

Testing Document

CSCI-310 Software Engineering

Professor: **Nenad Medvidović**

TA: **Shushan Arakelyan**

Fall 2017

Dream Team (Team 4)

Aarav Malpani
Boweï Chen
Prateek Bhatia
Shatrujeet Naruka
Tushar Singhal

8739-8865-03
5327-0036-21
2013-5693-40
2782-3669-61
2897-7369-04

malpani@usc.edu
boweiche@usc.edu
prateekb@usc.edu
naruka@usc.edu
tsinghal@usc.edu

Preface	3
Introduction	3
Instruction	3
White-box Tests	4
Base classes	4
Profile.java	4
ProfileInSchedule.java	6
Schedule.java	8
Server	10
DatabaseConnector.java	10
Black-box Tests	19
ArrangeAppsByName.java	19
MainActivity.java	19
Profiles	21
ProfilesActivity.java	21
CreateProfileActivity.java	22
EditProfileActivity.java	24
TimePicker.java	27
Schedules	28
SchedulesActivity.java	28
AddScheduleActivity.java	34
EditScheduleActivity.java	39
AddProfileToScheduleActivity.java	42
EditProfileInScheduleActivity.java	49
AddProfileToNewSchedule.java	50
Server	56
BackgroundService.java	56
Manual tests	61
Nonfunctional Tests (Extra)	62

1. Preface

This document outlines the test cases the original development team has created for the Android app *Focus!*, both white-box tests and black-box tests. This document specifies individual test cases, their rationale, and instructions to run all test cases. All information necessary to build test cases and execute them are included in the scope of this document.

The tests are created through JUnit 4. For Android Instrumented Tests, additional tools as Espresso and UIAutomator are also used to aid UI tests and to record actions.

NOTE: This document records the test case results as they have passed. All failed cases that have been discovered are promptly fixed then the test rerun to ensure the correctness. The bugs faced have been documented in the comments section but the test result screenshots are only produced in the document once that test has been regressively fixed and tested multiple times.

2. Introduction

Phones have constantly evolved and have today become an extension of people, notifying us about breaking news, social events, messages from friends, etc. Constant notifications are distracting and can make phone users unable to focus on the tasks at hand due to constant interference. *Focus!* seeks to alleviate this problem by blocking apps and notifications in a user-configured manner so that users can focus on their work and other activities.

Focus! is an Android application that allows users to block distracting applications and their notifications for a specified amount of time. The app allows users to define profiles to block certain applications and notifications at specified times and days.

3. Instruction

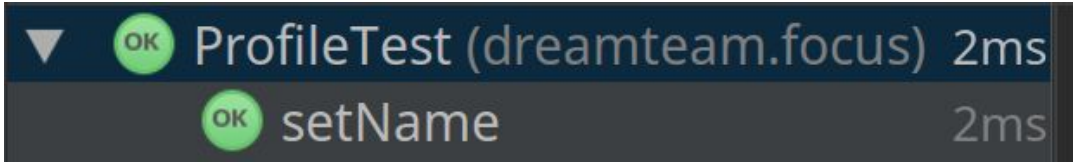
1. Sync Gradle file with project by invoking `Tools > Android > Sync Project with Gradle Files`
2. Point to the specific test to run. Individual test cases are grouped by the class it is run against.
3. Click on the "play" button on the left of class declaration to run the test.
 - a. If the test is a vanilla JUnit 4 test, the test will execute in the local machine.
 - b. If the test is an Android Instrumented Test, Android Studio will prompt for a device to run the specified test on. The user has the option to run the test on a virtual machine or a physical Android device.

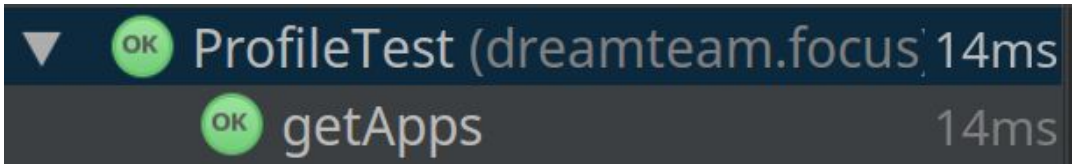
4. White-box Tests

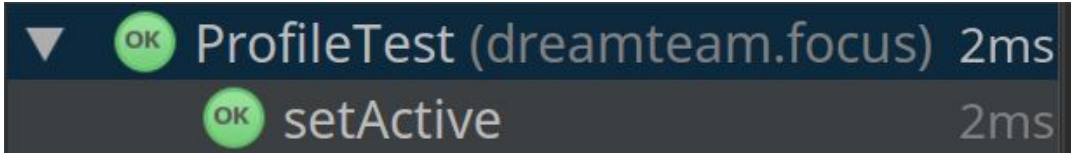
4.1. Base classes

4.1.1. Profile.java

To execute test cases specified on this class, compile and run:
app/src/test/java/dreamteam/focus/ProfileTest.java.

Tested function	Profile#getName() Profile#setName(String)
Test case location	dreamteam.focus.test.ProfileTest#setName()
Test case description	This test checks the getter and setter on the Profile class to see if they act correctly.
Rationale	Test strings are chosen with some random strings to see if the input string is correctly written to the class private member fields.
Results	 A screenshot of JUnit test results. It shows a green downward arrow, a green circle with 'OK', and the text 'ProfileTest (dreamteam.focus) 2ms'. Below this, another green circle with 'OK' is shown next to 'setName' and '2ms'.
Comments	No bugs were found while testing this case.

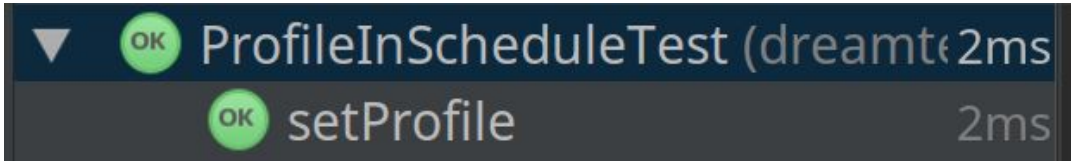
Tested function	Profile#getApps()
Test case location	dreamteam.focus.test.ProfileTest#getApps()
Test case description	This test verifies the method correctly extracts the apps to be blocked from the object.
Rationale	The member field is populated with random strings through the constructor and this case checks if the field is correctly populated with the right strings.
Results	 A screenshot of JUnit test results. It shows a green downward arrow, a green circle with 'OK', and the text 'ProfileTest (dreamteam.focus) 14ms'. Below this, another green circle with 'OK' is shown next to 'getApps' and '14ms'.
Comments	No bugs were found while testing this case.

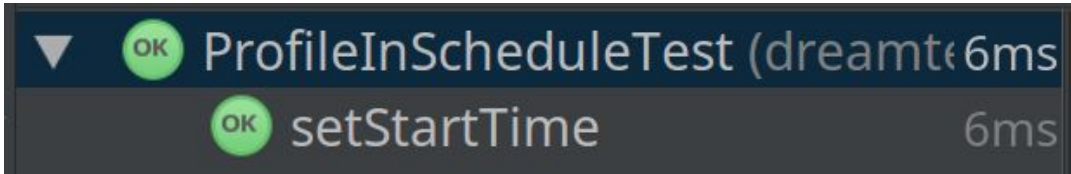
Tested function	Profile#isActive() Profile#setActive(boolean)
Test case location	dreamteam.focus.test.ProfileTest#setActive()
Test case description	This test verifies the method correctly sets the Profile internal active boolean value.
Rationale	A Profile defaults to false during instantiation, is set to true then tested, then set to false then tested again.
Results	 <p>The screenshot displays two test results. The first result is 'ProfileTest (dreamteam.focus)' with a duration of '2ms' and a green 'OK' status. The second result is 'setActive' with a duration of '2ms' and a green 'OK' status. Both results are preceded by a green circle containing the text 'OK'.</p>
Comments	No bugs were found while testing this case.

4.1.2. ProfileInSchedule.java

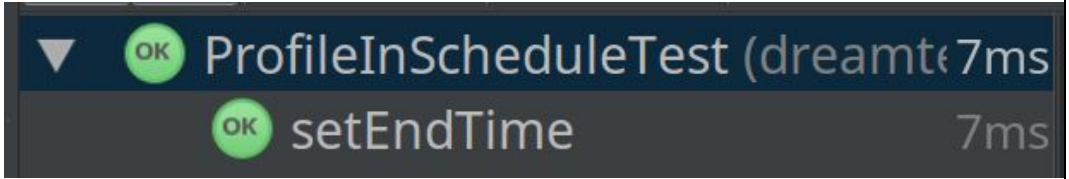
To execute test cases specified on this class, compile and run:

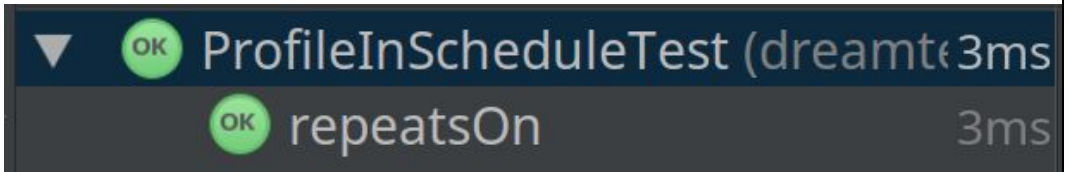
app/src/test/java/dreamteam/focus/ProfileInScheduleTest.java.

Tested function	ProfileInSchedule#getProfile() ProfileInSchedule#setProfile(Profile)
Test case location	dreamteam.focus.test.ProfileInScheduleTest#setProfile()
Test case description	This test checks the setProfile() method on ProfileInSchedule to see if overwriting member profile field in a ProfileInSchedule is possible.
Rationale	Two profiles are instantiated, compared to be not the same to ensure uniqueness. The test then goes through a series of setProfile(Profile) calls and tested along the way to ensure each call does what has been advertised.
Results	
Comments	No bugs were found while testing this case.

Tested function	ProfileInSchedule#getStartTime() ProfileInSchedule#setStartTime(Date)
Test case location	dreamteam.focus.test.ProfileInScheduleTest#setStartTime()
Test case description	This test checks the setStartTime(Date) method on ProfileInSchedule to see if they act as expected.
Rationale	Multiple Date instances are instantiated and put into setStartTime(Date) to ensure the fields are overwritten with the correct data.
Results	
Comments	No bugs were found while testing this case.

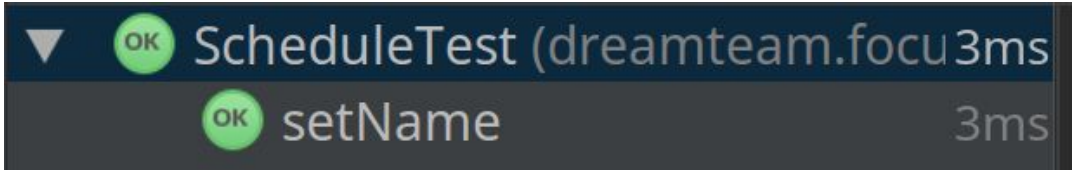
Tested function	ProfileInSchedule#getEndTime() ProfileInSchedule#setEndTime(Date)
-----------------	--

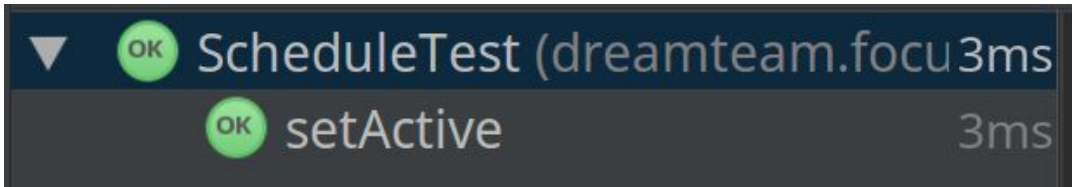
Test case location	dreamteam.focus.test.ProfileInScheduleTest#setEndTime()
Test case description	This test checks the setEndTime(Date) method on ProfileInSchedule to see if they act as expected.
Rationale	Multiple Date instances are instantiated and put into setEndTime(Date) to ensure the fields are overwritten with the correct data.
Results	
Comments	No bugs were found while testing this case.

Tested function	ProfileInSchedule#repeatsOn()
Test case location	dreamteam.focus.test.ProfileInScheduleTest#repeatsOn()
Test case description	This test checks the contents of repeatsOn field is as instantiated.
Rationale	The field is iterated through to verify against the instantiated data to see if all the expected information are present.
Results	
Comments	No bugs were found while testing this case.

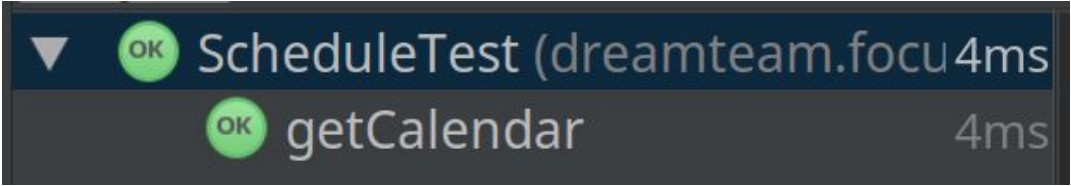
4.1.3. Schedule.java

To execute test cases specified on this class, compile and run:
app/src/test/java/dreamteam/focus/ScheduleTest.java.

Tested function	Schedule#getName() Schedule#setName(String)
Test case location	dreamteam.focus.test.ScheduleTest#setName()
Test case description	This test checks the getter and setter on the Schedule class to see if they act correctly.
Rationale	Test strings are chosen with some random strings to see if the input string is correctly written to the class private member fields.
Results	All pass.  The image shows JUnit test results for ScheduleTest. It displays a green 'OK' icon, a dropdown arrow, the test class name 'ScheduleTest (dreamteam.focu3ms', and the method name 'setName' with a duration of '3ms'.
Comments	No bugs were found while testing this case.

Tested function	Schedule#isActive() Schedule#setActive(boolean)
Test case location	dreamteam.focus.test.ScheduleTest#setActive()
Test case description	This test checks the getter and setter on the Schedule class to see if they act correctly.
Rationale	Test boolean values are chosen to see if the input string is correctly written to the class private member fields.
Results	All pass.  The image shows JUnit test results for ScheduleTest. It displays a green 'OK' icon, a dropdown arrow, the test class name 'ScheduleTest (dreamteam.focu3ms', and the method name 'setActive' with a duration of '3ms'.
Comments	No bugs were found while testing this case.

Tested function	Schedule#getCalendar()
Test case location	dreamteam.focus.test.ScheduleTest#getCalendar()
Test case	This test checks the getter on the Schedule class to see if they act correctly.

description	
Rationale	A getter is run and compared against the known ArrayList of expected values.
Results	<p>All pass.</p> 
Comments	No bugs were found while testing this case.


4.2. Server


4.2.1. DatabaseConnector.java


To execute test cases specified on this class, compile and run:

app/src/androidTest/java/dreamteam/focus/server/DatabaseConnectorTest.java


To run individual test case for a function **TESTED_FUNCTION** within DatabaseConnectorTest.java,

1. Navigate to: app/src/androidTest/java/dreamteam/focus/server/DatabaseConnectorTest.java
2. Click on the green arrow () next to the **TESTED_FUNCTION**


Tested function	DatabaseConnector#getProfiles()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getProfiles_isEmpty()
Test case description	The test case retrieves the list of profiles from a newly created database
Rationale	The test case ensures that a newly created database has no profiles stored and the number of profiles returned should be zero
Results	Successfully returned an empty array of profiles of size zero and passed the test case 
Comments	No bugs were found testing this case.


