CodeDispensary

ICT2105 Mobile Application Development Spring 2020

What needs to be done

Fork the repo **ict2105-quiz03-2020** and inspect the code within the project. Examine the skeleton code before working on the lab. Refer to the screenshots on the following page.

As we know, everyone on campus needs to report their temperatures due to the current epidemic situation. For visitors, having multiple personnel and equipment to check every entrant is costing a lot of resources and frustration in affected staff. We have devised an ICT solution for this.

We will just have the front desk officer to hand out unique codes to each visitor, who will then use the code to login into our existing temperature reporting website. A camera will be mounted to the front desk that automatically detects faces and takes photos of them. There will be a separate background Tagger app that tags the unique code with the photos taken, using timestamps. Your role is to create the CodeDispensary app. We will take care of the Tagger app (i.e., you don't have to worry about this).

With the pressing situation, we have to deploy this in 3 hours, hence this "Quiz". So, you definitely have to complete this in 2 hours, and we will spend 1 hour to test and choose the best implementation.

Since we're nice clients, we have already drafted our requirement into a user story as follows:

As a front desk officer,
I want to generate unique codes and click print,
So that I can hand out unique codes to campus visitors efficiently.

Acceptance criteria:

1. (1 mark) There is a constant stream of visitors to the campus, especially in peak hours. I need the CodeDispensary app to be ALWAYS ON. I may occasionally use other apps like Youtube or Fortnite in the rare occasion of no visitors, but I expect the CodeDispensary service to be always there when I need it. I have already provided an empty CodeDispensaryService class in the base repo.

2. (1 mark) In the background, the app will constantly generate 8-char codes at the rate of one number per 1678ms. I do not need to know what the past codes are, but I need this to be ALWAYS RUNNING as much as my phone allows.
   Each char in the code can be one of any characters in the string:
   "`abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ01234567890`"
   (see screenshots for examples)

3. (1 mark) I should be able to easily navigate back to the CodeDispensary app through a persistent notification (see screenshots). The notification title must be "CodeDispensary Notification". I should be able to simply click anywhere on the notification and it will bring me back to the CodeDispensary app.

4. (1 mark) On the same CodeDispensary notification, I should also constantly see the currently generated code (see screenshots).

5. (1 mark) There is only one view of the app, and in this view, I should see a **SHUTDOWN** button, a **PRINT COD3** button, and a **textViewPrint** text view that shows the printed layout for the visitor (see screenshots). You may wonder why is the print button name so strange? Well, **cos I'm the client and I like it that way.** Note that this printed layout includes a *feel good message* for the visitor so as to play our small part in improving the public mood in this dire environment.

6. (1 mark) On clicking the **PRINT COD3** button, a printable text that includes the code is generated in the **textViewPrint** element. The printable text includes a customized *feel good message* generated via a highly secretive yet effective algorithm. We will provide the algorithm in the base repo but it is a trade secret so please do not share with anyone. You MUST use the method named `highlySecretiveYetEffectiveMsgGenerator(int code)` to generate this message, which will be displayed in the view as mentioned in 5. Note that this method may be slightly resource-intensive due to its ingenuity, but I certainly do not wish to see any laggy unresponsiveness on the screen, let alone ANR…

7. (1 mark) Note that the above method should only be run if WIFI is on. This is because the actual printer communicates via WIFI. (note again we will handle the printing, and you do not have to actually make any print calls)

8. (1 mark) When the officer presses the **SHUTDOWN** button, everything with regards to the CodeDispensary app should cease to function, including any

code generation task, and any current and future notifications spawned from the app.

