Perform 2018
Mobile Performance Management for Native Mobile Applications

Hands-on Training Instructions

Lesson 3 – Instrumentation enhancement using the Dynatrace API for Android

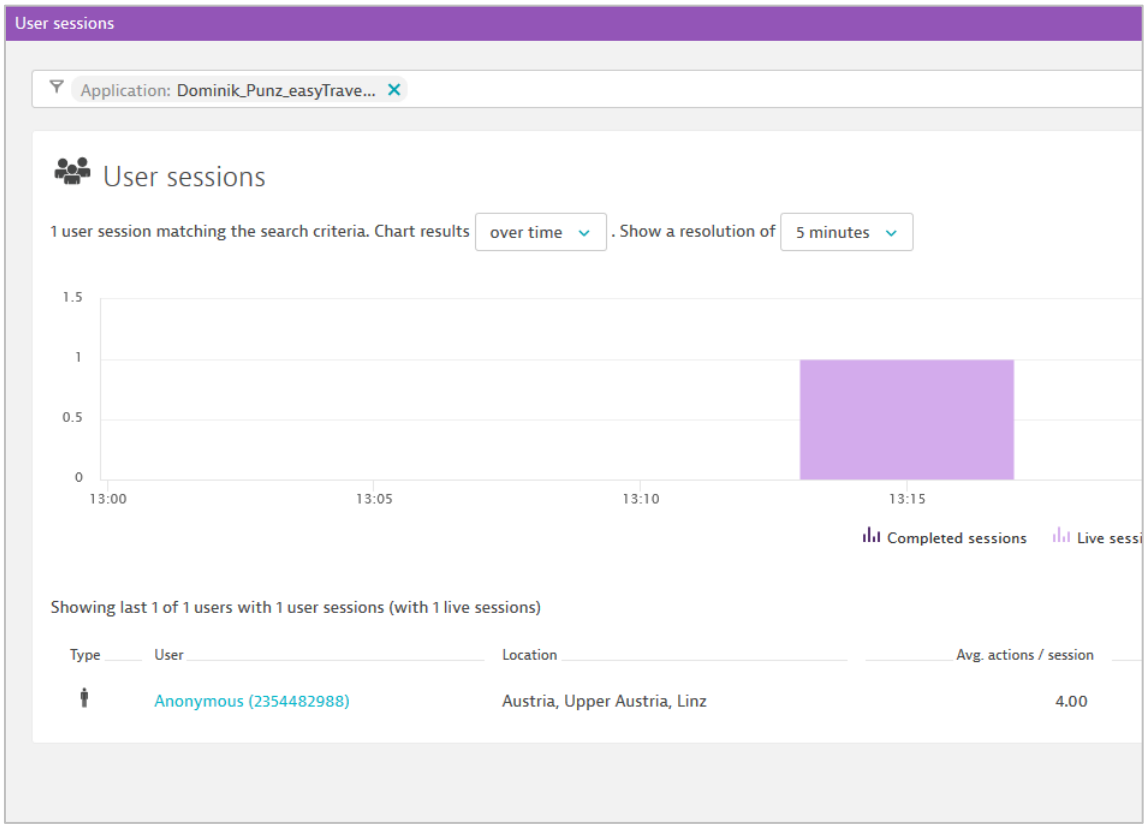
Preparation - Download the mobile applications projects