Tested function	DatabaseConnector#createProfile() DatabaseConnector#getProfiles()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getProfiles_addProfile()
Test case description	The test case adds a new profile to an empty database and retrieves the list of profiles from the database
Rationale	The test case ensures that a profile is successfully created in the database and retrieved from the database
Results	Successfully retrieved the added profile and passed the test case 

Comments	No bugs were found testing this case.
Tested function	DatabaseConnector#removeProfile()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getProfiles_removeProfile()
Test case description	The test case creates a profile in the database. Then it removes the profile
Rationale	The test case checks that the specified profile is successfully created in the database and then successfully removed from the database.
Results	<p>Successfully removed the only added profile from the database and returned an empty list of profiles and passed the test case</p> 
Comments	No bugs were found testing this case


Tested function	DatabaseConnector#activateProfile()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getProfiles_activateProfile()
Test case description	The test case creates a profile in the database. Then it activates the profile
Rationale	The test case checks that the specified profile is activated successfully from the database.
Results	<p>Successfully created a profile, activated it, retrieved all profiles from the database, checked its activation status and passed the test case</p> 
Comments	Initially activateProfile was crashing the app as it was treating the profile as a profileInSchedule and assumed it had an associated REPEAT_ENUM. Since REPEAT_ENUM for a profile is null, this gave the null pointer error.


Tested function	DatabaseConnector#deactivateProfile()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getProfiles_deactivateProfile()
Test case description	The test case creates a profile in the database. Then it deactivates the profile

Rationale	The test case checks that the specified profile is deactivated successfully from the database.
Results	<p>Successfully created a profile, activated it, then deactivated it, retrieved all profiles from the database, checked its activation status and passed the test case</p> 
Comments	No bugs were found testing this case

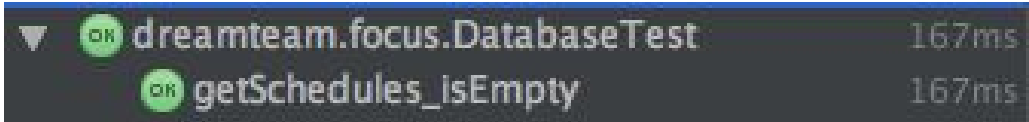
Tested function	DatabaseConnector#updateProfileName()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getProfiles_updateProfileName()
Test case description	The test case creates a profile in the database. Then it updates the name of the profile
Rationale	The test case checks that the specified profile's name has been successfully updated in the database.
Results	<p>Successfully created a profile, updated its name, retrieved all profiles from the database, checked that the name of the profile is that of the updated profile and passed the test case</p> 
Comments	Found a bug here. Initially updating a profile name created a new profile with the updated name, The problem was that the updateProfile function was not removing the old profile but adding a new profile with the updated information. So both the old and the updated profile would exist in the database

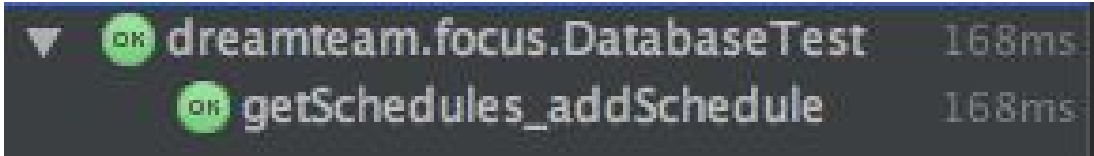
Tested function	DatabaseConnector#updateProfile()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getProfiles_updateProfileBlockedApps()
Test case description	The test case creates a profile in the database. Then it updates the apps blocked by the profile
Rationale	The test case checks that the specified profile's list of blocked apps has been successfully updated in the database.

Results	<p>Successfully created a profile, updated its blocked apps, retrieved all profiles from the database, checked that the list of the profile blocked apps is that of the updated profile and passed the test case</p> 
Comments	<p>Found a bug here. Initially updating a profile blocked apps created a new profile with the updated blocked apps, The problem was that the updateProfile function was not removing the old profile but adding a new profile with the updated information. So both the old and the updated profile would exist in the database</p>

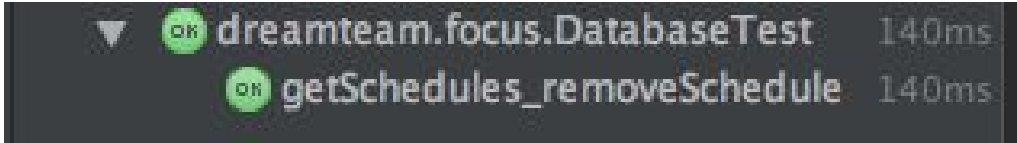
Tested function	DatabaseConnector#updateProfile()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getProfiles_updateProfileNameCheckSchedule()
Test case description	The test case creates a profile and a schedule in the database. Then it adds the profile to the schedule. Then it updates the name of the profile
Rationale	The test case checks that the specified profile's name has been successfully updated in the schedule as well.
Results	<p>Successfully created a profile and a schedule, added the profile to the schedule, updated the profile name, retrieved all profiles in the schedule from the database, checked that the name of the profile in schedule is that of the updated profile and passed the test case</p> 
Comments	<p>Found a bug here. Initially updating the profile name was not updating the profileInSchedule name from all the schedules associated with this profile. This was because updating the profile name was not updating the profileInSchedule table</p>

Tested function	DatabaseConnector#getSchedules()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getSchedules_isEmpty()
Test case description	The test case calls the getSchedules() method on a newly created database
Rationale	The test case ensures that a newly created database has no schedules stored and the number of schedules returned should be zero
Results	Successfully passed the test case

	
Comments	No bugs were found testing this case

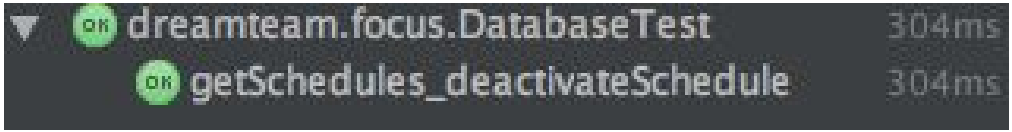
Tested function	DatabaseConnector#addSchedule()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getSchedules_addSchedule()
Test case description	The test case adds a schedule to a newly created database
Rationale	The test case ensures that the created schedule has been successfully added to the database
Results	<p>Successfully added a new schedule to the database, retrieved the list of schedules from the database, checked that the returned list contains the added schedule and passed the test case</p> 
Comments	No bugs were found testing this case

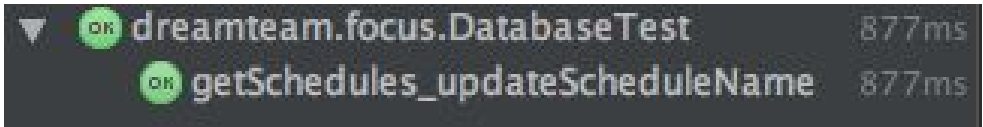
Tested function	DatabaseConnector#removeSchedule()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getSchedules_removeSchedule()
Test case description	The test case creates a schedule in the database. Then it removes the schedule
Rationale	The test case checks that the specified schedule is removed successfully from the database.
Results	Successfully created a schedule, removed it, retrieved all profiles from the database, checked that the list of schedules is empty and passed the test case

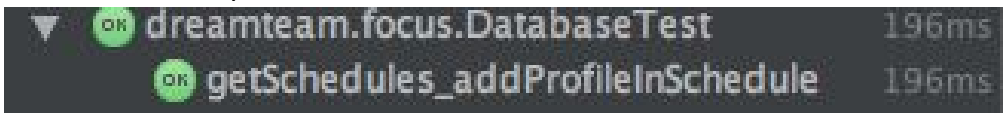
	
Comments	No bugs were found testing this case

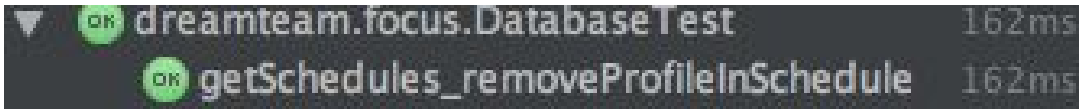
Tested function	DatabaseConnector#activateSchedule()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getSchedules_activateSchedule()
Test case description	The test case creates a schedule in the database. Then it activates the schedule
Rationale	The test case checks that the specified schedule is activated successfully in the database.
Results	<p>Successfully created a schedule, activated it, retrieved all schedules from the database, checked the activation status of the added schedule and passed the test case</p> 
Comments	No bugs were found testing this case

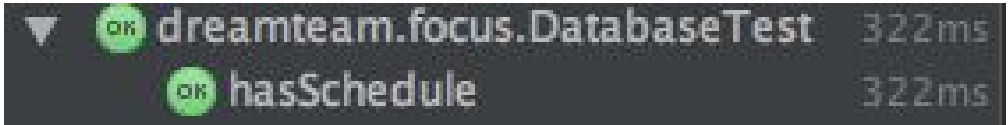
Tested function	DatabaseConnector#deactivateSchedule()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getSchedules_deactivateSchedule()
Test case description	The test case creates a schedule in the database. Then it deactivates the schedule
Rationale	The test case checks that the specified schedule is deactivated successfully in the database.
Results	Successfully created a schedule, activated it, deactivated it, retrieved all schedules from the database, checked the activation status of the added schedule and passed the test case

	
Comments	No bugs were found testing this case

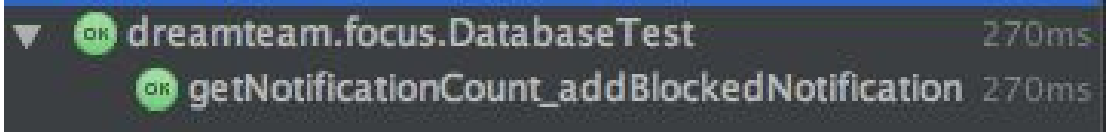
Tested function	DatabaseConnector#updateScheduleName()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getSchedules_updateScheduleName()
Test case description	The test case creates a schedule in the database. Then it updates the name of the schedule
Rationale	The test case checks that the specified schedule's name has been successfully updated in the database.
Results	<p>Successfully created a schedule, updated its name, retrieved all schedules from the database, checked the name of the schedule was that of the updated schedule and passed the test case</p> 
Comments	No bugs were found testing this case

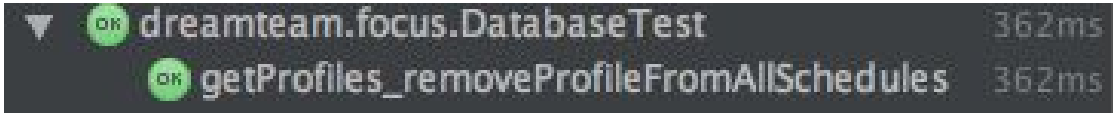
Tested function	DatabaseConnector#addProfileInSchedule()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getSchedules_addProfileInSchedule()
Test case description	The test case creates a profile in the database. Then it creates a schedule and adds the profile to the schedule.
Rationale	The test case checks that the specified profile has been successfully added to the schedule in the database.
Results	<p>Successfully created a profile and a schedule, added the profile to the schedule, retrieved all schedules from the database, checked the profilesInSchedule for the schedule and passed the test</p> 
Comments	No bugs were found testing this case

Tested function	DatabaseConnector#removeProfileInSchedule()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getSchedules_removeProfileInSchedule()
Test case description	The test case creates a profile in the database. Then it creates a schedule and adds the profile to the schedule. Then it removes the profile from the schedule.
Rationale	The test case checks that the specified profile has been successfully removed from the schedule in the database.
Results	<p>Successfully created a schedule and a profile, added the profile to the schedule, saved it to the database, removed the profile from schedule, retrieved all schedules from database and checked that the removed profile has been removed from the schedule and passed the test</p> 
Comments	Initially there was a bug here where removing a profile from the schedule on a specific day would remove all profiles in the schedule on other days as well. This was happening because the removeProfileInSchedule function was not taking into account the REPEAT_ENUM of the deleted profileInSchedule and was deleting any profileInSchedule with the same name, start time and endtime. Fixed it by taking into consideration the REPEAT_ENUM table by joining it in the SQL query

Tested function	DatabaseConnector#hasSchedule()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#hasSchedule()
Test case description	The test case creates a schedule in the database and then calls the tested function
Rationale	The test case checks that the tested function returns true for the already created schedule
Results	<p>Successfully created a schedule then called this function to check if it returns true for the same schedule and passed the test</p> 
Comments	No bugs were found testing this case

Tested function	DatabaseConnector#getNotificationCountForApp()
-----------------	--

	DatabaseConnector#addBlockedNotification()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#getNotificationCount_addBlockedNotification()
Test case description	The test case adds a few notifications to the database and retrieves them
Rationale	The test case ensures that the notifications are successfully added and retrieved from the database
Results	<p>Successfully added a few notifications for a particular app to the database, then retrieve the count of the notifications and checked that it was correct and passed the test</p> 
Comments	Found a bug here. Getting the notification count for an app would return 1 even when there were more than one notifications added. This was happening because the SQL query asked for SELECT COUNT(*) instead of SELECT * and then returned the row count instead of value, and the row count would always be 1

Tested function	DatabaseConnector#removeProfileFromAllSchedules()
Test case location	dreamteam.focus.androidTest.server. DatabaseConnectorTest#removeProfileFromAllSchedules()
Test case description	The test case creates a profile and a schedule and adds the profile to the schedule on Monday and Tuesday. It then deletes the profile
Rationale	The test case ensures that the deleted profile is removed from all schedules in the database
Results	<p>Successfully created a profile and a schedule, added a few instances of the profile to the schedule, removed the profile from the database and checked that the profile had been removed from all the schedules and passed the test case</p> 
Comments	No bugs were found testing this case

5. Black-box Tests

5.1. ArrangeAppsByName.java

To execute test cases specified on this class, compile and run:

`app/src/test/java/dreamteam/focus/client/ArrangeAppsByNameTest.java`.

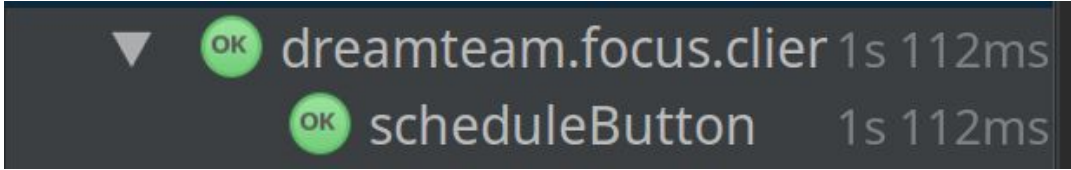
Tested function	ArrangeAppsByName#sortMapByValue
Test case location	<code>dreamteam.focus.test.ArrangeAppsByNameTest#sortMapByValueTest</code>
Test case description	The test case checks if the hashmap that is given as an argument in the class <code>ArrangeAppsByName</code> is returned as a treemap sorted by values. A manually sorted map is created and tested against an expected arraylist in which apps are sorted by name.
Rationale	The apps in the <code>CreateProfileActivity</code> and <code>AddProfileActivity</code> need to be sorted by name for the user's convenience to find a specific app that he/she needs to block in a profile. The specific input for this test case will be a hashmap of package names as key and corresponding app names as values.
Results	<div><p>The apps were arranged alphabetically successfully.</p></div>
Comments	During the initial run of the app, we noticed that it was really difficult to find a specific app to select in profile. We had to swipe through the entire list to find it. We then decided to arrange all the apps by name so as to make it user friendly. However, we couldn't straightaway use a <code>TreeMap</code> (a data structure that sorts its data based on keys) as app names in the phone could be same for multiple packages. So we needed an implementation of the treemap that is sorted by values, so that package names which are unique become keys and app names becomes values.

