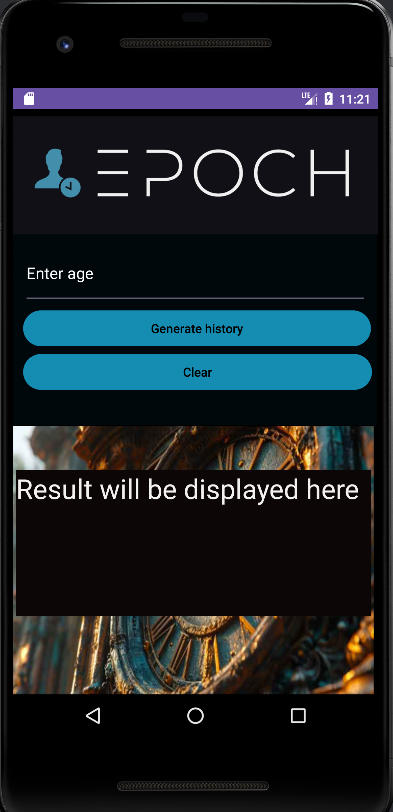
Introduction  
  
**Epoch** is a mobile application specifically designed for Android devices. The main task of this program is to take a person's age as input and match it with the ages of famous people who have died in history. The app provides users with information about famous people who died at the same age as the channel. The app also does take account of various considerations that may allow the users(student), to either generate the age of the famous person who has passed on. A range is set to create a restriction for outlying ages out of the range of (20-100). Users who enter or input invalid data that outlines these restrictions will then receive a prompt that will display an error message.

My report below looks at the purpose of the application, its design considerations, and the use of GitHub and GitHub Actions for version control and continuous integration/continuous deployment (CI/CD).  
  
**Purpose of the App**  
  
The Epoch App serves several purposes:  
  
**Educational Tool:** It provides an engaging way for students to learn about historical figures**,** in this case famous people who have passed on.  
  
**Fun: Epoch** provides a fun and exploratory experience. Users can discover interesting facts about famous people and compare their own age with those of historical icons. This feature encourages users to delve into their imagination and envision what life might have been like during various historical periods.

**Reference:** The app (Epoch) provides quick information about the ages of famous people, so it is useful for trivia games and history research.

Additionally, Epoch serves as a quick reference tool for trivia games and history research. Users can easily access information about the ages at which famous individuals passed away, making it a valuable resource for history enthusiasts and quiz aficionados alike.

**Design Considerations**

  
  
  
User Interface (UI)  
  
The application's user interface is designed to be user-friendly and visually appealing to the user, and includes the following components:

**EditText :** Users can enter their desired age.

**Generate History Button:** Runs a matching algorithm to find a famous person

who died at the same age.

**Clear Button:** Resets the input field and the result.  
  
**TextView for Result:** Shows the name of the correspondingperson and the age of the famous person at the time of death.  
  
The user interface uses a dark theme with custom background colors and font styles to improve the readability andaesthetics of the program, making it user-friendly. The use of contrasting colors ensures that the text is easy to readon the background.  
  
**Age Verification**  
  
To ensure data integrity and accuracy, the application validates age input before processing it. It checks if the age entered is within a reasonable range of historical figures in my application, 20-100 years old. If the input is outside this range, Epoch prompts the user to enter a valid age that falls within the range.  
  
**Matching algorithm**  
  
The main function of the application is a matching algorithm that compares the age entered by the user with a predefined list of famous people and their ge of death. The algorithm then takes the name a of a historical person who died at the same age as the input.  
  
The list of famous people and their ages are stored in anapplication data structure such as a map or database. This data structure allows for efficient data serving and retrieval during the reconciliation process.  
  
**Error handling**  
  
The application includes efficient error handling to ensure a smooth user experience. It displays informative error messages when:

**Please enter a valid age between 20-100.**

**The age input is invalid (e.g. outside the valid range).**

**No famous person found for the entered age**.  
  
These error messages guide the user to correct inputs or understand the limitations of the application's functionality.  
  
**GitHub integration and GitHub features**  
  
Version control with GitHub  
  
**Epoch** is hosted on GitHub, a version control and collaboration platform for developers. GitHub offers several key benefits:  
  
Version history: Developers can track changes to the code base over time, including who made the changes and when.  
  
**Forking** and **merging:** GitHub supports branching and merging workflows, allowing developers to work on features or fixes separately before integrating them into the main codebase.  
  
**Collaboration:** Multiple developers can work on the same project at the same time, making iteasier to coordinate activities and review code changes.  
  
**GitHub Actions for CI/CD**  
  
GitHub Actions are used for Continuous Integration (CI) and Continuous Deployment (CD) o**f** the Age Matcher project. The CI/CD pipeline automates the following processes:  
  
**Build process:** When changes are pushed to the repository, GitHub Actions initiates a build process to compile the code, resolve dependencies, and create an **Android package (APK)** file for the application.   
  
**Automated testing:** As part of the CI process, automated tests are performed to ensure that the application functions correctly and meets quality standards. These tests include unit tests, integration tests, and user interface tests.  
  
**Code quality checks:** GitHub Actions performs code quality checks, such as tape and static code analysis, to identify potential problems and meet code standards.  
  
**Deployment:** After successful testing and code validation, GitHub Actions automates the deployment of the application to a given environment, such as a test or production server.  
  
The CI/CD pipeline is designed to streamline development workflow, improve code reliability, and facilitate rapid iteration and deployment of new application versions.  
  
