Laboratorio di Applicazioni Mobili
Informatica, Università di Bologna

Matteo Celani

2019-2020

Docente:	Luciano Bononi
Docente:	Federico Montori
Tutor didattico:	Luca Sciullo
Mail:	matteo.celani@studio.unibo.it
Matricola:	0000804303

GitHub: www.github.com/matteocelani/PersonalHealthMonitor

Indice

1 Personal health monitor

Nel seguente progetto lo studente è tenuto a implementare un'applicazione interattiva per tenere traccia delle informazioni personali sulla propria salute da salvare nei report giornalieri. In particolare, l'applicazione dovrebbe essere in grado di gestire i report all'interno di un calendario, inviare notifiche e tracciare i dati in base a filtri specifici.

1.1 Creare e modificare report

L'applicazione deve essere in grado di creare, modificare ed eliminare i rapporti sulla salute. I rapporti sulla salute sono riassunti delle informazioni sulla salute tracciati dall'utente che devono essere salvati ogni volta che l'utente ritiene che sia una buona idea e almeno una volta al giorno. Ogni rapporto deve includere un numero minimo di due informazioni relative alla salute dell'utente (ad es. Temperatura corporea, pressione sanguigna, indice glicemico, ecc.). Ogni informazione ha un'importanza, cioè un indice che specifica il livello di attenzione che richiede tale parametro (da 1 a 5) e ogni rapporto ha una nota opzionale che può essere riempita con informazioni ausiliarie. I report vengono archiviati dall'applicazione (si consiglia vivamente di utilizzare un database) e deve esserci la possibilità di mostrare report su base giornaliera, come ad esempio all'interno di un calendario. Nel caso di più report per lo stesso giorno, è necessario creare un report di riepilogo, in cui le informazioni sulla salute sono la media di tutti i dati raccolti di quel giorno. Ci deve essere anche la possibilità di visualizzare i report in base ad alcuni filtri (ad esempio, solo i report con importanza impostata su 5).

1.2 Notifiche

L'applicazione deve avvisare l'utente se non ha ancora inserito un rapporto per quel giorno. In questo caso, l'utente può eseguire le seguenti azioni: rinviare il promemoria (in tal caso all'utente verrà richiesta un'altra ora e data dello stesso giorno) o aprire direttamente dall'interno della notifica il modulo per la compilazione del rapporto. Il tempo in cui la notifica viene inviata dall'applicazione può essere impostato dall'utente da una pagina delle impostazioni. L'applicazione deve inoltre informare l'utente se la media dei dati raccolti per un'informazione - con importanza maggiore di 3 - in un determinato periodo di tempo ha superato una soglia predefinita. L'utente può personalizzare i parametri precedenti da una pagina delle impostazioni, ovvero può decidere quali informazioni devono essere monitorate, per quanto tempo e quale soglia non deve essere raggiunta.

1.3 Grafici

L'applicazione dovrebbe essere in grado di raccogliere statistiche sull'utilizzo che visualizzano almeno due grafici di qualsiasi tipo (grafico a torta, diagramma a riquadri, istogramma, grafico a linee, ecc.) Che mostrano dati utili (ad esempio la variazione di informazioni sanitarie nell'arco di una settimana, la variazione di il numero di rapporti raccolti ogni giorno, ecc.).

2 HealtMonitor per iOS

Questa versione di HealtMonitor è stata pensata e sviluppa per **iOS 13** usando **Swift 5** su **xCode 11**.
L'app è divisa in 3 sezioni principali richiamate nel ContentView.swift :

- `Summary()`
- `CalendarTab()`
- `AddReport()`

Ogni view viene invocata tramite una `TabView`, in *"Riepilogo"* abbiamo i grafici dei valori inseriti e la lista di tutti i report, in *"Calendario"* abbiamo un calendario organizzato per mesi dove si può controllare i giorni in cui si è inserito il report, infine possiamo aggiungere un nuovo report nell'ultima tab *"Nuovo"*.

