**F2F – Families-To-Families Sharing Care of Children  
MIUR FISR 2020**

**Progetto di Ingegneria del Software 2021/22  
Università Ca’ Foscari Venezia**

**Documento di progettazione**

versione 1.0  
 **Black Buffalos**

30 novembre 2021

*Informazioni sul Documento*

| MUR FISR 2020 |  | Acronimo | | | F2F | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Coordinatore Progetto | Prof. Agostino Cortesi | | | | | | | |
| Deliverable | Documento di progettazione | | | | | | | |
| Data di Consegna | 30/11/2021 | |  | | |  | |  |
| Team Leader | Giovanni Costa | | |  | | | 880892@stud.unive.it | |
| Team members | Massimo Cailotto 880873@stud.unive.it  Filippo Di Gennaro 882795@stud.unive.it Matteo Minardi 880895@stud.unive.it  Andrea Munarin 879607@stud.unive.it | | | | | | | |

*Document History*

| Version | Issue Date | Stage | Changes | Contributors |
| --- | --- | --- | --- | --- |
| 1.0 | 30/11/2021 | Draft | First version | All members of the team |
| 1.1 |  | Draft |  |  |
| 2.0 |  | Final |  |  |

**Indice**

[***1.***](#bookmark=id.7keojf5x5ytl) ***Introduzione*** **4**

[***2.***](#bookmark=id.npim8zvt669p) ***Glossario 4***

[***3.***](#bookmark=id.a34eyu233d8z) ***Architettura del Sistema*** 6

[***4.***](#bookmark=id.warsbqnrluju) ***Modello dei Dati e del Controllo 8***

[**5.**   ***Modelli UML***](#bookmark=id.ofyt745k4dbt) ***10***

[5.1 Diagrammi di sequenza](#bookmark=id.fsrs9otdd0mc) 10

[**6.  *Progettazione dell’interfaccia utente***](#bookmark=id.86chlt3ocirj) ***13***

[6.1 Splash Screen](#bookmark=id.7jvhjordjaib) 16

[6.2 Fase di registrazione](#bookmark=id.di9wqzbhzkep) 17

[6.2.1 Sign Up - parte 1](#bookmark=id.eijaxpjexjtu) 17

[6.2.2 Sign Up - parte 2](#bookmark=id.g88pkbhl9zm6) 18

[6.3 Login](#bookmark=id.kfem0einyda2) 19

[6.4 Homepage](#bookmark=id.l8vo8xmuksmt) 20

[6.5 Gruppi](#bookmark=id.kj4b6hfetw9s) 21

[6.5.1 Creazione gruppi](#bookmark=id.gezmeh923pro) 21

[6.5.2 Attività del gruppo](#bookmark=id.49pa7mv5qmmx) 22

[6.5.3 Membri del gruppo](#bookmark=id.4x94pbaeyr0d) 23

[6.5.4 Info del gruppo](#bookmark=id.xi44vzvnzna3) 24

[6.6 Attività](#bookmark=id.enri7lcoglp2) 25

[6.6.1 Creazione attività - parte 1](#bookmark=id.1rjdev3uz0lm) 25

[6.6.2 Creazione attività - parte 2](#bookmark=id.q50bxz5y6ad7) 26

[6.6.3 Info attività](#bookmark=id.pf4nnjeon9ys) 27

[6.7 Info utente](#bookmark=id.nxxpgunqjlen) 28

[6.8 Menù e gestione](#bookmark=id.8i41jqeomr6m) 29

[6.9 Bacheca](#bookmark=id.tdjco8yqy260) 30

[6.10 Mercatino](#bookmark=id.40ezk976rrur) 31

[**7.  *Riferimenti***](#bookmark=id.15hfzkiisfbk) ***32***

1. **Introduzione**

Lo scopo di questo documento è descrivere nello specifico le scelte progettuali del sistema, come verranno implementate le varie funzionalità dell’app e come verranno soddisfatti i requisiti presenti all’interno dell’apposito documento di specifica.

Molta rilevanza è data all’esposizione dei modelli e paradigmi architetturali utilizzati e alla spiegazione mediante diagrammi UML dei vari flussi di progettazione del sistema. Inoltre, sono presenti delle sezioni in cui verranno mostrate le schermate dell'interfaccia utente dell'applicazione android, corredate da una spiegazione esaustiva delle funzioni messe a disposizione per l’utilizzatore sulla base del documento dei requisiti.

1. **Glossario**

Qui sono elencati i vari termini tecnici e parole chiave che si utilizzeranno all’interno del documento di progettazione, correlati da una breve descrizione:

**Front-end** : sezione dell’applicazione con cui l’utente interagisce direttamente. Si occupa della gestione dei flussi di dati in input da parte degli utilizzatori e della visualizzazione dei risultati delle operazioni effettuate.

**Back-end** : questa parte salva ed elabora gli input di dati che riceve e fondamentalmente svolge tutte quelle operazioni che sono necessarie per un corretto funzionamento dell’applicazione, al termine delle quali invia i risultati al front-end in modo che possa mostrare ciò che è stato richiesto

**Android** : Android è un sistema operativo mobile basato su una versione modificata del [kernel Linux](https://it.wikipedia.org/wiki/Linux_(kernel)) e altri software open source, progettato principalmente per dispositivi mobili touchscreen come smartphone e tablet.

È un software gratuito e open source ed il suo codice sorgente è noto come Android Open Source Project (AOSP)

Fonte: [Wikipedia](https://it.wikipedia.org/wiki/Android)

**JavaScript** : JavaScript è un [linguaggio di programmazione](https://it.wikipedia.org/wiki/Linguaggio_di_programmazione) [orientato agli oggetti](https://it.wikipedia.org/wiki/Programmazione_orientata_agli_oggetti) e agli [eventi](https://it.wikipedia.org/wiki/Programmazione_a_eventi), comunemente utilizzato nella [programmazione Web](https://it.wikipedia.org/wiki/Programmazione_Web) [lato client](https://it.wikipedia.org/wiki/Lato_client) (esteso poi anche al [lato server](https://it.wikipedia.org/wiki/Lato_server)) per la creazione, in [siti web](https://it.wikipedia.org/wiki/Sito_web) e [applicazioni web](https://it.wikipedia.org/wiki/Applicazione_web), di effetti dinamici [interattivi](https://it.wikipedia.org/wiki/Interattivit%C3%A0) tramite [funzioni](https://it.wikipedia.org/wiki/Funzione_(informatica)) di [script](https://it.wikipedia.org/wiki/Script) invocate da eventi innescati a loro volta in vari modi dall'utente sulla [pagina web](https://it.wikipedia.org/wiki/Pagina_web) in uso ([mouse](https://it.wikipedia.org/wiki/Mouse), [tastiera](https://it.wikipedia.org/wiki/Tastiera_(informatica)), caricamento della pagina ecc...).  
Fonte: [Wikipedia](https://it.wikipedia.org/wiki/JavaScript)

**Node.Js** : Node.js è un [runtime system](https://it.wikipedia.org/wiki/Runtime_system) [open source](https://it.wikipedia.org/wiki/Open_source) multipiattaforma [orientato agli eventi](https://it.wikipedia.org/wiki/Programmazione_a_eventi) per l'esecuzione di codice [JavaScript](https://it.wikipedia.org/wiki/JavaScript), costruita sul [motore JavaScript V8](https://it.wikipedia.org/wiki/V8_(motore_JavaScript)) di [Google Chrome](https://it.wikipedia.org/wiki/Google_Chrome).  
In origine JavaScript veniva utilizzato principalmente [lato client](https://it.wikipedia.org/wiki/Lato_client). In questo scenario gli script JavaScript, generalmente incorporati all'interno dell'HTML di una pagina web, vengono interpretati da un motore di esecuzione incorporato direttamente all'interno di un [Browser](https://it.wikipedia.org/wiki/Browser). Node.js consente invece di utilizzare JavaScript anche per scrivere codice da eseguire [lato server](https://it.wikipedia.org/wiki/Lato_server), ad esempio per la produzione del contenuto delle pagine web dinamiche prima che la pagina venga inviata al [Browser](https://it.wikipedia.org/wiki/Browser) dell'utente. Node.js in questo modo permette di implementare il cosiddetto paradigma "JavaScript everywhere" (JavaScript ovunque), unificando lo sviluppo di applicazioni Web intorno ad un unico linguaggio di programmazione (JavaScript).  
Fonte: [Wikipedia](https://it.wikipedia.org/wiki/Node.js)

