Relazione progetto  
Laboratorio avanzato di programmazione II  
Simone Guarnuccio X81000143

**Nome Progetto**: AppQuest

Lo scopo del progetto vèrge allo sviluppo di un applicazione mobile cross platform (Android e IOS) attraverso l’uso del framework React-Native basato su Javascript.  
L’applicazione da realizzare deve somministrare e

raccogliere risposte di questionari precompilati e disponibili in

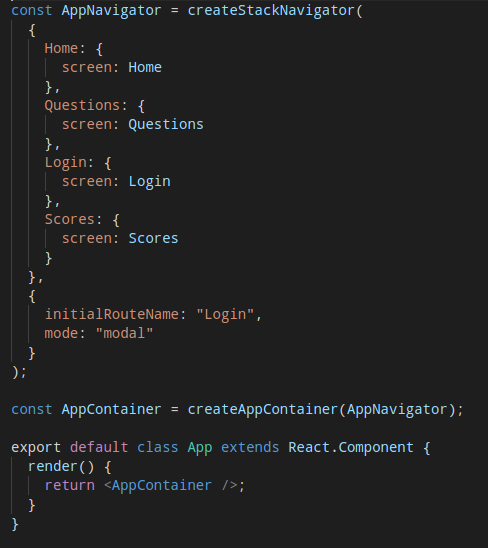
formato json su un server, che nel nostro caso scriviamo e/o carichiamo sul servizio di hosting Google Firebase. E’ sempre su questo che vengono caricati e aggiornati i file json relativi ai risultati del test scelto e che viene garantita e utilizzata l’autenticazione univoca dell’utente.  
Il link al progetto firebase è il seguente: <https://console.firebase.google.com/u/0/project/esempio-ae701/database/esempio-ae701/data>

L’applicazione è principalmente costituita da 5 file di tipo javascript:

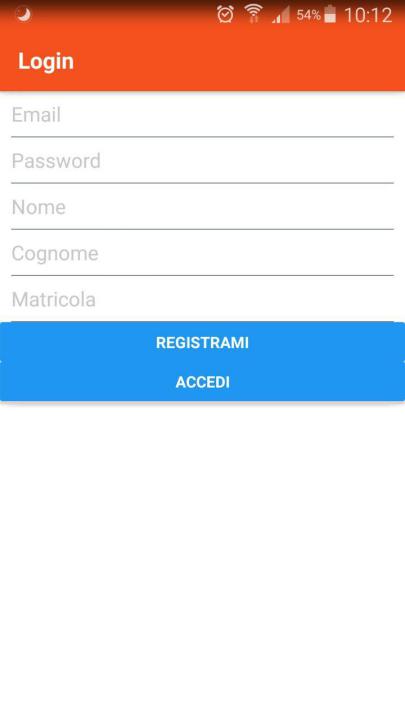
* 4 file corrispondenti a 4 screens (Login, Home, Question, Scores) di cui e’ composta l’app.
* 1 file {App} corrispondente al principale contenitore delle screen precedenti.

Procederò nella descrizione di quanto realizzato partendo dal file App.js e proseguirò poi nella descrizione delle varie screen seguendo la stessa logica di navigazione sequenziale reale dell’applicazione allo stato attuale.

***App***

File javascript che si carica principalmente dell’integrazione della libreria “React-Navigation” utile a permettere la navigazione tra screen diverse basandosi su una logica strutturale di pila. La createStackNavigator fornisce un modo per collegare logicamente le varie screen tra le quali l’applicazione potrà navigare e stabilisce anche quale screen sarà quello iniziale all’avvio dell’app.  
La createAppContainer crea il componente principale per il render dello screen da visualizzare.