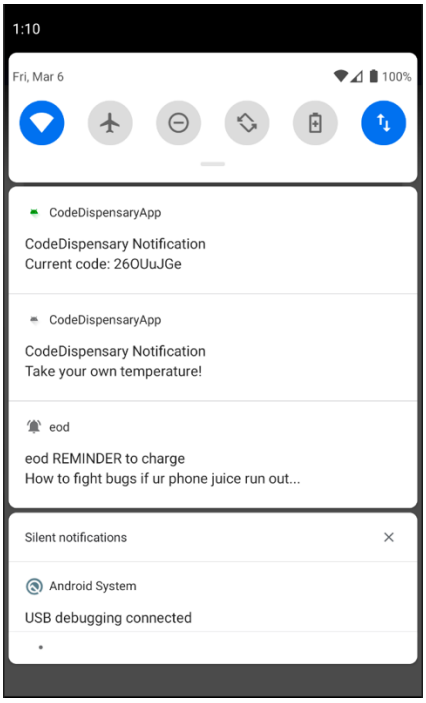
9. (1 mark) Well-commented code.

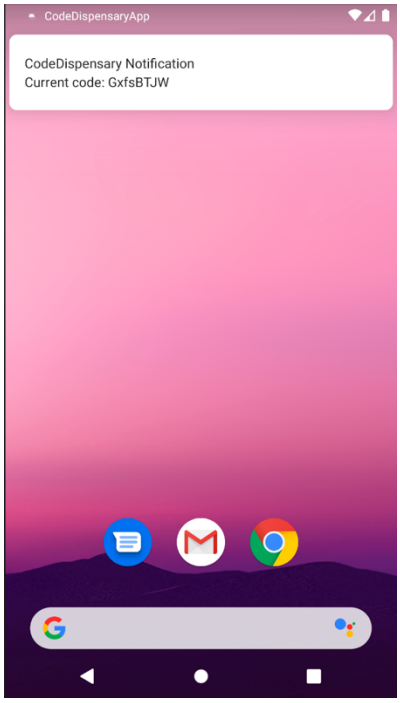
10. (1 mark) Good naming conventions for methods and variables.

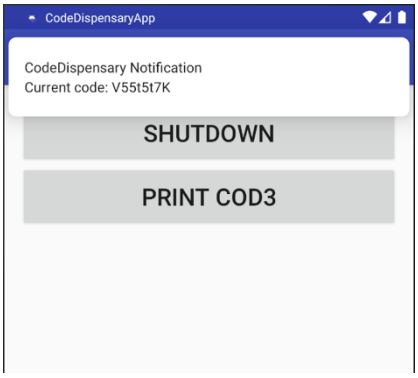
11. **[STRETCH GOAL]**
    (Bonus 1 mark) The front desk officer is often so busy that they will forget to report their own temperature. They are also at high risk, so we want them to take their temperature every 3 hours. Every 3 hours, the CodeDispensary app should display a notification with the text "Take your own temperature!" (see screenshots). We do not need it to be done exactly at the hour, but you must guarantee it will run when the system allows, as taking temperature is important in these times. On launching the CodeDispensary app, this notification should already appear to remind the officer to report his/her temperature at the start of the day.

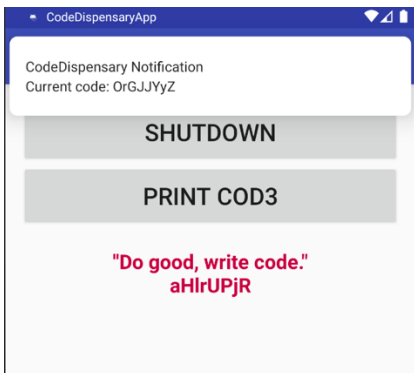
IMPORTANT:

Ensure that the ViewInstrumentedTest.kt test file compiles without errors (running and passing it will ensure even less errors). Do not edit any existing **package**, **class**, **variable**, **method** or **layout** file names. You may **add to, but do not edit existing dependencies** in, the gradle files.









Lab Quiz 3

1. Fork the repo **ict2105-quiz03-2020**, and then clone it. This code is incomplete.
2. Complete implementation of the code needed to run the **CodeDispensary** app that satisfies all the acceptance criteria listed.
3. Commit and push all changes to your forked repository **ict2105-quiz03-2020.**

END OF DOCUMENT