**Client-Server** : In [informatica](https://it.wikipedia.org/wiki/Informatica) il termine sistema client-server (letteralmente cliente-serviente) indica un'[architettura di rete](https://it.wikipedia.org/wiki/Architettura_di_rete) nella quale genericamente un [computer](https://it.wikipedia.org/wiki/Computer) [client](https://it.wikipedia.org/wiki/Client) o [terminale](https://it.wikipedia.org/wiki/Terminale_(informatica)) si [connette](https://it.wikipedia.org/wiki/Connessione_(informatica)) ad un [server](https://it.wikipedia.org/wiki/Server) per la fruizione di un certo servizio, quale ad esempio la condivisione di una certa [risorsa](https://it.wikipedia.org/wiki/Risorsa_informatica) [hardware](https://it.wikipedia.org/wiki/Hardware)/[software](https://it.wikipedia.org/wiki/Software) con altri client, appoggiandosi alla sottostante architettura [protocollare](https://it.wikipedia.org/wiki/Protocollo_di_rete).  
Più semplicemente, i sistemi client/server sono un'evoluzione dei sistemi basati sulla condivisione semplice delle [risorse](https://it.wikipedia.org/wiki/Risorsa_informatica): la presenza di un [server](https://it.wikipedia.org/wiki/Server) permette ad un certo numero di client di condividerne le risorse, lasciando che sia il server a gestire gli accessi alle risorse per evitare conflitti di utilizzazione tipici dei primi [sistemi informatici](https://it.wikipedia.org/wiki/Sistema_informatico)  
Fonte: [Wikipedia](https://it.wikipedia.org/wiki/Sistema_client/server)

**Three tier architecture** : è una particolare architettura software e hardware di per l'esecuzione di un'applicazione web che prevede la suddivisione dell'applicazione in tre diversi moduli o strati dedicati rispettivamente alla interfaccia utente, alla logica funzionale (business logic) e alla gestione dei dati persistenti

**MongoDB** : MongoDB è un [DBMS](https://it.wikipedia.org/wiki/Database_management_system) non relazionale, [orientato ai documenti](https://it.wikipedia.org/wiki/Base_di_dati_orientata_al_documento). Classificato come un database di tipo [NoSQL](https://it.wikipedia.org/wiki/NoSQL), MongoDB si allontana dalla struttura tradizionale basata su tabelle dei [database relazionali](https://it.wikipedia.org/wiki/Database_relazionali) in favore di documenti in stile [JSON](https://it.wikipedia.org/wiki/JSON) con schema dinamico (MongoDB chiama il formato [BSON](https://it.wikipedia.org/wiki/BSON)), rendendo l'integrazione di dati di alcuni tipi di applicazioni più facile e veloce.  
Fonte: [Wikipedia](https://it.wikipedia.org/wiki/MongoDB)

**Modello dei dati** : il modello dei dati stabilisce la maniera in cui i dati inviati al database vengono salvati

**Modello di controllo** : il modello di controllo determina il modo in cui l’intera applicazione gestisce la sue funzionalità

**UML** : in ingegneria del software, l’UML, acronimo che sta per unified modeling

language, è un linguaggio di modellazione e di specifica basato sul paradigma

orientato ad oggetti.

**Schermata** : una schermata è la rappresentazione grafica dei dati forniti dal monitor da parte di un elaboratore elettronico

**Splash Screen** : è una schermata contenente un’immagine, un logo o la versione corrente del software. Di solito viene visualizzato all’avvio di un'applicazione

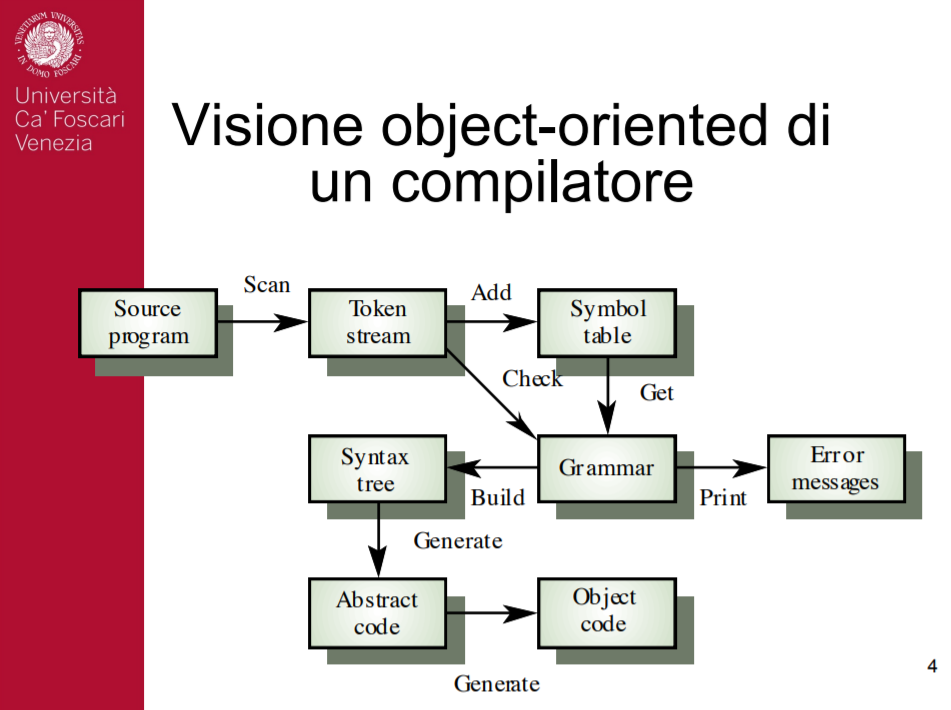
**UI** : il termine nel design sta per “interfaccia utente”. L’interfaccia utente è il layout grafico di un’applicazione. Consiste fondamentalmente nei pulsanti su cui gli utenti fanno clic, nel testo che viene inserito, nelle immagini, nei cursori, nei campi di immissione del testo e negli altri elementi con cui l’utente interagisce. Ciò include layout dello schermo, transizioni, animazioni dell’interfaccia e ogni singola micro interazione.

**UX** : il termine sta per “esperienza utente“. L’esperienza di un utente con l’app è determinata dal modo in cui questo interagisce con essa. Alcuni requisiti per una buona esperienza utente sono la fluidità, l’intuitività, la semplicità e l’efficienza.

**Compilatore** : è un programma informatico che traduce una serie di istruzioni scritte in un determinato linguaggio di programmazione (codice sorgente) in istruzioni di un altro linguaggio (codice oggetto): il processo di traduzione si chiama compilazione mentre l'attività inversa, ovvero passare dal codice oggetto al codice sorgente, è chiamata decompilazione ed è effettuata per mezzo di un decompilatore.

1. **Architettura del Sistema**

Il team ha deciso di adottare per la strategia di progettazione lo stile Object Oriented, in cui il sistema è visto come collezione di oggetti che interagiscono tra loro.