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

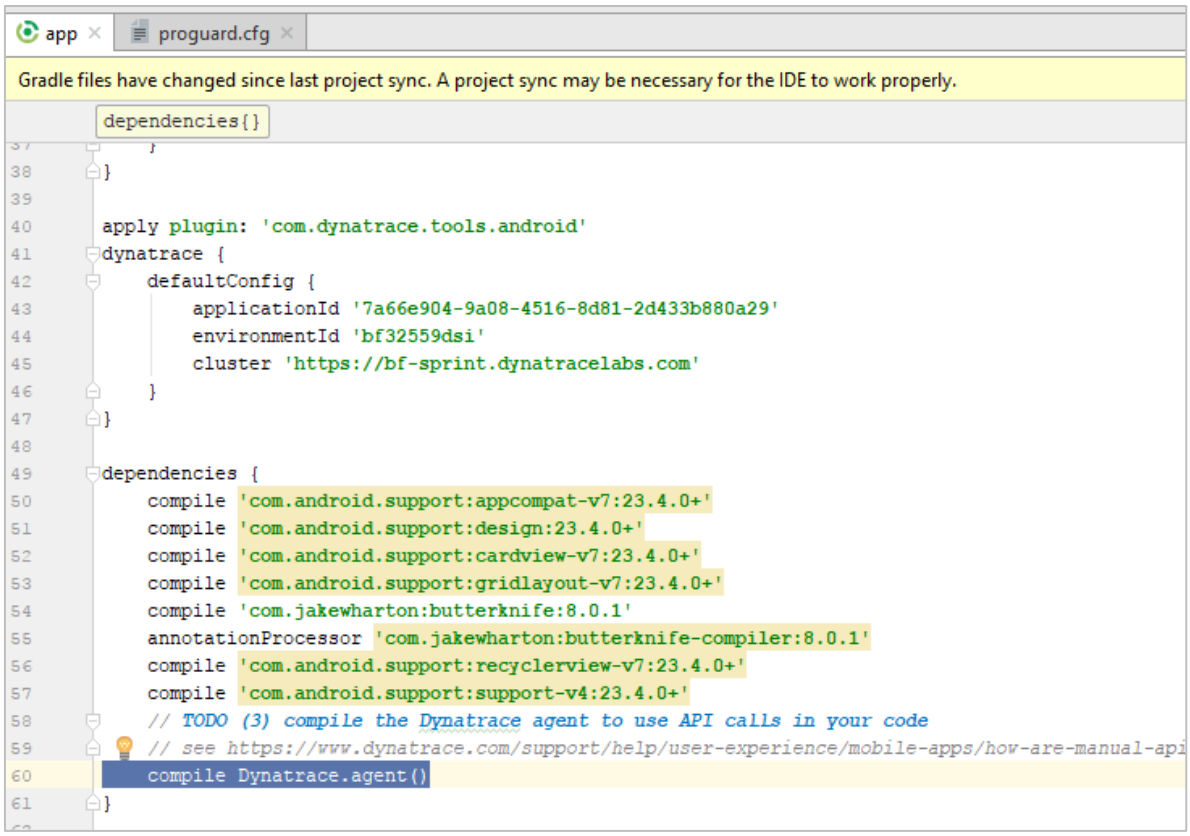
Lesson 3 - Instrumentation enhancement using Dynatrace API for Android