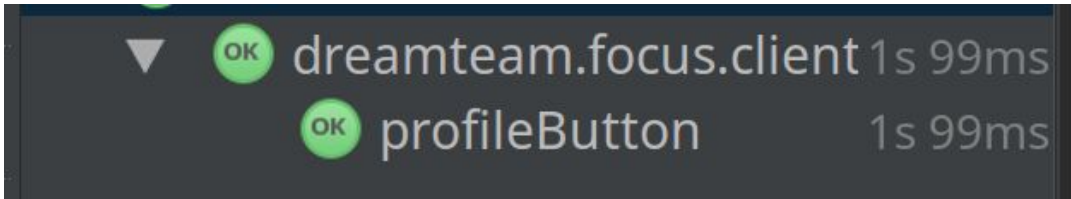
5.2. MainActivity.java

To execute test cases specified on this class, compile and run:

`app/src/androidTest/java/dreamteam/focus/client/MainActivityTest.java`.

Tested function	<code>MainActivity</code>
Test case location	<code>dreamteam.focus.androidTest.MainActivityTest#mainActivity</code>
Test case	When the "schedule" button is clicked, the view should be the list of schedule

description	screen, and Android back button should take the user back to the MainActivity.
Rationale	MainActivity should have two buttons, one leading to a list of Profiles and another leading to a list of Schedules.
Results	<p>All pass.</p> 
Comments	No bugs were discovered while this test is run.

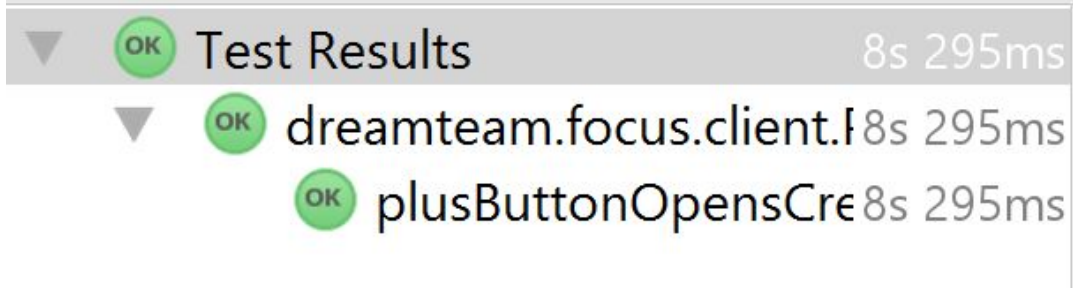
Tested function	MainActivity
Test case location	dreamteam.focus.androidTest.MainActivityTest#mainActivity
Test case description	When the “profile” button is clicked, the view should be the list of profile screen, and back should take the user back to the MainActivity.
Rationale	MainActivity should have two buttons, one leading to a list of Profiles and another leading to a list of Schedules.
Results	<p>All pass.</p> 
Comments	No bugs were discovered while this test is run.

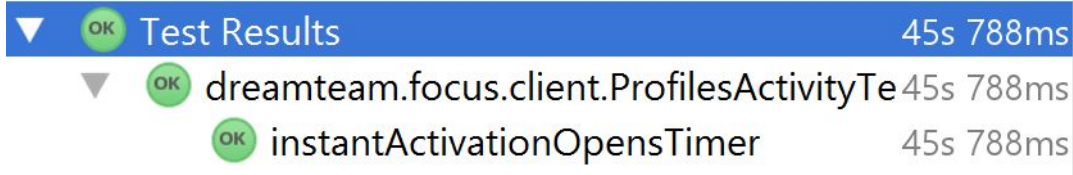
5.3. Profiles

5.3.1. ProfilesActivity.java

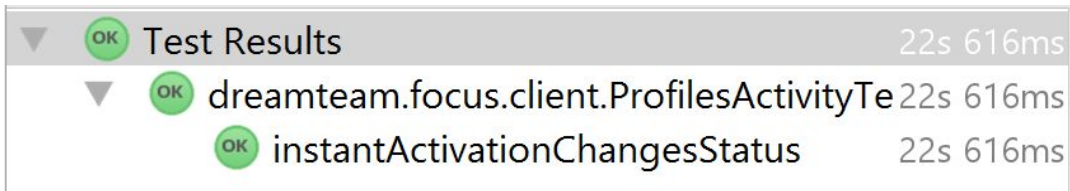
To execute test cases specified on this class, compile and run:

app/src/androidTest/java/dreamteam/focus/client/ProfilesActivityTest.java.

Tested function	ProfilesActivity
Test case location	dreamteam.focus.androidTest.client.profiles.ProfilesActivityTest#plusButtonOpensCreateProfile()
Test case description	The plus button on the Profiles Activity is pressed and then the test checks if the activity that opens up is Create Profile Activity by checking if the Edit Text view for entering profile Name is present.
Rationale	The Plus Button should direct the user to Create Profile Activity.
Results	 <p>The CreateProfile Activity opens up when the user clicks on the Plus Button.</p>
Comments	No bugs were discovered while this test is run.

Tested function	ProfilesActivity
Test case location	dreamteam.focus.androidTest.client.profiles.ProfilesActivityTest#instantActivationOpensTimer()
Test case description	The toggle button is pressed and the test checks if the timePicker Activity opened up by checking if the SetTime Button is Present.
Rationale	When the toggle button in front of Profile Name is pressed, it should direct the user to timePicker class.
Results	

	The TimePicker activity opens up when the user presses the toggle button for the user to choose the end time.
Comments	No bugs were discovered while this test is run.

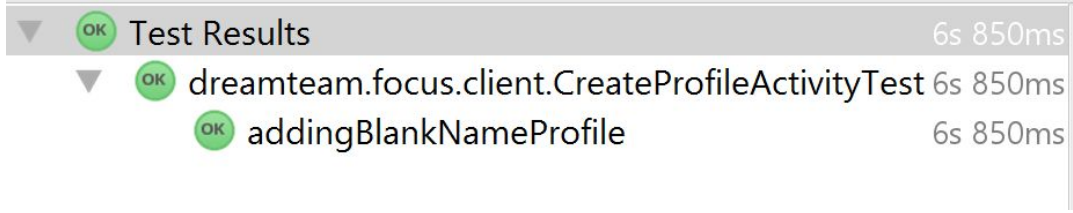
Tested function	ProfilesActivity
Test case location	<code>dreamteam.focus.androidTest.client.profiles.ProfilesActivityTest#instantActivationChangesStatus()</code>
Test case description	The toggle button is pressed and the timePicker Activity opens up, the timer is set to 11 minutes ahead of the current time and Set Time is pressed. The user is directed back to Profiles Activity and the Toggle button is checked if the state is "ON". Then the toggle is pressed again and the test checks if the toggle state is "OFF".
Rationale	When the toggle button in front of Profile Name is pressed, it should direct the user to timePicker class, and when you come back to the Profiles Activity, the toggle should change to ON and when clicked again it should be changed to OFF.
Results	 <p>When the user comes back to Profile Activity from the timePicker activity after selecting a valid time, the toggle changes to "ON".</p>
Comments	When the user returned to the Profiles Activity, the toggle was still "OFF" at the place of "ON", the reason being the Profile List was not updated. This bug was fixed by updating the list every time the activity restarts.

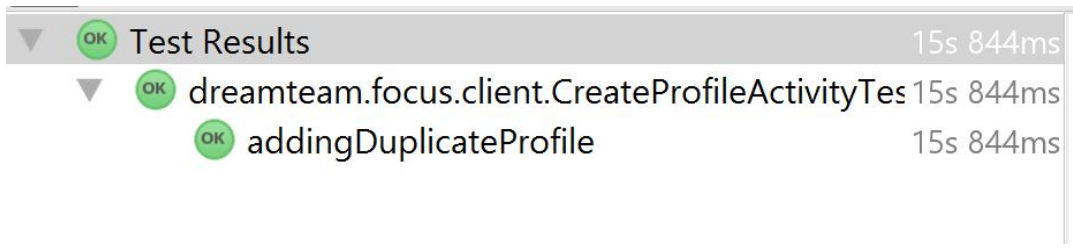
5.3.2. CreateProfileActivity.java

To execute test cases specified on this class, compile and run:

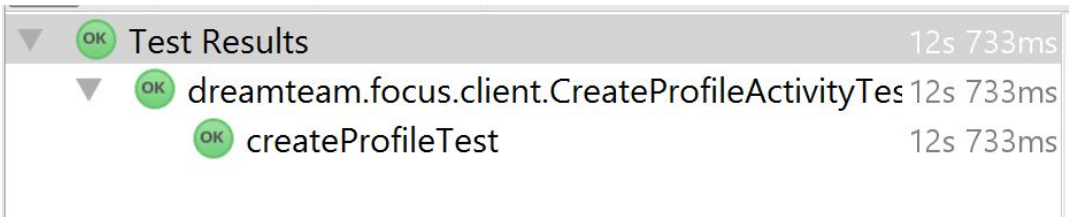
`app/src/androidTest/java/dreamteam/focus/client/profiles/CreateProfileActivityTest.java`.

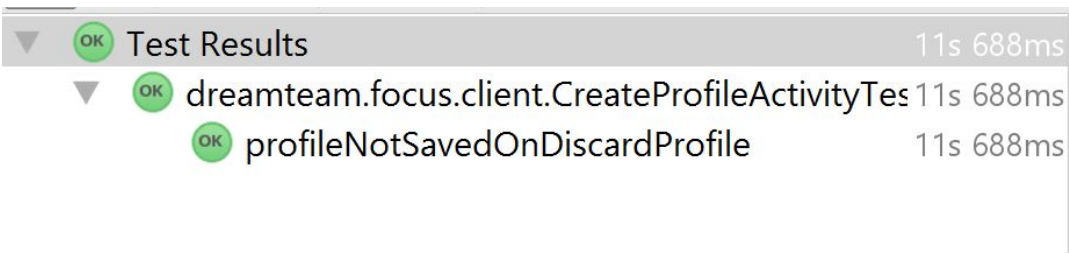
Tested function	CreateProfileActivity
Test case location	<code>dreamteam.focus.androidTest.client.profiles.CreateProfileActivityTest#addingBlankNameProfile()</code>
Test case description	The EditText for Profile Name is left empty and Create Profile Button is pressed. The test case checks if the user remains on the same activity or is directed to ProfileActivity.
Rationale	When the user doesn't enter a name for Profile Name, they should remain on the CreateProfileActivity as a profile without a name cannot exist.

Results	 <p>When the user tries to add a profile without a name,the user stays on the activity and a Toast pops up to give the error message.</p>
Comments	This test failed the first time because the Database was throwing a Unique exception but the GUI did not implement a catch functionality for that.Now,it pops up a toast when the exception is thrown in the catch block.

Tested function	CreateProfileActivity
Test case location	dreamteam.focus.androidTest.client.profiles.CreateProfileActivityTest#addingDuplicateProfile()
Test case description	The EditText for Profile Name is given an existing Profile's name and Create Profile Button is pressed,The test case checks if the user remains on the same activity or is directed to ProfileActivity.
Rationale	When the user enters a duplicate name for Profile ,they should remain on the CreateProfileActivity as two profiles with the same name cannot exist.
Results	 <p>The user remains on CreateProfileActivity if a duplicate Profile name is inserted and a toast pops up to notify the user.</p>
Comments	The earlier functionality did not check for duplicate name exception which was crashing the app when the user tried to create a new Profile with an existing Profile name.This test helped to rectify that error.

Tested function	CreateProfileActivity
Test case location	dreamteam.focus.androidTest.client.profiles.CreateProfileActivityTest#createProfileTest()
Test case description	The entries of CreateProfileActivity are filled with valid data and Create Profile button is pressed,then the test checks whether a new Profile has been added to the ProfilesActivity.

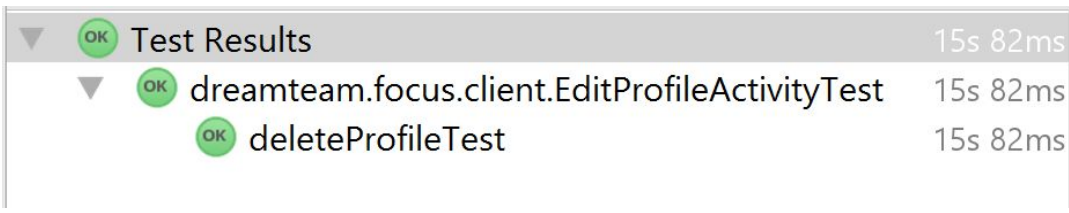
Rationale	When the user enters valid information for a new Profile and presses Create Profile, The user should be directed to Profiles Activity and a new Profile should be added to the ProfilesList.
Results	 <p>On the successful completion of the test, a new Profile is added to the Profiles List in the Profiles Activity.</p>
Comments	No bugs were discovered while this test is run.

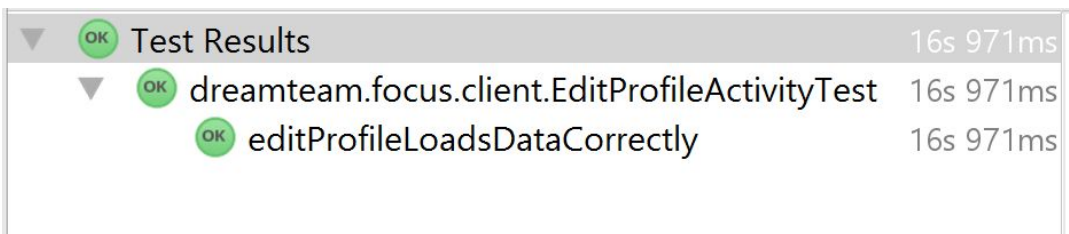
Tested function	CreateProfileActivity
Test case location	dreamteam.focus.androidTest.client.profiles.CreateProfileActivityTest#profileNotSavedOnDiscardProfile()
Test case description	The entries of Create Profile are filled with valid data and Discard Profile Button is pressed, the test checks whether the new Profile gets added to the Profiles Activity or not.
Rationale	When the user enters valid data for CreateProfileActivity and presses Discard Button, the application should not add it to the database and hence the new Profile should not be shown in the Profiles Activity.
Results	 <p>When the user presses discard Profile Button, a new Profile is not added to the Profiles list in ProfilesActivity.</p>
Comments	No bugs were discovered while this test is run.

5.3.3. EditProfileActivity.java

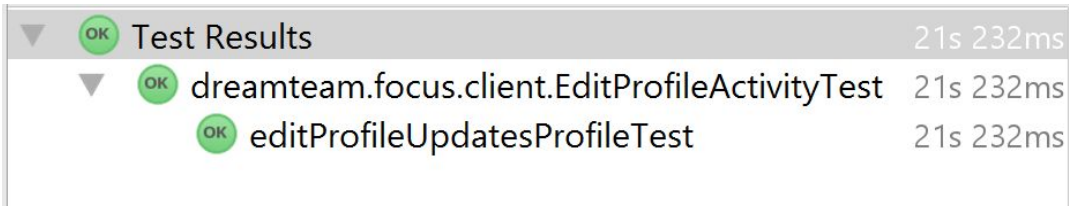
To execute test cases specified on this class, compile and run:

app/src/androidTest/java/dreamteam/focus/client/profiles/EditProfileActivityTest.
java.

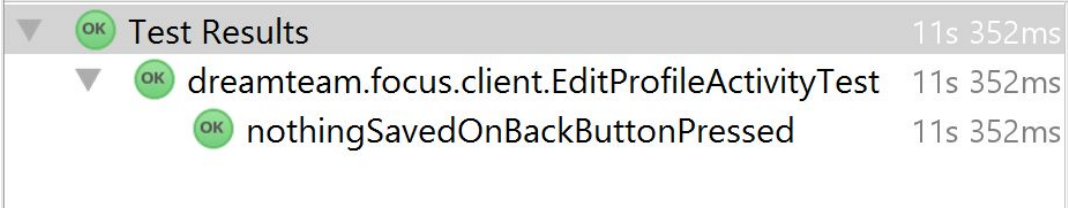
Tested function	EditProfileActivity
Test case location	dreamteam.focus.androidTest.client.profiles.EditProfilesActivityTest#deleteProfileTest()
Test case description	The Delete Profile Button in the EditProfileActivity is pressed and the test checks whether this specific Profile is removed from the list of Profiles in ProfilesActivity.
Rationale	When the user presses the delete Profile button,the profile should be deleted from the database and should not be shown in the list of Profiles in the ProfilesActivity.
Results	<div><p>When the user deletes a Profile,the profiles List in ProfilesActivity gets updated and reflects the given change.</p></div>
Comments	No bugs were discovered while this test is run.

