fill the void.

Porto Oculto

Software Development Report

Revision: 0.2

Client: Coletivo Oitoo and OASRN

Date of Revision: 05.31.2019

Revision History

REVISION					
REV.	**DATE**	**DESCRIPTION**	**BY**	**REVISION**	**APPROVED BY**
0.1	05.25.2019	Initial version	Carlos	0.1	Erica
0.2	05.31.2019	Adding iOS version	José	0.2	Erica

1. Purpose of the document

The purpose of this document is to convey relevant technical documentation for maintenance purposes for Coletivo Oitoo and OASRN. The document describes the Porto Oculto iOS and Android app's structure and offers suggestions for future maintenance.

2. Definitions and Abbreviations

Initials	Description
MVVM	Model-View-ViewModel, an architectural pattern
DAO	Data Access Object, a coding pattern
SDR	Software Development Report
API	Application Programming Interface

Table 1 - Definitions and Abbreviations

3. System Architecture

Diagram 1 depicts core interactions between the application and external dependencies, namely Google Maps and the Firebase service.

The Google Maps API is responsible for providing a real-time map of the city of Porto, along with an accurate representation of the user's current location in the area.

The Firebase API handles most account-related information, including but not limited to logging in, creating new accounts, storing images and related information, email validation and personal map markers.

Diagram 2 showcases the main form of interaction between system components, which makes full use of the MVVM architectural pattern.

For a more in-depth description of the rules and structure of database communication and component interaction used in the development of this application, please check section 5.1. "Primary structures and rules" of this SDR.

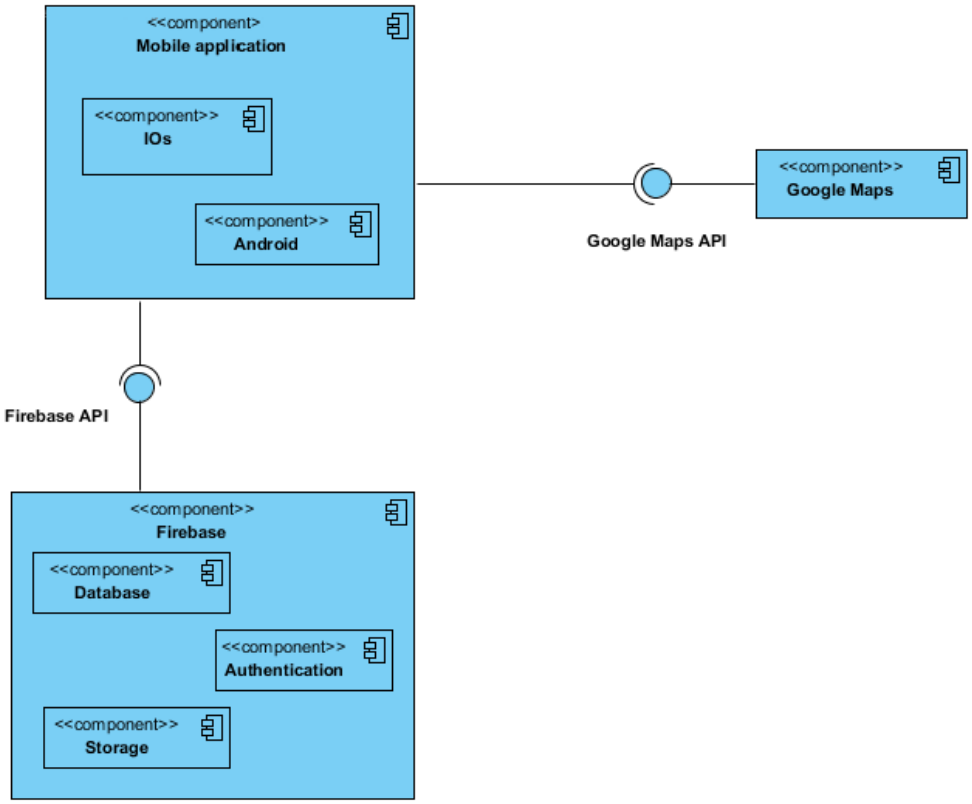

Diagram 1 – Communication with external services. Both Google Maps and Firebase expose APIs that our application consumes.