User Tagging

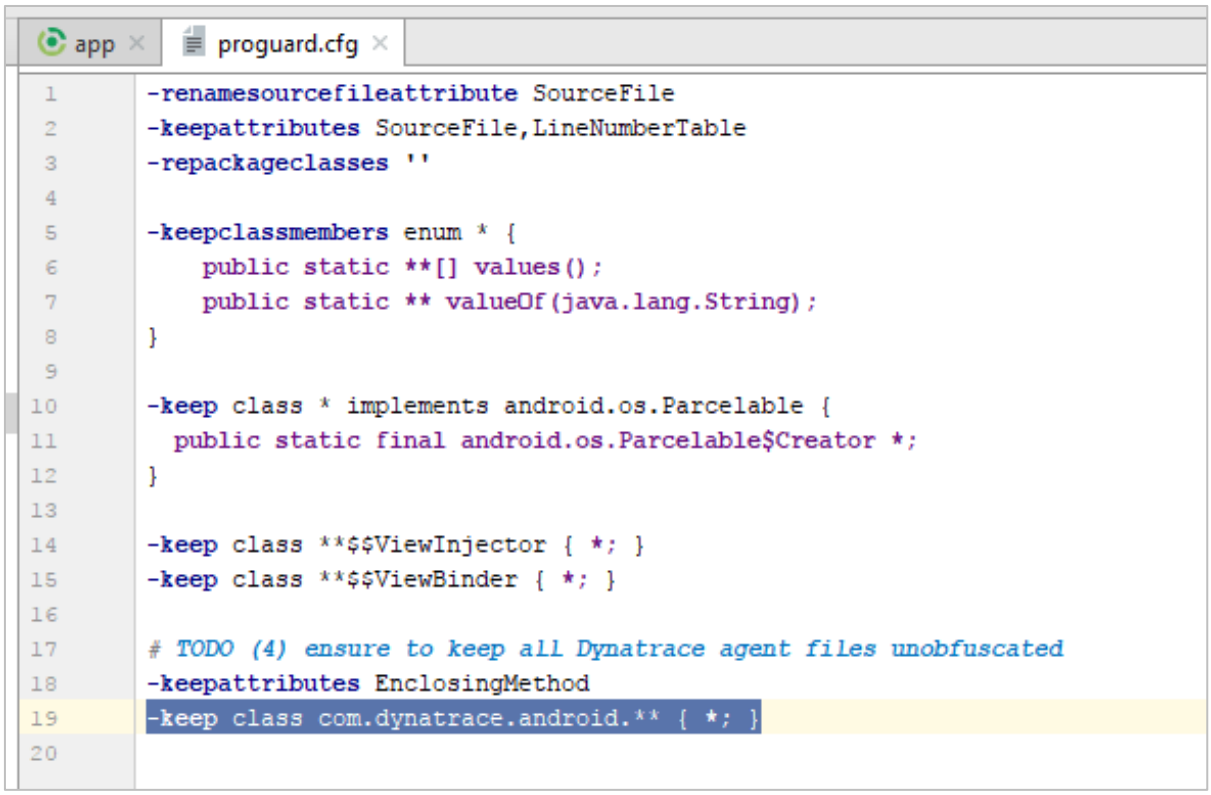
1. Currently, user session are anonymous.



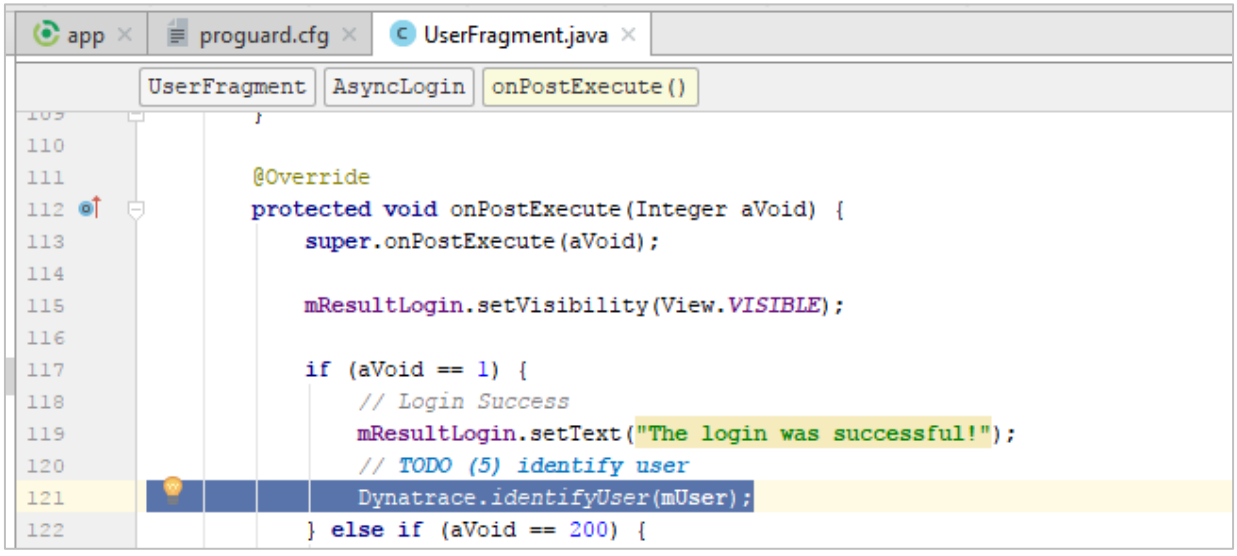
2. It would be useful to see the real user name in the session search. User tagging requires us to use the Dynatrace API for Android which means adding some code to the application. Therefore the Dynatrace agent needs to be available as a library during development. In Android Studio, find **TODO (3)** in the module’s **build.gradle** script, uncomment the line **compile dyntrace.agent()** and sync gradle.