Tested function	EditProfileActivity
Test case location	dreamteam.focus.androidTest.client.profiles.EditProfilesActivityTest#editProfileLoadsDataCorrectly()
Test case description	The profile name in the ProfilesActivity is pressed and the test checks whether the EditProfileActivity opens up with the correct credentials of that specific profile by comparing it with data that was entered when this profile was created.
Rationale	The EditProfileActivity should open up with the specific Profile's data so the user can decide upon the changes to be made.
Results	<div><p>When the EditProfileActivity opens up for a ProfileInSchedule,it loads the correct name of the Profile and the selected apps.</p></div>

Comments	The test failed the first time because the correct apps were not loaded when the EditProfile window opened up because of the Recycler Views in ListView. So I had to disable them to allow desired functionality.
----------	---

Tested function	EditProfileActivity
Test case location	dreamteam.focus.androidTest.client.profiles.EditProfilesActivityTest#editProfileUpdatesProfileTest()
Test case description	The credentials of a Profile are changed in EditProfileActivity and update Profile button is pressed. The updated Profile is pressed from the Profiles Activity and the test checks whether the EditProfileActivity opens up with updated Profile data.
Rationale	When the user presses the Update Profile button, the Profile is updated in the database so when the user selects the updated Profile again from the ProfileActivity, the user should be directed to the EditProfile Activity with the corresponding updated Profile data.
Results	 <p>The updated changes in a Profile are reflected when the EditProfile opens up again.</p>
Comments	No bugs were discovered while this test is run.

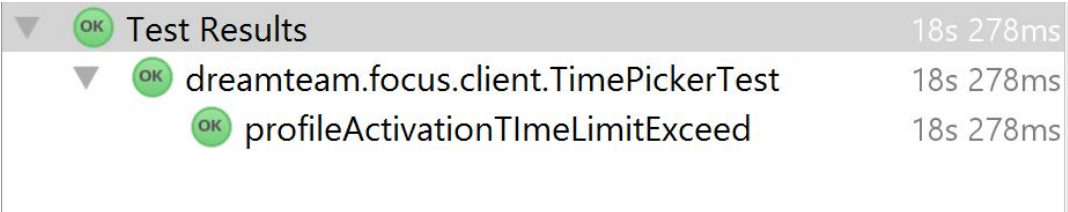
Tested function	EditProfileActivity
Test case location	dreamteam.focus.androidTest.client.profiles.EditProfileActivityTest#nothingSavedOnBackButtonPressed()
Test case description	The credentials of a Profile are changed in the EditProfileActivity and the backButton is Pressed, The test checks whether the Profile gets updated or not.
Rationale	When the user changes the Profile credentials in EditProfile and presses backButton rather than Update Profile Button, the changes are not saved on the database, so when the EditProfileActivity opens up again, it still contains the old Profile data.

Results	 <p>The Profile does not get updated if the user presses back button</p>
Comments	No bugs were discovered while this test is run.

5.3.4. TimePicker.java

To execute test cases specified on this class, compile and run:

app/src/androidTest/java/dreamteam/focus/client/profiles/TimePickerTest.java.


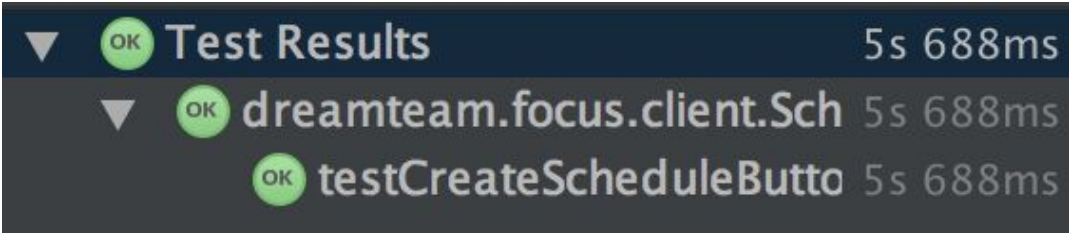
Tested function	TimePicker
Test case location	dreamteam.focus.androidTest.client.profiles.TimePickerTest#profileActivationTimeLimitExceed()
Test case description	The Set Time in TimePicker button is pressed, the tests check whether the difference between the current time and the time given by the user falls in the ten minute to ten hour window.
Rationale	When the user presses the Set Time Button and the time difference does not fall in the valid range, the user should remain on the same activity and a toast should be displayed to notify the user.
Results	 <p>The activity does not allow the user to instantly activate a profile if the time difference does not fall in the valid window of ten minute to ten hour.</p>
Comments	The timePicker did not allow the user to instantly activate the profile if the difference between the current time and set time was ten minutes due to my comparisons. I updated those comparisons to accommodate this case.

5.4. Schedules


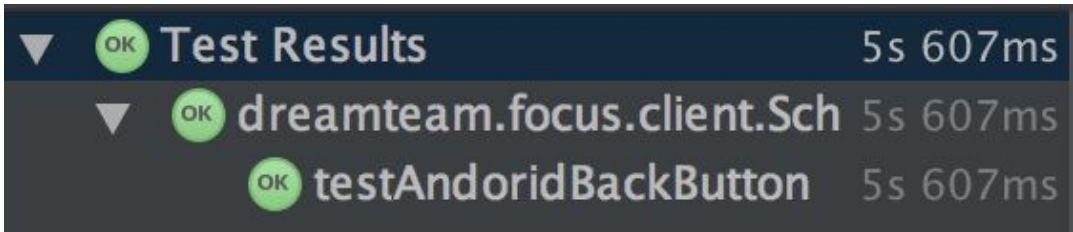
5.4.1. SchedulesActivity.java


To execute test cases specified on this class, compile and run:

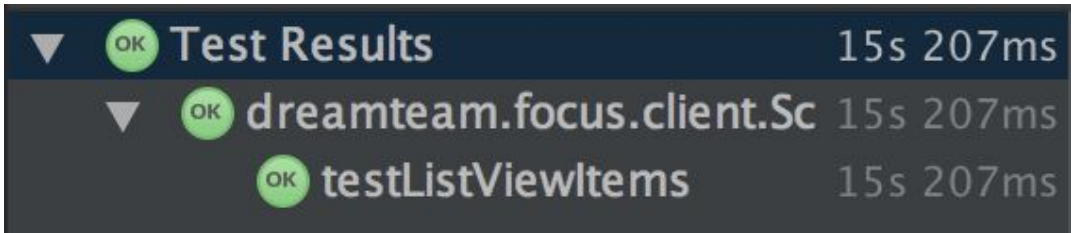
app/src/androidTest/java/dreamteam/focus/client/schedules/SchedulesActivityTest.java.


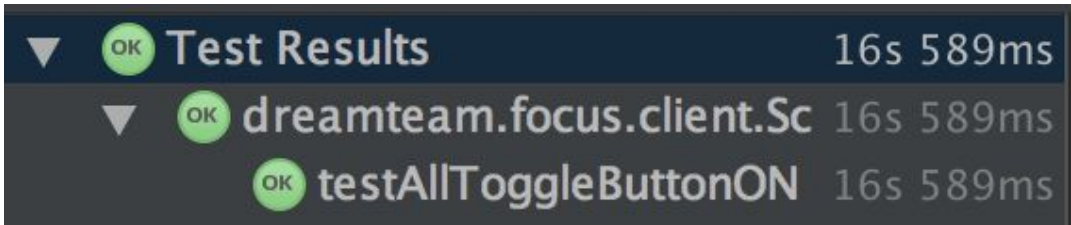
Tested function	SchedulesActivtiy
Test case location	<code>dreamteam.focus.androidTest.client.schedules.SchedulesActivityTest#testCreateScheduleButton()</code>
Test case description	<p>This test verifies the “+” button leads the user to a form where he can create a new schedule for future use. After clicking the button, the user should be directed to a new Activity where he can make a schedule.</p> <p>To Run the test:</p> <ol style="list-style-type: none">1. Navigate to: <code>app/src/androidTest/java/dreamteam/focus/client/schedules/SchedulesActivityTest.java</code>2. Click on the green arrow () next to the function <code>testCreateScheduleButton()</code>
Rationale	The user should be able to navigate through the app swiftly and should be able to create a new schedule which is the main purpose of the app. Hence this test is important to the fundamental operations of the add and the “+” button should be tested.
Results	<div></div> <p>Successfully launched the AddScheduleActivity after the “+” button was clicked.</p>
Comments	No bugs were discovered during this test run

Tested function	SchedulesActivtiy
Test case location	<code>dreamteam.focus.androidTest.client.schedules.SchedulesActivityTest#testAndroidBackButton()</code>


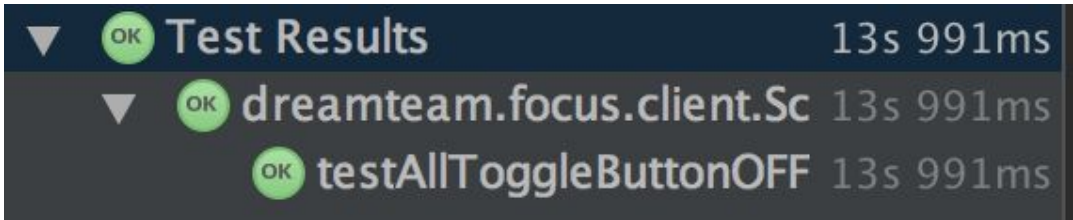
Test case description	<p>This test verifies that the back button leads to the home page (MainActivity) where two buttons, the “Schedules” button and the “Profiles” button are found. The back button is located on every screen in Android and can cause serious problems if not tested.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/SchedulesActivityTest.java 2. Click on the green arrow () next to the function testAndroidBackButton()
Rationale	<p>Often times programmers forget to code and consider other interactions outside the programmed screen; to have a robust and reliable app is to see how the app reacts to such inputs and hence this test is necessary and the android back button must be tested so that the app behaves exactly how it should in all situations. The test also verifies that the programmers aren’t building a stack by starting new intents and not finishing them which makes the overall device slow!</p>
Results	 <p>The back button successfully navigated to the MainActivity once pressed.</p>
Comments	No bugs were discovered during this test run

Tested function	SchedulesActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.SchedulesActivityTest #testListViewItems()
Test case description	<p>This test verifies that all the schedules present in the database are shown in a ListView with toggle buttons in front of them indicating if they are turned on/off. This case also tests the connection of the GUI to the Database and makes sure that the information passed is complete</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/SchedulesActivityTest.java 2. Click on the green arrow () next to the function testListViewItems()
Rationale	All schedules that exist in the database should be shown in this list, so that the


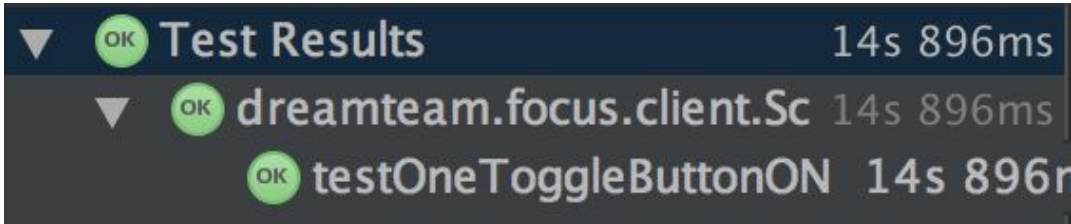
	user can be sure of what his day looks like and won't be disturbed during important meetings/classes!
Results	 <p>The List had all the Schedules in the database including the ones which we add before the test.</p>
Comments	No bugs were discovered during this test run


Tested function	SchedulesActivitiy
Test case location	<code>dreamteam.focus.androidTest.client.schedules.SchedulesActivityTest</code> <code>#testAllToggleButtonON()</code>
Test case description	<p>This test verifies that the schedules which are active have a toggle button which display "ON". By default, when a Schedule is made, the button displayed should be "OFF" as this schedule is not active, but once the user toggles the button, the button should change to show the current status which is "ON" and send the same information to the Database as well so next time the user decides to check the status of his schedules they see this schedule "ON"</p> <p>To Run the test:</p> <ol style="list-style-type: none"> Navigate to: <code>app/src/androidTest/java/dreamteam/focus/client/schedules/SchedulesActivityTest.java</code> Click on the green arrow () next to the function <code>testAllToggleButtonON()</code>
Rationale	All schedules that are turned on should be shown in this list with the "ON" indicator on the right. This is an important case as it tests the connection to the database, gui and background services which make sure that the app is doing its job of blocking only when it is supposed to.
Results	 <p>This test successfully toggled all the Schedules to turn "ON" once clicked.</p>

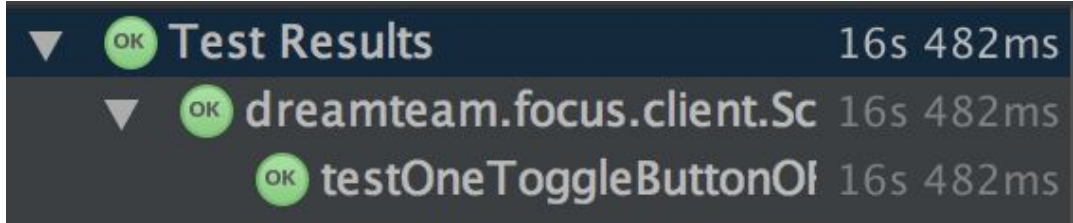
Comments	No bugs were found in this case
----------	---------------------------------


Tested function	SchedulesActivitiy
Test case location	<code>dreamteam.focus.androidTest.client.schedules.SchedulesActivityTest</code> <code>#testAllToggleButtonOFF()</code>
Test case description	<p>This test verifies that the schedules which are inactive have a toggle button which display “OFF”. By default, when a Schedule is made, the button displayed should be “OFF” as this schedule is not active, but once the user toggles the button, the button should change to show the current status which is “ON” and send the same information to the Database as well so next time the user decides to check the status of his schedules they see this schedule “ON” and vice versa.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> Navigate to: <code>app/src/androidTest/java/dreamteam/focus/client/schedules/SchedulesActivityTest.java</code> Click on the green arrow () next to the function <code>testAllToggleButtonOFF()</code>
Rationale	All schedules that are turned off should be shown in this list with the “OFF” indicator on the right. This test ensures that Schedules which aren’t needed are not running in the background so that the user interactions aren’t limited.
Results	 <p>The test successfully toggled all the Toggle buttons “ON” and then “OFF” and assured that they were working properly.</p>
Comments	No bugs were found in this case

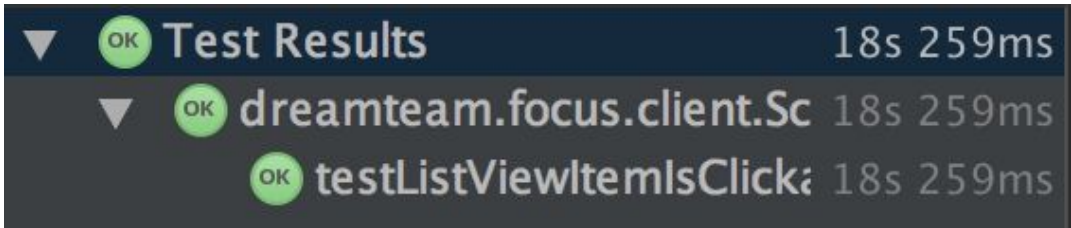
Tested function	SchedulesActivitiy
Test case location	<code>dreamteam.focus.androidTest.client.schedules.SchedulesActivityTest</code> <code>#testOneToggleButtonON()</code>
Test case	This test verifies that the when the toggle button associated with a particular

description	<p>Schedule is toggled when it is “OFF” initially, it activates the Schedule and changes the State of the button to “ON”. This status should not change even after the user closes and starts the app again or goes to the home screen and opens the app again.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/SchedulesActivityTest.java 2. Click on the green arrow () next to the function testOneToggleButtonON()
Rationale	<p>Clicking the “OFF” button should make the button read “ON” and send a signal to the database so that the schedule is activated so that the app can do its job of blocking notifications. This is different from the other two cases mentioned above as we are just testing and seeing one button is being pressed and all other buttons are unaffected by it.</p>
Results	 <p>The test successfully only toggled the one button that was required and checked the status of that button and the others to ensure that the buttons were working independently.</p>
Comments	<p>No bugs were discovered during this test run</p>

