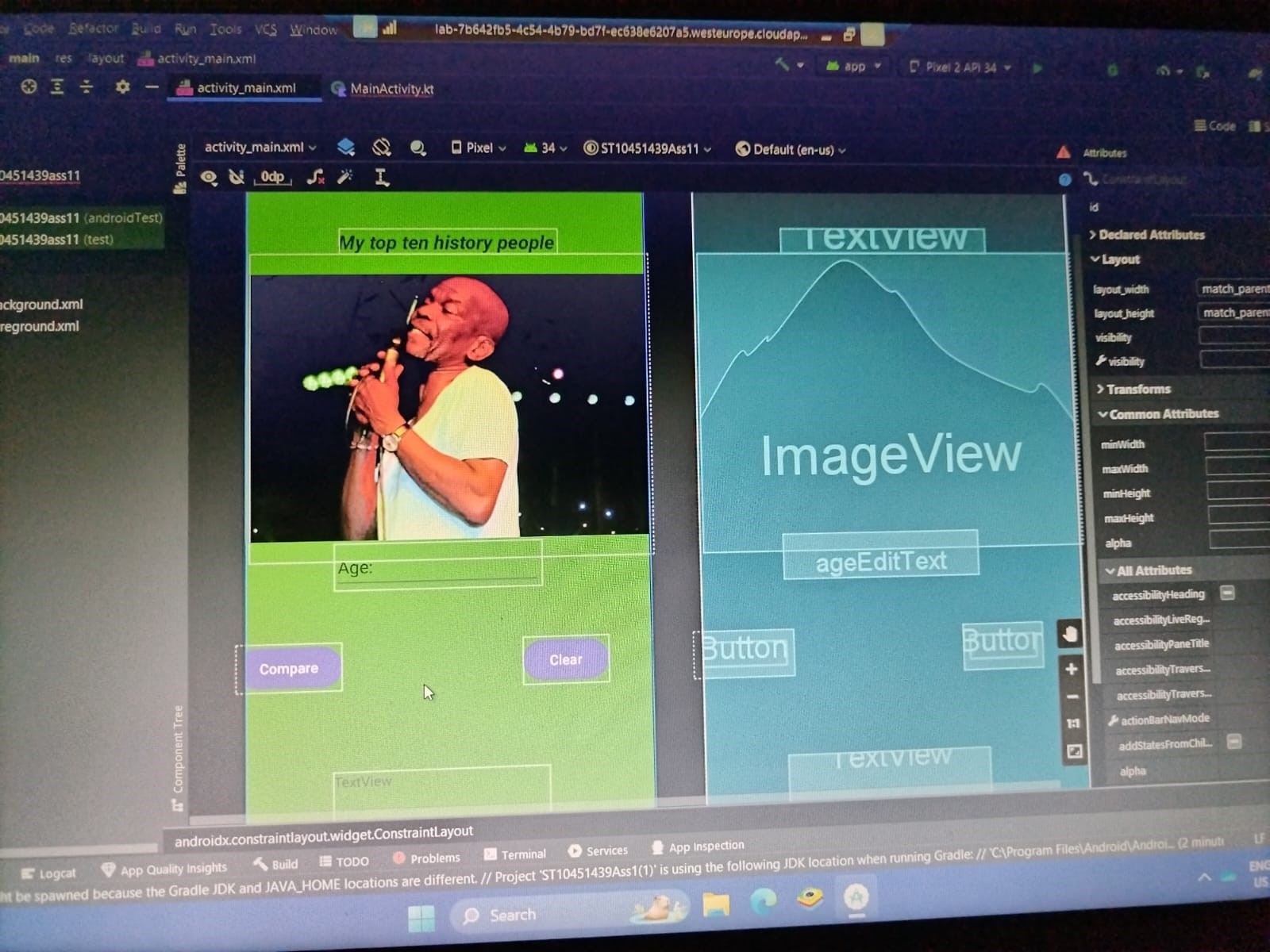
ST10451439                                   Charles Mamabolo

**1.Purpose**

My application will be based on my top ten historical figures. My app will share knowledge on the historical figures in football, music, and science. My app will tell when my historical figures died, and it will also explain what my historical figures were Famous for. My app will compare your age that you insect with my historical figures age and display what they were famous for.

**2.Design Considerations**



**Intuitive design**

My design is easy to follow because it starts forward and self- explanatory. Everything is clear for the user to understand in my design from top to bottom.

**Content prioritisation**

Yes, I have included the necessary interface elements such, all my val and var, my identification for all my var and val. I would not say I display essential content and functionalities because my design is more like and educational experience where you get to know about old historical figures that were famous back in the day.

**Legible Text Content**

Yes, I did use fonts that are easy to read, they all above size 16 and I believe and think that okay for someone who even wears glasses. The font text that I used is Abadi cause its clear and does not make the words hard to read.

**Make Interface elements clearly visible**

Yes, the user can see all components because they are all visible and clear and they also stand out. Everything in my design is spaced up well its not to crowded because I did not use many pictures and it is also not to spacious where you must fully extend your fingers to type.

**Hand Position Controls**

Yes, both my buttons are positioned in a horizontal form and they and next to each other. Where you will use one button to compare your age that was typed in to the information or data in the application. Whereby the other button you will use to clear text so you can insert text to compare a different person with a different age.

**Images**

In my design I used one picture. This picture is a cover of my report. Ray Phiri is the man in the picture because this man inspires me and just maybe he might inspire you to if you go look him up.