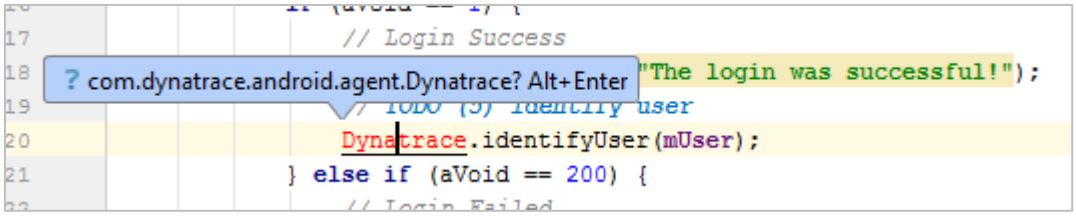
3. Find **TODO (4)** in the **proguard.cfg** file and notice that there is a rule set up that preserves all classes inside the **com.dynatrace.android** package, which means that the Dynatrace agent (API library) will be excluded from minification. This is important as the auto-instrumentation which will run later at build time would otherwise not be able to find the agent.

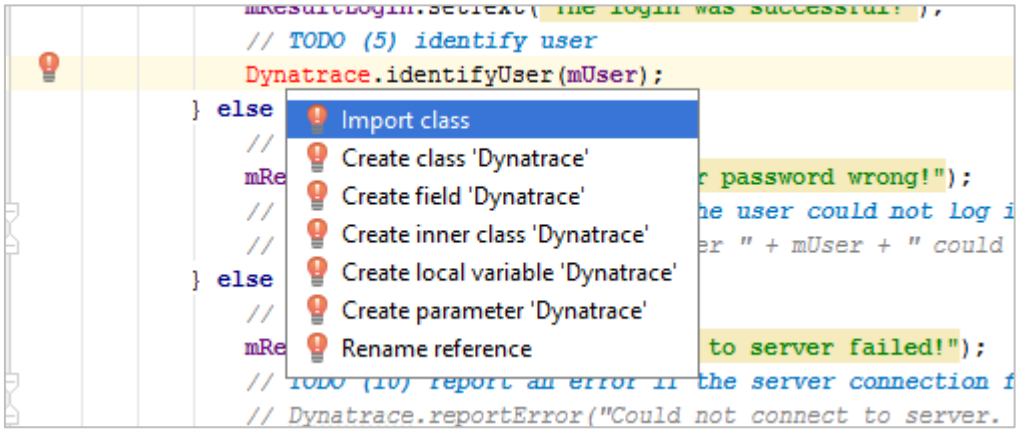


4. Now, lets do some coding… **TODO (5)** in the **UserFragment** class shows you the place in the code where a user successfully logs in. The API method to tag a user is named **identifyUser()**. We will add this method call there.



