**Corso di Laurea : Informatica per il Management**

***Progetto del corso di  
Laboratori di Applicazioni Mobili***

**Roberto Mitugno   
0000970439***roberto.mitugno@studio.unibo.it*

**Docenti : Prof. Luciano Bononi, Prof. Federico Montori  
Tutor : Prof. Luca Sciullo**

**a.a. 2023/2024**

**Descrizione del progetto :**

Il progetto " Cellular Connectivity and Noise Map" è un'applicazione mobile che monitora la connettività cellulare e il rumore acustico all'interno di aree geografiche e colora la mappa in modo da ottenere una mappa termica discreta.   
L'applicazione monitora le frequenze di segnale LTE, WiFi e il rumore acustico. L'applicazione visualizza la posizione dell'utente sulla mappa e, man mano che l'utente si sposta, il valore della misurazione viene rappresentato sulla mappa sotto forma di colore associato all'area in cui si trova l'utente.   
L'applicazione può visualizzare tre mappe diverse (LTE, WiFi e rumore acustico). I colori della mappa rappresentano diverse condizioni di connettività o rumore acustico, rosso per "scarso", giallo per "medio" e verde per "buono".   
L'utente può scegliere la frequenza con cui vengono effettuate le misurazioni (attivamente o in background) e può impostare la durata minima tra due misurazioni. L'applicazione avvisa l'utente quando si trova in un'area in cui non sono state effettuate nuove misurazioni.

**Architettura generale :**

Il progetto è stato sviluppato su Android Studio con Java e richiede Android SDK versione 27 o successiva. Il dispositivo deve utilizzare un sistema operativo Android versione 8.0 (Oreo) o superiore e avere la capacità di connessione a internet per accedere ai servizi di localizzazione e mappatura.

Per quanto riguarda le librerie, sono state aggiunte diverse dipendenze, tra cui Room per la gestione del database, AppCompat per garantire la compatibilità con versioni precedenti di Android, Material Design per un'interfaccia utente coerente, ConstraintLayout per la gestione del layout, JUnit ed Espresso per i test, Google Maps e Google Location Services per la geolocalizzazione e WorkManager per la gestione dei lavori in background.

**Design e funzionalità :**

Immagine che contiene testo, mappa, schermata, atlante

Descrizione generata automaticamenteAll'avvio, l'applicazione visualizza una breve schermata di caricamento che dura pochi secondi. Una volta caricate tutte le funzionalità, viene visualizzata una mappa di Google Maps al centro della schermata.   
In alto, sono presenti tre pulsanti disposti orizzontalmente, che rappresentano le tre tipologie di misurazione: LTE, WiFi e rumore acustico.   
Subito sotto i pulsanti, è presente una barra di ricerca per cercare località sulla mappa.

Nella parte inferiore della schermata, è presente un testo che invita l'utente a selezionare una misurazione e un pulsante per accedere alla schermata delle impostazioni.   
Immediatamente sopra questi due elementi, sono presenti due pulsanti: un pulsante "play" a sinistra, che consente di acquisire una nuova misurazione per il segnale selezionato e un pulsante a destra, che consente di trovare la posizione corrente dell'utente.

Selezionata una delle tre misurazioni viene visualizzata la mappa corrispondente con la relativa griglia colorata.

Immagine che contiene testo, mappa, schermata, atlante

Descrizione generata automaticamente

In alto a destra viene visualizzata una legenda che indica il significato dei colori utilizzati per rappresentare i valori misurati.  
In basso alla mappa viene visualizzato il valore aggiornato ogni secondo della misurazione che è stata scelta.

*Colorazione mappa :*

Il colore della mappa viene determinato in base al valore misurato. In particolare, sono stati utilizzati i seguenti criteri:

* Immagine che contiene testo, schermata, Carattere, Elementi grafici

  Descrizione generata automaticamenteLTE:
  + Verde: valore maggiore a 3 dBm
  + Giallo: valore pari a 3 dBm
  + Rosso: valore inferiore a 3 dBm
* Immagine che contiene testo, schermata, Carattere, Elementi grafici

  Descrizione generata automaticamenteWiFi:
  + Verde: valore maggiore o uguale a 50 Mb/s
  + Giallo: valore compreso tra 30 e 50 Mb/s
  + Rosso: valore inferiore a 30 Mb/s
* Immagine che contiene testo, schermata, Carattere, design

  Descrizione generata automaticamenteRumore acustico:
  + Verde: livello inferiore o uguale a 30 dB(A)
  + Giallo: livello compreso tra 30 e 50 dB(A)
  + Rosso: livello superiore a 50 dB(A)

Tuttavia, il colore delle caselle della mappa viene in realtà determinato facendo una media degli ultimi *x* valori misurati. Questo comporta una colorazione non definita solamente dai tre colori principali, ma che dipende dalla media dei valori misurati.  
Se non sono ancora state prese misurazioni per la relativa zona la casella assume il colore di default, grigio.

