Gesaven Pillay

ST10459254

IMAD Assignment 1 Report

**Purpose of Application**

This app is a way to show a little history of our world in a fun and interactive way, by correlating the user’s age to the age of a historical figure’s age when they passed!

**General function of Application**

A user enters their age, that age is then validated to see first that it a whole number, if not it will display a message saying that user must enter only whole number, then it checks whether the ages is too young, below 20, or too old, above 100, if it is without this range then a message will be displayed telling the user that it is out of range and the number has to be from 20 to 100.

If the age is validated then it goes through 10 historical figures, all South African historical figures. I chose to keep it within South Africa as we are South Africans and it’s a way to show a little of South African history.

If the age entered by the user is the same as anybody on the South African historical figures list, then a message pops up saying how the user is the same age as that figure and it also gives the user a bit of background on that user, for example if you are 95 years old then it would display, “You are 95 years old! You share the age of Nelson Mandela, the first president of post-Apartheid South Africa!”, I added little historical information to make the app more fun to the user, and they learn a little of our South African heroes.

Everything is displayed in a textview, all error messages and historical figure information is displayed in bold and in big enough text to attract the reader once they press the button.

Once the user is done, or they wish to try another age, they may click the “Clear” button which clears the edit text and the text view so that the user may type again.

**Graphical Design**

The background I made light blue to contrast and blend with the purple, peach and brown of the buttons, textbox and images. The contrast and blend make it more appealing to the user.

The images must be appealing to the eye as well as relating to what the app entails, so I chose a picture that’s eye catching, with the word “History” on it to show that the app relates to history.

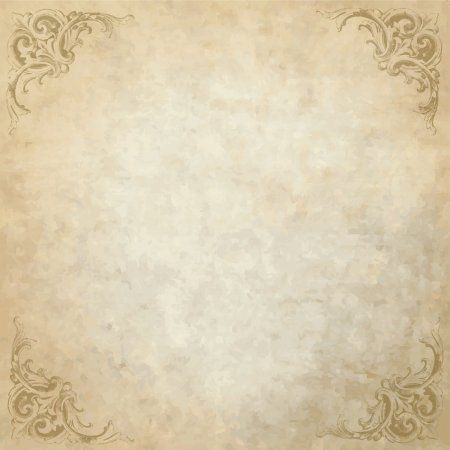


**Figure 1. Charles Marchand, 2022, *What is History? Truth vs. Perception* (Marchand, 2022)**

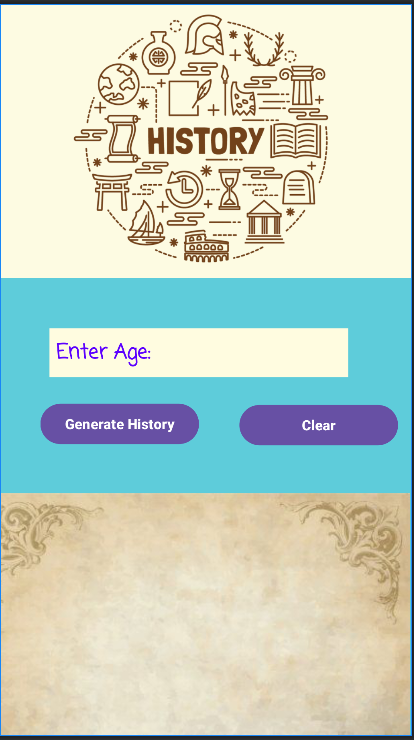
The edit box has fun looking text that tells the user to enter age, it is also displayed as a hint so that it is easier for the user to enter their age without needing to delete the “Enter age:” text. It is in purple to match with the peach, brown and light blue throughout the application. It is in clear, easily readable text so that the user knows exactly what to enter.

The buttons I kept in default purple as it matches well with the whole application and contrasts to the pictures making it pop out to the user. It also has easily readable and understandable text so that the user can easily understand what the button does.

The second image is a blank peach parchment with small engravings which relates to old parchment used in history. The peach also matches with the whole application. The parchment is blank for the text view to be on, so it is as if the results are written on parchment like in the past! The words are in bold and are big to attract the user’s attention when the output is displayed.