Tested function	SchedulesActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.SchedulesActivityTest #testOneToggleButtonOFF()
Test case description	<p>This test verifies that the when the toggle button associated with a particular Schedule is toggled when it is “ON” initially, it deactivates the Schedule and changes the State of the button to “OFF”. This status should not change even after the user closes and starts the app again or goes to the home screen and opens the app again.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/SchedulesActivityTest.java 2. Click on the green arrow () next to the function

	testOneToggleButtonOFF()
Rationale	Clicking the “ON” button should make the button read “OFF and send a signal to the database so that the schedule is deactivated so that the app can do its job of sending any missed notifications and not blocking further notifications. This is different from the other two cases mentioned above as we are just testing and seeing one button is being pressed and all other buttons are unaffected by it.
Results	 <p>The test successfully only toggled the one button that was required and checked the status of that button and the others to ensure that the buttons were working independently.</p>
Comments	No bugs were discovered during this test run


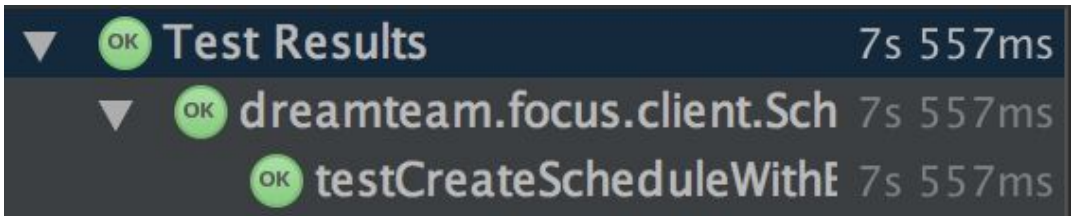
Tested function	SchedulesActivtiy
Test case location	dreamteam.focus.androidTest.client.schedules.SchedulesActivityTest #testListViewItemIsClickable()
Test case description	<p>This case verifies that a user can edit a previously made Schedule by clicking on the Schedule name. The action should lead us to the EditScheduleActivity where everything about the schedule from its name to each Profile in Schedule should be editable.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/SchedulesActivityTest.java 2. Click on the green arrow () next to the function testListViewItemIsClickable()
Rationale	Clicking on the profile name leads to EditScheduleActivity of the specific schedule. This test is important as we don't want the user to have to delete the entire schedule in case they make a mistake or one thing changes during their week. They should be able to only remove and alter the desired time without having to create a new schedule and deleting this one.

Results	 <p>Clicking a ListView item (Schedule Name) led to the EditScheduleActivity successfully.</p>
Comments	No bugs were found in this case

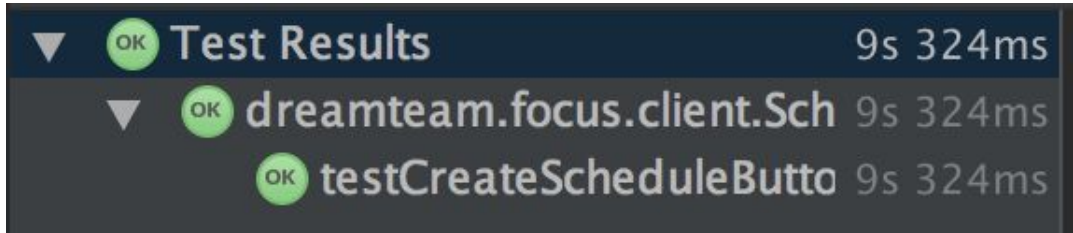
5.4.2. AddScheduleActivity.java

To execute test cases specified on this class, compile and run:

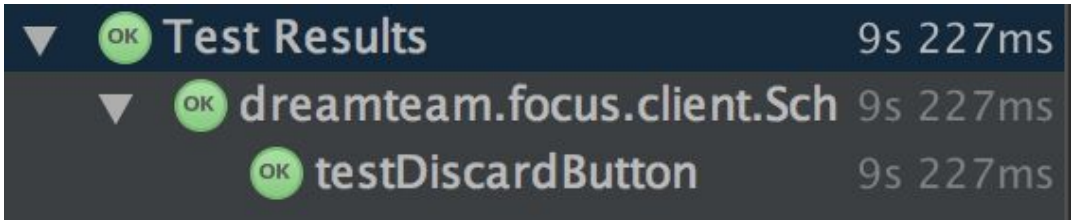
app/src/androidTest/java/dreamteam/focus/client/schedules/AddScheduleActivityTest.java.


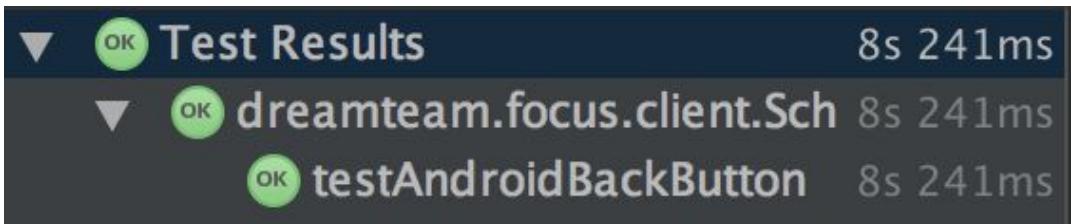
Tested function	AddScheduleActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddScheduleActivity#testCreatScheduleWithEmptyName()
Test case description	<p>This test verifies that all constraints on fields must be satisfied before a schedule is to be instantiated in the database especially the one asking for a name. A schedule can be created without having a single other item except for a name.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddScheduleActivityTest.java 2. Click on the green arrow () next to the function testCreatScheduleWithEmptyName()
Rationale	A schedule cannot be created without a name. Having a schedule without a name would be confusing for the user. This should not be allowed because it's hard to keep track of unique schedules.
Results	 <p>The test Successfully tried created a new schedule without a name and received an appropriate Toast message asking the user to enter a name to create a schedule.</p>

Comments	No bugs found
----------	---------------

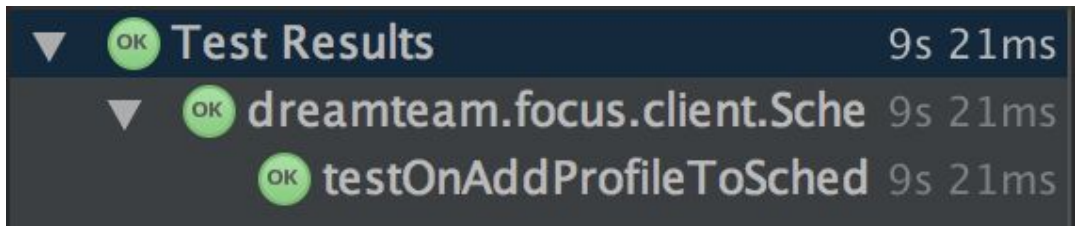
Tested function	AddScheduleActivtiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddScheduleActivity#testCreateScheduleButton()
Test case description	<p>This test verifies that if all constraints and fields are satisfied and the schedule is created which shows up in the list and appears each time the app is opened as long as the user does not delete the Schedule.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddScheduleActivityTest.java 2. Click on the green arrow (▶) next to the function testCreatScheduleButton()
Rationale	When the “create schedule” button is clicked, the schedule should be created and changes written to the database. This is important to see that user inputs are received and passed on to the database as expected.
Results	 <p>The test successfully created a new Schedule with all the required parameters and the ListView was populated with the Schedule immediately after.</p>
Comments	No bugs were found

Tested function	AddScheduleActivtiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddScheduleActivity#testDiscardButton()
Test case description	<p>This test verifies that changes aren't made if the discard button is pressed and the new Schedule doesn't appear in the list on all Schedules in the Database on the SchedulesActivity page.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddScheduleActivityTest.java 2. Click on the green arrow (▶) next to the function testDiscardButton()


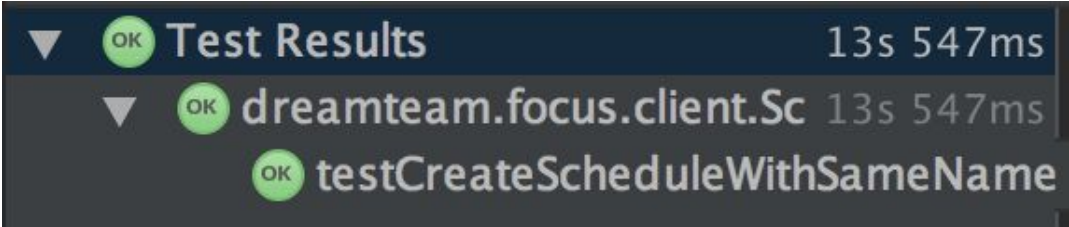
Rationale	When the “discard changes” button is clicked, any changes should be discarded and the database should not be modified.
Results	 <p>The Discard button did not create the schedule and led back to the SchedulesActivity where the listview did not have the schedule.</p>
Comments	A bug was found while testing which was adding the “TemporaryScheudleToCreate” in the Schedule view in some cases which wasn’t supposed to be added. The code was then fixed to check and delete the temp schedule if it existed in the database. The temp schedule is created in the back end to ease the process of adding profiles in Schedule immediately.

Tested function	AddScheduleActivtiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddScheduleActivity#testAndroidBackButton()
Test case description	<p>This test verifies that changes aren’t made if the back button is pressed the changes aren’t saved to the database and the correct activity is displayed to the user by checking the intent.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddScheduleActivityTest.java 2. Click on the green arrow () next to the function testAndroidBackButton()
Rationale	Often times programmers forget to code and consider other interactions outside the programmed screen; to have a robust and reliable app is to see how the app reacts to such inputs and hence this test is necessary and the android back button must be tested so that the app behaves exactly how it should in all situations. The test also verifies that the programmers aren’t building a stack by starting new intents and not finishing them which makes the overall device slow!
Results	 <p>The back button did not create the schedule and led back to the SchedulesActivity where the listview did not have the schedule.</p>

Comments	A bug was found at first when pressing the backbutton as we didn't explicitly code what would happen if the user pressed back while creating a schedule. The program was layering up intents on the stack which was later fixed by overriding the back button functionality and making it more like a discard button where it finished the activity before going back.
----------	--

Tested function	AddScheduleActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddScheduleActivity#testOnAddProfileToScheduleButton()
Test case description	<p>The add profile to schedule button should lead us to the AddProfileToScheduleActivity and the intent should have the particular activity in it.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddScheduleActivityTest.java 2. Click on the green arrow (▶) next to the function testOnAddProfileToScheduleButton()
Rationale	Testing the add profile to schedule button so user can add Profiles to start using the Schedules feature and can make his calendar so that he can use the app.
Results	 <p>The "+" button successfully led to the AddProfileToNewSchedule page as intended.</p>
Comments	No bugs were found during this test


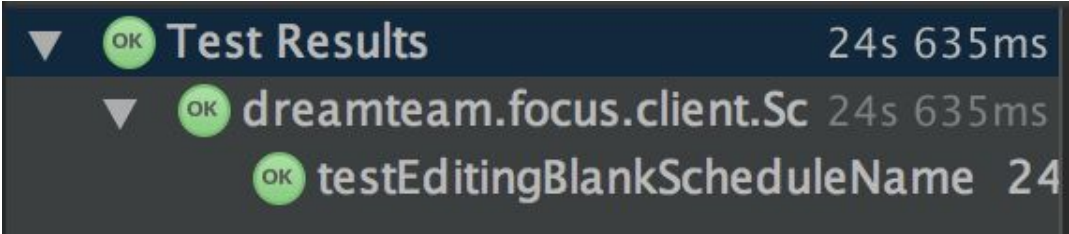
Tested function	AddScheduleActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddScheduleActivity#testCreateScheduleWithSameName()
Test case description	<p>Creating a schedule with a same name of Schedule which is already present in the database should not be allowed and the user should see some kind of error message if he tries to create the schedule with the same name.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddScheduleActivityTest.java

	<p>2. Click on the green arrow () next to the function <code>testCreatScheduleWithSameName()</code></p>
Rationale	<p>Adding a schedule with an existing name should fail on create schedule button pressed as two schedule cannot share the same name. This would not only be confusing for the user but also for the background services and database to figure out which schedule is on and what are its respective profiles. This would lead to mixed notification blocking and unexpected behavior from the app which should not be the case.</p>
Results	 <p>While trying to create the Schedule with the same name, a Toast message saying that the name already exists was thrown successfully.</p>
Comments	<p>No bugs were found during this test.</p>


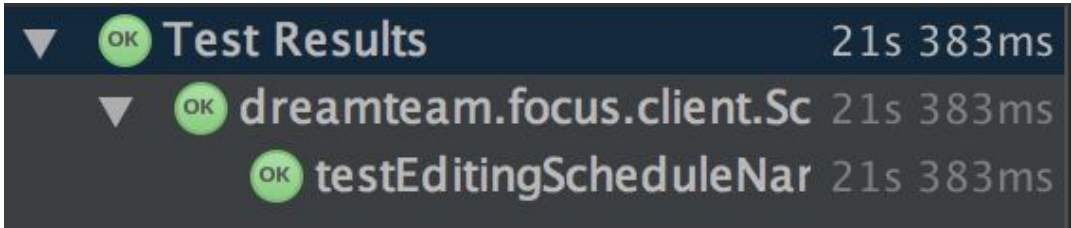
5.4.3. EditScheduleActivity.java


To execute test cases specified on this class, compile and run:

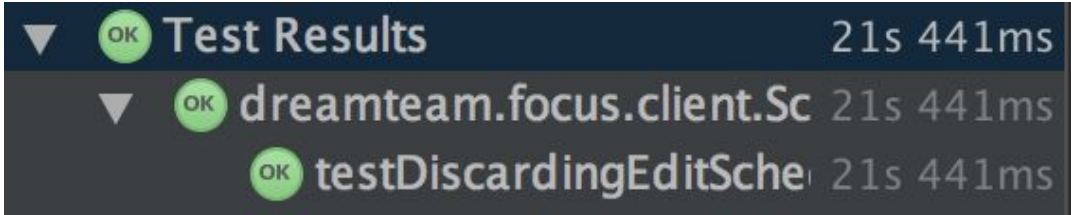
app/src/androidTest/java/dreamteam/focus/client/schedules/EditScheduleActivityTest.java.


