Mobile Application Development - a.y. 2012/2013

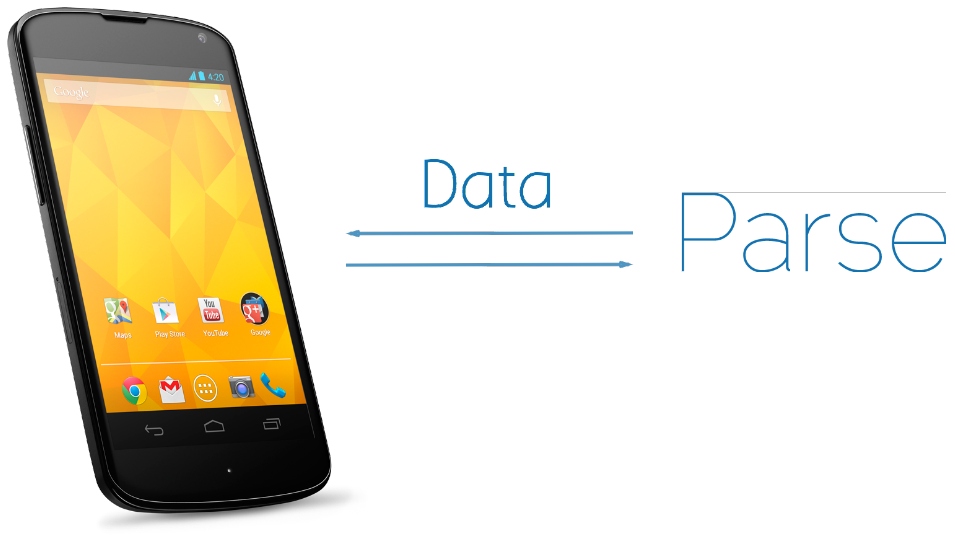
Hintsite

Technical document

Name	Student no.	email addr.
Andrea Leopardi	202879	an.leopardi@gmail.com
Federico Cicerone	218335	federico.cicerone@gmail.com
Lucio Baglione	xxxxxx	info@dehlic.com

Architecture

Architecture overview



Data sources

We used no external data sources or services. Everything has been created from scratch - the database we use has been filled with data we generated while developing the application, and will continue to be populated by user-generated data.

Technologies

The main technology we used in order to support Hintsite has been Parse.
We used Parse as a database and we used the Push Notifications service they provide in order to build the push notifications system Hintsite relies on.

We used the following set of JavaScript libraries:

- **jQuery**: Parse is developed upon the skeleton of Backbone.js, which in turn requires jQuery in order to work properly. We took advantage of this dependency and used jQuery for lots of additional stuff (CSS-style selectors, DOM manipulation, etc.).
- **Handlebars.js**: we used this template engine along with Backbone-based Parse Views.
- **Require.js**: we used Require.js in order to develop a modular application based on the latest standards (AMD modules).
- **Moment.js**: this small library converts a timestamp into an "ago"-style string; we decided to include the library and to avoid developing this functionality since the library is very light-weighted.
- **Leaflet.js**: we used Leaflet as a valid open source alternative to Google Maps. Using Leaflet, we have no limitations on daily requests (as in Google Maps) and the main functionalities are the same.
- **Hammer.js**: Hammer helped us with touch gestures. We chose to use Hammer over writing custom touch events ourselves since we needed several different gesture events.

We used Sass (SCSS) as a CSS preprocessor.

Development activities

Smart solutions

We tried to maintain our application as light as possible, in order to be able to use it on a wide range of devices.

Here's a more practical example: most of the views in Hintsite needed to connect to the Parse back-end (be it for retrieving data or uploading new ones), . So, we needed a loading feedback so that the user would know that something was actually happening. Since this would happen quite often, we decided to create a single loading view at launch time, and to just "hide" it when the other views were in the foreground. Doing this way, we reduced the creation time for the loading view, with practically no drawback (having a simple, static view in the background didn't affect performance, especially because the only animation we needed in the loading view was paused while the view was in the background, and resumed when it came to the foreground).

We also had to get our hands dirty in order to take advantage the Push Notification system provided by Parse. In fact, Parse (stille) doesn't provide a JavaScript SDK dedicated to push notifications.
Since our main testing platform was Android, we built a simple Java plugin that registers which user is related to a specific "installation" (meaning an application instance on a given device) as soon as the user logs in for the first time.
This way, when sending push notifications, we've been able to filter based on users, which allowed maximum flexibility.

Difficulties

The main difficulty we found was in the size of the application. We let ourselves go along during the planning and brainstorming phases, resulting in a pretty large application with lots of functionalities and lots of pieces to implement.

It took us some time to get used to Parse (hence Backbone) way of handling things, but in the end we really appreciated the power and flexibility provided by the combined use of the Backbone skeleton along with the Handlebars template engine, and took advantage of the separation of logic and presentation it brought us.

Log

Activity	Duration (hrs)	Location	Team members (no.)	Iterations (no.)	Outcome	Tools
BRAIN	8	home	3	4	ideas	pen & paper
COMP	3	home	3	2	business effectiveness	web
MAP	6	home	3	2	sitemap	http://draw.io, Google+
LOFI	12	home	3	5	lo-fi wireframes, usability insights	pen & paper
HIFI	8	home	3	2	hi-fi wireframes	Adobe Illistrator CS5, Inkscape
REQ	2	home	3	1		brain
FEAT	10	home	3	2	features list	brain

Activity	Duration (hrs)	Location	Team members (no.)	Iterations (no.)	Outcome	Tools
STUDY	20	home	3	10	Backbone (+Parse) background, MVC techniques tuning, clean JavaScript code	web, some books
PROOF	4	outdoors	3	1	usability tuning	mobile devices
DEV	250	home	3	45	code	*dev tools
TEST	4	home	3	2	code revisions	*testing tools
SERVER	4	home	3	2	/	http://parse.com

*dev tools

- Sublime Text 2/3
- vim
- Google Chrome, Google Chrome Canary
- git (along with GitHub)
- Ripple (Chrome extension)
- JSHint for code linting
- Grunt (task management)

*testing tools

Software
- Google Chrome, Google Chrome Canary (DevTools)
- Ripple (Chrome extension)
- XAMPP and `python SimpleHTTPServer`

Hardware
- Asus Nexus 7
- LG Nexus 4

- Samsung Galaxy Nexus
- Samsung Galaxy S2
- Samsung Galaxy Note 2