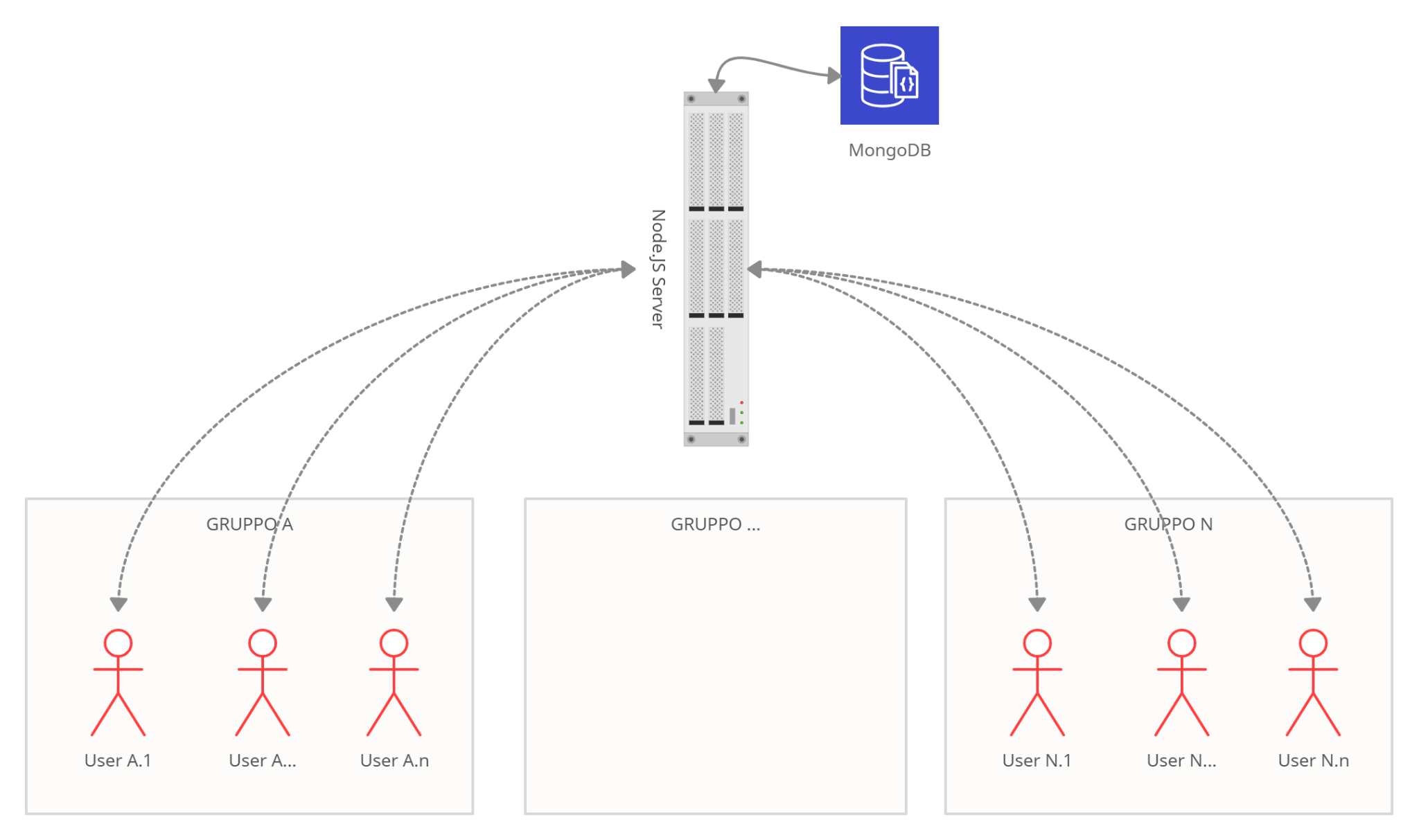
In questa architettura il sistema è di tipo decentralizzato ed ogni oggetto gestisce il proprio stato. Gli oggetti che possono essere istanze di una classe e comunicano scambiandosi le informazioni attraverso i propri metodi.

*Esempio di progettazione di un compilatore in stile Object Oriented*

Il sistema è quindi diviso in sottosistemi (oggetti) che cooperano tra loro ed in particolare l’architettura è suddivisa nelle seguenti componenti principali:

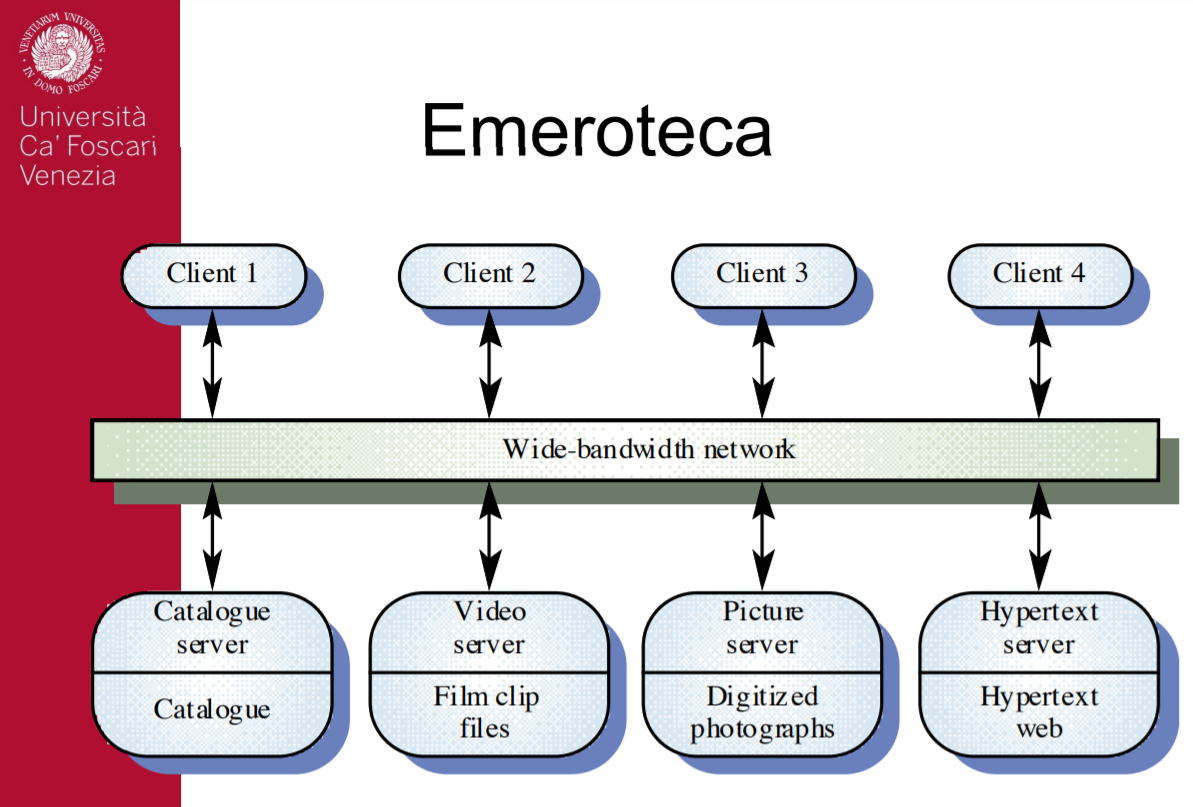
* **Database MongoDB**: si occupa del mantenimento dei dati all’interno del sistema
* **Server Node.js**: si interfaccia sia con il database che con l’applicazione Android. Permette alle due entità di comunicare computando le richieste ed elaborando le risposte necessarie.
* **Applicazione Android**: l’interfaccia utente per il sistema. Mediante questa componente l’utente ha la possibilità di effettuare tutte le richieste necessarie attraverso appositi metodi: queste sono modellate come richieste HTTP al server.

Il modello di collaborazione (descritto più nello specifico nel documento dei requisiti) quindi è il seguente:



1. **Modello dei Dati e del Controllo**

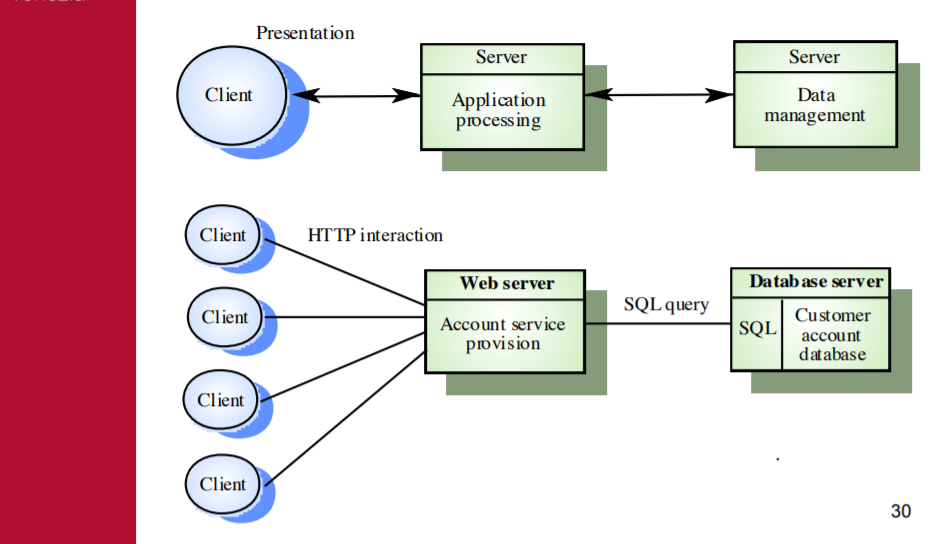
Il modello dei dati che è stato scelto dal team è di tipo Client-Server, nello specifico con architettura Three Tier, per le sue seguenti caratteristiche:

