Perform 2018
Mobile Performance Management for Native Mobile Applications

Hands-on Training Instructions

Lesson 2 – Auto-instrumentation in Android Studio

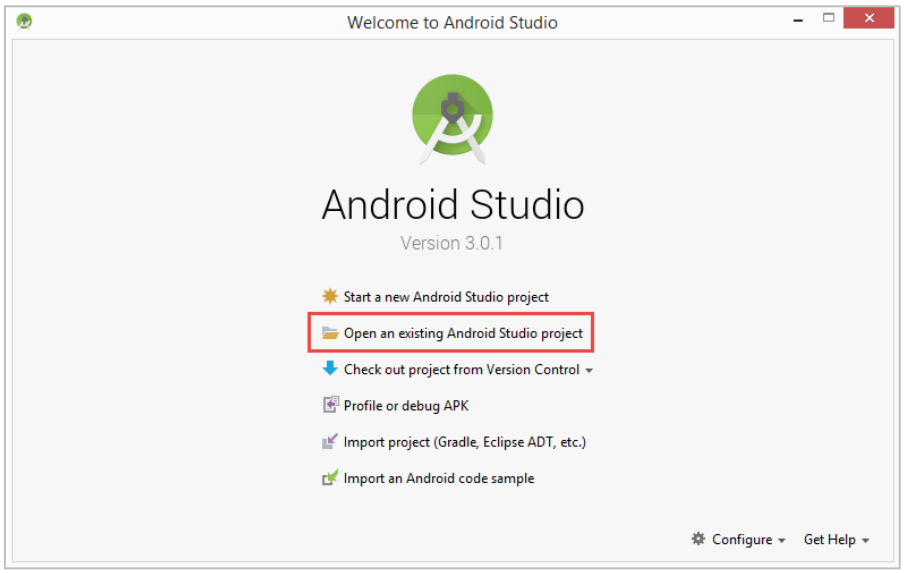
Preparation - Download the mobile applications projects

1. Launch your command line interface
   a. Windows : Windows button -> type “cmd” at the prompt
   b. MacOS : Search for “Terminal” in the Spotlight Search
2. Create a directory structure for your projects. **It is important that the directory path does not contain any space**. For example, create directly from the C drive or from your MacOS drive
   a. Windows : **md c:\projects**
   b. MacOS : **mkdir projects**
3. Switch to your projects directory :
   a. Windows : **cd c:\projects**
   b. MacOS : **cd projects**
4. Clone the mobile-hotday-2018 repository from Github
   a. git clone https://github.com/Dynatrace/mobile-hotday-2018.git

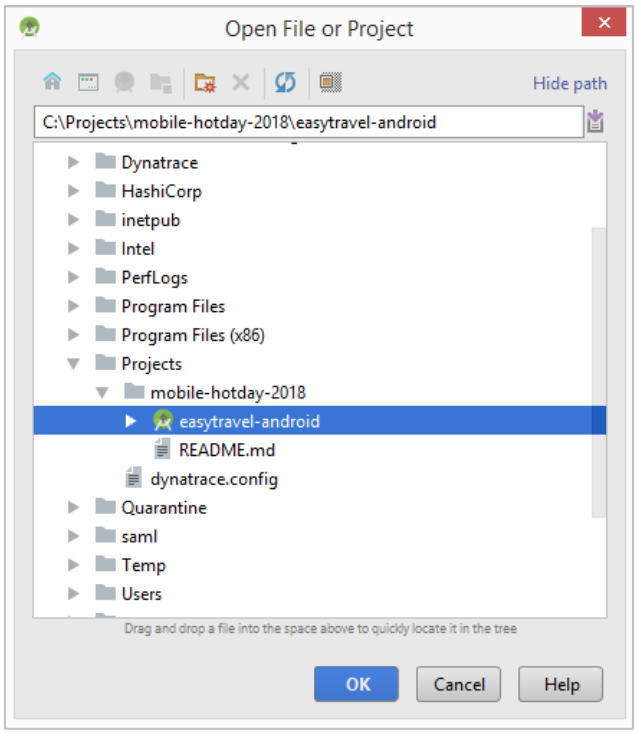
Lesson 2 - Auto-instrumentation in Android Studio

Importing the project in Android Studio

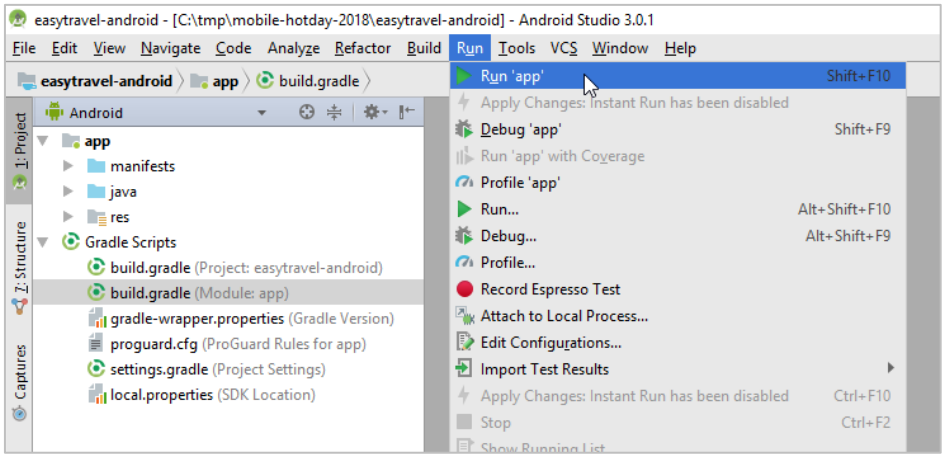
1. Launch Android Studio
2. Select **Open an existing Android Studio project**