Conclusion

Epoch stands as a versatile and educational mobile application that seamlessly blends technology with historical knowledge. Its user-friendly design, robust functionalities, and integration with GitHub and CI/CD pipelines exemplify modern app development practices.The Epoch app represents a fusion of education, entertainment, and technology, providing users with a unique and engaging way to learn about historical figures and events. By leveraging a user-friendly interface, a robust matching algorithm

By adhering to best practices in UI/UX design, data validation, error handling, version control, and automated testing, Epoch delivers a compelling user experience while maintaining code integrity and reliability. As an educational tool, Epoch bridges the gap between learning and exploration, making historical insights accessible and engaging for users of all ages and backgrounds.

**REFERECING**

//Code Attribution//

//This method was taken from stackoverflow

// <https://stackoverflow.com/questions/59370915/how-to-call-private-functions-of-a-class-from-outside-the-class-in-kotlin>

//Sergio

// <https://stackoverflow.com/users/1731626/sergio?tab=profile>



//Code Attribution//

//This method was taken from stackoverflow

// <https://stackoverflow.com/questions/22654932/what-is-the-purpose-of-unit-returning-in-functions>

//Dmitry Romanov

// https://stackoverflow.com/users/548894/dmitry-romanov



//Code Attributions

// This method was taken from stackoverflow

// stackoverflow.com/questions/73891861/how-can-via-kotlin-mapof-map-buttons-key-to-value

//Slava

// https://stackoverflow.com/users/2046451/slava



<https://youtu.be/miODyMpQAQ4> (Youtube video:Return on Functions)

<https://youtu.be/sOpS0mMN-sg> (Making a Toast)

<https://youtu.be/JXe6CFrYByU>( (Visibility Modifiers)

[](https://youtu.be/JXe6CFrYByU)

<https://youtu.be/a2ueQgedZpU> (Functions for Data Collections)

(APP LOGO DESIGN)

<https://app.brandmark.io/v3/#logo_data=%7B%22id%22%3A%22logo-7b6e4002-39a4-4a15-916c-ddd47508d69d%22%2C%22page%22%3A%220%22%2C%22name%22%3A%22logo-basic%22%2C%22template%22%3A%22Logo%22%2C%22layout%22%3A0%2C%22businessName%22%3A%22End%20Epoch%22%2C%22description%22%3A%22End%20Epoch%22%2C%22title%22%3A%22ENDEPOCH%22%2C%22titleFamily%22%3A%22Brandmark1%20Bold%20Color%22%2C%22titleVariant%22%3A%22700%22%2C%22titleColor%22%3A%5B%7B%22hex%22%3A%22%23c2bfba%22%2C%22location%22%3A0%7D%2C%7B%22hex%22%3A%22%23b7b3ad%22%2C%22location%22%3A1%7D%5D%2C%22titleScale%22%3A1.75%2C%22titleLetterSpace%22%3A0%2C%22titleLineSpace%22%3A1.25%2C%22titleBoldness%22%3A0%2C%22titleX%22%3A0%2C%22titleY%22%3A0%2C%22titleAlign%22%3A%22left%22%2C%22slogan%22%3A%22%22%2C%22sloganFamily%22%3A%22Brandmark1%20Bold%20Color%22%2C%22sloganVariant%22%3A%22700%22%2C%22sloganColor%22%3A%5B%7B%22hex%22%3A%22%23c2bfba%22%7D%5D%2C%22sloganScale%22%3A1.75%2C%22sloganLetterSpace%22%3A6%2C%22sloganLineSpace%22%3A1.25%2C%22sloganBoldness%22%3A0%2C%22sloganAlign%22%3A%22left%22%2C%22sloganX%22%3A0%2C%22sloganY%22%3A0%2C%22icon%22%3Anull%2C%22showIcon%22%3Afalse%2C%22iconScale%22%3A1%2C%22iconColor%22%3A%5B%7B%22hex%22%3A%22%23b7b3ad%22%7D%5D%2C%22iconContainer%22%3Anull%2C%22showIconContainer%22%3Afalse%2C%22iconContainerScale%22%3A1%2C%22iconContainerColor%22%3A%5B%7B%22hex%22%3A%22%233d3b39%22%7D%5D%2C%22iconSpace%22%3A1%2C%22iconX%22%3A0%2C%22iconY%22%3A0%2C%22nthChar%22%3A0%2C%22container%22%3Anull%2C%22showContainer%22%3Afalse%2C%22containerScale%22%3A1%2C%22containerColor%22%3A%5B%7B%22hex%22%3A%22%233d3b39%22%7D%5D%2C%22containerX%22%3A0%2C%22containerY%22%3A0%2C%22backgroundColor%22%3A%5B%7B%22hex%22%3A%22%23000000%22%7D%5D%2C%22palette%22%3A%5B%22%23000000%22%2C%22%23c2bfba%22%2C%22%23bebbb5%22%2C%22%23bab7b1%22%2C%22%23b7b3ad%22%5D%2C%22hasDomain%22%3Atrue%2C%22domains%22%3A%22com%22%2C%22autoScale%22%3Afalse%2C%22keywords%22%3A%22EPOCH%22%7D>

Kotlin(2024)demontrates how.

Kotlin

2024

Kotlin[v1.9.23](https://github.com/JetBrains/kotlin/releases/tag/v1.9.23)

https://kotlinlang.org/api/latest/jvm/stdlib/kotlin.collections/map-of.html