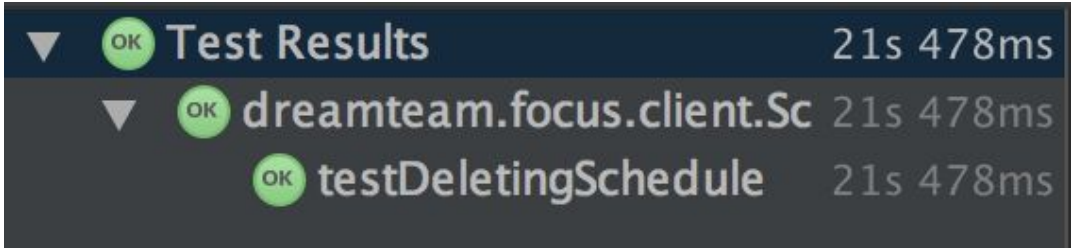
Tested function	EditScheduleActivtiy
Test case location	<code>dreamteam.focus.androidTest.client.schedules.EditScheduleActivity#testEditingBlankScheduleName()</code>
Test case description	<p>This test verifies that the user cannot delete the old name without entering a new name for the Schedule. If the user removes the old name and leaves the name field blank, the user should see an error message asking to enter a name for the schedule.</p> <p>To Run the test:</p> <ol style="list-style-type: none">1. Navigate to: <code>app/src/androidTest/java/dreamteam/focus/client/schedules/EditScheduleActivityTest.java</code>2. Click on the green arrow () next to the function <code>testEditingBlankScheduleName()</code>
Rationale	A schedule cannot be created without a name. Having a schedule without a name would be confusing for the user. This should not be allowed because it's hard to keep track of unique schedules.
Results	<div></div> <p>When an existing schedule was clicked and the Schedule name was deleted and the changes were trying to be saved, the app successfully threw a Toast message asking for a name before saving the changes.</p>
Comments	No bugs were found during the test


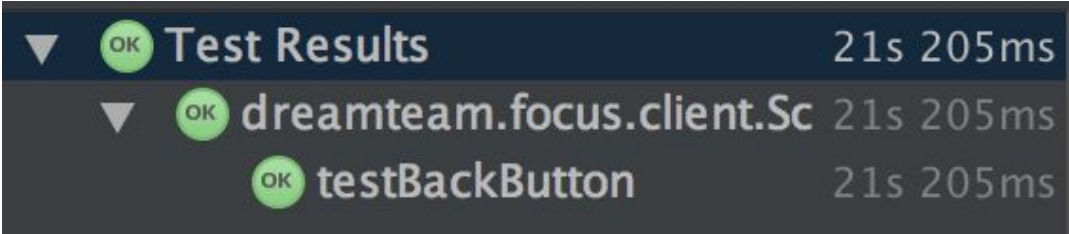
Tested function	EditScheduleActivtiy
Test case location	<code>dreamteam.focus.androidTest.client.schedules.EditScheduleActivity#testEditingScheduleName()</code>
Test case description	This test verifies that the user changes, such as new name and new Profiles in Schedule are saved in the Database which are reflected if the user views the

	<p>Schedule again. The old name should not be seen and the new name should be associated with the old profile instead.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/EditScheduleActivityTest.java 2. Click on the green arrow () next to the function testEditingScheduleName()
Rationale	<p>When the “save changes” button is clicked, all changes should be written to the database and should be reflected in the GUI. The list with all the Schedules should have the new name as the user will now associate the Schedule with the new name instead of the old name.</p>
Results	 <p>When a Schedule name was changed on the EditSchedulesActivity, the changes were immediately reflected on the Schedules List with the new name accurately.</p>
Comments	<p>No bugs were found during the test</p>

Tested function	EditScheduleActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.EditScheduleActivity#testDiscardingEditiScheduleName()
Test case description	<p>This test verifies that the user changes, such as new name and new Profiles in Schedule are NOT saved in the Database and the Schedule object is not modified if the user re-opens it again.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/EditScheduleActivityTest.java 2. Click on the green arrow () next to the function testDiscardingEditScheduleName()
Rationale	<p>When the “discard changes” button is clicked, any changes should be discarded and the database should not be modified. This case was taken into consideration as we wanted the user to have more functionality and be able to change his mind and revert to the old version. The discard button interestingly also changed some fundamentals in the package design as well.</p>

Results	
Comments	No bugs were found during this test


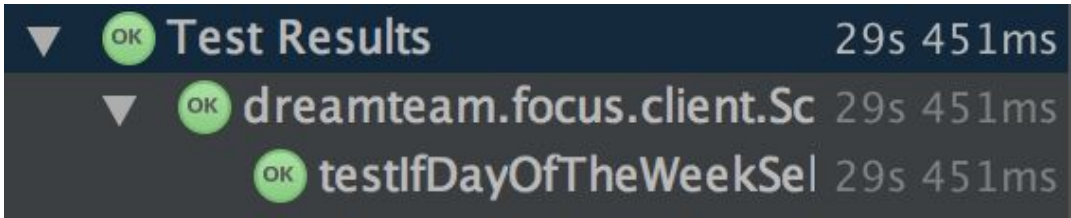
Tested function	EditScheduleActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.EditScheduleActivity#testDeletingSchedule()
Test case description	<p>This test verifies that the Schedule has been deleted from the database and is not seen on the Schedules list. The Schedule should be deleted from the database and should not be seen again.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/EditScheduleActivityTest.java Click on the green arrow () next to the function testDeletingSchedule()
Rationale	When the “delete schedule” button is clicked, the schedule should be deleted and changes written to the database. This is an important case as we need to make sure that deleted schedules aren’t blocking notification and should not appear on the GUI as this is not what the user wanted.
Results	
Comments	A bug was found while doing this test which was making the app crash. The bug was caused because of the database delete function (then, on line 807 of DatabaseConnector.java). The function previously did not account for the Profiles In schedule to be zero and was trying to delete something that wasn’t there and the app was crashing. It was later fixed and the same problem was present in some other functions which were realised during this test. The input during the crash was a schedule with just a name and nothing else.

	<pre> 92/dreamteam.focus I/updateFromServer(): Completed update. 92/dreamteam.focus D/AndroidRuntime: Shutting down VM 92/dreamteam.focus E/AndroidRuntime: FATAL EXCEPTION: main Process: dreamteam.focus, PID: 15992 java.lang.IndexOutOfBoundsException: Invalid index 0, size is 0 at java.util.ArrayList.throwIndexOutOfBoundsException(ArrayList.java:255) at java.util.ArrayList.get(ArrayList.java:308) at dreamteam.focus.server.DatabaseConnector.removeSchedule(DatabaseConnector.java:807) at dreamteam.focus.client.Schedules.EditScheduleActivity\$4.onClick(EditScheduleActivity.java:114) at android.view.View.performClick(View.java:5198) at android.view.View\$PerformClick.run(View.java:21147) </pre>
Tested function	EditScheduleActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.EditScheduleActivity#testBackButton()
Test case description	<p>This test verifies that if the user clicks the back button without saving the Schedule, the changes are not made in the Database and if he re-open the Schedule. This should also not make the app crash.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/EditScheduleActivityTest.java Click on the green arrow () next to the function testBackButton()
Rationale	<p>When the Android back button is clicked, all changes should be discarded, the database should not be modified, and the view should be returned to the list of schedules. Often times programmers forget to code and consider other interactions outside the programmed screen; to have a robust and reliable app is to see how the app reacts to such inputs and hence this test is necessary and the android back button must be tested so that the app behaves exactly how it should in all situations. The test also verifies that the programmers aren't building a stack by starting new intents and not finishing them which makes the overall device slow!</p>
Results	
Comments	No bugs were found during this test


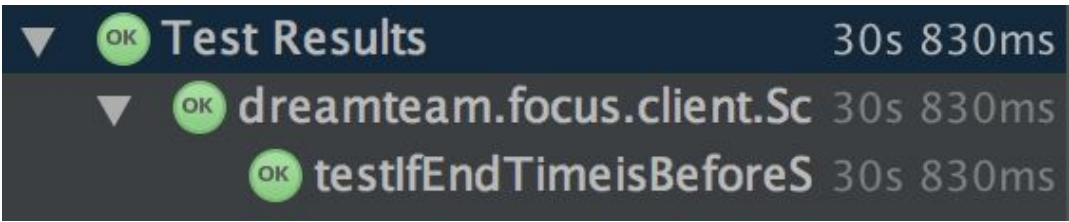
5.4.4. AddProfileToScheduleActivity.java


To execute test cases specified on this class, compile and run:

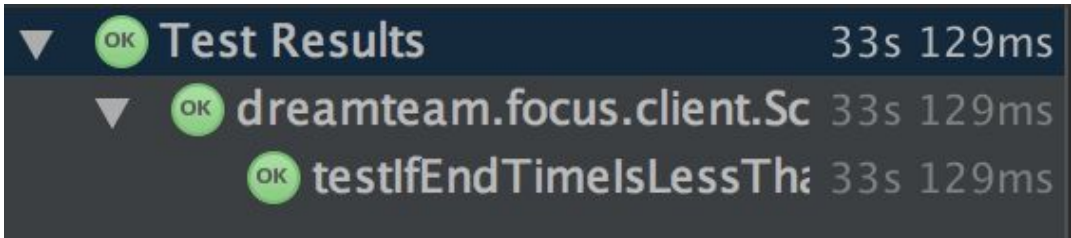
app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToScheduleActivityTest.java.


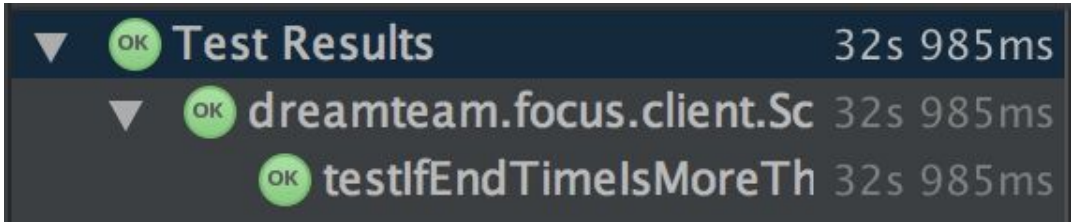
Tested function	AddProfileToScheduleActivtiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddProfileToScheduleTest#testIfDayOfTjeWeekSelected()
Test case description	<p>This test verifies that all constraints on fields must be satisfied before a profile is to be associated with a schedule. There should be at least one day selected while making the Schedule otherwise the user should see an error message asking to select at-least one day to add the profile.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToScheduleActivityTest.java 2. Click on the green arrow () next to the function testIfDayOfTheWeekSelected()
Rationale	A profile cannot be added to a schedule without specifying a day to repeat on/to run on. A schedule functions according to each day of the week and without specifying the day while adding a profile doesn't make sense.
Results	
Comments	This test gave some problems in the beginning which were to do with Espresso trying to select a profile in Schedule when no profiles had been made before. This problem was then solved by introducing some profiles in the setup before running any other tests on lines 71,72 and 73 of in AddProfileToScheduleTest.java

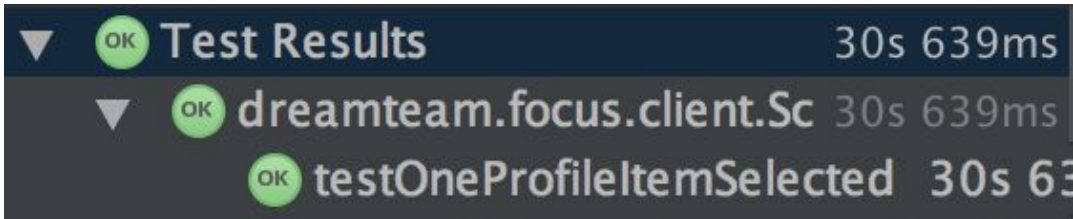
Tested function	AddProfileToScheduleActivtiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddProfileToScheduleTest#testIfEndTimeBeforeStartTime()
Test case description	<p>The user should select a reasonable start Time and end time such that the start time is earlier end time otherwise the user should see an error message asking him to change the time and try again once he clicks on the Add Profile button after entering all the other constraints.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to:

	<p>app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToScheduleActivityTest.java</p> <p>2. Click on the green arrow () next to the function testIfEndTimeBeforeStartTime()</p>
Rationale	The start time must be earlier than the end time and this is a fundamental case. The end time can never be before the start time as this is not possible. This test ensures that those constraints have been met.
Results	
Comments	No bugs were found during this test


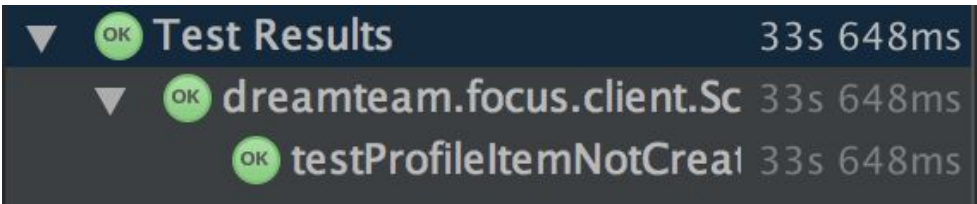
Tested function	AddProfileToScheduleActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddProfileToScheduleTest#testIfStartTimeIsLessThan10Minutes()
Test case description	<p>The user should select a reasonable start Time and end time such that the start time and end Time are at-least 10 mins apart otherwise the user should see an error message asking them to revise the time.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToScheduleActivityTest.java Click on the green arrow () next to the function testIfStartTimeIsLessThan10Minutes()
Rationale	The end time must not be lesser than 10 minutes of the start time as this was one of the customer requirements and should be taken into consideration. This test verifies and validates that requirement and hence is necessary.


Results	
Comments	No bugs were found during this test

Tested function	AddProfileToScheduleActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddProfileToScheduleTest#testIfStartTimeIsMoreThan10Hours()
Test case description	<p>The user should select a reasonable start Time and end time such that the end time is 10 hours after the start time otherwise the user should see an error message asking them to revise the time.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToScheduleActivityTest.java Click on the green arrow () next to the function testIfStartTimeIsMoreThan10Hours()
Rationale	The end time must not be more than 10 hours after the start time as this was one of the customer requirements and should be taken into consideration. This test verifies and validates that requirement and hence is necessary.
Results	
Comments	No bugs were found during this test

Tested function	AddProfileToScheduleActivtiy
Test case location	<code>dreamteam.focus.androidTest.client.schedules.AddProfileToScheduleTest#testOneProfileItemSelected()</code>
Test case description	<p>The user should select at-least one Profile which he would like to add in the Schedule before clicking the “Add Profile” button or else the user should an error message asking them to select at-least one profile to add to the mentioned time and day.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: <code>app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToScheduleActivityTest.java</code> 2. Click on the green arrow (▶) next to the function <code>testOneProfileItemSelected()</code>
Rationale	A profile cannot be added to a schedule without specifying a profile to be added as it wouldn't make sense to add and block a time slot on the calendar without having any apps to block. This would not serve the purpose of the app but would rather be seen as a calendar instead to our app.
Results	
Comments	No bugs were found during this test.

Tested function	AddProfileToScheduleActivtiy
Test case location	<code>dreamteam.focus.androidTest.client.schedules.AddProfileToScheduleTest#testProfileItemNotCreatedInSchedule()</code>
Test case description	<p>This test checks that the changes aren't made in the Database if the user clicks the “Discard” button. If the user navigates back to the Schedule, the profile which was supposed to be added shouldn't be seen on the calendar.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to:

	<p>app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToScheduleActivityTest.java</p> <p>2. Click on the green arrow () next to the function testProfileItemNotCreatedInSchedule()</p>
Rationale	When the discard button is clicked, all changes should be discarded, the database should not be modified, and the view should be returned to the original activity to EditScheduleActivity without seeing the changes.
Results	
Comments	No bugs were found during this test

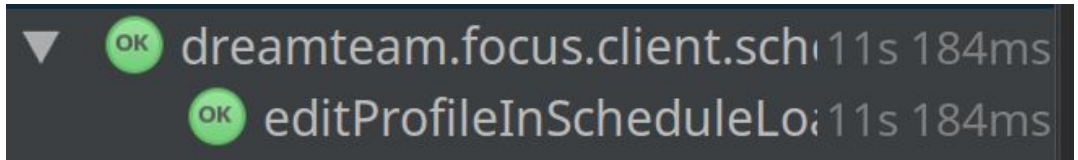
Tested function	AddProfileToScheduleActivitiy
Test case location	dreamteam.focus.androidTest.client.schedules.AddProfileToScheduleTest#testProfileItemCreatedInSchedule()
Test case description	<p>This test checks that the user can see the Profile in Schedule which he just created in the right place. The user should see it in the designated calendar view and should the display must contain the same profile name and time that the user entered while adding the Profile to Schedule.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToScheduleActivityTest.java Click on the green arrow () next to the function testProfileItemCreatedInSchedule()
Rationale	When the user enters all the correct steps to create a profile in Schedule, the user should see the Profile appear on the calendar in the right place. This test case is important as this was one of the requirements to see all the profiles in an organised way. Once it has been added to the calendar the user should see all the Profiles in Schedule in a weekly calendar.

