**Mobile computing**

**course**

**a.y. 2015/2016**

**LOVEITALY**

**Technical documentation[[1]](#footnote-1),[[2]](#footnote-2)**

|  |  |  |
| --- | --- | --- |
| **Team Members[[3]](#footnote-3)** | | |
| **Name** | **Student Number** | **E-mail address** |
| **Giulio Lastra** | *228457* | [*giulio.lastra@student.univaq.it*](mailto:giulio.lastra@student.univaq.it) |
| Antonio Di Pietro | *229650* | *antonio.dipietro@student.univaq.it* |
| Alessandro Liberato | *228261* | [*alessandro.liberato@student.univaq.it*](mailto:alessandro.liberato@student.univaq.it) |
| Samuele Orsini | *221916* | [*samuele.orsini@student.univaq.it*](mailto:samuele.orsini@student.univaq.it) |

Architecture

# MVC architecture



Figura - Diagramma MVC

I **model** che compongono la nostra architettura MVC realizzata con l’uso di Backbone sono i seguenti:

* *CartModel*
* *CategoryModel*
* *LoginModel*
* *ManufacturerModel*
* *ProductModel*
* *RegistrationModel*
* *SettingsModel*

Questi modelli, insieme alle collection, gestiscono tutta la parte dati dell’applicazione, infatti in essi troviamo tutta la parte relativa all’uso delle API di PrestaShop con le quali si interfacciamo con il database.

I dati ricevuti vengono quindi parsati ed utilizzati, infine è presente la gestione delle tasse relative ai prodotti messi in vendita. Inoltre, è possibile trovare il codice che ci permette di salvare i dati di login, della geolozalizzazione, del carrello e delle impostazioni nel local storage. Infine, sempre nella parte dei modelli è stata implementata la gestione della posizione dell’utente.

Le **collection** che compongono l’architettura MVC sono le seguenti:

* *CategoryCollection*
* *CategoryProductsCollection*
* *ManufacturerProductsCollection*
* *ManufacturersCollection*
* *ProductsCollection*
* *SearchCollection*
* *StructureCollection*

In questa parte troviamo di nuovo l’utilizzo delle API di PrestaShop per ottenere collezioni di modelli. Viene inserita anche la funzione che permette la ricerca del prodotto desiderato.

Le **view** che sono state realizzate per l’applicazione sono le seguenti:

* *StructureView*
* *CartView*
* *CategoryView*
* *ContactView*
* *FaqView*
* *HomeView*
* *InfoView*
* *LoginView*
* *ManufacturerView*
* *ProductView*
* *SearchView*
* *SettingsView*

Nelle view viene implemetata tutta la user interface dell’app. Si trovano qui quindi tutte quelle funzioni di gestione degli eventi scatenati dell’utente per mezzo della GUI.

E’ rilevante sottolineare che in questa sezione è presente l’interazione con il template engine.

# Data sources



Figura - Diagramma Data sources

**API SERVER PrestaShop**

Queste APIs ci hanno permesso di gestire le richieste RESTful GET, POST, PUT verso il database per ottenere e manipolare i dati Loveitaly.

**API SERVER Nominatim**

Questa API ci permette di comunicare al server di Nominatim le coordinate geografiche ottenute dal dispositivo, da elaborare per ottenere come risultato, le informazioni sulla posizione dell’utente.

**Local storage**

Questa porzione di memoria ci permette di salvare in locale (sulla memoria di massa del dispositivo) dei dati strategici da tenere sempre disponibili (carrello, coordinate, login, impostazioni, indirizzo).

**Geolocalizzazione**

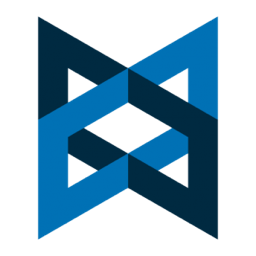
Questa richiesta utilizza un plugin di Cordova (Geolocation) per ottenere dal dispositivo la latitudine e la longitudine.

