Relazione Programmazione di Sistemi Mobili

Cardinali Rocco [331815], Vescovi Francesco Maria [331235]

Ingegneria Informatica, Elettronica e delle Telecomunicazioni

Contents

1 Requisiti

L'applicazione è stata progettata e testata utilizzando la API 34 e la versione di Android 14.0. L'applicazione è stata scritta in Android Studio Jellyfish 2023.3.1, con AGP 8.4.0

Per l'implementazione del server è stato utilizzato Flask, il quale utilizza il linguaggio Python ed alcune sue librerie per operare. Suddette librerie andranno appositamente installate tramite terminale, come poi affrontato nella sezione: "Guida all'installazione e all'utilizzo".

2 Scopo dell'applicazione

Il progetto si basa sul modello delle "Tracking App", ovvero applicazioni in grado di monitorare varie attività della vita quotidiana dell'utente che la utilizza. I parametri di cui è tenere traccia sono: il numero di passi percorsi durante il giorno, le ore di sonno e le pagine di libri lette. In merito alle ultime due attività, è prevista l'opzione di aggiungere una nota personale e, tramite un algoritmo di Natural Language Processing, si può ottenere un resoconto relativo alla qualità del sonno o a quanto siano state gradite le pagine lette. Ulteriore scopo dell'applicazione è quello di fornire dei grafici, ad istogramma, riassuntivi degli ultimi sette giorni, relativi alle singole attività.

3 Guida all'installazione e utilizzo

3.1 Installazioni necessarie

L'applicazione per essere utilizzata necessita dell'esecuzione un server Flask in Python, linguaggio che bisogna installare, il quale si occupa della generazione dei grafici e della Sentiment Analysis task dell'NLP. Gli script relativi al server sono situati nella cartella "Backend" del progetto. Per permettere il corretto funzionamento del server bisogna installare, tramite terminale, i seguenti pacchetti:

- pip install Flask
- pip install Flask-CORS
- pip install matplotlib
- pip install textblob

Per far funzionare correttamente l'app bisogna prima eseguire il server Flask, tramite apposito comando nel terminale bisogna spostarsi nella cartella "backend" e poi digitare il comando "python app.py" in modo da eseguire il file. Una volta che il server è in esecuzione, quindi pronto a ricevere le richieste del client, possiamo eseguire anche l'applicazione Android premendo apposito pulsante di "run".

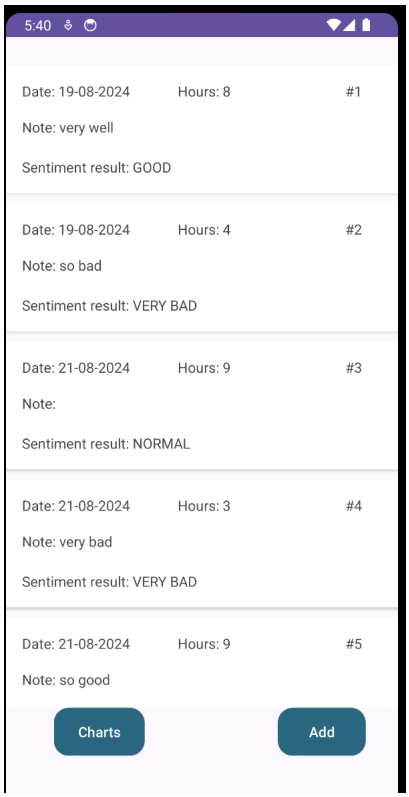
3.2 Utilizzo - Navigazione tra le pagine

3.2.1 Pagina iniziale

La pagina iniziale è composta da tre pulsanti, relativi alle 3 attività: "STEPS", "SLEEP", "BOOK PAGES". Cliccando su uno dei pulsanti sarà possibile visualizzare la pagina relativa ad esse.

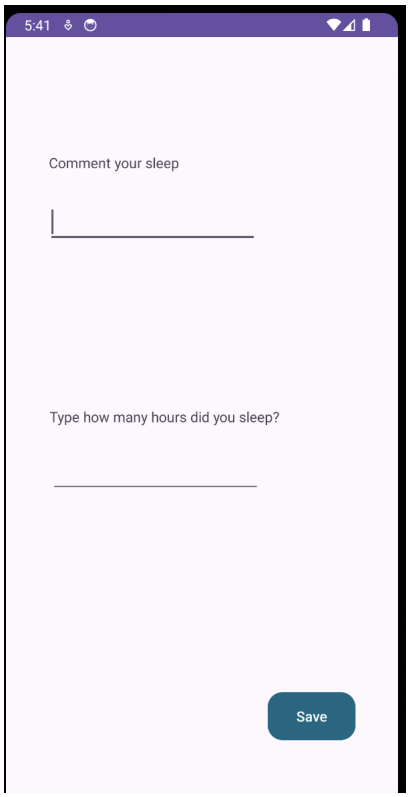
3.2.2 Pagine relative all'attività

Ogni pagina presenta una lista a scorrimento verticale, la quale contiene tutte le informazioni sulle note precedentemente scritte per l'attività selezionata. In basso sono presenti due pulsanti: il primo, posto a destra, ci direziona alla pagina apposita per inserire una nuova nota; sulla sinistra, invece, troviamo il bottone che ci indirizza nella pagina dedicata alla visualizzazione del grafico con andamento settimanale. La pagina relativa alle note dei passi non presenta il pulsante di aggiunta di una nuova nota, poiché l'inserimento dei dati nel database avviene in automatico.