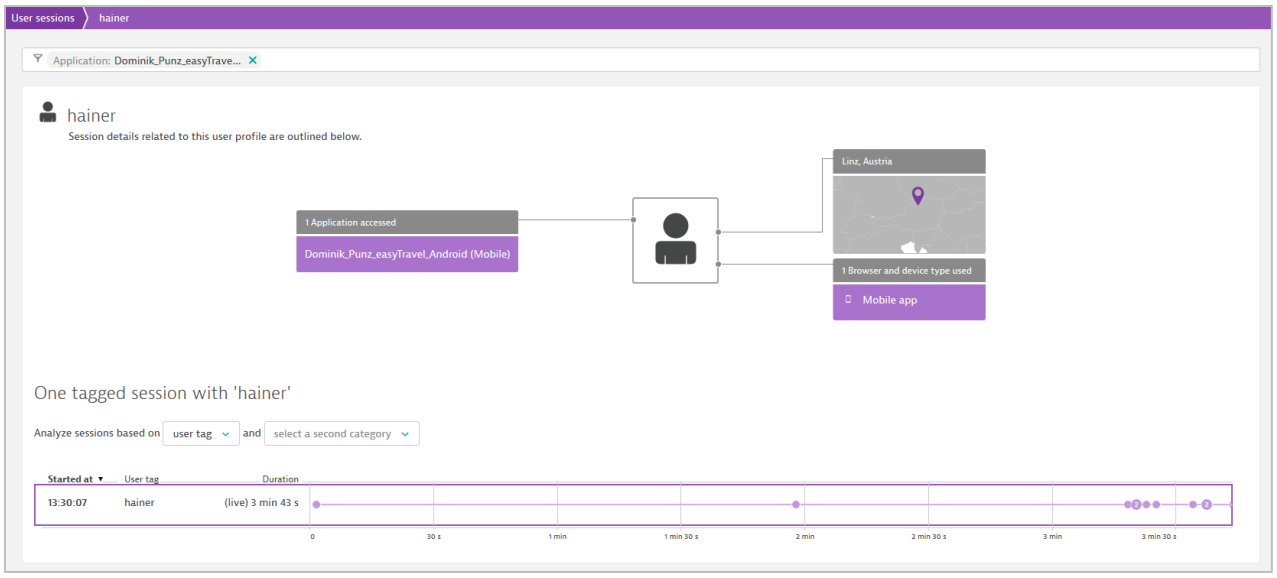
5. If you get this notification in Android Studio, you need to import the Dynatrace library by typing Alt+Enter and selecting Impot class





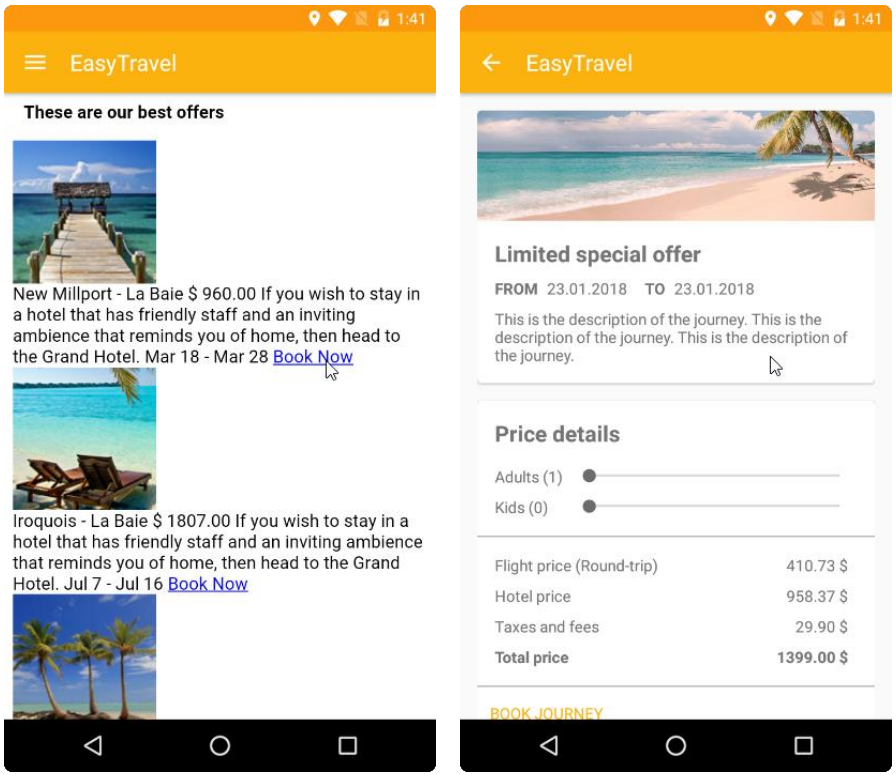
6. Run the application again. Make sure to disable the **Crash on Login** option in Settings, go to **User Account** and then log in. You can try any of these username / password
   - hainer / hainer
   - maria / maria
   - monica / monica
   - barbara / barbara
   - randy /randy