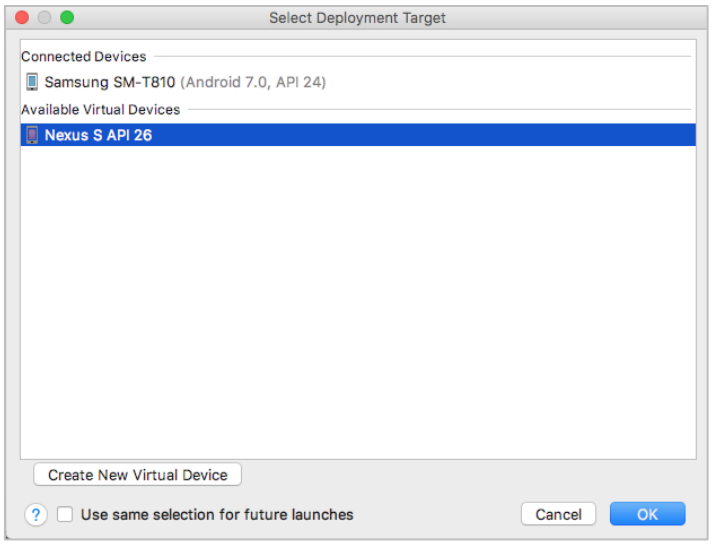
3. Browse to your project directory and select the **easytravel-android** directory

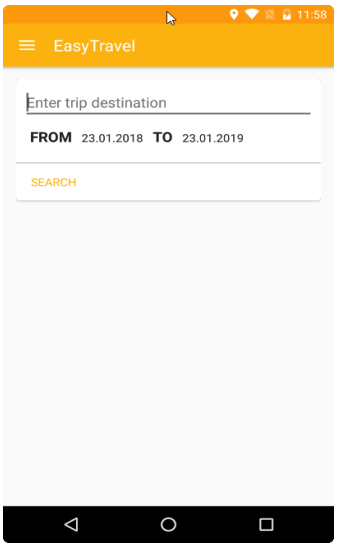


4. Wait for Gradle to complete the build (look for “Gradle build finished” at the bottom left, in the status bar). Run the **easyTravel** application by clicking on the **Run** button.
5. 

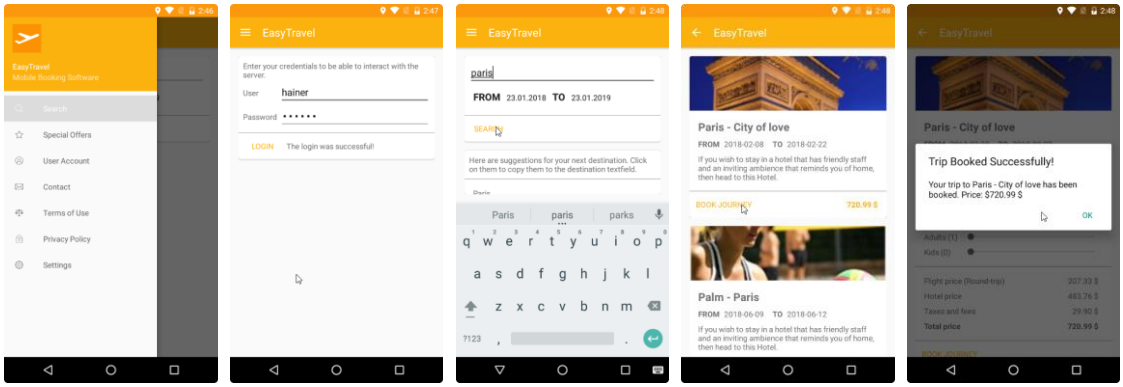


6. Select a device (either physical or virtual) to deploy the application.





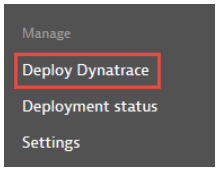
7. Navigate and explore the functionalities of the application
    - Log in
    - Book a journey (e.g. search for Paris)
    - View the special offers
    - Settings will be used later in the exercise to simulate problems