***Visualizzazione della griglia :***

La visualizzazione della griglia sulla mappa consente di suddivide l'area in una serie di celle di dimensioni uniformi. Per visualizzare la griglia, lo zoom della mappa deve essere compreso tra 3 e 16. Questa scelta è stata fatta per garantire una visualizzazione delle misurazioni più accurata e discreta. **Utilizzando uno zoom inferiore a 3, la colorazione di una singola cella comprenderebbe un'area troppo ampia, rendendo difficile l'interpretazione dei dati. Al contrario, con uno zoom superiore a 16, la colorazione potrebbe non essere precisa a causa della tolleranza nella misurazione della posizione dell'utente**.

**La creazione della griglia viene effettuata mediante l'utilizzo dell'API di Google Maps, attraverso la libreria “TileProvider”, che consente di** gestire la visualizzazione della griglia sulla mappa, in modo da adattarla alle dimensioni e alla risoluzione dello schermo. Abbiamo quindi una rappresentazione delle singole porzioni di mappa, in base alle coordinate geografiche specificate.

La griglia viene suddivisa in celle quadrate di dimensioni uniformi. La grandezza di ciascun quadrato è definita da costante, misurata in pixel di densità (density pixels), adattandosi così alla risoluzione del dispositivo.   
Ogni quadrato è ulteriormente suddiviso in un numero fisso di celle, specificato dalla variabile ‘gridSize’, nel nostro caso impostata su 6. Pertanto, ogni quadrato è composto da una sottogriglia di dimensioni 6x6. La suddivisione viene eseguita all'interno del metodo ‘drawGridTile’ attraverso un doppio ciclo for, iterando sia sulle colonne che sulle righe della griglia.

*int tileSize = (int) (TILE\_SIZE\_DP \* scaleFactor);   
int cellSize = tileSize / gridSize;*

*...*

*// in every Tile, create a Grid [gridSize:gridSize]   
int zoomMin = 3;   
if (zoom > zoomMin) {   
 for (int row = 0; row < gridSize; row++) {   
 for (int col = 0; col < gridSize; col++) {   
 int color = Color.TRANSPARENT;   
 int xSubCell = x \* gridSize + col; int ySubCell = y \* gridSize + row;*

Ogni cella della griglia è individuata in base alla sua posizione all'interno del quadrato, utilizzando le coordinate x e y ricevute come input dai cicli for e il livello di zoom. Perciò ogni quadratino è individuato in modo univoco.

Le coordinate geografiche dell'utente vengono inizialmente convertite in coordinate metriche per la loro corretta rappresentazione sulla mappa. Ottenute le coordinate metriche dell'utente, la funzione ‘getSubTileByCoordinate’ determina il la casellina della griglia in cui si trova l'utente. Questo avviene in base ai seguenti criteri:

* pointX e pointY rappresentano le coordinate del punto,
* zoomLevel indica il livello di zoom sulla mappa.

*double convertedLat =* ***inMetersLatCoordinate****(currentLocation.getLatitude());   
double convertedLng =* ***inMetersLngCoordinate****(currentLocation.getLongitude());   
TileDataInfo myTile = GridTileProvider.****getSubTileByCoordinate****(convertedLng, convertedLat, zoom, 0);*

*public static TileDataInfo* ***getSubTileByCoordinate****(double pointX, double pointY, int zoomLevel, long date) {   
 double tileDim = MAP\_SIZE / Math.pow(2d, zoomLevel);   
 tileDim = tileDim / gridSize; int tileX = (int) ((pointX - TILES\_ORIGIN[0]) / tileDim);   
 int tileY = (int) ((TILES\_ORIGIN[1] - pointY) / tileDim);   
 return new TileDataInfo(tileX, tileY, zoomLevel, date);   
}*

*public static double* ***inMetersLatCoordinate****(double latitude) {   
 if (latitude < 0) {   
 return -inMetersLatCoordinate(-latitude);   
}*  
*public static double* ***inMetersLngCoordinate****(double longitude) {  
 return longitude \* ORIGIN\_SHIFT / 180.0;  
}*

La griglia viene sovrapposta alla mappa di Google Maps. Quando si seleziona una nuova misurazione, la griglia corrente viene sostituita da quella della misurazione selezionata.

