mappa e seleziona un pin 3. Il nome e l'indirizzo del luogo vengono visualizzati nella pagina 4. A1 preme il bottone per visualizzare il luogo su Google Maps
Postconditions: • Viene aperta l'app di Google Maps
Alternative paths: 1. Se Google Play Services non è installato verrà visualizzato un messaggio di errore 2. Se Google Maps non è installato viene visualizzata una notifica Toast che comunica l'errore