7. Go back to the Dynatrace console and wait for 2 minutes until the new data shows up. Go to the session search and you'll find your user tag in the list.



Custom User Actions

8. The special offers page in the application displays everything in a web view. Clicks on the “Book now” links are intercepted and an activity gets started. Dynatrace does not detect this as a user action out-of-the-box.



9. Let’s create a custom user action for this so this gets recorded by Dynatrace. Find **TODO (6)** and create a new user action by calling the **enterAction** method, providing the action name as an argument. The method returns a

DTXActionb object. The action is stored in the **mApp** object for persistence (see next step). We can also report the URL of the web request triggered by that click by using the **reportValue** method of the action object.

```
98      mWebView.setWebViewClient(shouldOverrideUrlLoading(view, url) → {
101             // TODO (6) create custom user action and store it to be accessible from the DetailJourneyFragment
102             DTXAction action = Dynatrace.enterAction( actionName: "Loading details for special offer");
103             action.reportValue( s: "url", url);
104             mApp.setCurrentAction(action);
105
```

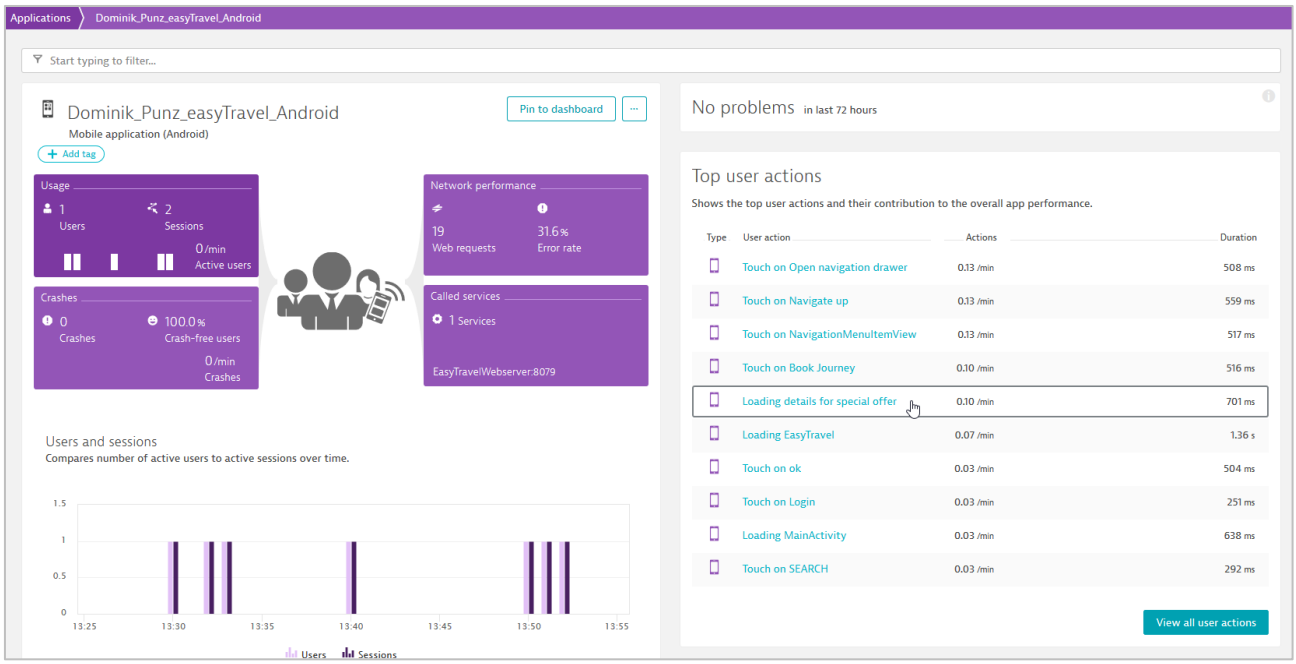
10. The place where we want to close the action is in a different fragment, so we must find a way to pass the newly created action. We will create an instance variable to store the action in the Application class and associated get/set methods to access the variable **(TODO (7))**.

```
166 // TODO (7) provide a getter/setter for the currently open custom user action
167 DTXAction mCurrentAction;
168 public void setCurrentAction(DTXAction action) {
169     this.mCurrentAction = action;
170 }
171 public DTXAction getCurrentAction() {
172     return mCurrentAction;
173 }
```