# Used libraries and frameworks

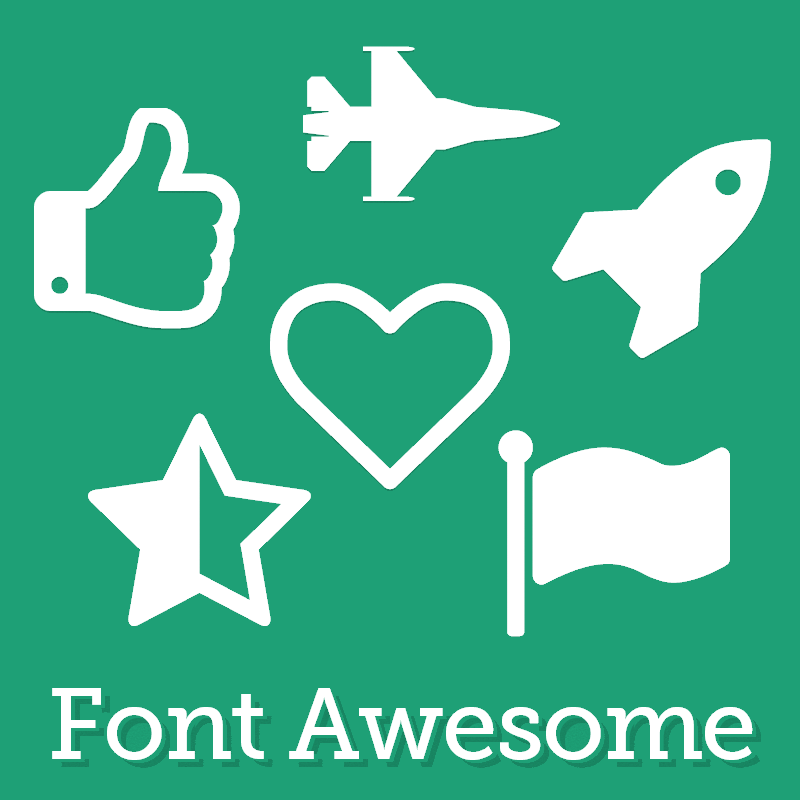
Per lo sviluppo si è deciso di utilizzare il Cordova Boilerplate fornito dal Professor Malavolta, a cui sono state rimosse le seguenti componenti: IceCream, Leaflet, Ratchet, spin.js e Zepto.

E’ stata aggionrata la versione di jQuery alla versione 2.4.

Abbiamo aggunto:

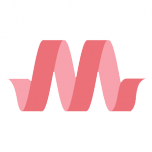
**Backbone**

Utilizzato per implementare l’architettura MVC.



**Font Awesome**

Si occupa di gestire le icone tramite font.

**Materialize** E’ stato caricato in versione AMD ed è utilizzato per dare un aspetto material design

all’ applicazione e per le animazioni e per gli effetti js (menu trascinabili, modali …).



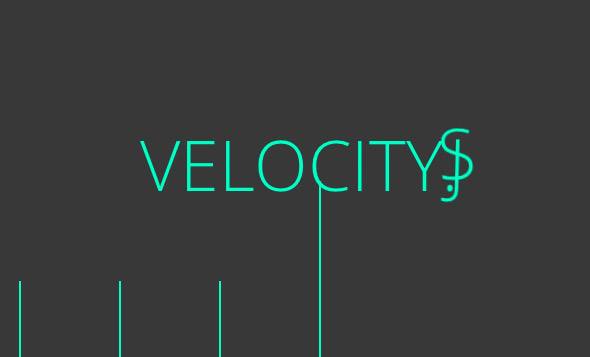
**Hammer.js** (richiesto da Materialize)

Serve a definire le gesture.



**MD5**

Serve per cifrare in MD5 la password dell’utente.

**Velocity.js**

Per alleggerire le animazioni jQuery.

****Tools

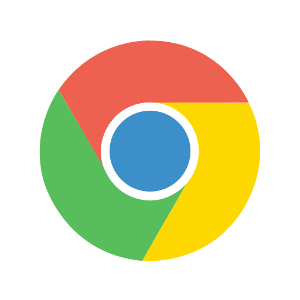
**Cordova**

**Atom**

Editor di testo per codice HTML, CSS, JS.

**Sublime Text 3**

Editor di testo per codice HTML, CSS, JS.



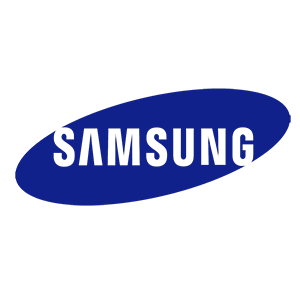
**Google Chrome**

Web browser con inspector integrato.



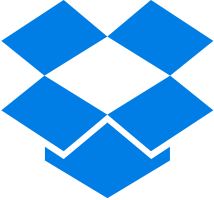
**VirtualBox**

Utilizzato per accedere alla macchina virtuale di Loveitaly.



**Smartphones**

Xiaomi mi4i, Samsung Galaxy Note 3.



**Dropbox**

Cloud per condividere i file del progetto.

**Android Studio**

Per fare il deploy dell’applicazione su sistema operativo Android.

**draw.io**

Tool online per realizzare diagrammi di ogni genere.



**Microsoft Word 2016**

Per realizzare la documentazione.

**Moqups**

Tool online per realizzare lo-fi wireframes.

\*In allegato si possono trovare le immagini a dimensioni normali

1. The max length of this document is 10 pages [↑](#footnote-ref-1)
2. The structure of this document is fixed, it cannot be changed in any way [↑](#footnote-ref-2)
3. The team leader is listed as first member in this table [↑](#footnote-ref-3)