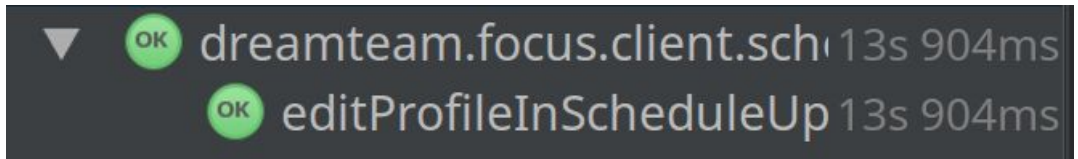
Results	<div><div>▼</div><div>OK</div><div>Test Results</div><div>33s 573ms</div></div> <div><div>▼</div><div>OK</div><div>dreamteam.focus.client.Sc</div><div>33s 573ms</div></div> <div><div>OK</div><div>testProfileItemCreatedI</div><div>33s 573ms</div></div>
Comments	No bugs were found during this test

5.4.5. EditProfileInScheduleActivity.java

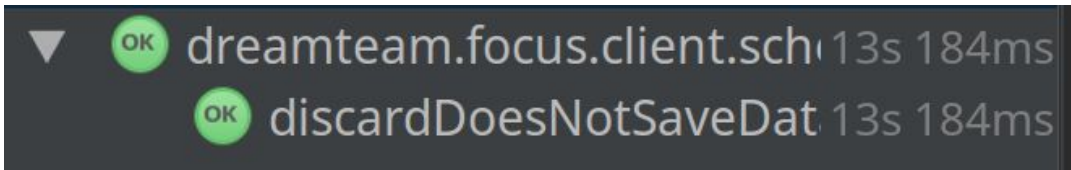
To execute test cases specified on this class, compile and run:

app/src/androidTest/java/dreamteam/focus/client/schedules/EditProfileInScheduleActivityTest.java.

Tested function	EditProfileInScheduleActivity
Test case location	dreamteam.focus.androidTest.client.schedules.editProfileInScheduleLoadsDataCorrectly()
Test case description	The EditProfileInScheduleActivity loads with the data from the ProfileInSchedule, the test checks whether the data corresponds to the specific ProfileInSchedule clicked.
Rationale	When the user presses the ProfileInSchedule in EditSchedule, he should be able to see the current state of that ProfileInSchedule to help him decide if he wants to make any changes
Results	 <p>The EditProfileInSchedule loads with the corresponding ProfileInSchedule's data. The start time, end time, day of the week and profile name match with the ProfileInSchedule.</p>
Comments	No bugs were found during this test

Tested function	EditProfileInScheduleActivity
Test case location	dreamteam.focus.androidTest.client.schedules.editProfileInScheduleUpdatesPIS()
Test case description	The credentials of a ProfileInSchedule are changed in EditProfileInScheduleActivity and update button is pressed. The updated ProfileInSchedule is pressed from the EditScheduleActivity and the test checks whether the EditProfileInScheduleActivity opens up with updated ProfileInSchedule data.
Rationale	When the user presses the Update button, The ProfileInSchedule is updated in the database so when the user selects the updated ProfileInSchedule again from the EditScheduleActivity, the user should be directed to the EditProfileInScheduleActivity with the corresponding updated ProfileInSchedule data.
Results	

	The EditProfileInSchedule loads with the updated ProfileInSchedule data and the position of the ProfileInSchedule changes in EditScheduleActivity if the day is changed.
Comments	The problem encountered in this part was that when you create a copy of a profileInSchedule on the same day, The application allows that but that should not be the case. So we had to update the database to make ProfileInSchedule unique in every schedule.

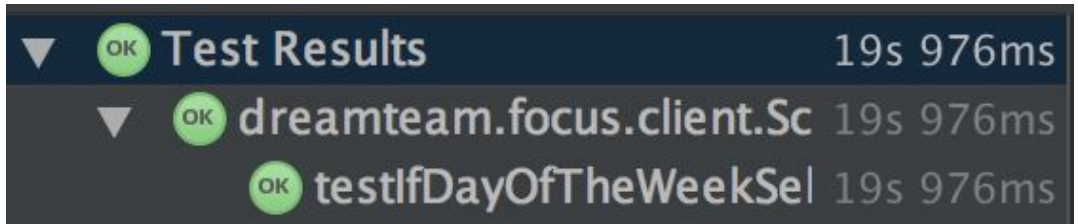
Tested function	EditProfileInScheduleActivity
Test case location	dreamteam.focus.androidTest.client.schedules.discardDoesNotSaveData()
Test case description	The credentials of a ProfileInSchedule are changed in the EditProfileInScheduleActivity and the discard Button is Pressed, The test checks whether the ProfileInSchedule gets updated or not.
Rationale	When the user changes the ProfileInSchedule credentials in EditProfileInScheduleActivity and presses Discard Button rather than Update Button, the changes are not saved on the database, so when the EditProfileInScheduleActivity opens up again, it still contains the old ProfileInSchedule data.
Results	 <p>The EditProfileInSchedule loads with the corresponding ProfileInSchedule's old data. The start time, end time, day of the week and profile name match with the old ProfileInSchedule.</p>
Comments	No bugs were found during this test

5.4.6. AddProfileToNewSchedule.java

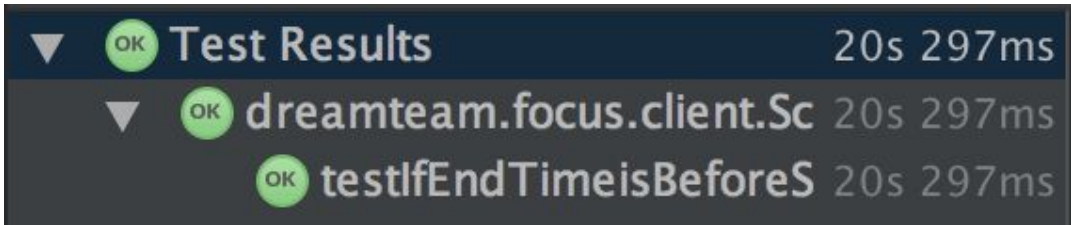
To execute test cases specified on this class, compile and run:

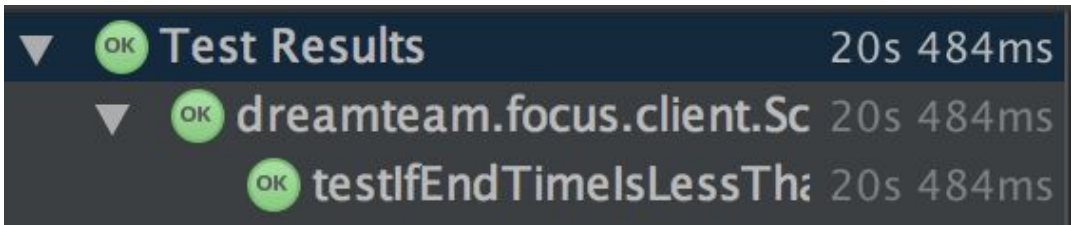
app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToNewScheduleTest.java.

Tested function	AddProfileToNewSchedule
Test case location	dreamteam.focus.androidTest.client.schedules.AddProfileToNewScheduleTest#testIfDayOfTheWeekSelected()


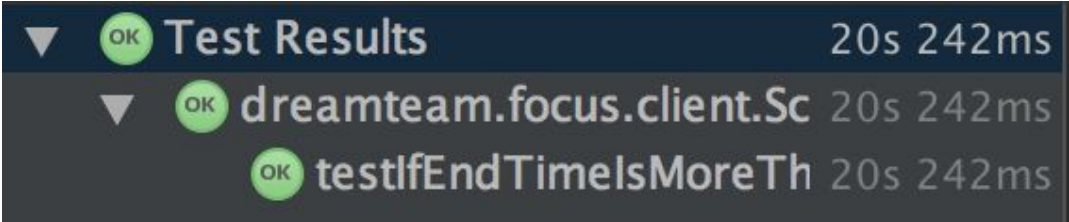
Test case description	<p>This test verifies that all constraints on fields must be satisfied before a profile is to be associated with a schedule. There should be at least one day selected while making the Schedule otherwise the user should see an error message asking to select at-least one day to add the profile.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToNewScheduleTest.java 2. Click on the green arrow (▶) next to the function testIfDayOfTheWeekSelected()
Rationale	A profile cannot be added to a schedule without specifying a day to repeat on/to run on. A schedule functions according to each day of the week and without specifying the day while adding a profile doesn't make sense.
Results	
Comments	This test gave some problems in the beginning which were to do with Espresso trying to select a profile in Schedule when no profiles had been made before. This problem was then solved by introducing some profiles in the setup before running any other tests.


Tested function	AddProfileToNewSchedule
Test case location	dreamteam.focus.androidTest.client.schedules.AddProfileToNewScheduleTest#testIfEndTimeBeforeStartTime()
Test case description	<p>The user should select a reasonable start Time and end time such that the start time is earlier end time otherwise the user should see an error message asking him to change the time and try again once he clicks on the Add Profile button after entering all the other constraints.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToNewScheduleTest.java 2. Click on the green arrow (▶) next to the function testIfEndTimeBeforeStartTime()
Rationale	The start time must be earlier than the end time and this is a fundamental case. The end time can never be before the start time as this is not possible. This test ensures that those constraints have been met.

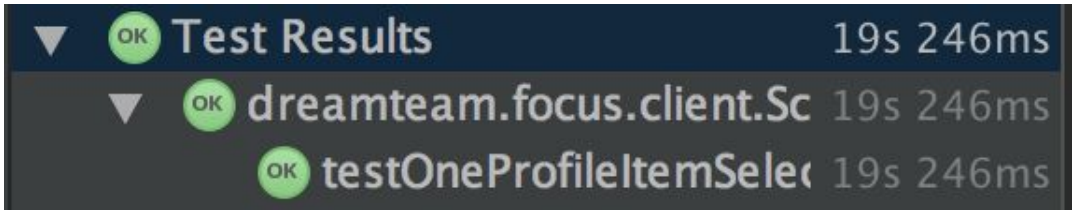
Results	
Comments	No bugs were found during this test

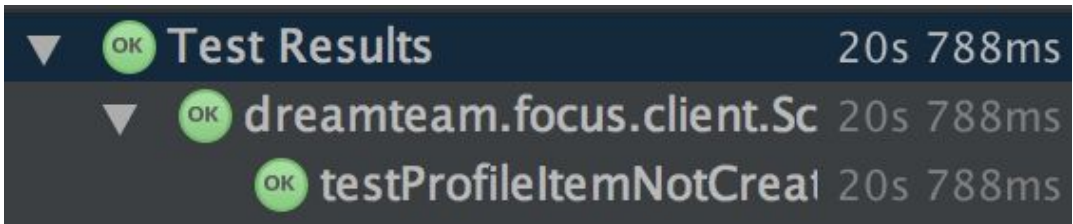
Tested function	AddProfileToNewSchedule
Test case location	<code>dreamteam.focus.androidTest.client.schedules.AddProfileToNewScheduleTest#testIfStartTimeIsLessThan10Minutes()</code>
Test case description	<p>The user should select a reasonable start Time and end time such that the start time and end Time are at-least 10 mins apart otherwise the user should see an error message asking them to revise the time.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: <code>app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToNewScheduleTest.java</code> 2. Click on the green arrow (▶) next to the function <code>testIfEndTimeIsLessThan10Minutes()</code>
Rationale	The end time must not be lesser than 10 minutes of the start time as this was one of the customer requirements and should be taken into consideration. This test verifies and validates that requirement and hence is necessary.
Results	
Comments	No bugs were found during this test

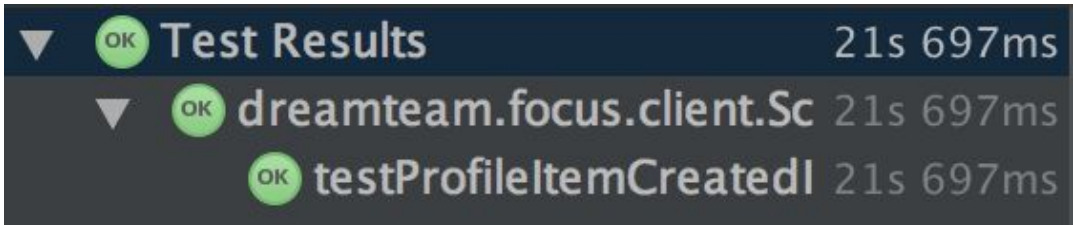
Tested function	AddProfileToNewSchedule
Test case location	<code>dreamteam.focus.androidTest.client.schedules.AddProfileToNewScheduleTest#testIfStartTimeIsLessThan10Minutes()</code>

	<code>leTest#testIfStartTimeIsMoreThan10Hours()</code>
Test case description	<p>The user should select a reasonable start Time and end time such that the end time is 10 hours after the start time otherwise the user should see an error message asking them to revise the time.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: <code>app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToNewScheduleTest.java</code> 2. Click on the green arrow () next to the function <code>testIfEndTimeIsMoreThan10Hours()</code>
Rationale	The end time must not be more than 10 hours after the start time as this was one of the customer requirements and should be taken into consideration. This test verifies and validates that requirement and hence is necessary.
Results	
Comments	No bugs were found during this test

Tested function	<code>AddProfileToNewSchedule</code>
Test case location	<code>dreamteam.focus.androidTest.client.schedules.AddProfileToNewScheduleTest#testOneProfileItemSelected()</code>
Test case description	<p>The user should select at-least one Profile which he would like to add in the Schedule before clicking the “Add Profile” button or else the user should an error message asking them to select at-least one profile to add to the mentioned time and day.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: <code>app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToNewScheduleTest.java</code> 2. Click on the green arrow () next to the function <code>testOneProfileItemSelected()</code>
Rationale	A profile cannot be added to a schedule without specifying a profile to be added as it wouldn't make sense to add and block a time slot on the calendar without

	having any apps to block. This would not serve the purpose of the app but would rather be seen as a calendar instead to our app.
Results	
Comments	No bugs were found during this test

Tested function	AddProfileToNewSchedule
Test case location	<code>dreamteam.focus.androidTest.client.schedules.AddProfileToNewScheduleTest#testProfileItemNotCreatedInSchedule()</code>
Test case description	<p>This test checks that the changes aren't made in the Database if the user clicks the "Discard" button. If the user navigates back to the Schedule, the profile which was supposed to be added shouldn't be seen on the calendar.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 3. Navigate to: <code>app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToNewScheduleTest.java</code> 4. Click on the green arrow (▶) next to the function <code>testProfileItemNotCreatedInSchedule()</code>
Rationale	When the discard button is clicked, all changes should be discarded, the database should not be modified, and the view should be returned to the original activity to <code>EditScheduleActivity</code> without seeing the changes.
Results	
Comments	No bugs were found during this test

Tested function	AddProfileToNewSchedule
Test case location	dreamteam.focus.androidTest.client.schedules.AddProfileToNewScheduleTest#testProfileItemCreatedInSchedule()
Test case description	<p>This test checks that the user can see the Profile in Schedule which he just created in the right place. The user should see it in the designated calendar view and should the display must contain the same profile name and time that the user entered while adding the Profile to Schedule.</p> <p>To Run the test:</p> <ol style="list-style-type: none"> 1. Navigate to: app/src/androidTest/java/dreamteam/focus/client/schedules/AddProfileToNewScheduleTest.java 2. Click on the green arrow (▶) next to the function testProfileItemCreatedInSchedule()
Rationale	When the user enters all the correct steps to create a profile in Schedule, the user should see the Profile appear on the calendar in the right place. This test case is important as this was one of the requirements to see all the profiles in an organised way. Once it has been added to the calendar the user should see all the Profiles in Schedule in a weekly calendar.
Results	
Comments	No bugs were found during this test


5.5. Server

5.5.1. BackgroundService.java


In tests pertaining to BackgroundService.java, both Espresso and UIAutomator tools have been used. UIAutomator allows to check notifications received by a device, which is a key component of BackgroundService.java.

