Rollin With My Homies SDD

Organized Classes – Previous versions are found at the bottom of the document, in ascending order from oldest to newest.

Version 0.01 (reflects “Must” requirements). Model was created before introduction Android Studio.

* Main – class responsible for running the program.
  + Testbed – Class containing tests for program.
    - GenerateIntegerTest – Class responsible for testing the functionality of the GenerateInteger class.
    - RollTest – Class responsible for testing the functionality of the Roll class.
  + Roll – Class responsible for returning an integer
    - GenerateInteger – Class responsible for generating an integer within a given range.

Requirements are listed in rough order of priority. If a requirement is made undone from adding a new feature, further development will be stopped until the undone requirement is fixed.

“Must” Requirements (i.e. first priority)

* The system must not have a feature created for it without first having a test written for that feature.
* The system must have tests that ensure the overall functionality of the system after each change made to the system.
* The system must have tests for each function and feature of the system, with an explanatory comment for each test.
* The system must return an integer that, for the purposes of games played with dice, is random, within a given range, such as 1 through 20, when prompted by the user through the system’s interface.
* The system must not crash if an unsupported number is chosen by the user.
* The system must not store any data other than saved profiles (which would consist of a file containing the name of the function and the formula for the function, a function being the range of numbers along with the variables input by the user).
* The system must have access to create, edit, and store files related to its function, i.e. the files that store saved profiles.
* The system must be able to be installed onto a Galaxy A20 phone
* The system must not be accessible without first unlocking the phone.

“Shall” Requirements (i.e. second priority)

* The system shall be able to be installed onto all Android phone models released since 2015.
* The system shall not interfere with the performance of other systems in its environment.
* The system shall not leave artifacts after uninstallation.
* The main functions of the system (selecting a profile, entering the menu for generating integers) shall be visible and available from the main menu shown when the system is launched.

“Should” Requirements (i.e. third priority)

* The system should have tests that check the performance of the system, to see if it meets the criteria listed under time behavior and resource utilization within this document.
* The system should be able to be installed onto all Android phone models and I-phone models released since 2015.
* The classes used by the system should be able to be used in other environments with no reliance on other classes from this system.
* The system should be available through the app store.
* The system should not crash upon interaction with any part of the screen.
* The system should update the “runtime” file (i.e. the file edited during the time the system is running) containing its profile after each edit made by the user.
* The system should be able to save profiles to different files stored in the environment, reducing the impact of a system or environment failure.
* The system should not be editable by users.
* The system should be authenticated through the app store.
* The system should be able to store the hexadecimal code after returning it.
* The system should not have functions beyond returning generated integers as well as creating, storing, and editing saved profiles.
* The system should adhere to the coding conventions listed here: <https://kotlinlang.org/docs/reference/coding-conventions.html>.
* The system should adhere to the standards required by the Android app store.
* The system should adhere to the standards required by the Apple app store.
* The system should respond to user input within 0.1 seconds in all situations.
* The system’s launch time (i.e. the time it takes to start running) should be under 1 second.
* The system should not require more than 5 megabytes of storage from its environment.
* The system should not require more than 1 gigabyte of ram from its environment.
* The system should have a 5 page user guide available within the first menu of the system after it is launched, with each page displaying a function of the system.
* The system should have a depth of 3 commands. Depth refers to how many choices the user can make going from the main menu before reaching a screen with no choices other than to return.
* The menus of the system should be composed of rectangles with rounded edges.
* The system’s color scheme should be composed of light pastel colors.
* The system’s font should be sans serif and not stylized (i.e. not cursive or textured).
* The system’s interface should have buttons that each use about 1/4th of the screen space.
* The tests for the system should be documented using json files.
* The system should be created using object oriented programming, i.e., there should be low to no level of coupling between the application’s functions.