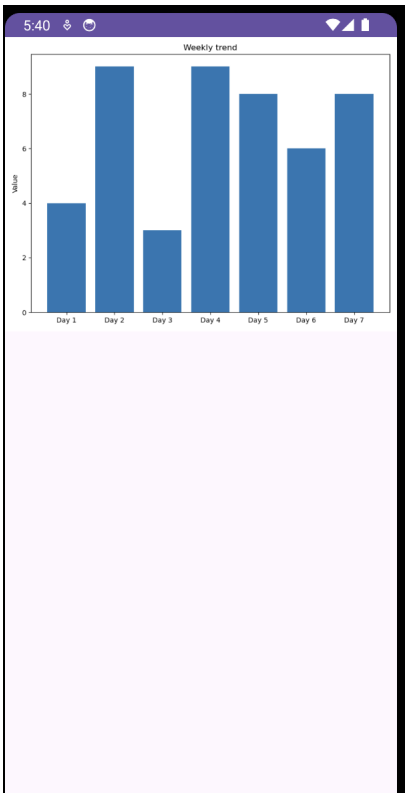
3.2.3 Pagina di aggiunta di una nuova nota

La pagina si presenta con due campi vuoti da compilare, rispettivamente una breve nota di commento all'attività, da dover scrivere in inglese, e la quantità numerica associata ad essa. Salvando la nota, essa viene inserita nel database e si viene riportati alla pagina iniziale.



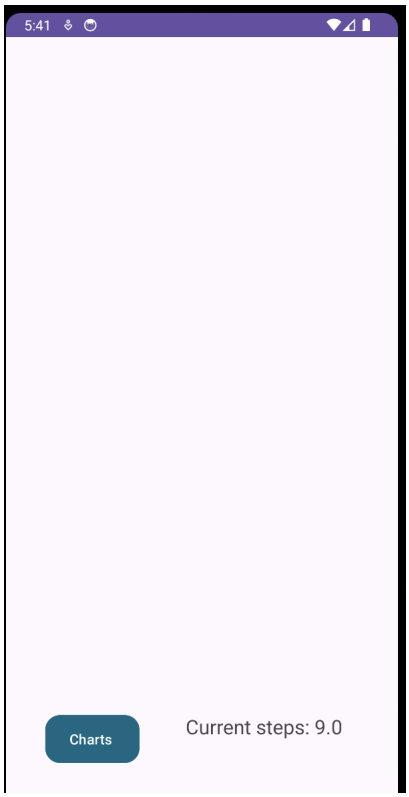
3.2.4 Pagina dei grafici

La pagina relativa alla visualizzazione dei grafici è comune per tutte e tre le attività, si compone unicamente di un istogramma che mostra l'andamento settimanale dell'attività selezionata.



3.3 Utilizzo - Conteggio passi

Per poter aumentare il numero dei passi bisogna andare in "Running devices", poi andare su "Extended Controls", "Virtual Sensors" ed infine su "Device Pose"; una volta in questa pagina si seleziona l'opzione "Move" e si muove il device che ci troviamo di fronte lungo l'asse X e Y per simulare il movimento eseguito dai passi.



4 Test effettuati

I test effettuati hanno lo scopo di identificare eventuali aree che potrebbero generare errori e causare problemi con il funzionamento dell'app. Sono stati effettuati i seguenti test: inserimento dati, comunicazione con il server e rappresentazione degli stessi.

4.1 Inserimento dati

Quando si preforma questo tipo di operazione bisogna far si che i dati siano corretti, che le variabili inserite siano del tipo corretto e che il tutto sia fatto in regime asincrono, essendo operazioni che possono richiedere tempo per essere impiegate.

Si prova a inserire dei dati di tipo "Sleep" all'interno del database:

4.2 Comunicazione HTTP

Quando si comunica con il server c'è la necessità di controllare che la comunicazione avvenga tramite il corretto indirizzo e, nel caso di metodi POST, è necessario controllare che i dati siano presenti e in formato corretto, per far si che vengano processasti correttamente dal server.