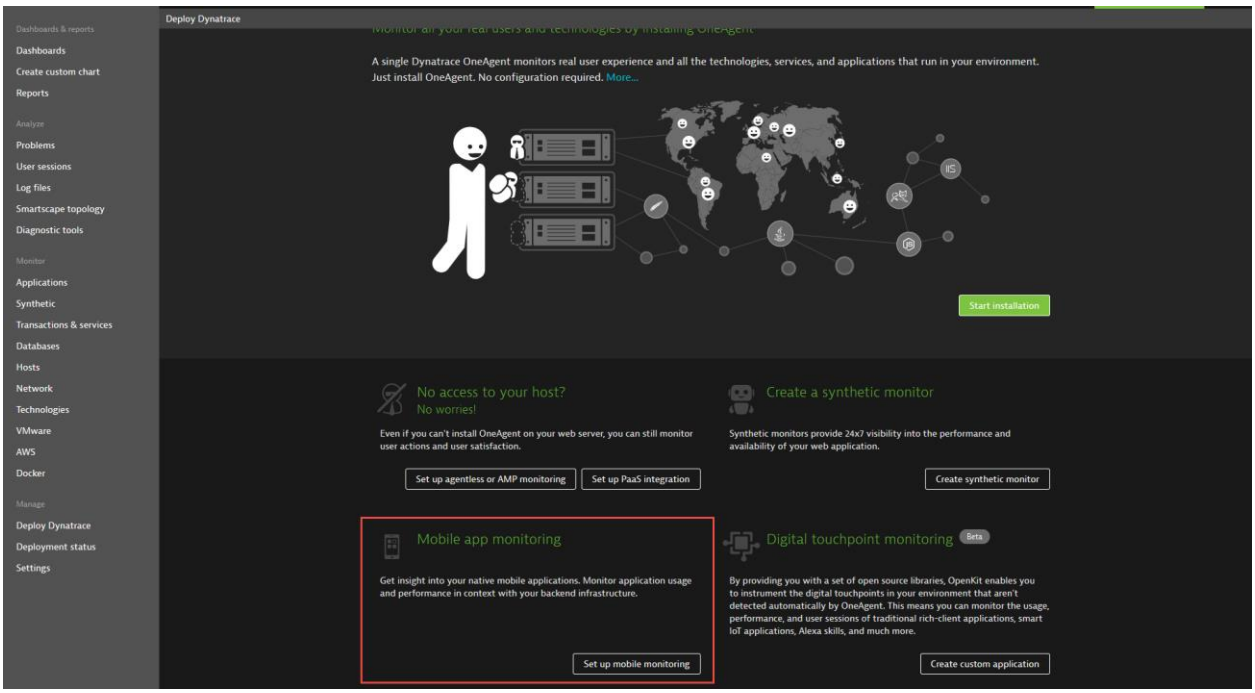
8. Although it is optional, you might want to uninstall the application from the device before we instrument it.

Instrumenting the application with Dynatrace

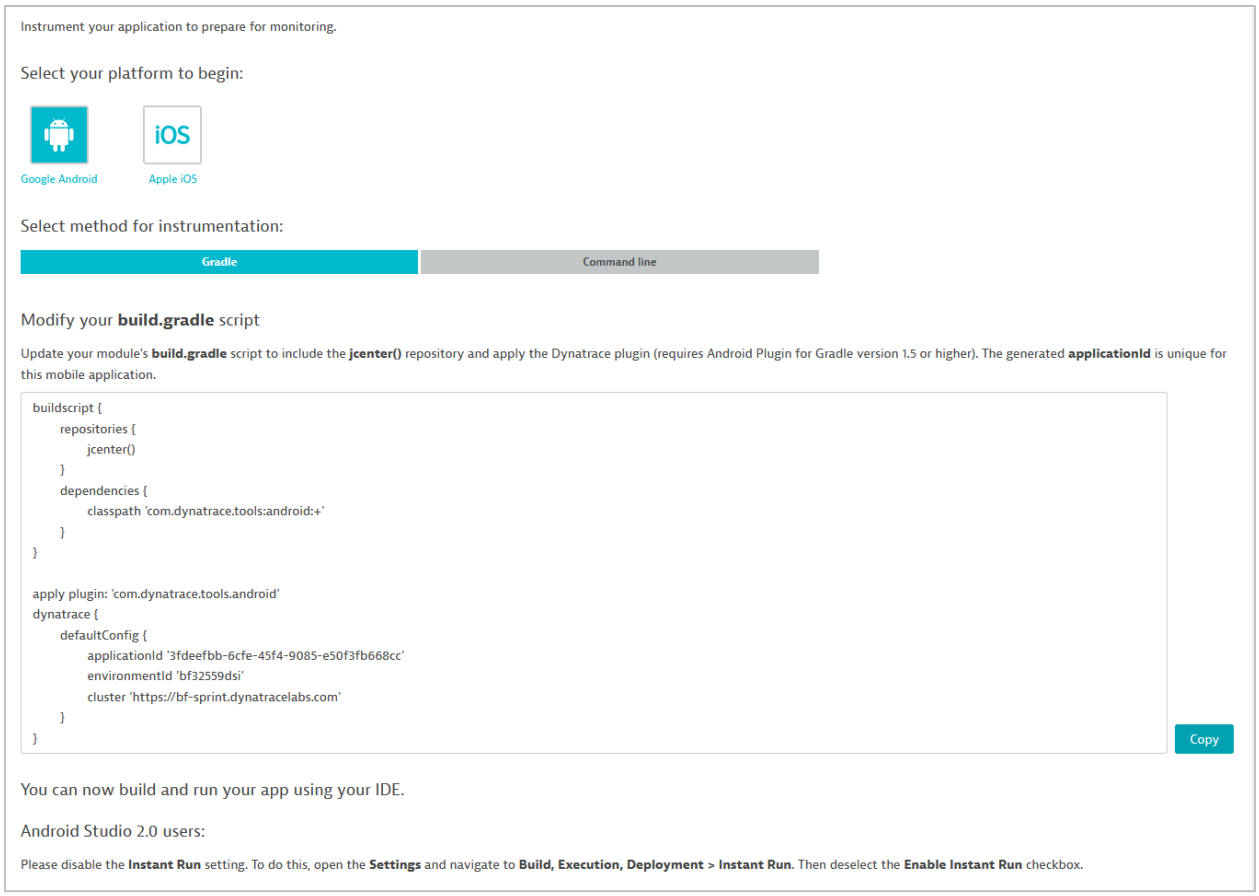
9. Now, you will add the configuration to have Dynatrace monitor the application. First, we will define the native mobile application. In the Dynatrace portal, on the left menu at the bottom, click on **Deploy Dynatrace**