**Figure 2. Anon, n.d. *realistic old paper texture with blank space***

**

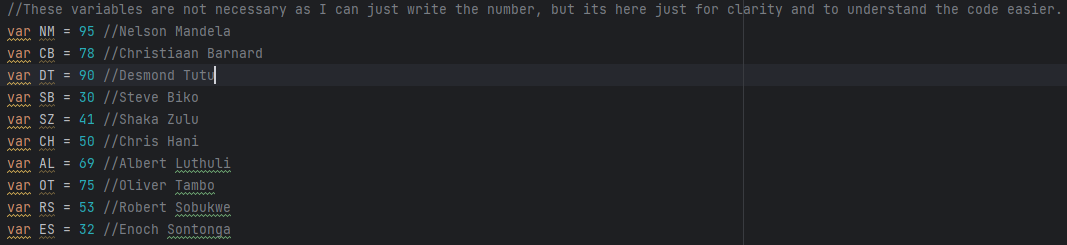
**Figure 3. Screenshot from Android Studio of user interface.**

**Coding Design**

The user must be able to enjoy the interface, we covered how it is visually appealing, but now let us look at the actual feel of the application. The user should be able to go through the application fluently without any glitches or annoying interface issues.

Which my application accomplishes. It is fluent in how it works. The user can clearly see what they need to do: Enter the age where it says to do so, then click on the “Generate History” button and then the results are displayed in a textview on the parchment, it is written in bold and pops out to the user.

It checks through 10 historical figures, anybody with the same age will immediately be shown in the textview. The clear button is available to clear both the textview and the age, resetting the application to the beginning.



**Figure 4. Screenshot displaying the list of the historical figures assigned to variables.**

What about user error? We must ensure that if the user makes an error, it should not break the program and that an error message should be displayed to correct the user.

I have checked that any type of data will not break the program, also if the user enters a non-number or a decimal, the program puts a message in the textview saying that the data can only be whole numbers. If the user enters a number out of range, then the textview displays that the numbers are not in range.

**Testing in Android Studio**

Through testing there is no glitches or problems with any data entered. Also, it is clear to the user how the application works which is good because when an interface becomes too complex it can make the user dislike the application or even uninstall it.

**Utilisation of GitHub and GitHub Actions**

It was easy to set up a repository and the fact that you can easily push your code, to make any changes even if its already uploaded is an amazing feature for anybody having questions: They can upload the code for everybody to see and comment on, either to improve, to learn and/or get help from other coders. GitHub Actions is also useful to test that other people can run your code without any errors, which is extremely helpful for both fun and business use.

**Logging of Code and Whole Assignment**

4 April 2024

* Made user interface, added images and picked right colours that make the app appealing.
* Started coding.
* Line 15-24 Found 10 South African historical figures and assigned them a variable for easier understanding of the code not because it was necessary.
* Line 26-29 Tried directly using the EditText box and buttons which did not work, added values and assigned them to used components through the findViewById.
* Line 35-36 Took value from editText but it is a string which cannot be used to check if it is in a range from 20 and 100 so I had to change any data entered to an integer for it to be used.
* Line 38-70 caused an error, I forgot to put it under the button onclick event so I added it.
* Line 38-70 there was still issues as it was not displaying the correct output even if a number was entered.

5 April 2024

* Line 38-39 realized that I put a wrong if statement (logic was wrong incorrect which led to all entries being seen as null), removed that part and the code worked.
* Line 38-39 issue with validation, separated the if statement to two parts, first checking if it the data entered is a number then if it is, checking if it was in range.
* Validation was correct, and everything works.
* Line 72-75 Clear button created wrote the code and it worked perfectly.
* Testing of whole application, user interface worked perfectly no issues, buttons worked, validation worked for any type of data entered, and output was completely right every time.
* Code completed.

**References**

Line 36 and line 38 I used toIntOrNull and ageInt !=null

Kotlin. (n.d.). toIntOrNull – Kotlin Programming Language. [online] Available at: <https://kotlinlang.org/api/latest/jvm/stdlib/kotlin.text/to-int-or-null.html>.

For the list of south African people:

Wikipedia. (2021). *Great South Africans*. [online] Available at: <https://en.wikipedia.org/wiki/Great_South_Africans>.

**Video presentation link:**

<https://youtu.be/v6Da3COxmyw>

**GitHub link:**

<https://github.com/MrSmiley777/IMAD-Assingment-1-ST10459254.git>