***Misurazioni :***

L’applicazione consente di prendere le misurazioni attivamente, ovvero è l’utente che decide quando effettuare la misurazione. È possibile utilizzando il bottone ‘play’ presente nella parte inferiore sinistra della schermata principale.   
Una volta richiesta la misurazione, il valore della misurazione scelta viene ottenuto in modi diversi a seconda del tipo di misurazione :

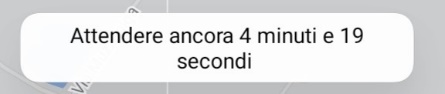
* LTE

Nel caso della misurazione del segnale LTE, il valore viene prelevato utilizzando il metodo getSignalStrength() dell'oggetto TelephonyManager. Se la versione del sistema operativo Android è uguale o superiore ad Android 9, il codice può utilizzare il metodo getLevel() per ottenere un valore compreso tra 0 e 4, che rappresenta la forza del segnale LTE. Se la versione del sistema operativo è inferiore a 9, il codice restituisce un valore di default di 0.

* WiFi  
  Per misurare il segnale WiFi, è possibile accedere alle informazioni sulla connessione Wi-Fi corrente utilizzando il metodo getConnectionInfo(). Questo metodo restituisce un'istanza della classe WifiInfo, che contiene diversi dettagli sulla connessione Wi-Fi, tra cui l'intensità del segnale (RSSI).  
  L'intensità del segnale Wi-Fi è rappresentata da un valore numerico chiamato RSSI (Received Signal Strength Indicator), che indica la potenza del segnale ricevuto dal dispositivo. Per convertire il valore RSSI in un valore compreso tra 0 e 100, è necessario aggiungere 127 al valore RSSI.
* Rumore  
  in questo caso è necessario consentire il permesso per l’accesso al microfono del dispositivo.  
  Una volta ottenuta l'autorizzazione, il codice utilizza il microfono del dispositivo per catturare le vibrazioni sonore nell'ambiente circostante. Queste vibrazioni sono convertite in un segnale elettrico attraverso il microfono. Il dispositivo elettronico utilizzato per elaborare questo segnale è denominato fonometro, il quale determina il livello di pressione sonora (SPL) in decibel (dB). Il valore del rumore acustico è calcolato su una scala compresa tra 0 e 160 dB.

Ottenuti i segnali relativi ad ogni misurazione, abbiamo un ulteriore controllo per verificare se l’ultima misurazione presa per quel quadratino è stata effettuata con un intervallo di tempo maggiore di *x*, dove *x* è una preferenza scelta dall'utente nelle impostazioni.

Se il tempo è maggiore di *x*, la misurazione viene inserita all’interno del database e viene mostrato a schermo un messaggio Toast che conferma l’avvenuta misurazione. In caso contrario viene mostrato a schermo un messaggio Toast che mostra i minuti e/o secondi rimanenti per poter effettuare una nuova misurazione.



Infine, viene aggiornata la mappa con la griglia risultante dalla misurazione presa dall’utente

È possibile ottenere le misurazioni anche in modo passivo, con l'opzione di configurazione disponibile nelle impostazioni dell'applicazione.

AGGIUNGERE FOTOOOOO IMPOSTAZIONI BACKGROUND MEASURE

Durante la creazione della classe MainActivity, viene istanziato un Periodic Worker. Questo componente rappresenta un elemento del sistema operativo Android che si attiva ciclicamente, nel nostro caso ogni 15 minuti. Inoltre, vi è applicato il vincolo di avere una batteria non scarica.

Il worker, nel contesto della nostra applicazione, verifica se l'applicazione è attualmente in primo piano (foreground), controlla se l'utente sta utilizzando l'app, e verifica se la cella in cui si trova l'utente non è stata ancora soggetta a misurazioni. Se la posizione è valida e non è stata ancora soggetta a misurazioni, il worker procede con l'effettuare la misurazione del segnale per LTE, WiFi e segnale acustico.

***Gestione database :***

La gestione del database è affidata alla libreria Room, un componente che offre un livello di astrazione superiore rispetto a SQLite. Questa libreria semplifica notevolmente la gestione dei dati locali.