***Login***



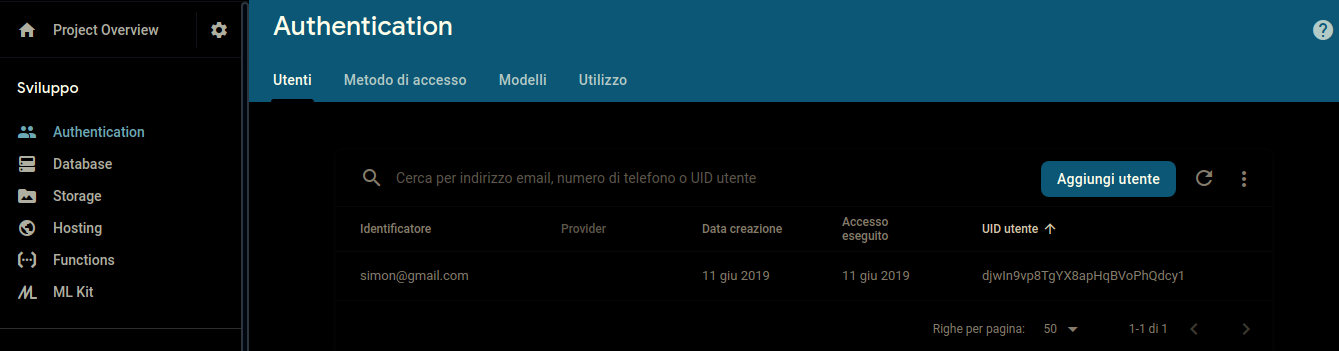
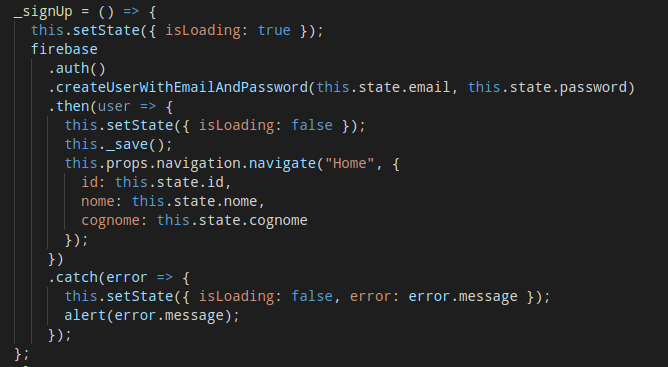
All’avvio dell’applicazione l’utente è indirizzato principalmente alla screen “Login” nella quale potere inserire i dati attraverso l’uso di Input dedicate alla modifica dello stato del componente per salvare dinamicamente i dati inseriti (Email e Password sono campi obbligatori) e successivamente scegliere tra due componenti Button per l’accesso o registrazione. I tre metodi principali di funzionamento di questa screen sono:

* \_signUp();
* \_logIn();
* \_save();

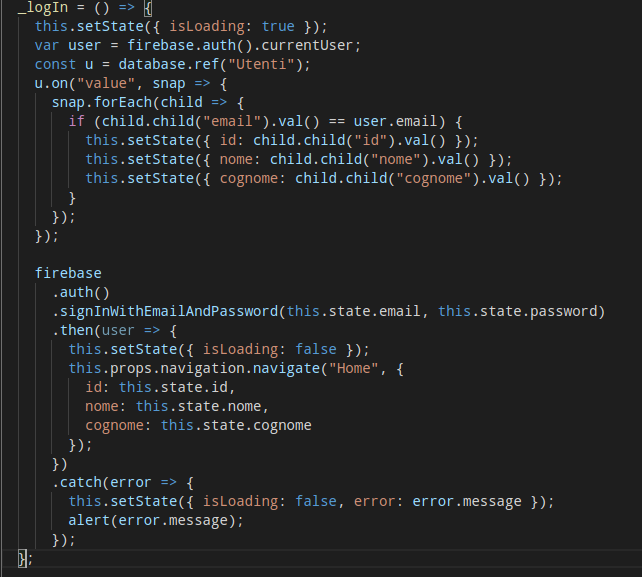
**\_save**

Metodo necessario a creare direttamente un oggetto utente partendo dai valori dello “state”, già modificati all’inserimento sui componenti input della screen.  
L’oggetto “utenti” nello state punta e chiama automaticamente una push di inserimento con id univoco al database di farebase sotto il nodo padre Utenti che se non ancora presente viene creato. La \_save memorizza prima di tutto la chiave univoca creata dalla push e creato l’oggetto newUser questo viene scritto nel nodo relativo alla chiave univoca attraverso la funzione .set() che permette la scrittura su uno specifico nodo del database.

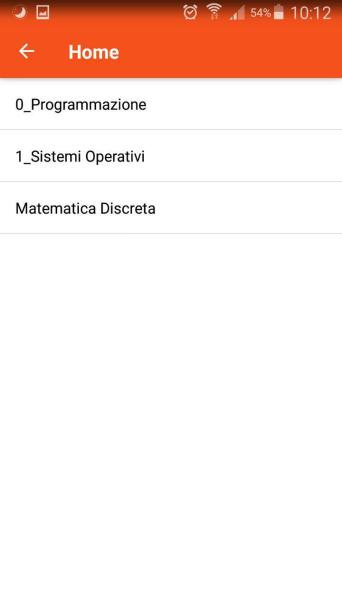
**\_signUp**

Metodo richiamato dal Button “Registrami” per effettuare la creazione di un nuovo utente tramite mail nella sezione Authentication di firebase. Verificata l’esattezza della mail inserita e andata a buon fine verrà richiamata il metodo \_save per la creazione di un nuovo utente nella sezione Database di firebase e successivamente indirizzamento alla screen Home attraverso l’uso delle navigation prop di React-Navigation che usiamo per portare tra screen diverse dati utente utili ad aggiornare il database e in particolar modo il nodo utente relativo a conclusione del test scelto ed eseguito.