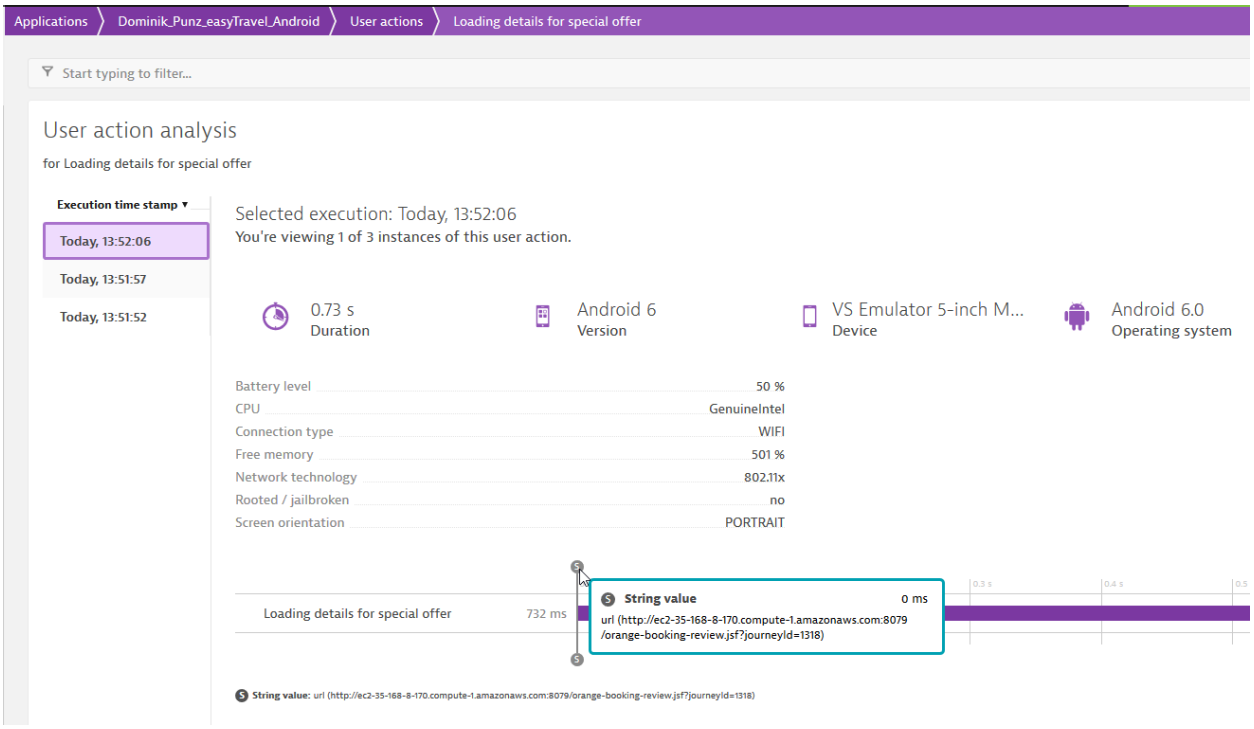
11. Find **TODO (8**) at the end of the **onCreateView** method definition in . The code check if there is an action open and if it is the case, closes it by calling the **leaveAction()** method.

```
113     // TODO (8) end the open action if the fragment was launched with a special offer
114     DTXAction action = mApp.getCurrentAction();
115     if (action != null) {
116         action.leaveAction();
117         mApp.setCurrentAction(null);
118     }
119
```