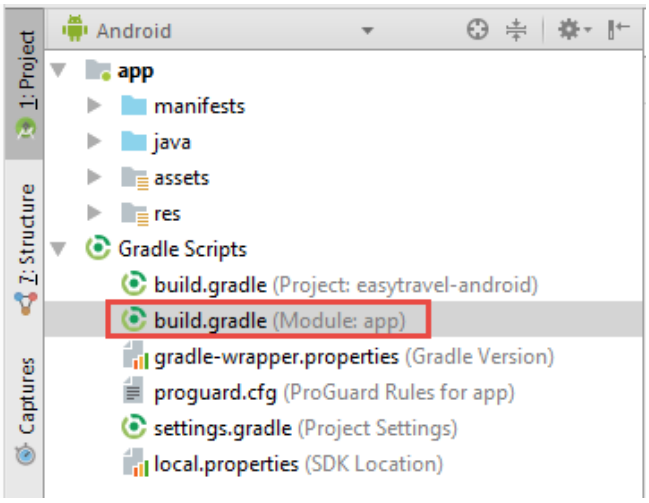
10. On the Deploy Dynatrace screen, go to Mobile app monitoring and click on **Set up mobile monitoring**



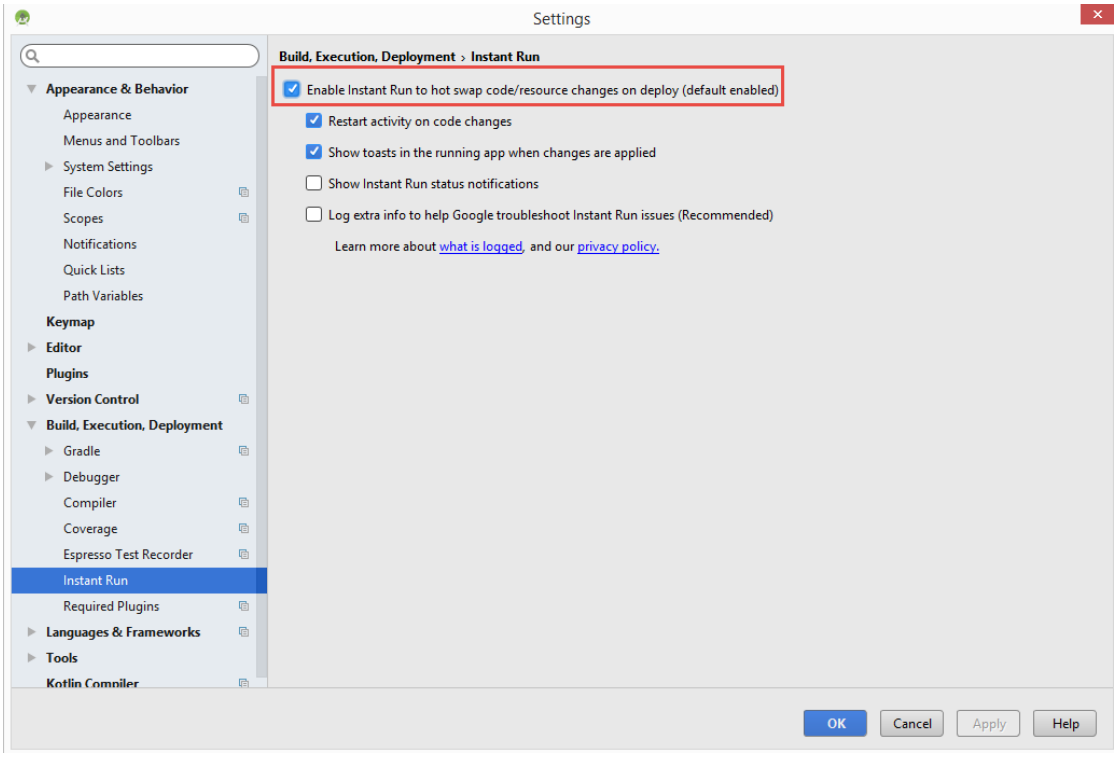
11. Enter an application name. For consistency, please use the following syntax: YourName_Native (replace the YourName string by your First_LastName ☺). Click on **Create mobile app**.
12. Click on the Google Android platform button. The screen will display the snippet that needs to be copied in your Android project **build.gradle** script



13. In Android Studio, open the **build.gradle** script.
    1. There will be 2 **build.gradle** scripts. Use the one for the module.

