Software Engineering 2

Travlendar+

Installation **G**uide

Menchetti Guglielmo
Norcini Lorenzo
Scarlatti Tommaso

January 07, 2018

Contents

1 Introduction

This document is a brief guide that allows for a easy setup of both the client and the back-end application, specifically it refers to the installation on a Linux Ubuntu environment.

This is only a setup guide, more information about the implementation is available in the associated Implementation and Testing Document.

In order to allow easier testing of its functionalities the application is also currently online and reachable at the 168.235.96.252 ip address.
The APIs respond on the port 8080 and their usage is explained in the aforementioned document.
The client application provided is already setup to interact with this server.

For any other request or clarification feel free to reach us at:

lorenzonorcini@icloud.com
scarlattitommaso@gmail.com
guglielmomenchetti.gm@gmail.com

2 Database Setup

In this paragraph we illustrate a basic postgres setup, replace the <> fields with your information.

Install the postgres package

```
$ sudo apt-get install postgresql postgresql-contrib
```

Create a new postgres user

```
$ sudo -u postgres createuser <username>
```

Create a new database

```
$ sudo -u postgres createdb <dbname>
```

Open postgres console

```
$ sudo -u postgres psql
```

Assign a password to the user

```
$ (postgres) alter user <username> with encrypted password <password>;
```

Grant privileges to the user over the database

```
$ (postgres) grant all privileges on database <dbname> to <username>;
```

if you plan to install the DBMS on a different machine from that of the application you may want to enable port forwarding and change postgres setup to allow external requests.

3 Mail Account

You may want to create a new email account which will be used to send email on behalf of the application, in our configuration we used Gmail but any mail provider should be fine.

4 External Services Setup

In this paragraph we illustrate the necessary setup for the external services API used.

4.1 Google API

In order to get the Google API key follow the instructions in the following link

https://developers.google.com/maps/documentation/directions/get-api-key

4.2 MapBox API

In order to get the MapBox API key sign up using the following link

https://www.mapbox.com

Once logged in, go to account→dashboard and create a new token selecting the scopes "DATASETS:READ", "DATASETS:LIST" and "DATASETS:WRITE".

4.3 Uber API

In order to get the Uber API key sign up using the following link

https://developer.uber.com

Once logged, create a new App. From the dashboard is possible to get the *Server Token.*
Once the iOS application is created, the following changes have to be applied:

- In the Dashboard Auth page, edit the *redirect URI* field with

  <APPLICATION BUNDLE ID>://oauth/consumer

- In the Dashboard→Settings→Security, edit the *App Signatures* with

  <APPLICATION BUNDLE ID>

In order to use Uber with more accounts, in the Dashboard→Developers settings the Uber accounts must be added.

4.4 Usage of the keys

Once all the keys are retrieved, open the file *ApiKeys.php* (*app/Http/Helper/ApiKeys.php*) and edit the variable value:

```
$googleKey = <YOUR GOOGLE API KEY>

$mapboxKey = <YOUR MAPBOX API KEY>

$uberServerToken = <YOUR UBER SERVER TOKEN>

$uberClientToken = <YOUR UBER CLIENT TOKEN>
```

N.B. This key is used only used in order to test the application. This key is available only after the authentication in the Uber application. The Authentication protocol is omitted.

5 Application Back-end Setup

In this paragraph we illustrate how to install the dependencies and initialize the application.

5.1 Instructions

Install Laravel dependencies

```
$ sudo apt-get install php

$ sudo apt-get install openssl

$ sudo apt-get install php7.1-pgsql

$ sudo apt-get install php7.1-mbstring

$ sudo apt-get install php7.1-xml
```

Install other dependencies

```
$ sudo apt-get install curl

$ sudo apt-get install php7.1-curl
```

Install PHP dependency manager

```
$ sudo apt install composer
```

Install Python package manager

```
$ sudo apt install python3-pip
```

Install python package for CSP

```
$ pip3 install python-constraint
```

Decompress the zip file containing the application files.
Configure your '.env' file in the root folder of the project filling the <> fields with your information, the other fields may be left empy or with the default values.

```
APP_NAME=Travlendar
APP_ENV=local
APP_KEY=
APP_DEBUG=true
APP_LOG_LEVEL=debug
APP_URL=http://localhost

DB_CONNECTION=<dbms identifier or pgsql for postgres>
DB_HOST=<dbaddress or 127.0.0.1 if db is local>
```

```
DB_PORT=<dbport or 5432 as default>
DB_DATABASE=<dbname>
DB_USERNAME=<username>
DB_PASSWORD=<password>

BROADCAST_DRIVER=log
CACHE_DRIVER=file
SESSION_DRIVER=file
SESSION_LIFETIME=120
QUEUE_DRIVER=database

REDIS_HOST=127.0.0.1
REDIS_PASSWORD=null
REDIS_PORT=6379

MAIL_DRIVER=smtp
MAIL_HOST=<mail host or smtp.gmail.com for gmail>
MAIL_PORT=587
MAIL_USERNAME=<mailaccount>
MAIL_PASSWORD=<mailpassword>
MAIL_ENCRYPTION=tls
MAIL_FROM_ADDRESS=<mailaddress>
MAIL_FROM_NAME=Travlendar+

PUSHER_APP_ID=
PUSHER_APP_KEY=
PUSHER_APP_SECRET=
```

From the root folder of the project.