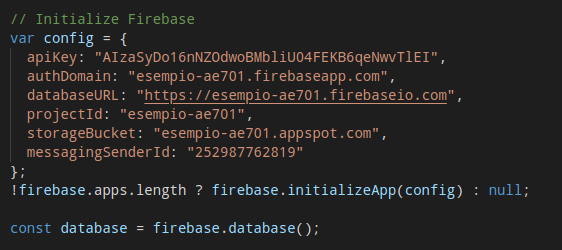
**\_logIn**

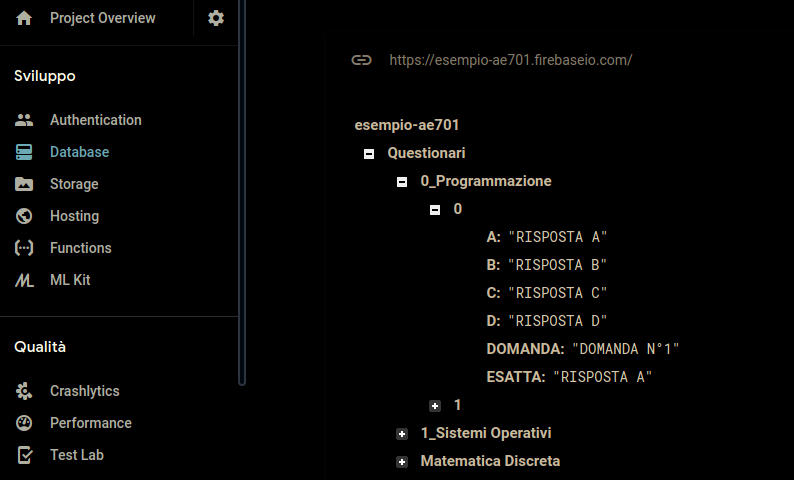
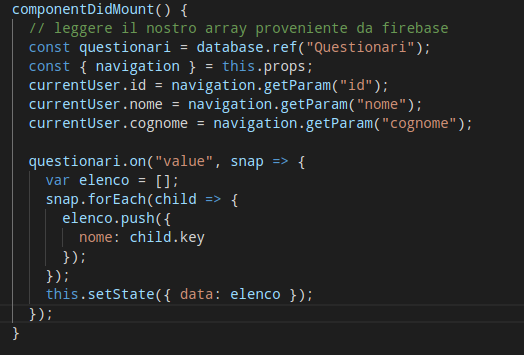


Metodo necessario a far eseguire l’accesso all’utente se già precedentemente registrato.  
Considerato il fatto che di firebase utilizziamo due sezioni (Authentication e Database) divise, quello che il metodo in un primo momento verifica è se esiste un istanza dello stesso utente autenticato nell’albero del file json in database. Se si, aggiorna in state i valori di identificazione relativi all’utente loggato perchè necessari da portare alla screen successiva per il futuro aggiornamento del nodo json.

***Home***

Rappresenta la screen alla quale l’utente visualizzerà la lista di questionari disponibili. E’ qui che viene eseguita l’inizializzazione del database di firebase attraverso il seguente codice:



Dopo di ciò, quel che il componente della screen farà sarà spacchettare i dati ricevuti tramite navigate props dalla screen login usando l’istruzione navigation.getParam e in seguito effettua la lettura e upload dei questionari scritti e disponibili nell’albero json del database di firebase, upload eseguito temporaneamente su variabile locale che useremo subito dopo per settare lo state proprio con questo elenco, perchè è da qui che nella fase di render del componente verrà effettuata la lettura dei dati.

Proprio il render è costituito da una scrollView così da prevedere anche liste abbastanza corpose di test che saranno visualizzati e selezionabili grazie al componente ListItem inserito all’interno del metodo map richiamato dall’array di test caricato nel nostro stato.  
Alla selezione del questionario che l’utente vuole svolgere verrà indirizzato sempre con i consueti metodi alla screen Questions per l’effettiva esecuzione del test.