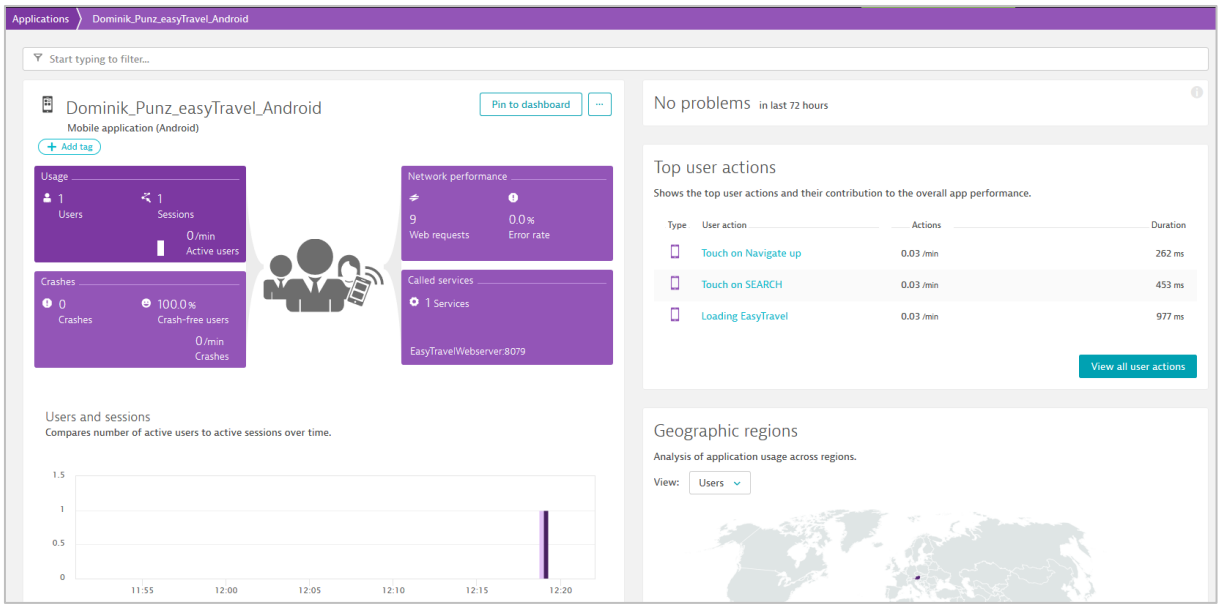


14. Find **TODO (1)** in the gradle script and insert the snippet you copied from Dynatrace.
15. **Warning:** the Android agent is not compatible with the Android Studio **Instant Run** feature. Before running the application on a device, you need to make sure **Instant Run** is disabled. In the Android Studio menu go to **Files->Settings**. In the settings vertical menu, go to **Build, Execution, Deployment->Instant Run**. Uncheck the **Enable Instant Run…** check box. Click OK.

