Perform 2018
Mobile Performance Management for Native Mobile Applications

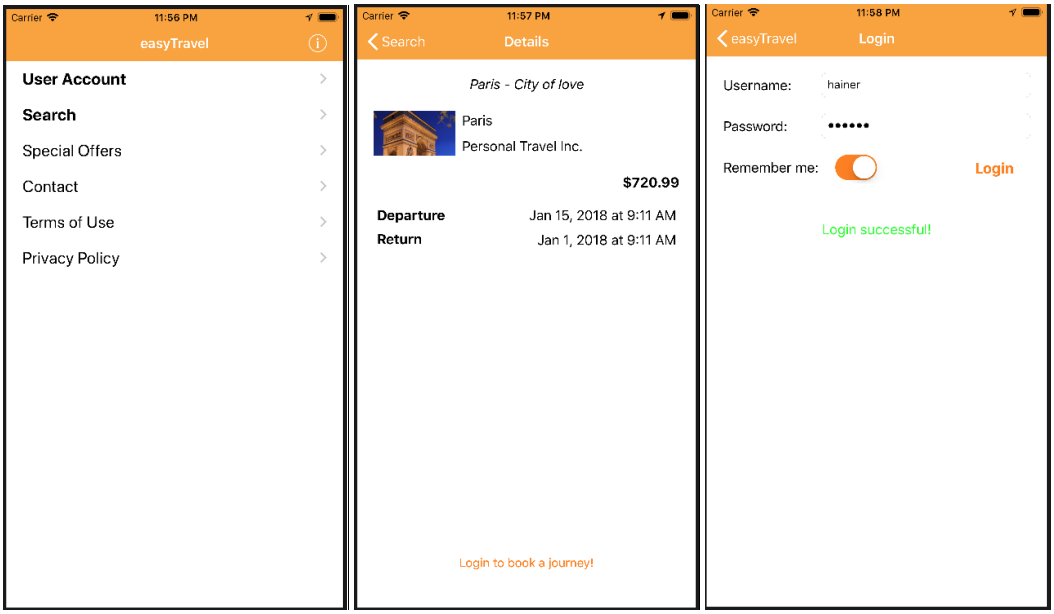
Hands-on Training Instructions

Lesson 5 – Auto-instrumentation for iOS Native Mobile Application

Lesson 2 - Auto-instrumentation for iOS Native Mobile Application

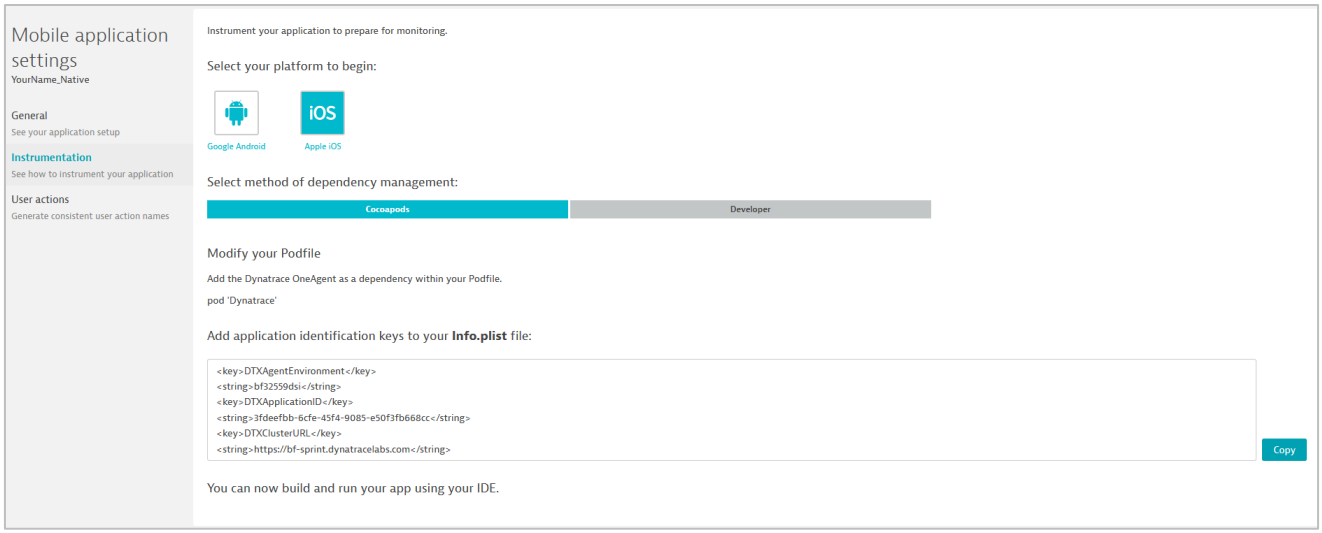
Importing the project in Xcode

1. Launch Xcode
2. Open the easyTravel project (navigate to the **easyTravel-ios** directory, file **easyTravel.xcodeproj**)
3. Run the application. Navigate in the application; you will see it has the same features as the Android application.



Instrumenting the application with Dynatrace

4. In Dynatrace, you can use the same application that you created for Android Native (lesson 2) or you can create a new one. The procedure is the same what we did for Android.
5. Get to the Application Settings -> Instrumentation screen (if you reuse your existing Application, this is accessible by clicking the **…** button and then **Edit**). Select the Cocoapods tab.



6. If you don't have **Cocoapods** installed on your Mac, you will need it. **Cocoapods** is built with Ruby and is installable with the default Reuby available on OS X.
    - Open a Terminal window and execute the following command : **sudo gem install cocoapods**

7. From the command line interface, get to your **easytravel-ios** project directory.
8. Add the Dynatrace Pod to your project.
    - Create a Podfile by executing this command : **pod init**
    - Open the Podfile with a text editor (e.g. **nano Podfile**)
    - Add the following line : **pod 'Dynatrace'**
    - Save and exit
    - Execute this command : **pod install**

9. Close your Xcode project and re-open it, but this time using **easytravel.xcworkspace**
10. In your project, find the **easyTravel-Info.plist** file (in the Supporting Files), open it as source code (ctrl-click **Open as source code**) and add the application identification keys copied from the I**nstrumentation** screen in Dynatrace (see step 5).
11. Run the application
12. Navigate in the application, executing a few actions such as log in (User Information), booking destinations either from Search or Special Offers, etc. You can also crash the app using one of the available option.
13. Send the application in the background, wait around 2 minutes and analyze the data collected by Dynatrace the same way as we did in lesson 2 for Android.

CHALLENGE

In your Xcode project, try to do the rest of the exercises that we did for Android

- Crash reporting, symbolication
- User tagging
- Customer user actions
- Error reporting

The Dynatrace API for iOS is similar to what you have seen with Android. Use the documentation as a reference : https://www.dynatrace.com/support/help/user-experience/mobile-apps/how-are-manual-api-calls-used-to-enrich-mobile-user-experience-data/

Additional documentation and code examples are available in **Settings -> Web and mobile monitoring -> Custom user actions** and also here : https://ruxitdev-repo-m5.s3.amazonaws.com/pipeline/sprint/js_agent/1.137.58.20180117-071433/dynatraceapi-1.137.0.20171222-

133618.zip?AWSAccessKeyId=AKIAJH2YTPUAP6LDUUMQ&Expires=1893456000&Signature=5B41GSnOodnmjfw0Ryb8A8b7fl4%3D