12. Run the application again and book some special offers. Send the application to the background to ensure that all data gets sent right away and wait for 2 minutes. You will then see this new customer user action in the list of user actions in your Application view.



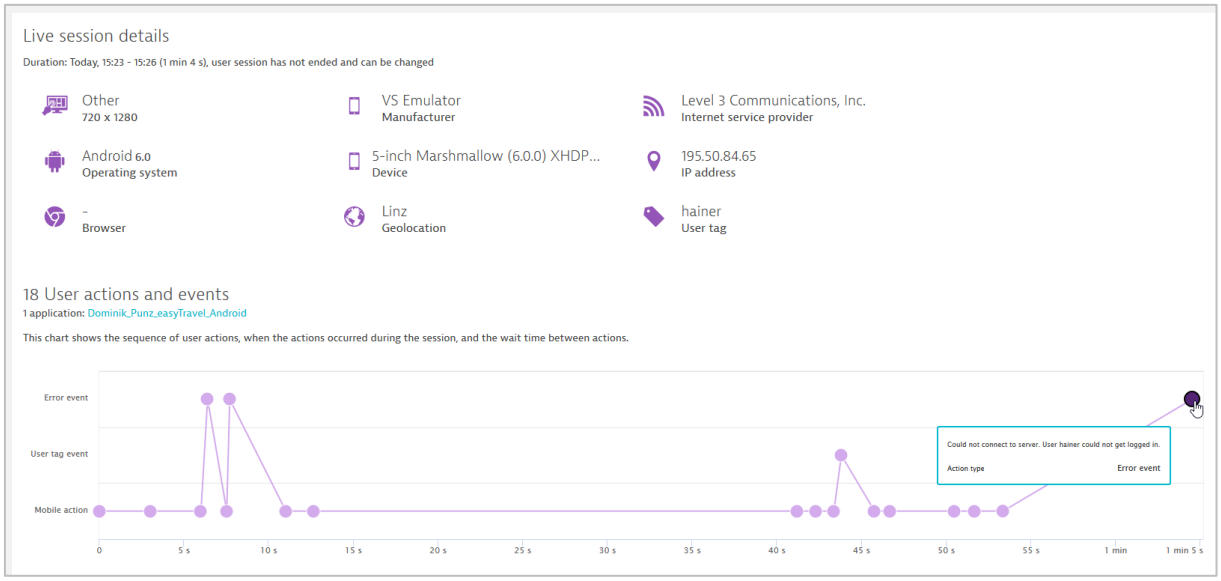
13. Click on this customer user action to get the list of individual executions of this action, click on the timestamp and the execution details will be displayed. The reported string value will be shown in the loading details.



Error Reporting

14. Find **TODO (9)** and **TODO (10**) in the **UserFragment** class. Here we can use **Dynatrace.reportError** method to report to Dynatrace the information about failed logins.

15. Run your application again, go to **User Account** and try log in in with a wrong user name. Then go to the settings and select the easyTravel not reachable problem.
16. Try to log in again and it will run into a timeout.
17. Send the app to the background and wait 2 minutes for the data to be available in Dynatrace. Find your live session and look at the details.



18. It is also possible to report handled exceptions. Find **TODO (11)** and **TODO (12)** and run the app again.
19. Disable the **EasyTravel not reachable** event and enable **Errors on search and book problem**.
20. Try to search for a journey and send the application in the background. Wait for the data to show up in your user session in Dynatrace.