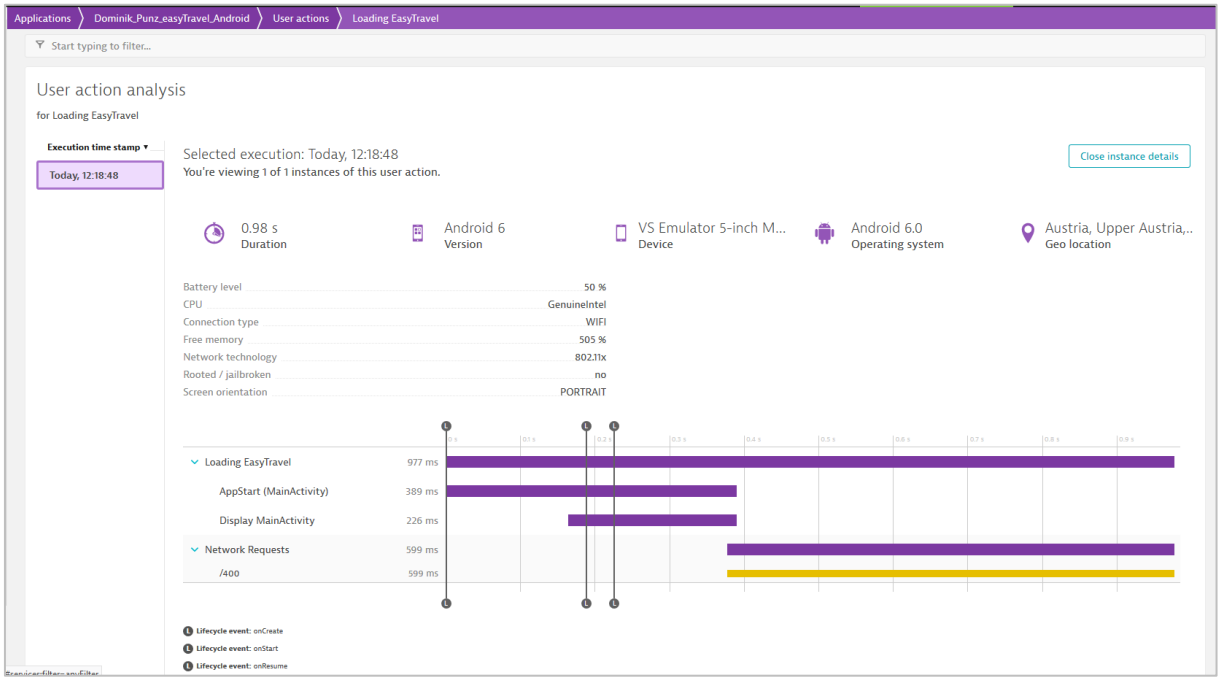


16. Wait for the Gradle sync to complete and launch the application by clicking on the run button and select a device to deploy it.
17. In the Gradle console you will notice output from the auto-instrumentor.
    - Search for Paris and try to book the journey.
    - Then close the app.
    - This will trigger the mobile agent to send the data to Dynatrace.

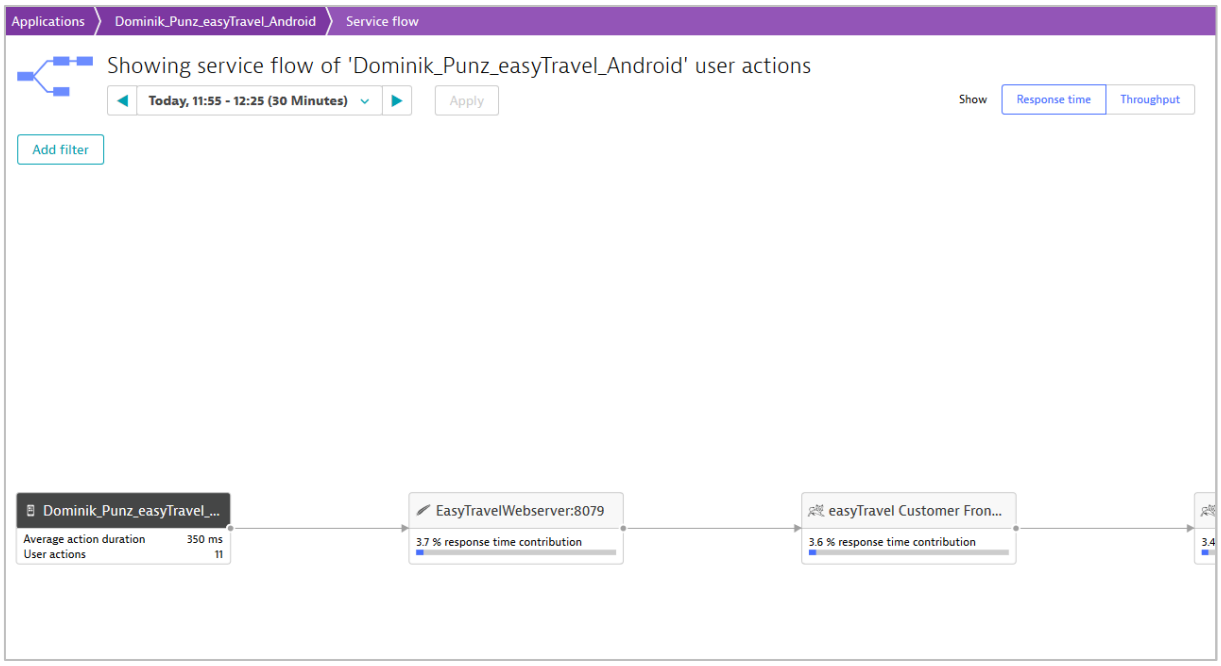
18. Return to the Dynatrace console, close the settings and go to your application screen by clicking the bread crumb and wait for about 2 minutes. You will then see the first details flowing into the app screen.
19. View the usage statistics, network performance, crashes and services section.



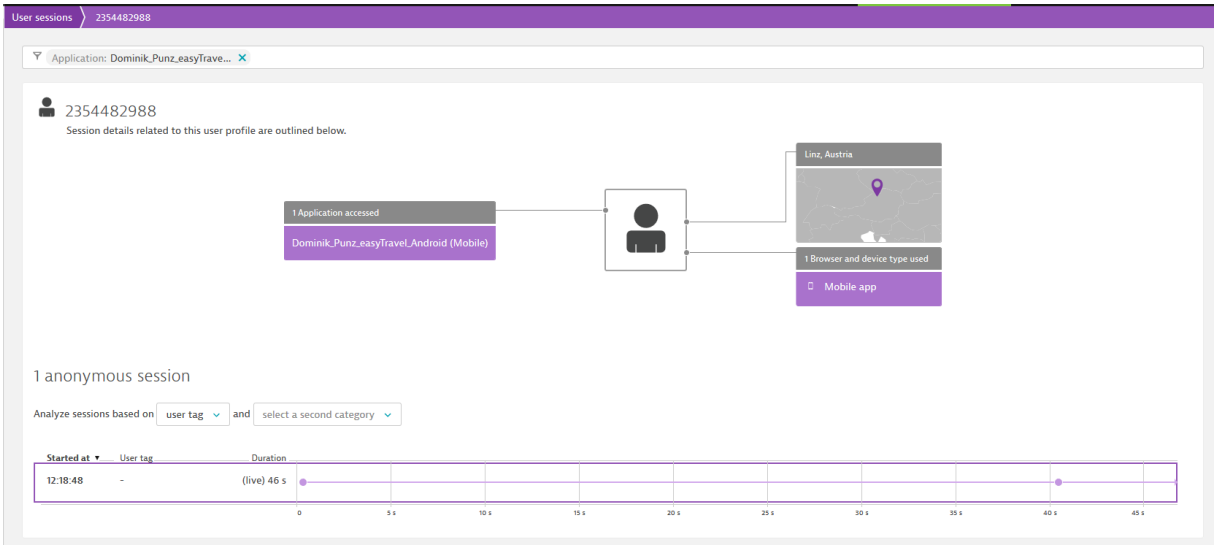
20. Analyze the user actions and locate the **Loading EasyTravel** user action and view the details.