Generate an application key

```
$ php artisan key:generate
```

Migrate the tables to the database

```
$ php artisan migrate --seed
```

Generate oauth keys

```
$ php artisan passport:install
```

Remove from the 'composer.json' file

```
"phpunit/phpunit": "~6.0"
```

Run

```
$ composer update
```

Add the previously deleted line and rerun the update.
(This may seem unnecessary but it fixes a missing symlink caused by migrating the project and allows to run the test suite).

Start Laravel Development Server

```
$ php artisan serve --host=<host_addr> --port=<port>
```

Run the tests

```
$ vendor/bin/phpunit
```

6 Client Application Setup

In this section, all the relevant information to build the iOS application are provided as clearly as possible.

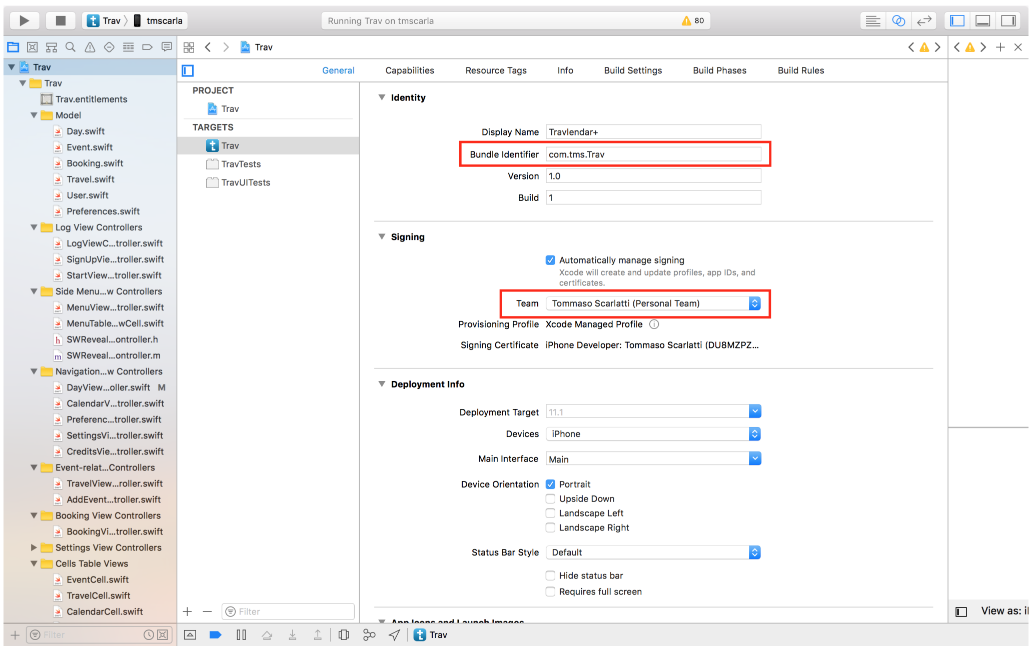
6.1 Requirements

- OSX with Xcode 9.x
- Apple ID
- Internet connection
- (Optional) iPhone with iOS 10+

6.2 How to build

In order to build the application, you need to open with Xcode the file *Trav.xcworkspace* inside the iOS/Trav folder.

The indexing phase may take up to one or two minutes. Then, choose a simulator device at the top left corner of the Xcode and press the play button to build. Alternatively, you can connect your own iPhone and run the app on it directly.



Identity

Display Name: Travlendar+
Bundle Identifier: com.tms.Trav
Version: 1.0
Build: 1

Signing

✓ Automatically manage signing
Xcode will create and update profiles, app IDs, and certificates.
Team: Tommaso Scarlatti (Personal Team)
Provisioning Profile: Xcode Managed Profile
Signing Certificate: iPhone Developer: Tommaso Scarlatti (DU8MZPZ...

Deployment Info

Deployment Target: 11.1
Devices: iPhone
Main Interface: Main
Device Orientation: ✓ Portrait, ☐ Upside Down, ☐ Landscape Left, ☐ Landscape Right
Status Bar Style: Default
☐ Hide status bar
☐ Requires full screen