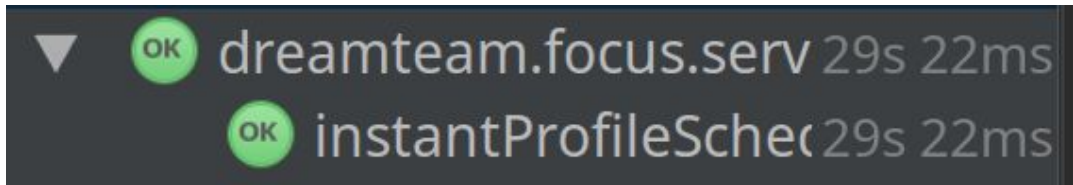
To execute test cases specified on this class, compile and run:

```
app/src/androidTest/java/dreamteam/focus/client/schedules/BackgroundServiceTest.java.
```

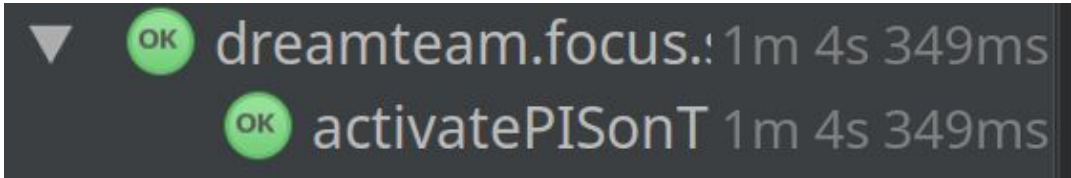
Tested function	BackgroundService#tick BackgroundService#sendNotification
Test case location	dreamteam.focus.androidTest.server.BackgroundServiceTest#instantProfileActivate
Test case description	The test case checks if a notification is thrown when the user activates a profile as a single-time timer activation. This test case first activates a profile from the ProfilesActivity by toggling a profile on and then waits for a few seconds for the notification to pop up in the status bar which explicitly tells the user that the particular profile is now active.
Rationale	When a user activates a profile directly from the list of profiles, the user should receive a notification that the profile has gone active. The specific input from the user for this feature is just for him/her to activate a profile from the list of profiles.
Results	 <p>The notification regarding profile being active was thrown successfully.</p>
Comments	No bugs were discovered while this test is run.

Tested function	BackgroundService#tick BackgroundService#sendNotification
Test case location	dreamteam.focus.androidTest.server.BackgroundServiceTest#instantProfileDeactivate
Test case description	The test case checks if a notification is thrown when the user deactivates a profile which was made active as a single-time timer activation. This test case first activates a profile from the ProfilesActivity, then waits for a few seconds (assuming the user might possibly do so in real-time usage). Then the same profile that was activated is toggled off and then waits for a few seconds for the

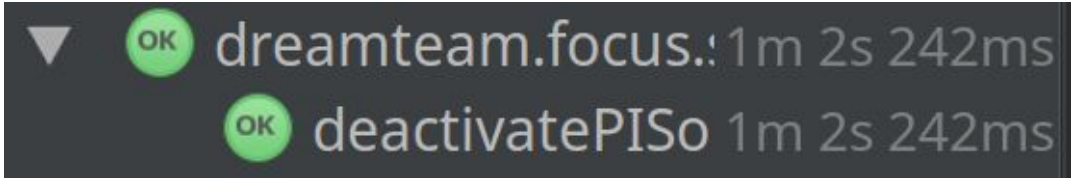
	notification to pop up in the status bar which explicitly tells the user that the profile is now inactive.
Rationale	When a user deactivates a profile directly from the list of profiles that was earlier active, the user should receive a notification that the profile has gone inactive. The specific input from the user for this feature is just for him/her to deactivate a profile from the list of profiles that was earlier made active.
Results	 <p>The notification regarding profile being inactive was thrown successfully.</p>
Comments	No bugs were discovered while this test is run.

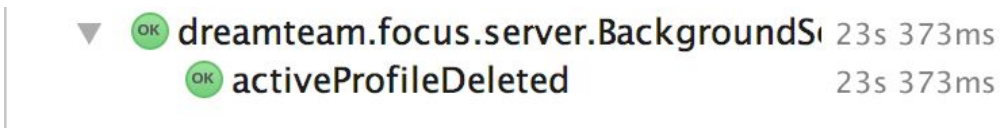
Tested function	BackgroundService#tick BackgroundService#sendNotification
Test case location	dreamteam.focus.androidTest.server.BackgroundServiceTest#instantProfileScheduledDeactivate
Test case description	The test case checks if a notification is thrown when the app deactivates a profile which was made by the user as a active as a single-time timer activation at its scheduled end time. This test case first activates a profile in the database with start time as 9 minutes prior to current system time and end time as 1 minute post system time. Then the test waits for a minute (time to end time of this activated profile) for the notification to pop up in the status bar which explicitly tells the user that the profile is now inactive.
Rationale	When a profile that was activated as single-timer activation reaches it ends time, the user should receive a notification that the profile has gone inactive. The specific input chosen by us in this case is the least time possible for instant activation i.e 10 minutes but since we cannot wait to see the test case run for 10 minutes, we set the start time for the profile as 9 minutes before system time and 1 minute post system time, simply so that the test case runs for almost around a minute.
Results	 <p>The notification regarding profile being inactive was thrown successfully.</p>
Comments	We noticed that an instantly active profile was not getting deactivated at the scheduled time. This bug was found in the tick() function in BackgroundService. The window of time that checks whether the scheduled time lies close to the start time was too small and hence there were instances when a notification was not being sent. We decided to expand this window to compensate for the misses and

	fixed the bug. On regression testing of other elements, everything worked.
--	--

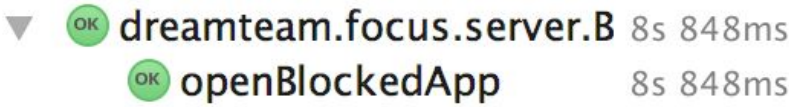
Tested function	BackgroundService#tick BackgroundService#sendNotification
Test case location	dreamteam.focus.androidTest.server.BackgroundServiceTest#activatePISonTime
Test case description	The test case checks if a notification is thrown when an active schedule has a profile that has gone active. The test case adds a profile onto a schedule that has an activation time one minute past the current system time. The test then waits for one minute to check for the expected notification.
Rationale	When a profile inside an active schedule is activated during a scheduled time, a notification should be thrown. In the test case, we shrink the test time by making a scheduled activation one minute after the current system time to decrease the total test time.
Results	 <p>The notification regarding profile being active was thrown successfully.</p>
Comments	We noticed that an instantly active profile was not getting deactivated at the scheduled time. This bug was found in the tick() function in BackgroundService. The window of time that checks whether the scheduled time lies close to the start time was too small and hence there were instances when a notification was not being sent. We decided to expand this window to compensate for the misses and fixed the bug. On regression testing of other elements, everything worked.

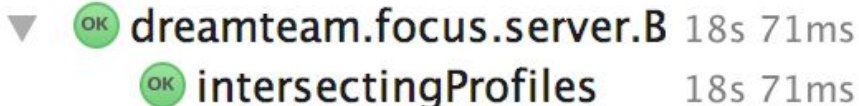
Tested function	BackgroundService#tick BackgroundService#sendNotification
Test case location	dreamteam.focus.androidTest.server.BackgroundServiceTest#deactivatePISonTime
Test case description	The test case checks if a notification is thrown when an active schedule has a profile that is deactivated. The test case adds a profile onto a schedule that has an deactivation time one minute past the current system time. The test then waits for one minute to check for the expected notification.
Rationale	When a profile inside an active schedule is deactivated during a scheduled time, a notification should be thrown. In the test case, we shrink the test time by making a ten-minute schedule that has gone active nine minutes before the current system time to decrease the total test time.

Results	 <p>The notification regarding profile being inactive was thrown successfully.</p>
Comments	<p>We noticed that an instantly active profile was not getting deactivated at the scheduled time. This bug was found in the tick() function in BackgroundService. The window of time that checks whether the scheduled time lies close to the start time was too small and hence there were instances when a notification was not being sent. We decided to expand this window to compensate for the misses and fixed the bug. On regression testing of other elements, everything worked.</p>

Tested function	BackgroundService#tick BackgroundService#sendNotification
Test case location	dreamteam.focus.androidTest.server.BackgroundServiceTest#activeProfileDeleted
Test case description	<p>The test case checks if a notification is thrown when the user deletes a profile which was made active as a single-time timer activation. This test case first activates a profile from the ProfilesActivity, then waits for a few seconds (assuming the user might possibly do so in real-time usage). Then the same profile that was activated is deleted by going to the EditProfileActivity. Now the test waits for a few seconds for the notification to pop up in the status bar which explicitly tells the user that the profile is now inactive.</p>
Rationale	<p>When a user deleted a profile directly from the list of profiles that was earlier active, the user should receive a notification that the profile has gone inactive. The specific input from the user for this feature is just for him/her to delete a profile from the list of profiles that was earlier made active.</p>
Results	 <p>The notification regarding profile being inactive was thrown successfully.</p>
Comments	<p>No bugs were discovered while this test is run.</p>

Tested function	BackgroundService#blockApps
Test case location	dreamteam.focus.androidTest.server.BackgroundServiceTest#openBlockedApp
Test case description	<p>The test case checks if a user is disallowed from opening a blocked app. This test case first activates a profile from the ProfilesActivity that is blocking the Messenger app on the phone*. Then it tries to open the Messenger app on the</p>

	<p>phone and checks if the home screen is reached or not because if the home screen isn't reached that means user is still on messenger app which is blocked.</p> <p>*NOTE: For this test to pass the tester must have the "Messenger" app of facebook installed on the device on which the test takes place.</p>
Rationale	<p>When a user tries to access an app blocked by one of the active profiles/schedules, the user should be disallowed from accessing the app and get redirected to the home screen of the phone. The specific input from the user for this feature is just for him/her to try to access a blocked app.</p>
Results	 <p>Messenger wasn't allowed to open when profile was active.</p>
Comments	<p>No bugs were discovered while this test is run.</p>

Tested function	<p>BackgroundService#tick BackgroundService#blockApps</p>
Test case location	<p>dreamteam.focus.androidTest.server.BackgroundServiceTest#intersectingProfiles</p>
Test case description	<p>The test case checks if a user is disallowed from opening a blocked app on intersecting profiles. This test case first activates two profiles in the database that are both blocking the Messenger app on the phone* and some other apps. Then it tries to open the Messenger app on the phone and checks if the home screen is reached or not because if the home screen isn't reached that means user is still on messenger app which is blocked. Once this is passed, profile1 is deactivated and messenger is accessed again and again this should be blocked as the other profile which is still active, continues to block it.</p> <p>*NOTE: For this test to pass the tester must have the "Messenger" app of facebook installed on the device on which the test takes place.</p>
Rationale	<p>When a user tries to access an app blocked by two intersecting active profiles, the user should be disallowed from accessing the app. Further, when one of those profiles gets deactivated, the restriction from still be applied from the other active profile. The specific input from the user for this feature is just for him/her to try to access a blocked app when two active profiles intersect and when one of them becomes inactive.</p>
Results	

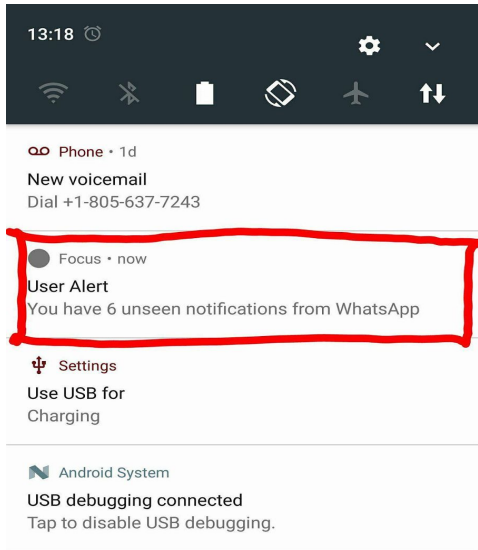
	Messenger was blocked even after interesting profile was both active or inactive.
Comments	No bugs were discovered while this test is run.

5.6. Manual tests

The following test case require two instruments, hence a manual test with screenshots is attached since an independent unit test isn't possible.

To execute test cases specified on this class, compile and run:

app/src/androidTest/java/dreamteam/focus/server/BackgroundServiceTest.java.

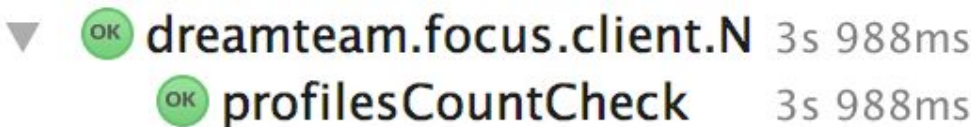
Tested function	BackgroundService#tick BackgroundService#sendNotification
Test case description	The test case checks if a notification regarding suppressed notifications are thrown when a profile goes inactive. This test case first deactivates a profile from the ProfilesActivity by toggling a profile off and then waits for a few seconds for the notification to pop up in the status bar which explicitly tells the user that notifications of app "Whatsapp" were missed.
Rationale	When a profile goes inactive, the user should receive a notification regarding the missed notifications blocked by an app. The specific input for this test is that another device sends a number of messages on 'Whatsapp' (exactly 6 in this screenshot) to the user, and whatsapp is blocked currently by an active profile on the user's device.
Results	 <p>The notification regarding number of notifications missed was thrown successfully.</p>
Comments	No bugs were discovered while this test is run.

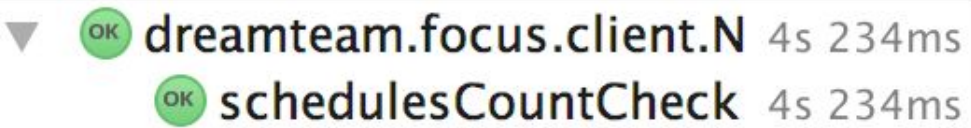
5.7. Nonfunctional Tests (Extra)

We have also done two tests for non functional requirements.

To execute test cases specified on this class, compile and run:

`app/src/androidTest/java/dreamteam/focus/client/NonFunctionalTest.java`.

Tested function	ProfilesActivity
Test case location	<code>dreamteam.focus.androidTest.client.NonFunctionalTest#profilesCountCheck</code>
Test case description	Firstly 20 profiles are added to the database. Then the test takes user to ProfilesActivity and tries to add another profile but isn't allowed to do so as the limit is reached.
Rationale	The user isn't allowed to add more than 20 profiles.
Results	 <p>The user wasn't able to redirect to add profile activity to add another profile.</p>
Comments	No bugs were discovered while this test is run.

Tested function	SchedulesActivity
Test case location	<code>dreamteam.focus.androidTest.client.NonFunctionalTest#schedulesCountCheck</code>
Test case description	Firstly 20 schedules are added to the database. Then the test takes user to SchedulesActivity and tries to add another schedule but isn't allowed to do so as the limit is reached.
Rationale	The user isn't allowed to add more than 20 schedules.
Results	 <p>The user wasn't able to redirect to add schedule activity to add another schedule.</p>
Comments	No bugs were discovered while this test is run.

TODO:

Aarav : implement test cases:

If schedule is active, next to any of the PIS that is supposed to be active, a running timer replaces the timings which shows the time remaining.

- Timer changes back to timings when end time of PIS is reached.