21. Go back to your application screen (again, you can use the breadcrumb), click on **Called Services** in the infographic and view the **Service Flow**.

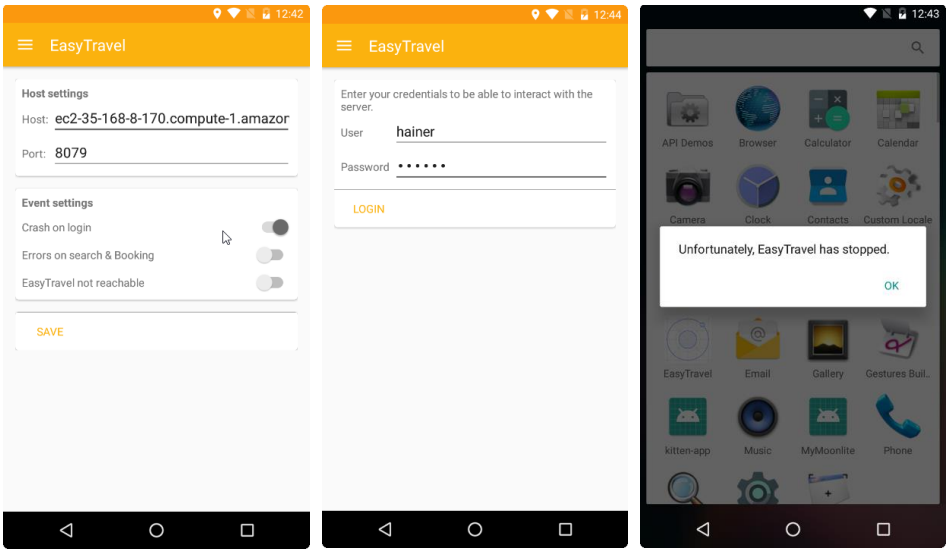


22. Go back to the application main screen and click on **Analyze user sessions** (button at the bottom – you might need to scroll down).

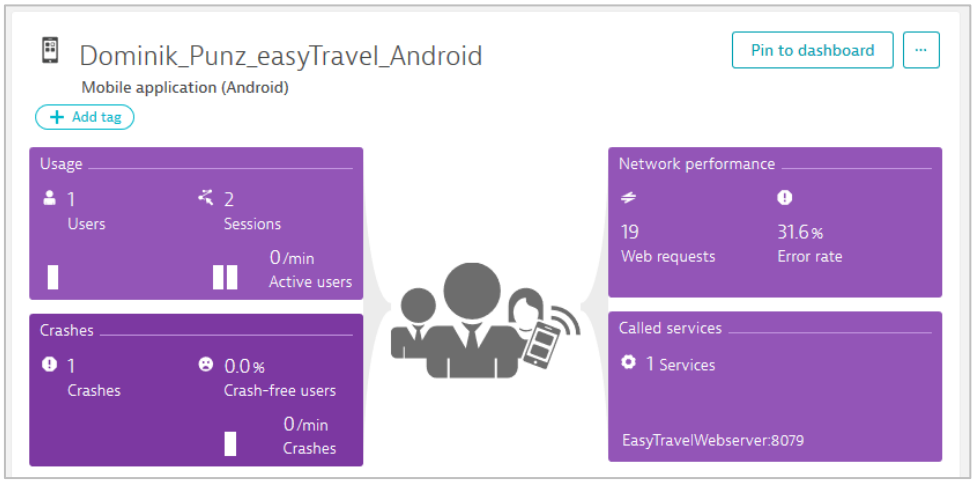


Crash Reporting

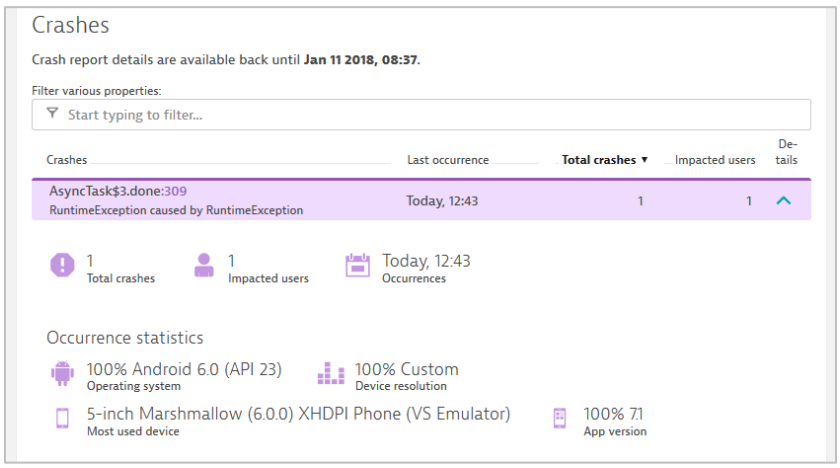
23. Return to Android Studio and run the application again.
24. In the menu, select **Settings** and enable **Crash on login** and click the **Save** button.
25. In the menu, select **Login**, enter a user name and password (for example, maria / maria) and click **Login**.