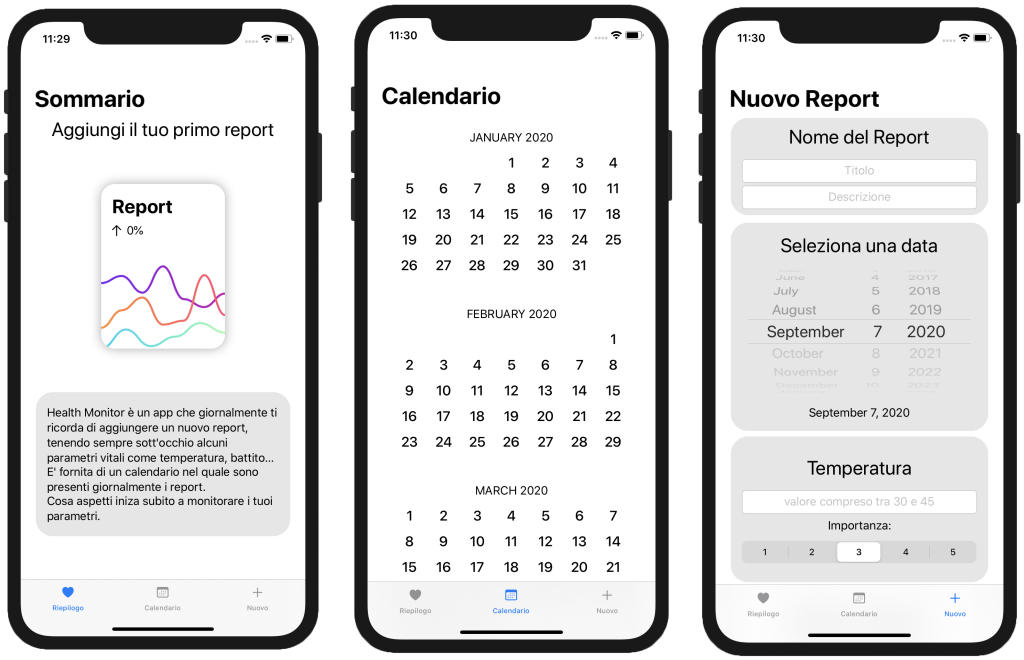

Figura 1: Schermate Iniziali

2.1 I Report

Nella nostra applicazione iOS è possibile aggiungere, modificare ed eliminare dei report giornalieri.
I report sono composti da 4 valori e ogni valore ha un importanza che va da 1 a 5. In particolare i valori da inserire possono essere:

- temperatura

- battito cardiaco
- glicemia
- frequenza respiratoria

Inoltre ogni report è caratterizzato da una data, un titolo (obbligatori) e una descrizione (facoltativa).

3 Riepilogo

In riepilogo vengono generati i grafici relativi ai dati inseriti giorno per giorno inoltre è presenta una lista dove si possono consultare tutti i Report, essi possono essere visualizzati tramite alcuni filtri, possono essere modificati ed eliminati.
Se non è presente nessun dato viene caricata una schermata iniziale di benvenuto.

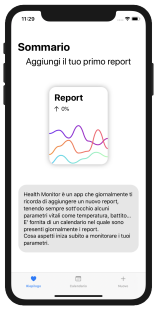

Figura 2: Schermata di benvenuto

3.1 Grafici

Quando si inseriscono i dati, si iniziano a generare 4 grafici uno per ogni valore presente nel Report.
I grafici vengono generati usando un pacchetto Swift dal nome ***SwiftUICharts***, esso ha diverse configurazioni, in particolare in HealthMonitor troviamo `LineChartView()` che prende in input array di dati.
I valori dei grafici sono ordinati per data, basta trascinare il dito sopra al grafico per visualizzare i valori.

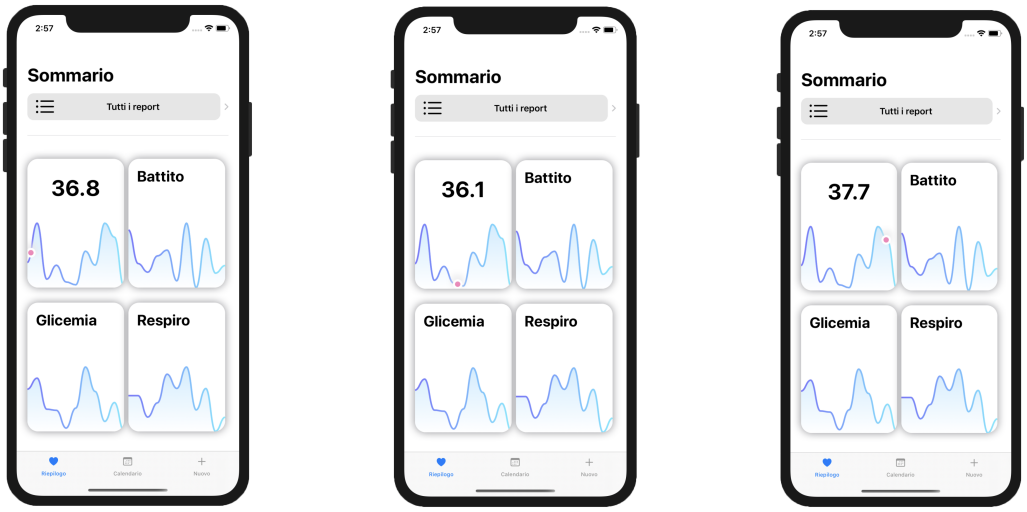

Figura 3: Grafici con valori

3.2 Tutti i Report

Nella schermata iniziale è presente un `NavigationLink` che ti permette di spostarti tra le View, toccandoci sopra si caricherà `AllReport()` nel quale troviamo una lista completa dei Report.