* E’ il modello tipico dei sistemi distribuiti;
* I dati sono gestiti in modo decentralizzato da singoli servizi;
* Le diverse componenti richiedono accesso ai dati tramite queste primitive;
* La rete permette ai client di accedere ai server;

*Esempio di modello client/server per un emeroteca*

Per quanto riguarda l’architettura Three Tier, si descrivono i tre strati funzionali di cui è composta:

* **Presentation layer**: ha il compito di presentare i risultati della computazione agli utenti del sistema e di gestire gli input degli stessi;
* **Application processing layer**: si occupa di offrire le funzionalità specifiche dell’applicazione;
* **Data management layer**: è responsabile della gestione della comunicazione con il DBMS;



*Esempio di Three Tier Architecture*

Per queste caratteristiche il modello descritto si adatta perfettamente al sistema da progettare, caratterizzato da clienti che, attraverso le loro applicazioni android, fanno richieste per i vari servizi offerti dai server Node.js, che si interfacceranno con il database.

Il modello di controllo scelto è quello centralizzato di tipo call-return.

Questo paradigma è adatto per una gerarchia di procedure, in cui il controllo viene gestito con il metodo top-down, ed è quindi applicabile a sistemi sequenziali.   
Il modello si adatta esattamente al sistema da sviluppare, in cui l’utente esegue le varie operazioni in sequenza. Alcuni esempi sono: registrazione, login, il successivo accesso ad un gruppo, la creazione di un'attività e la successiva aggiunta dei bambini che vi parteciperanno.

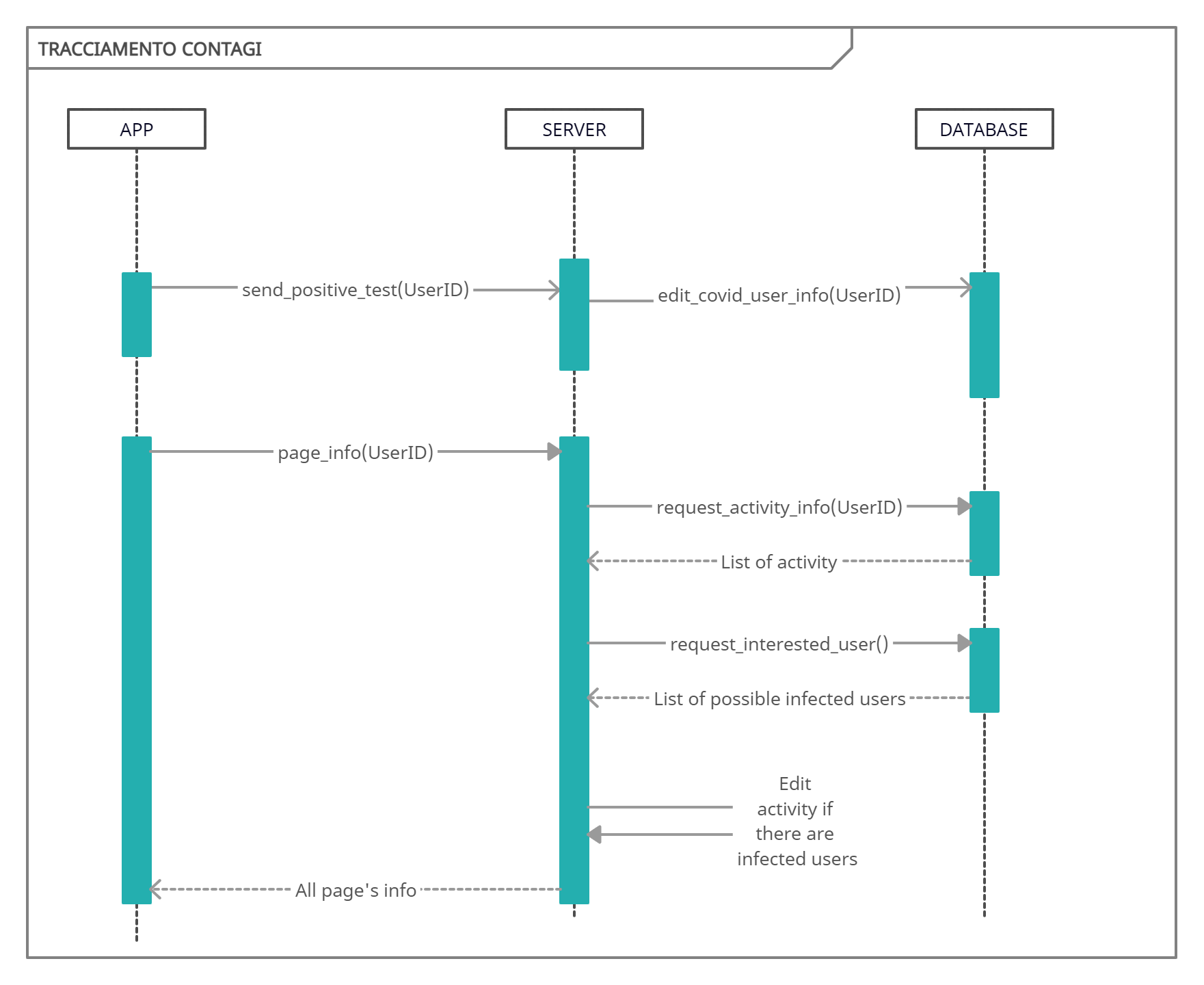
Infine, si è pensato di servirsi un ulteriore livello strutturale: la scomposizione modulare basata sul modello ad oggetti, dove il sistema e i sottosistemi sono scomposti in elementi che interagiscono.   
Un esempio può essere il sottosistema della parte client, scomposta nelle varie activity che interagiscono e si scambiano messaggi tra loro.