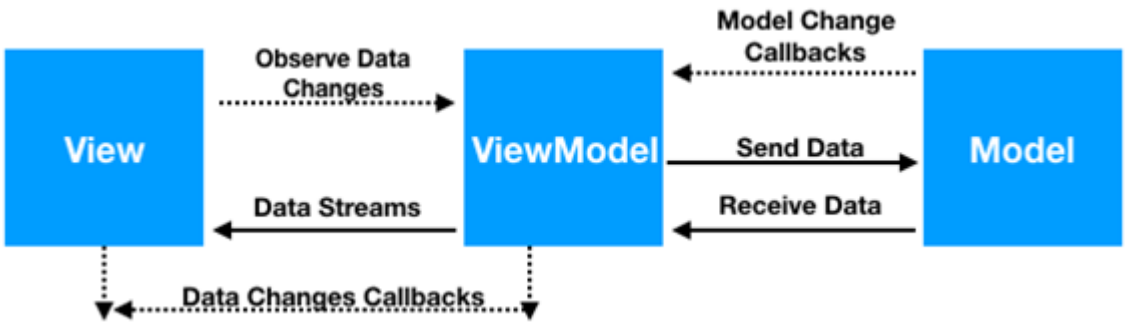

Diagram 2 – MVVM architectural pattern. "View", "ViewModel" and "Model" are packages, each containing multiple classes.

4. Database documentation

The database used by the application is the cloud-based Firebase service, which encompasses all of Firebase Storage, Firebase Database and Firebase Authentication. Calls made to any of these asynchronous-friendly services follow the DAO pattern (see section 5.1. "Primary structures and rules" of this SDR for more).

Firebase Authentication lists existing accounts and their status (e.g. email validation) and links other services such as Facebook and Gmail to the application, providing the option to log in with accounts from those platforms.

Firebase Storage serves as an efficient collection of pictures takes by users while using the application, allowing for addition, retrieval and deletion of large blobs of data at any time.

Firebase Database holds the bulk of the information used by Porto Oculto. Building data, map markers, properties, and other user account information are stored in this sub-service.

5. Code structure

5.1 Primary structures and rules

The code follows the MVVM architectural pattern. This gives the application a low amount of coupling between views and business logic, which makes it easier and faster for developers to implement changes to user interfaces, features and testing processes. In the MVVM pattern, there are three main roles. Each component of the application must have one and only one of those roles.

- Model – anything that stores information and performs processing of data.
- View – the user interface of the application. It presents the data to the user and allows for interaction with it.
- ViewModel – a link between views and models. It transforms data from models and updates views.

Interactions with the online database follow the DAO pattern. This pattern provides a high level of abstraction for services that rely on low-level operations and API calls. High-level business logic is thus both decoupled from lower level services and simplified, providing extra flexibility in the event that the database structure may change.

5.2 Classes and packages

The Engine package contains the DAO interfaces and their respective implementations. It abstracts interaction with the database for account registration, logging in and map marker handling.

Model, View and ViewModel packages hold all classes with those respective roles, following the MVVM pattern.

The Util package includes a multitude of abstractions and helper methods that are used in many other files.

6. API

For the Porto Oculto applications, there is no need to develop a custom API. Google Maps and Firebase provides all the APIs necessary to build both iOS and Android applications.

7. Unit tests

For the Android application, unit tests were added, and a CI service is in place to run all automated tests after a git push is made.

8. Any other relevant topics

8.1 Transparency

The entire codebase and structure of the iOS and Android applications are consistently updated and stored in an online repository on GitLab. The repository can be accessed through the URLs:

- https://gitlab.com/lgp-3b/lgp-3b-android
- https://gitlab.com/lgp-3b/lgp-3b-android (permission must be given)
- https://git.fe.up.pt/lgp2019/lgp-3/lgp-3b-ios (permission must be given)
- https://gitlab.com/lgp-3b/lgp-3b-ios

In it, all changes made to the code are recorded and connected to issues (i.e. individual problems that require a solution).

8.2. Related information

The Software Architecture Document also covers some of the topics that are described in this report, such as system and code structure. As such, it may potentially provide additional information not included in this SDR.

9. Appendixes

<link to software architecture document?>