**3. Git Hub actions and steps**

1. Create a New GitHub Repository:

Go to the GitHub website (<https://github.com/>) and sign in to your account.

Click on the "+" icon in the top right corner and select "New repository".

Give your repository a name (your student number and name – in one word), add a description (use IMAD5112 Assignment 1) and choose public.

Click on the "Create repository" button.

2. Initialise the Repository with a README File:

After creating the repository, you'll see an option to "Initialize this repository with a README". Check this option to create a README file.

Click on the "Create repository" button to finalize the creation of the repository.

3. Commit and Push Your Project Files to the GitHub Repository:

In Android Studio, go to VCS (Version Control System) -> Import into Version Control -> Share Project on GitHub.

Log in to your GitHub account if prompted, and select the repository you created earlier.

Click on the "Share" button to push your project files to the GitHub repository.

4. Regularly Commit and Push Your Code as You Make Progress:

After the initial push, continue making changes to your project in Android Studio.

Whenever you make significant progress or changes, commit your changes locally in Android Studio using VCS -> Commit Changes.

Once committed, push your changes to the GitHub repository using VCS -> Git -> Push.

**Testing and Automated Testing**:

1. Conduct Manual Testing:

Manually test your app to ensure it functions seamlessly and offers an enjoyable educational experience for learners.

To test various features and user interactions do the following:

1. Create a New Test Class:

In your Android project, navigate to the tests directory (or create it if it doesn't exist).

Create a new Kotlin file for your test class. Name it appropriately to indicate what component or functionality you are testing.

2. Write Test Methods:

Inside the test class, write test methods like the example below.

3. Use assertions to verify the expected behaviour of your code.

4. Run the Tests:

Run the tests using the testing framework's tools provided by Android Studio or through the command line.

Sample of a test class using JUnit:

import org.junit.Assert.\*

import org.junit.Test

class MyUnitTest {

    @Test

    fun testWhenStatement() {

        // Test case for a when statement

        val result = when (25) {

            21 -> "Some Body, famous for …, died at this age"

            34 -> "Another Some Body, famous for ……., died at this age"

            63 -> "Someone Else, famous for …..., died at this age"

            else -> "Nobody famous known to me died at this age"

        }

        assertEquals("Nobody famous known to me died at this age", result)

      }

    }

2. GitHub Actions for Automated Testing:

Set up GitHub Actions to automatically run tests and build your code whenever changes are pushed to the repository.

Create a GitHub Actions workflow (.github/workflows/tests.yml) to run tests automatically on every push:

Create a .github/workflows directory in your project repository.

Inside this directory, create YAML files defining your GitHub Actions workflows for testing and building.

**Sample GitHub Actions Workflow for Testing (tests.yml):**

name: Run Tests

on: [push]

jobs:

  test:

    runs-on: ubuntu-latest

    steps:

      - name: Set up JDK

        uses: actions/setup-java@v2

        with:

          distribution: 'adopt'

          java-version: '11'

      - name: Check out code

        uses: actions/checkout@v2

      - name: Build and test

        run: ./gradlew test

Set Up Automated Build:

Create a GitHub Actions workflow (.github/workflows/build.yml) to build the APK automatically on every push.

**Sample GitHub Actions Workflow for Building (build.yml):**

name: Build APK

on: [push]

jobs:

  build:

    runs-on: ubuntu-latest

    steps:

      - name: Set up JDK

        uses: actions/setup-java@v2

        with:

          distribution: 'adopt'

          java-version: '11'

      - name: Check out code

        uses: actions/checkout@v2

      - name: Build APK

        run: ./gradlew assembleDebug

Test your workflows by pushing changes to your repository and observing the actions running in the "Actions" tab on GitHub.

[https://github.com/CharlesSiya/ST10451439CharlesMamabolo](https://github.com/CharlesSiya/ST10451439CharlesMamabolo )

**4. YouTube Video**

<https://youtu.be/uSBiV5F8lFE>

**5. Reference List**

Alpion, G., 2006. *Mother Teresa: saint or celebrity?*. Routledge.

Asante, M.K., 2021. The Remarkable Curvature of the Mind of Abdias do Nascimento. *Journal of Black Studies*, *52*(6), pp.577-587.

Fast, S., 2010. Difference that Exceeded Understanding: Remembering Michael Jackson (1958–2009).

Howard, D.A., 2005. Albert Einstein as a philosopher of science. *Physics today*, *58*(12), pp.34-40.

Iwamoto, D., 2003. Tupac Shakur: Understanding the identity formation of hyper-masculinity of a popular hip-hop artist. *The Black Scholar*, *33*(2), pp.44-49.

Mandela, N., 1990. *Nelson Mandela: the struggle is my life: his speeches and writings brought together with historical documents and accounts of Mandela in prison by fellow-prisoners*. Popular Prakashan.

Markovich, S.G., 2022. Legacy of Queen Elizabeth II: The symbolism of the British Monarchy and the challenges it faces. *Politički život*, (23), pp.7-17.

Reese, R., 2021. The Life and Legacy of Kobe Bryant: Reflections from Cal Poly Pomona. *Journal of African American Studies*, *25*, pp.339-352.

Ruse, M., 2007. Charles Darwin. In *Philosophy of Biology* (pp. 1-35). North-Holland.

Shakespeare, W., 1989. *William Shakespeare: the complete works*. Barnes & Noble Publishing.