1. **Modelli UML**

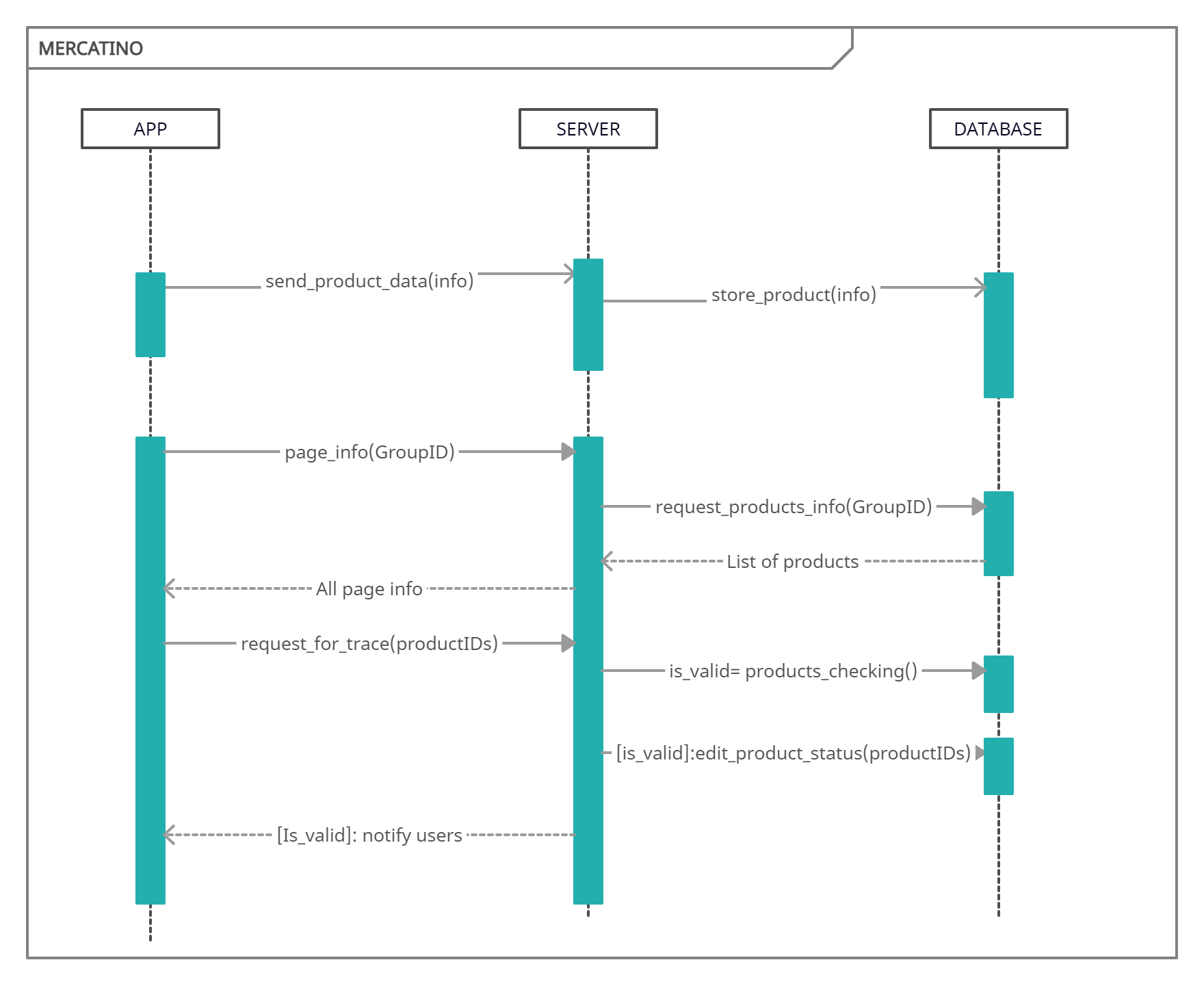
Il modello UML ritenuto necessario per lo sviluppo del sistema è il diagramma di sequenza. Questo è stato scelto soprattutto perchè come descritto nei capitoli precedenti il sistema è di tipo sequenziale e secondo il team questo diagramma è la migliore scelta per la rappresentazione degli use-cases e dei vari componenti presenti.

* 1. **Diagrammi di sequenza**

**Tracciamento dei contagi**

Un utente può segnalare il suo stato di positività a Covid19 ad altri utenti mediante un’apposita schermata dell’applicazione Android. Quest'ultima si occuperà di inviare le informazioni dell’utente infetto al server, che modificherà nel database alcuni parametri sull’utente e sulle attività future (disiscrizione) e quelle a cui ha partecipato. 

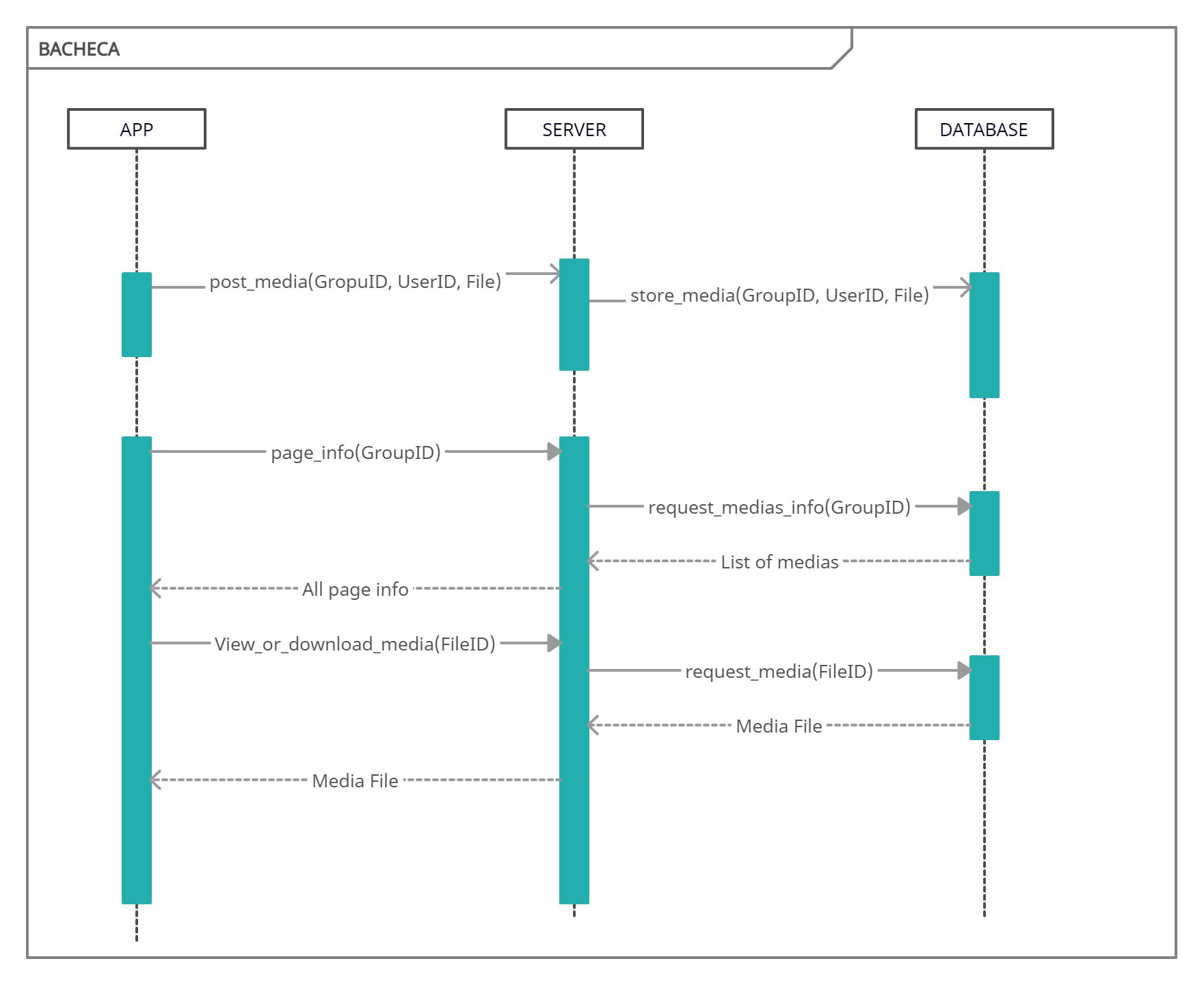
A tutti gli utenti che hanno partecipato ad una delle attività in cui era presente anche l’utente positivo, nei 4 giorni precedenti la segnalazione, riceveranno un avviso di possibile contagio sulla homepage dell’app e nello specifico, l’attività in cui c’è stato il contatto verrà marcata di rosso.

**Mercatino**

Un utente può pubblicare un nuovo annuncio di scambio o prestito, anche gratuito, sulla pagina dell’applicazione relativa al mercatino. Per postare l’elemento è necessario passare al server tutti i dati necessari per poterli inserire nel database, come l’oggetto messo in mostra e quelli richiesti.  
Quando un altro utente che è iscritto allo stesso gruppo entra nella sezione mercatino dedicata, attraverso una richiesta al server otterrà tutti i dati e gli oggetti aggiornati, e potrà visionare i prodotti in tempo reale.

Nel caso in cui venga effettuato uno scambio gli ID degli oggetti in questione e memorizzati nel database, verranno inviati al server che contatterà il database ed aggiornerà lo stato di questi prodotti.

L’utente infine riceverà una notifica di scambio avvenuto correttamente se ha eseguito tutte le operazioni come da specifica e troverà il riassunto dell’operazione in un'apposita sezione.

**Bacheca**

Un utente può postare un qualunque media, come foto, video, documenti o file audio all’interno del gruppo nell’apposita sezione di bacheca. Per questo è necessario inviare al server, che poi si connetterà col database, le informazioni relative all’utente che ha pubblicato i media, il gruppo in cui li ha pubblicati, ed il file in questione.