Figure 1: General options

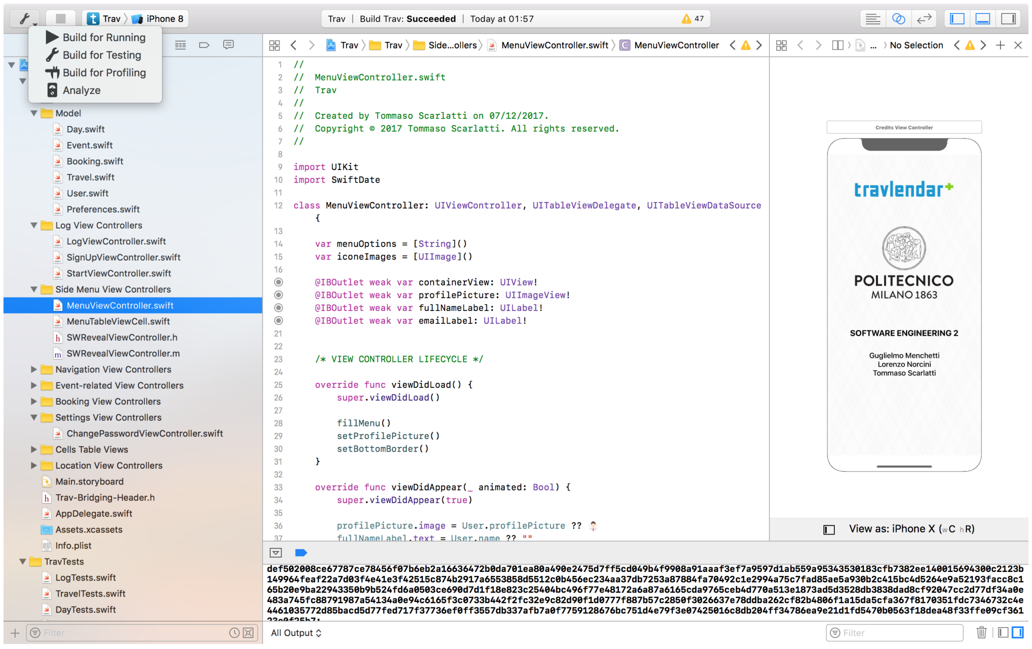
You have to change the team name with your personal Apple ID (or a team name, if you have one) and consequentely the Bundle Identifier, as shown in the image above.

Info.plist

Inside the project folder you can find a file called *Info.plist*, which is a structured text file that contains essential configuration information for a bundled executable. It is basically a dictionary with a [key : value] mapping.

At the bottom of the list you can find to keys: *Server Name* and *Travlendar Client Secret* that can be both easily replaced in case you want to set up the back end on a different server.

Here you can also find the keys related to the Uber iOS sdk setup if you want to change them in order to make your account elegible to perform rides requests.



```swift
//
//  MenuViewController.swift
//  Trav
//
//  Created by Tommaso Scarlatti on 07/12/2017.
//  Copyright © 2017 Tommaso Scarlatti. All rights reserved.
//

import UIKit
import SwiftDate

class MenuViewController: UIViewController, UITableViewDelegate, UITableViewDataSource
    {

    var menuOptions = [String]()
    var iconeImages = [UIImage]()

    @IBOutlet weak var containerView: UIView!
    @IBOutlet weak var profilePicture: UIImageView!
    @IBOutlet weak var fullNameLabel: UILabel!
    @IBOutlet weak var emailLabel: UILabel!


    /* VIEW CONTROLLER LIFECYCLE */

    override func viewDidLoad() {
        super.viewDidLoad()

        fillMenu()
        setProfilePicture()
        setBottomBorder()
    }

    override func viewDidAppear(_ animated: Bool) {
        super.viewDidAppear(true)

        profilePicture.image = User.profilePicture ??
        fullNameLabel.text = User.name ?? ""
```

```
def502008ce67787ce78456f07b6eb2a16636472b0da701ea80a490e2475d7ff5cd049b4f9908a91aaaf3ef7a9597d1ab559a95343530183cfb7382ee140015694300c2123b
149964feaf22a7d03f4e41e3f42515c874b2917a6553858d5512c0b456ec234aa37db7253a87884fa70492c1e2994a75c7fad85ae5a930b2c415bc4d5264e9a52193facc8c1
65b20e9ba22943350b9b524fd6a0503ce690d7d1f18e823c25404bc496f77e48172a6a87a6165cda9765ceb4d770a513e1873ad5d3528db3838dad8cf92047cc2d77df34a0e
483a745fc88791987a54134a0e94c6165f3c0733b442f2fc32e9c82d90f1d0777f887b57c2850f3026637e78ddba262cf82b4806f1a15da5cfa367f8170351fdc7346732c4e
4461035772d85bacd5d77fed717f37736ef0ff3557db337afb7a0f7759128676bc751d4e79f3e07425016c8db204ff34786ea9e21d1fd5470b0563f18dea48f33ffe09cf361
```

Figure 2: Build options

6.3 How to test

If you want to perform the integration tests for the HTTP request to the server you can hold the play button, as in the figure above, and choose the "Build for Testing" option. Doing like this will build all the tests located in the "TravTest" folder.

If you want to perform a single test, you can swap the left view form the current *Project navigator* to the *Test navigator*.

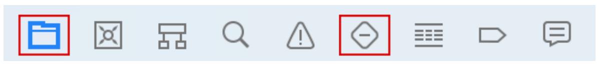


Figure 3: Project navigator and Test navigator

6.4 Known issues

- **Auto-layout warnings**: there are plenty of warnings due to UI constraints.
- **UI issues**: the application was tested on the iPhone 8 simulator and on a physical iPhone 7. Due to constraints warnings, the view may not perform as expected.

- **Uber requests**: due to limitations imposed by Uber, you need to setup your personal Uber developer account in order to make Uber rides requests, as explained in section 4.3.