Per garantire un'organizzazione strutturata del database, sono state create tre diverse entità, ciascuna dedicata a una specifica misurazione. L'utilizzo delle annotazioni, come @Entity, permette di definire il nome della tabella, i campi e le eventuali chiavi primarie. Nel nostro caso, le variabili sono comuni a tutte e tre le tipologie di misurazione e includono latitudine, longitudine, zoom, valore e data.

È essenziale che la classe astratta estenda RoomDatabase e contenga un metodo astratto per ottenere un'istanza del DAO (Data Access Object) associato al database.   
Il DAO, un'interfaccia che fornisce i metodi per accedere e manipolare i dati nella tabella del database. La sua implementazione generata automaticamente da Room. All'interno di questa classe, sono definite le query utilizzate per accedere ai dati nel database.

***Notifiche :***

Il processo di gestione delle notifiche è affidato al Periodic Worker, impiegato anche per le misurazioni passive. Pertanto, nel momento in cui viene creata la MainActivity, si procede con l'inizializzazione e l'avvio del worker periodico mediante l'utilizzo del WorkManager.

All'interno della classe BackgroundWorker, il metodo doWork() costituisce la fase iniziale per l'esecuzione di operazioni in background. Inizialmente, si verifica che non sia in corso alcuna attività nell'applicazione, si effettua un controllo sull'acquisizione dei permessi di posizione e si richiede la determinazione della posizione attuale del dispositivo.

Dopo aver ottenuto con successo le coordinate di posizione, si procede con una serie di verifiche finalizzate all'assenza di segnali LTE, Wi-Fi o rumore acustico in una specifica area. Se tali condizioni risultano soddisfatte, si attiva l'invio di una notifica all'utente attraverso l'invocazione del metodo sendNotification().

All'interno di quest'ultimo metodo, viene creato un canale di notifica (esclusivamente per versioni Android >= Oreo) e viene costruita la notifica mediante l'uso di NotificationCompat.Builder. Successivamente, la notifica viene visualizzata all'utente, consentendo la possibilità di aprire l'applicazione con un semplice clic grazie all'utilizzo di PendingIntent.

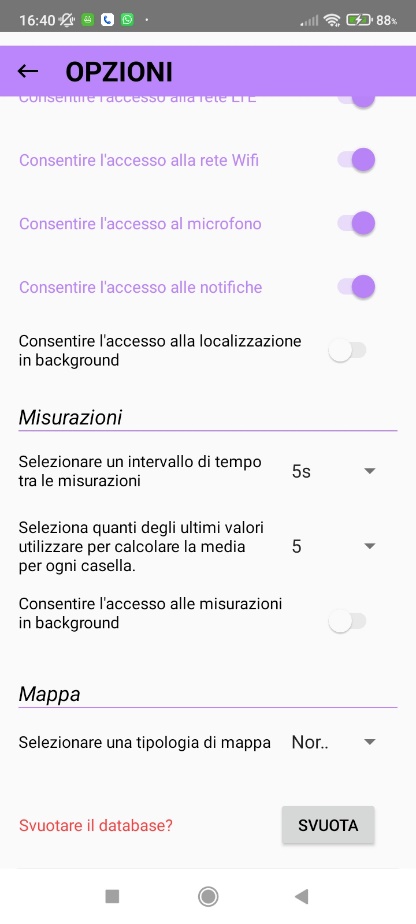
AGGIUNGI FOTO NOTIFICHEEEEE

Immagine che contiene testo, mappa, schermata, atlante

Descrizione generata automaticamente***Impostazioni :***

Per accedere alle impostazioni è necessario cliccare sul bottone in basso a destra col simbolino dell’ingranaggio che rimanda alla apposita pagina.

La schermata dell’applicazione si suddivide in 3 sezioni principali

Immagine che contiene testo, schermata, software, Pagina Web

Descrizione generata automaticamente

La prima sezione è dedicata alla gestione dei permessi dell'utente per accedere e fruire delle diverse funzionalità dell'applicazione. Successivamente, si affronta la configurazione delle impostazioni, derivanti dalle misurazioni fornite dall'utente, nonché il consenso per acquisire tali misurazioni in modo passivo. La terza sezione offre la possibilità di personalizzare la visualizzazione della mappa, consentendo di optare tra una modalità standard o satellitare.

Come opzione finale, troviamo la facoltà di svuotare il database. Questa funzionalità permette l'eliminazione completa e definitiva di tutti i dati associati alle misurazioni precedentemente acquisite.