Quando un altro membro del gruppo accederà alla schermata della bacheca attraverso una richiesta al server, potrà visualizzare tutti gli elementi condivisi, ricercarli, modificarli se è stato questo a caricarli e in base alle proprie esigenze richiedere di visualizzare/scaricare il media all’interno del proprio smartphone.

1. **Progettazione dell’interfaccia Utente**

La creazione delle interfacce seguirà dei principi di progettazione rigidi, ma che porteranno ad ottenere un risultato finale che renda l’esperienza utente al massimo delle possibilità.

Secondo O’Brien et al. (2010), i Parametri focali per definire una buona **User Experience** sono i seguenti:

-Focused Attention:   
 il coinvolgimento cognitivo dell’utente, non deve essere troppo   
 elevato per non creare stress ma neanche troppo basso per non   
 abbassare troppo la soglia dell’attenzione

-Perceived Usability (intuitiva):  
 l’usabilità percepita dall’utente; se un oggetto ha una buona usabilità,   
 allora capire a cosa serva non è ambiguo

- Aesthetichs:   
 l’interfaccia deve essere esteticamente piacevole in modo che   
 l’utilizzatore non si stanchi di osservarla

- Endurability:   
 in grado di rendere facilmente attuabili anche operazioni che, se   
 isolate fuori contesto, sembrano difficili o non piacevoli da svolgere,   
 facendo crescere all'utilizzatore il desiderio di ripeterle

- Novelty:   
 il grado di novità percepito dall’utente durante l’utilizzo   
 dell’applicazione in questione

- Felt Involvement:   
 il coinvolgimento emotivo dell’utente; il suo stato emotivo ha effetto   
 sulla durata, concentrazione e comprensione delle sue interazioni, e   
 di conseguenza anche su ciò a cui portano

Per l’applicazione di una buona **usabilità** il gruppo si baserà sulle 10 euristiche proposte dal ricercatore in ambito di interazione uomo-macchina Jackob Nielsen:

- stato del sistema:   
 deve essere sempre visibile in che stato si trova il sistema, se sta   
 funzionando correttamente o se si è verificato un errore che ha   
 portato ad un blocco

- analogie con il mondo reale:   
 bisogna introdurre metafore ed analogie tra il sistema sviluppato ed   
 il mondo reale, per far comprendere il funzionamento anche di   
 funzionalità meno intuitive ad alcune categorie di utenti

- controllo utente e libertà:   
 aggiungere funzioni che permettano di annullare, ricominciare o   
 abbandonare certe operazioni che sono state iniziate, senza dover   
 perdere eventuali progressi accumulati da operazioni precedenti   
 fino a quel momento

- consistenza e standard:   
 gli elementi delle pagine, come ad esempio un bottone, devono   
 rimanere sempre conformi senza cambiare aspetto e/o   
 funzionamento perchè altrimenti l’utente potrebbe premerli   
 portando l’app ad assumere comportamenti indesiderati

- prevenzione errori:   
 inserimento di messaggi di aiuto per prevenire ambiguità, errori o   
 malfunzionamenti

- riconoscimento anziché ricordo:   
 le componenti delle interfacce devono assumere un aspetto sempre   
 più simile all’aumentare della loro correlazione, portando gli utenti   
 alla creazione di un’associazione visiva tra gli elementi e le loro   
 relazioni

- estetica e design minimalisti:  
 un design minimale permette di focalizzare facilmente quali siano i   
 punti chiave di una determinata schermata, senza creare confusione   
 e dispersione

- flessibilità ed efficienza d'uso:   
 bisogna introdurre comandi e shortcut che permettano ad utenti   
 esperti dell’applicazione di effettuare operazioni semplici in un   
 numero ridotto di passaggi di bassa complessità

- aiuto per utente:  
 inserimento di messaggi di errore dedicati all’utente al verificarsi di   
 eventi non corretti, con relativa indicazione su come risolverli

- documentazione:  
 creazione di un documento che elenchi le diverse interfacce inserite   
 nell’applicazione, affiancate da una loro descrizione testuale che   
 chiarisca eventuali dubbi

Il sistema si impegnerà ad acquisire dall’utente il minor numero di dati necessari possibile, in modo da garantire un servizio sicuro e semplice da interpretare.  
I dati di maggiore rilevanza che saranno richiesti, soprattutto in fase di registrazione, includono:

* mail;
* password;
* Nome;
* Cognome;

E saranno inseriti in delle textbox, che semplificano l’interazione con l’utilizzatore.

Il sistema comunica unicamente i dati che gli utenti inseriscono direttamente, e svolge il ruolo di intermediario nelle loro comunicazioni.

**Progettazione del sistema di help**

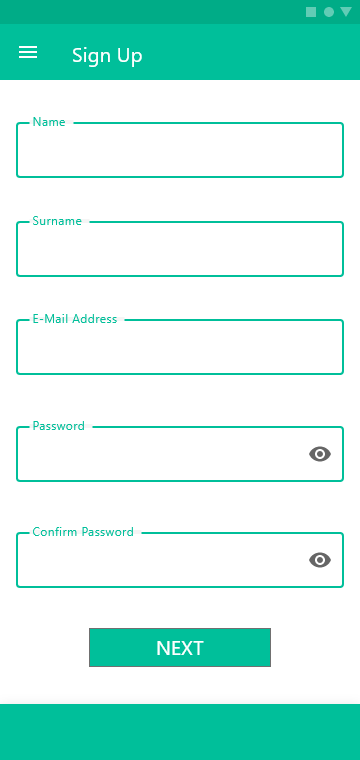
Aiuta gli utenti a comprendere come utilizzare al meglio le diverse funzionalità implementate nell’applicazione.  
Risolve eventuali dubbi che gli utilizzatori possono avere, possibilmente espandendo il target a coloro che potrebbero scaricare il prodotto pur non essendo nativi digitali.

Nei prossimi sottocapitoli saranno esposte e descritte le diverse interfacce e schermate che saranno presenti nell’applicazione, a cui sarà affiancata una loro descrizione sia dell’aspetto grafico che del funzionamento.

* 1. **Splash Screen**

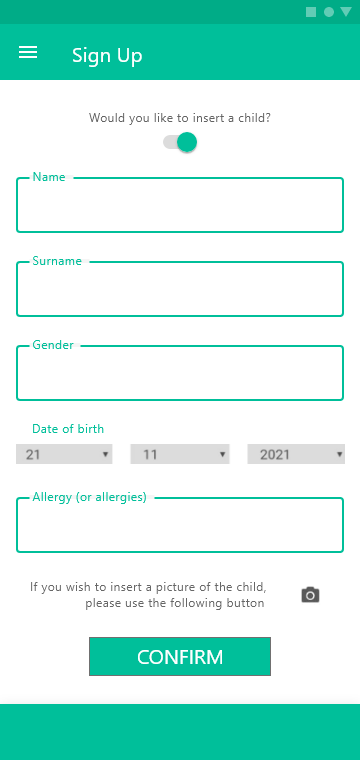
Schermata che appare per un brevissimo intervallo di tempo ad ogni avvio dell’applicazione.  
Ha lo scopo principale di informare l’utente che l’applicazione sia stata avviata, mostrandone il logo ed il team di sviluppo.

* 1. **Fase di registrazione**

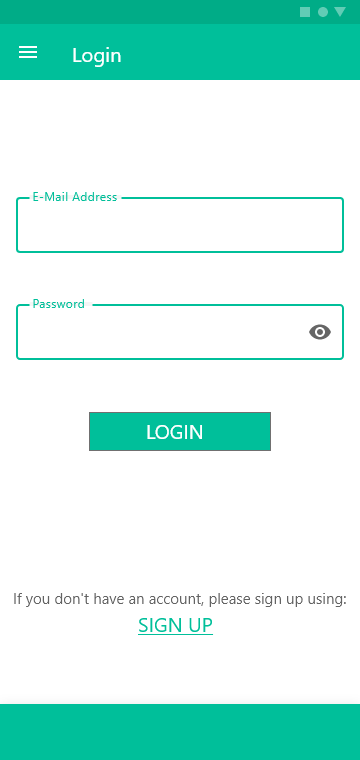
* + 1. **Sign Up - parte 1**

Schermata che appare quando un nuovo utente sceglie di creare un account all’interno dell’applicazione.  
In questa schermata vengono richieste alcune informazioni personali che serviranno ad arricchire il profilo dell’utilizzatore, e verrà scelta la password, che potrà essere nascosta o mostrata durante la sua scrittura attraverso l’apposita icona con l’occhio, con cui si potrà accedere all’account una volta creato.  
È presente un bottone che permette di proseguire con la seconda parte della registrazione.