La lista completa dei report in `AllReport()` è ordinata per data all'interno di una `List` nel quale ritroviamo i `NavigationLink` che ci permettono di accedere ai dettagli dei report, inoltre si possono cancellare i report trascinando l'elemento verso sinistra.
I report possono essere visualizzati anche tramite dei filtri, che calcolano la media dell'importanza dei 4 valori inseriti e restituisce una lita più corta. La funzione che si occupa del filtraggio dei report è definita nel seguente modo:

```
private func filterReport() -> [FetchedResults<Report>.Element] {
    return self.reports.filter({
        avgImp(tempImp: $0.tempImportance, heartImp: $0.heartImportance, glyImp
            : $0.glycemiaImportance, breImpo: $0.breathImportance) >= self.
            avgImportance
    })
}

func avgImp(tempImp: Int16, heartImp: Int16, glyImp: Int16, breImpo: Int16) ->
     Int16 {
    let avgImp : Int16 = (tempImp + heartImp + glyImp + breImpo)/4
    return avgImp
}
```

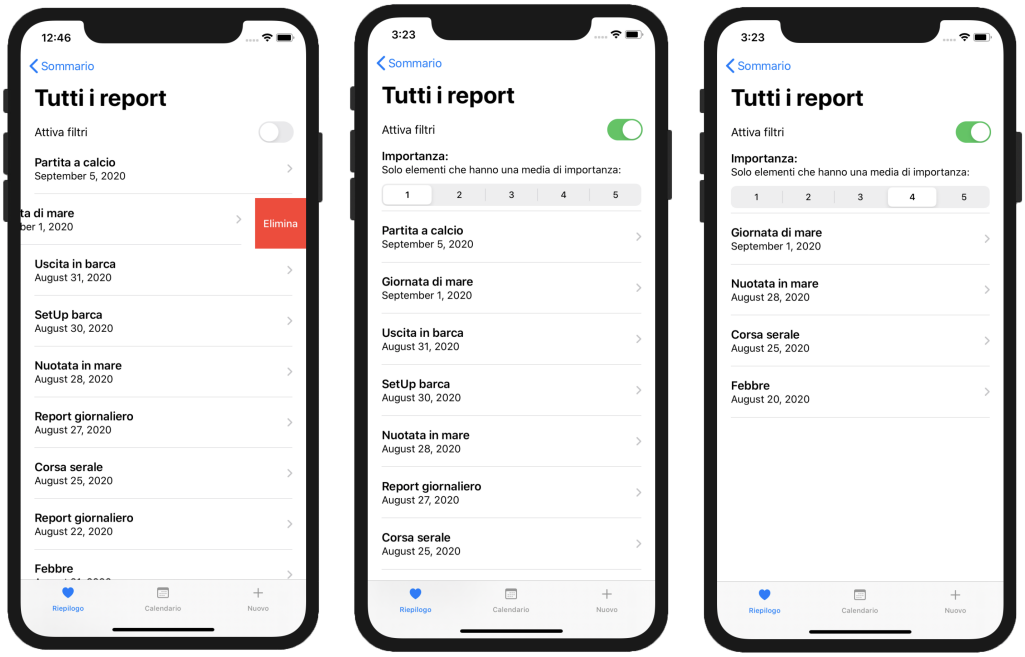

Figura 4: Lista Report

Toccando un oggetto della lista verrà caricata un nuova view `ReportView()` nel quale troviamo i dettagli del report selezionato. In `ReportView()` è possibile modificare il report premendo sul bottone "Modifica il Report", esso caricherà a schermo una nuova `sheet` e prendendo in input tutti i dati del report corrente verrà mostrato `EditViewSheet()`. Tramite questa `sheet` è possibile salvare le modifiche con un bottone a fine pagina, se invece si cambia idea basterà trascinare in basso la `sheet` o toccare il bottone "Chiudi".

Figura 5: Dettagli del report/Modifica del report/Report modificato

4 Calendario

Al centro delle tre `TabView` troviamo il Calendario che, stilisticamente parlando, è l'oggetto più complesso dell'app.
Tramite il calendario possiamo aggiungere eliminare e modificare i report. Inoltre la grafica colorata del calendario ci permette di comprendere i giorni che presentano i report e i giorni senza report.

La grafica del calendario è stata costruita sulla base di ***RKCalendar***, codice è stato ripulito ed adattato per lo sviluppo di HealthMonitor.
Il Calendario è invocato tramite
`CalendarController(reports: self.reports, CalendarManager: self.CalManager)`
prendendo in input i report salvati e una classe nel quale sono state inseriti e la data di inizio, di fine calendario ed il tipo di calendario.
In CalendarController per ogni mese presente viene richiamata la view `CalendarMonth()` che è il cuore del nostro calendario, essa non solo si occupa di costruire graficamente il calendario, ma si occupa anche di controllare i giorni e quindi segnalare il giorno corrente, se è presente un report ed il giorno selezionato. I giorni verranno colorati in base a determinate proprietà:

- nessun colore → nessun report presente → è possibile aggiungere un nuovo report
- giallo → report presente → è possibile visualizzare, modificare ed eliminare il report
- rosso → giorno corrente
- verde → giorno selezionato

5 Nuovo

Aggiunta nuovi report

6 Notifiche

Notifiche