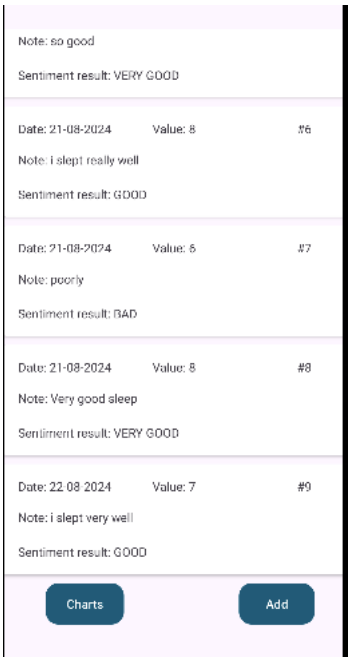
Riprendendo il test precedente, si può notare che i dati sono stati correttamente elaborati dal server:

```
{'note': 'i slept very well'}
0.2
127.0.0.1 - - [22/Aug/2024 09:45:49] "POST /api/sentiment_analysis HTTP/1.1" 200 -
```

4.3 Rappresentazione dati

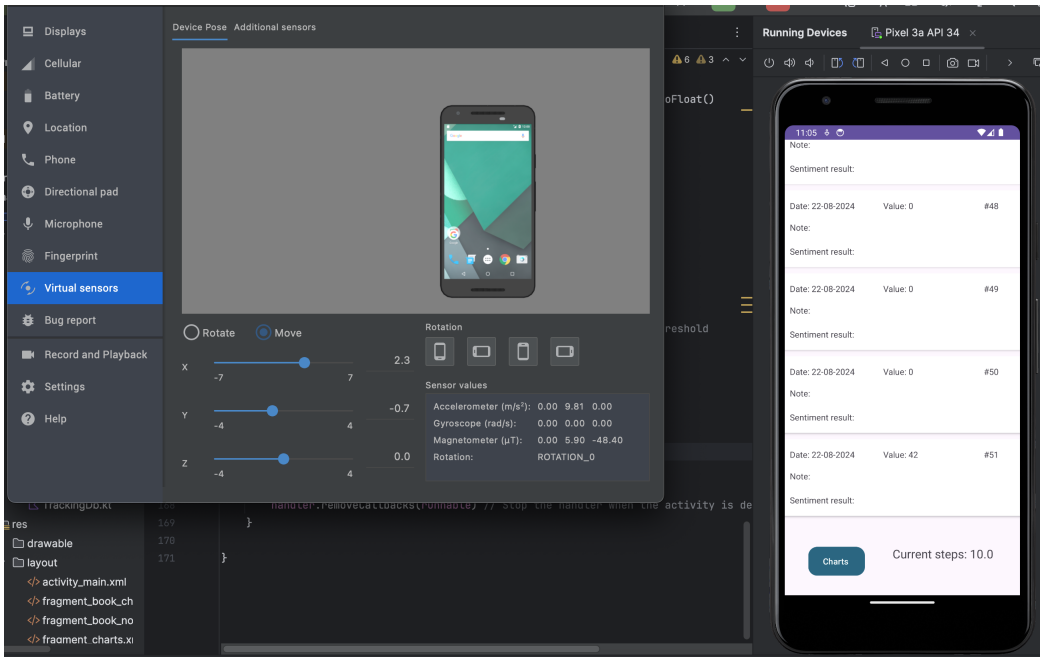
Al fine di avere una visualizzazione corretta dei dati bisogna assicurarsi che i dati vengano presi correttamente dal database, siano completi e che tutte le operazioni avvengano in maniera coerente e asincrona in modo da non bloccare altre porzioni di codice che potrebbero compromettere il funzionamento corretto dell'applicazione.

I dati relativi alla nota inserita precedentemente sono ora visualizzabili nell'apposita lista:



4.4 Conteggio passi

Per testare il movimento dei passi abbiamo utilizzato gli "Extended controls" citati nella sezione "Guida all'installazione e all'utilizzo: Utilizzo - Conteggio passi". Una volta accertato che i movimenti del dispositivo vengono memorizzati dall'applicazione, è bastato cambiare temporaneamente l'orario relativo al caricamento dei dati nel database, dalla mezzanotte a circa uno o due minuti dopo l'avvio dell'app. Una volta trascorso l'orario impostato manualmente, i dati sono stati caricati correttamente nel database e sono stati resi visibili all'interno della RecyclerView.



4.5 Gestione degli errori

Tutti i possibili errori vengono gestiti tramite blocchi di codice Try-Catch per rilevare eventuali problematiche e notificare l'utente dell'accaduto tramite brevi pop-up.