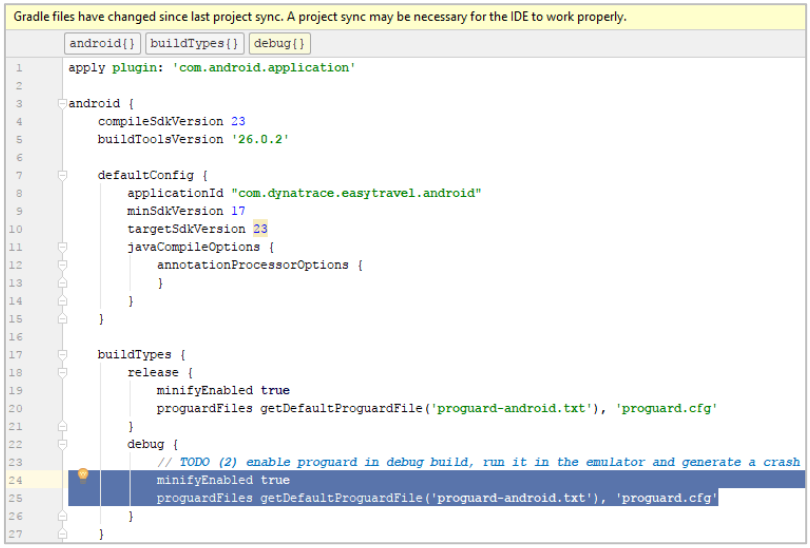
26. Return to the Dynatrace console and wait 2 minutes. The crash will show up in the infographic.



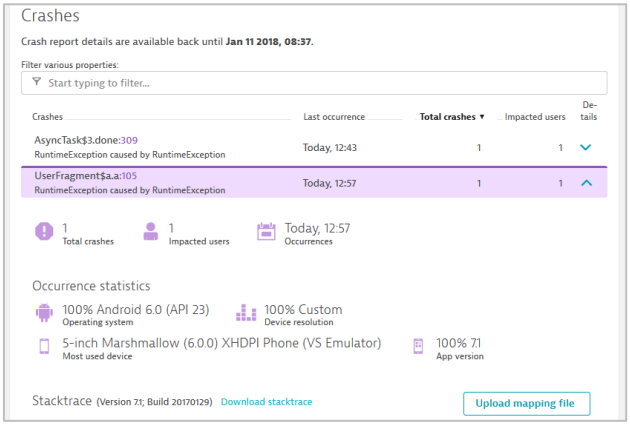
27. View the crash details and notice that the stack trace is readable.



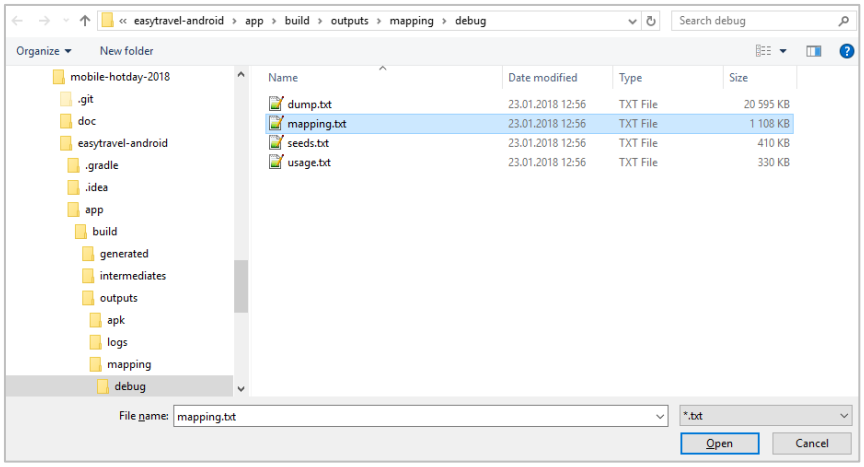
28. Often, mobile applications are obfuscated so the crash stack trace is not readable. Let's modify the application and use Proguard to obfuscate it. In Android Studio, go to the module gradle script and locate the **TODO (2)**.
29. Enable Proguard in the **debug** build by uncommenting the 2 lines (remove the //) under **TODO (2)** and sync Gradle.



30. Run the app again, trigger the crash and go back to Dynatrace. After 2 minutes the crash will show up. Notice the difference to the first crash. This time it is obfuscated.



31. Click on **Upload mapping file** and locate the **mapping.txt** file in the **easytravel-android/app/build/outputs/mapping/debug** directory.



32. View the unobfuscated stack trace in Dynatrace

33. View the list of available mapping files