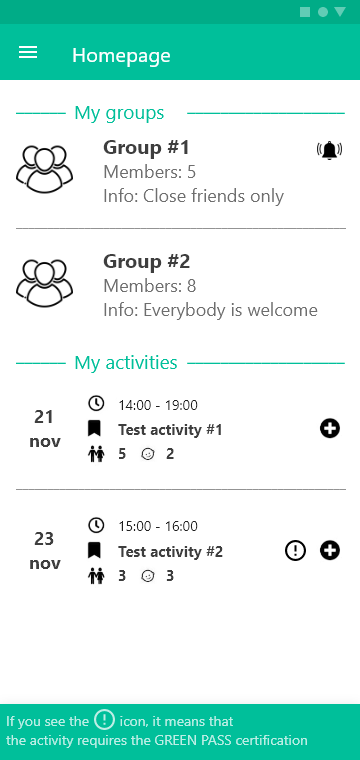
* + 1. **Sign Up - parte 2**

Appare solo dopo che l’utente avrà cliccato sul bottone con scritto “next” presente nella schermata precedente.  
Viene chiesto se si intende inserire le informazioni relative ad un figlio già durante la propria registrazione; in caso affermativo si dovranno inserire alcune informazioni personali e volendo si potrà anche inserire un’immagine del figlio stesso.  
Il processo terminerà quando il bottone con scritto “confirm” sarà premuto. 

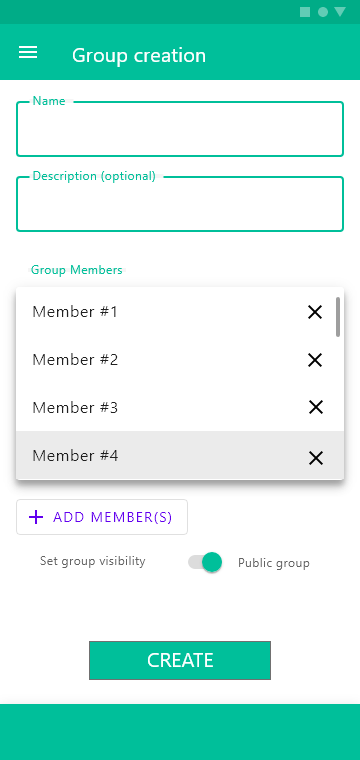
* 1. **Login**

Schermata che appare all’avvio dell’app, in seguito alla Splash Screen, nel caso in cui l’utente abbia già creato il proprio account in passato.  
Richiede l’indirizzo E-Mail e la password del proprio profilo, decise al momento della registrazione. Permette di accedere alle vere e proprie funzionalità dell’app una volta inserite le giuste credenziali e premuto il bottone con scritto “login”.

* 1. **Homepage**

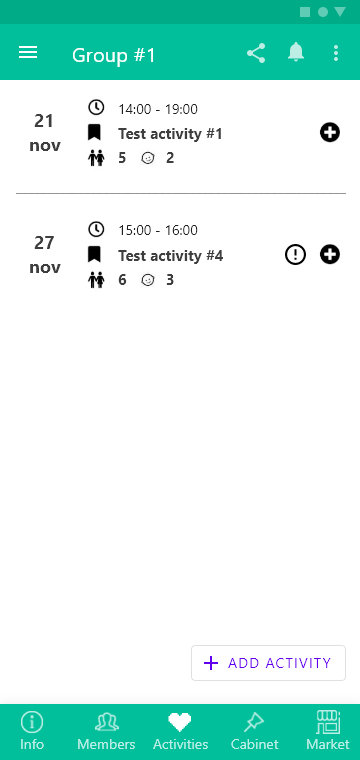
Schermata che appare dopo il corretto login da parte dell’utente.  
Presenta un elenco dei gruppi a cui l’utilizzatore fa parte ed un elenco delle attività a cui dovrà partecipare, con alcune relative informazioni in modo tale da attirare l’attenzione ed agevolare la comprensione dell’utente.  
È possibile cliccare sugli elementi degli elenchi per ottenere più informazioni di quante siano visualizzate normalmente.  
A fondo pagina è presente una legenda che spiega il significato dell’icona con il punto esclamativo.

* 1. **Gruppi**

* + 1. **Creazione gruppi**

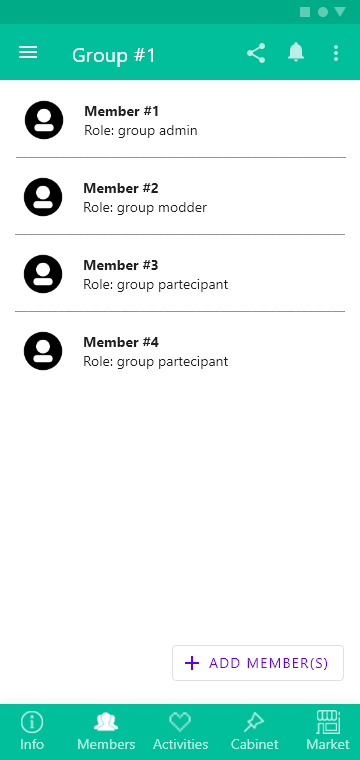
Schermata che viene mostrata durante la creazione di un gruppo. Quando un utente decide di creare un nuovo gruppo verranno richiesti un nome per il gruppo, una breve descrizione facoltativa e dei partecipanti. L’aggiunta di nuovi partecipanti può essere fatta direttamente durante la creazione del gruppo, premendo il pulsante [ADD MEMBER(S)] ma anche successivamente tramite un link di invito o nella schermata relativa ai partecipanti di un gruppo. Si deve inoltre settare la visibilità del gruppo rispetto agli altri utenti, decidendo quindi tra pubblico o privato. Una volta completato il procedimento, premendo [CREATE] verrà creato il gruppo e verrà mostrata la schermata iniziale per la gestione dello stesso.

* + 1. **Attività del gruppo**

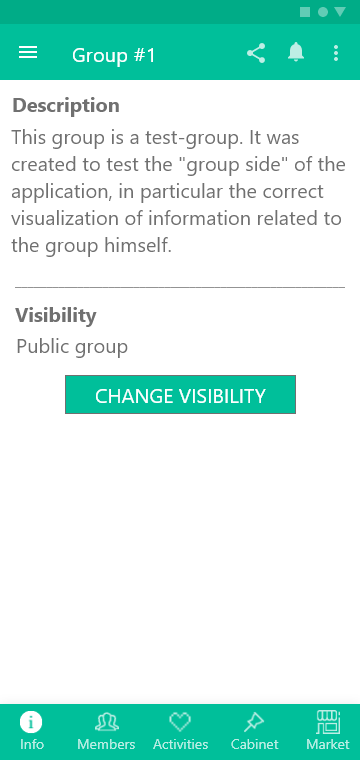


Schermata relativa alla sezione “Activities” di un gruppo e che appare subito dopo una corretta creazione. In questa schermata vengono visualizzate tutte le attività relative ad un gruppo. È possibile gestire la singola attività cliccandola e la schermata permette anche di crearne di nuove tramite il pulsante [ADD ACTIVITY].

* + 1. **Membri del gruppo**



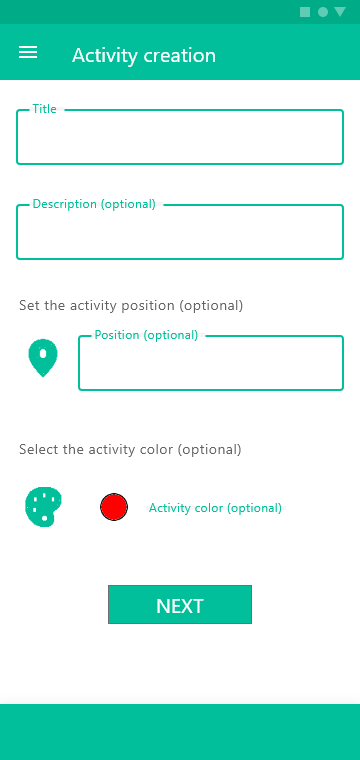
Schermata che viene visualizzata quando si clicca nella sezione “Members” di un gruppo. In questa sezione vengono visualizzati e gestiti i partecipanti del gruppo. È possibile aggiungere nuovi partecipanti tramite il pulsante [ADD MEMBER(S)].