5 Tecnologie esterne

5.1 Flask

La tecnologia esterna utilizzata è quella di un server esterno scritto in Flask Python. Per generare il grafico e inviarlo al client si usa una libreria Python chiamata Matplotlib, mentre per eseguire lo script di Sentiment Analysis si fa uso della libreria TextBlob; entrambe le librerie, al fine di essere utilizzate nel codice, vanno precedentemente installate con apposito comando "pip" tramite terminale.

6 Architettura software

6.1 Grafica e navigazione tra pagine

L'interfaccia dell'applicazione è basata sull'utilizzo di "Fragment", collegati tramite un Navigation Graph. Per ogni attività ci sono tre fragment dedicati, due nel caso dei passi, più pagina iniziale.

Gli spostamenti tra le pagine avvengono tramite la pressione degli appositi bottoni, mentre per tornare alla schermata precedente si possono usare le gesture del dispositivo o il relativo tasto "indietro" presente su Android. [Unica eccezione avviene quando nella pagina per aggiungere una nota al database si preme sul pulsante di aggiunta, tramite il quale si viene riportati automaticamente alla pagina di selezione iniziale].

6.2 Database

I dati immessi vengono memorizzati e immagazzinati all'interno di un database locale, accessibile tramite un DAO. Il database contiene la data class "Tracking", avente come parametri:

- trackingType: tipo dell'attività ("Steps", "Sleep", "Book") di tipo stringa e non nullable.
- note: nota da aggiungere all'attività, sulla quale viene effettuata la Sentiment Analysis; di tipo stringa e nullable poichè potrebbe essere assente (sia perché utente non vuole inserirla, sia per la pagina dedicata ai passi).
- number: valore numerico intero relativo all'attività, ovvero il numero di passi, ore di sonno o pagine lette, anch'esso non nullable.
- sentimentResult: valore numerico assegnato dall'algoritmo di Sentiment Analysis; di tipo double e nullable.
- date: la data in cui è stato inserito l'oggetto nel database, ottenuta tramite codice durante l'invio delle informazioni al database.

6.3 Comunicazione HTTP

Il server comunica tramite protocollo HTTP ricevendo le request da parte del client, applicazione Android, e mandando una response legata al tipo di richiesta fatta. Il server si compone di due API, entrambe di tipo POST, ovvero che ricevono dati dal client per poi mandare una risposta, e servono rispettivamente per: generare il grafico settimanale dell'attività scelta, selezionando gli ultimi 7 dati del database; elaborare e inviare il risultato della Sentiment Analysis effettuato sulla nota testuale inviata.

```
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with watchdog (fsevents)
 * Debugger is active!
 * Debugger PIN: 799-772-467
Received data: {'numeric_value': [8, 4, 9, 3, 9, 8, 6, 8]}
Processed chart data: [4, 9, 3, 9, 8, 6, 8]
127.0.0.1 - - [21/Aug/2024 17:40:40] "POST /api/receive_charts HTTP/1.1" 200 -
```

6.3.1 Generazione grafici

Nella funzione dedicata alla generazione dei grafici per prima cosa si ricevono i dati dal client, un oggetto JSON con chiave "numeric_value" e un array che contiene tutti i valori salvati nel database. Una volta ricevuti i dati bisogna andare a selezionare solo gli ultimi sette valori, dopodiché possiamo andare a creare la figura che conterrà i nostri dati, nominiamo gli assi e andiamo a generare effettivamente il nostro grafico adattandolo alla dimensione dei bordi. Passando attraverso un buffer di byte traduciamo il grafico in una immagine, in formato .png, che poi inviamo al client tramite apposito metodo "send_file" come response.

6.3.2 Sentiment Analysis

Il server utilizza la libreria TextBlob, un'API specializzata nel Natural Language Processing. La libreria ottiene come input una stringa di testo e, analizzando le parole contenute in essa, restituisce un valore double "polarity" compreso nel range [-1, 1]. Il dato numerico viene poi interpretato dall'applicazione, convertendolo in una stringa all'interno della RecyclerView che può variare da "VERY BAD" a "VERY GOOD".

6.4 Contapassi

L'applicazione si avvale dell'accelerometro presente nel dispositivo per memorizzare e aggiornare i passi percorsi dall'utente tramite la funzione onSensorChanged, visualizzando i passi attuali all'interno della schermata relativa all'attività. L'applicazione si occupa inoltre di controllare l'orario del dispositivo e, una volta giunta la mezzanotte, invia i passi raccolti nelle 24 ore al database, rendendone possibile la visualizzazione nella RecyclerView.

7 Suddivisione dei ruoli

Il codice Kotlin per l'applicazione è stato scritto tutto in collaborazione. Cardinali Rocco ha posto maggiore attenzione per quanto concerne l'implementazione e l'utilizzo del sensore (accelerometro) legato al rilevamento dei passi. Vescovi Francesco Maria ha posto maggiore attenzione per quanto concerne l'implementazione e la gestione del server Flask.