* + 1. **Info del gruppo**

Questa schermata viene visualizzata cliccando nella sezione “Info” di un gruppo. Vengono qui mostrate la descrizione e la visibilità, che sono anche modificabili.

* 1. **Attività**

* + 1. **Creazione attività - parte 1**

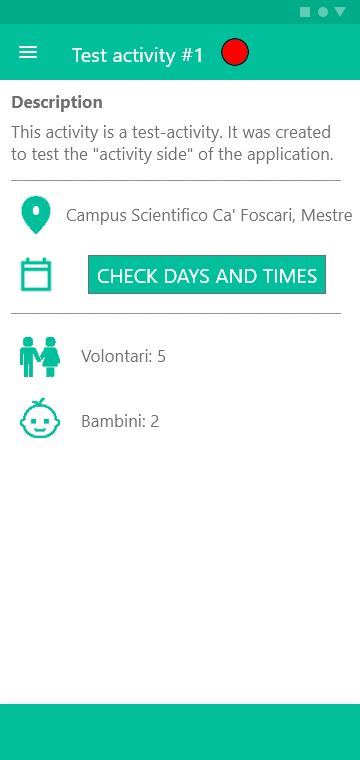


Schermata che viene visualizzata durante la creazione di una nuova attività. Quando un utente di un gruppo decide di creare un nuova attività verranno richiesti un titolo (nome) per l’attività, una breve descrizione facoltativa, una posizione (facoltativa) ed un colore per identificarla (facoltativo). Dopo aver inserito i dati si passa alla schermata successiva per completare la creazione.

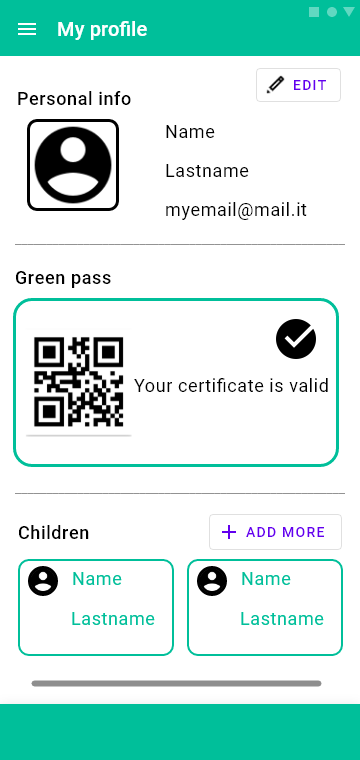
* + 1. **Creazione attività - parte 2**

In questa schermata vengono impostati i giorni e gli orari della nuova attività, successivamente alla configurazione di nome, colore, descrizione e posizione. Una volta inseriti correttamente i dati, premendo [CREATE] verrà creata la nuova attività e si potrà gestire tramite la schermata di visualizzazione dell’attività stessa.

* + 1. **Info attività**

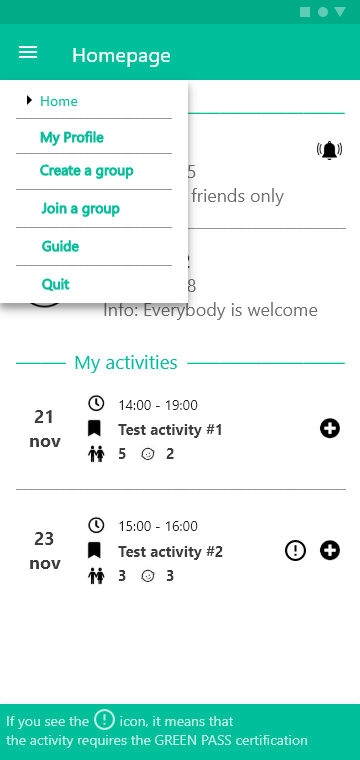


Questa schermata appare quando si clicca su una determinata attività e subito dopo una corretta creazione. Qui si possono visualizzare tutti i dati relativi ad un’attività come: la descrizione, gli orari ed i giorni, i volontari e i bambini.

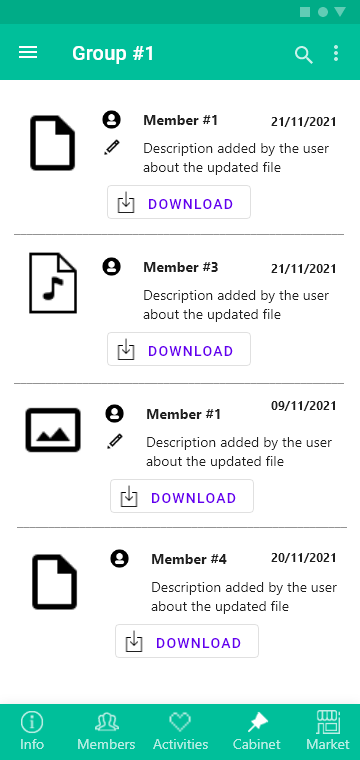
* 1. **Info utente**  
     

Questa schermata viene visualizzata quando si clicca sulla sezione “profilo” del menù. Qui vengono mostrate le informazioni generali del profilo come nome, cognome e email. Inoltre è possibile caricare e visionare l’immagine del proprio GreenPass, aggiungere e modificare le informazioni dei figli a carico.

* 1. **Menù e gestione**

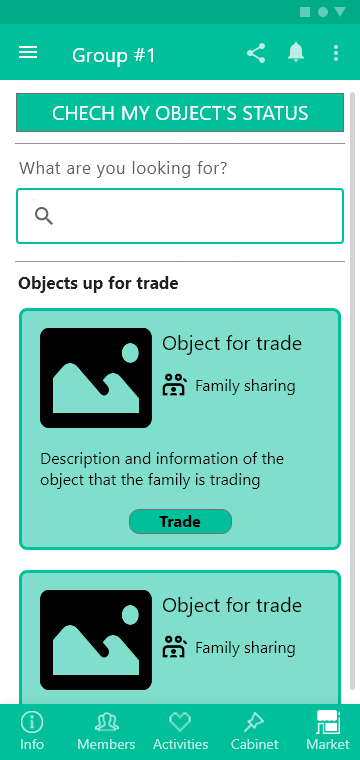


È possibile accedere al menù di navigazione premendo l’icona collocata in alto a sinistra, che mostrerà l’elenco delle sezioni dell’app in cui si potrà navigare.  
La pagina in cui ci si trova attualmente è indicata con una freccia nera, mentre quelle accessibili sono scritte in grassetto.

* 1. **Bacheca**

Schermata che viene mostrata quando si accede alla sezione “Cabinet” all'interno di un gruppo. Qui è possibile postare, scaricare e, se si è il creatore del post, modificare e cancellare il media postato. I file condivisi con il gruppo possono essere di formato diverso, avere un nome, una descrizione, l’autore del post e la data di caricamento.

I media possono essere anche cercati mediante l’apposita icona in alto a destra

* 1. **Mercatino**

“Mercatino” è la sezione creata per interfacciarsi nella condivisione o nello scambio di oggetti tra famiglie dello stesso gruppo. Ogni elemento ha una descrizione e un'immagine che rappresenta l'oggetto. In questa schermata è possibile cercare oggetti utilizzando parole chiave oppure cercali manualmente nella lista degli oggetti. È possibile contattare il possessore dell’oggetto, effettuare lo scambio premendo il bottone “Trade”, annullare uno scambio o un prestito, visualizzare lo stato di un oggetto .

1. **Riferimenti**

Per creare questo documento sono stati utilizzati come riferimento:

* “Documento di progettazione” di alcuni gruppi degli anni precedenti
* Piano di Progetto
* Documento dei requisiti
* materiale messo a disposizione dal professore