

9/26/2024

# User Documentation

COUNTIFY



OPSC7312 POE  
VARSITY COLLEGE

# Countify

**Countify** is a simple Android counter app developed using Kotlin in Android Studio. The app allows users to create and manage custom counters, either by registering an account with their email or by using Google SSO (Single Sign-On). Counters are persisted using Firestore Database, and the app integrates various user preferences, including notifications, theme, and language settings. The app also displays random advice fetched from a third-party API.

## Table of Contents

Group Members: .....	1
Purpose .....	2
Features .....	2
Design Considerations .....	4
Setup and Installation .....	7
Usage .....	8
GitHub .....	9
Further Details .....	12

## Group Members:

Ethan Schoonbee (**PM**) – **ST10036509**

Leonard Bester – **ST10026396**

Ross Harper – **ST10048758**

Cameron Brooks – **ST10032014**

# Purpose

Countify was built to provide users with a simple and intuitive way to create, edit, and track multiple counters. It enables users to increment values easily for a variety of personal or professional use cases. Additionally, the app offers settings customization for a personalized user experience, as well as integration with external APIs to enhance functionality.

## Features

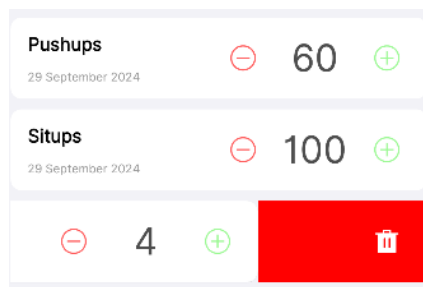
### User Authentication:

- Register using email and password or with Google SSO.

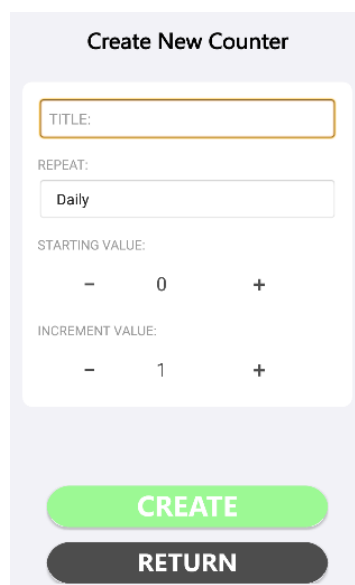


### Counters:

- View a list of created counters.



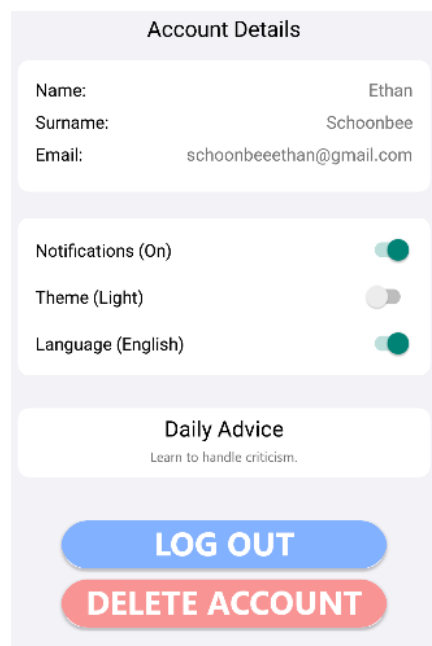
- Add new counters with custom titles, descriptions, and increment values.

A form titled 'Create New Counter'. It contains a 'TITLE:' input field, a 'REPEAT:' dropdown menu set to 'Daily', a 'STARTING VALUE:' section with a minus sign, the number '0', and a plus sign, and an 'INCREMENT VALUE:' section with a minus sign, the number '1', and a plus sign. At the bottom are two buttons: a green 'CREATE' button and a dark grey 'RETURN' button.

- Data is stored and retrieved from Firestore Database.

### Settings Page:

- Manage account details (name, email, surname).
- Toggle settings for:
  - Notifications (On/Off).
  - Theme (Dark/Light).
  - Language (English/Afrikaans).
- Random Advice API integration: Displays a new random piece of advice every time the settings page is loaded.



The screenshot displays a mobile application settings page with a light purple background. It is organized into three main sections. The top section, titled 'Account Details', contains a white card with three rows of text: 'Name: Ethan', 'Surname: Schoonbee', and 'Email: schoonbeeethan@gmail.com'. The middle section contains a white card with three toggle switches: 'Notifications (On)' which is turned on (green), 'Theme (Light)' which is turned off (grey), and 'Language (English)' which is turned on (green). The bottom section features a white card titled 'Daily Advice' with the text 'Learn to handle criticism.' Below this card are two large, rounded rectangular buttons: a blue one labeled 'LOG OUT' and a red one labeled 'DELETE ACCOUNT'.

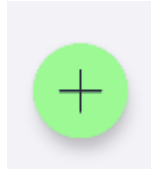
### Persistent Data:

- All user preferences (settings, counters) are saved to the Firestore Database and reloaded when the app starts.

# Design Considerations

## User Interface:

The user interface is designed to be minimalistic and intuitive, with a Floating Action Button (FAB) for creating new counters and easy navigation to the settings page.



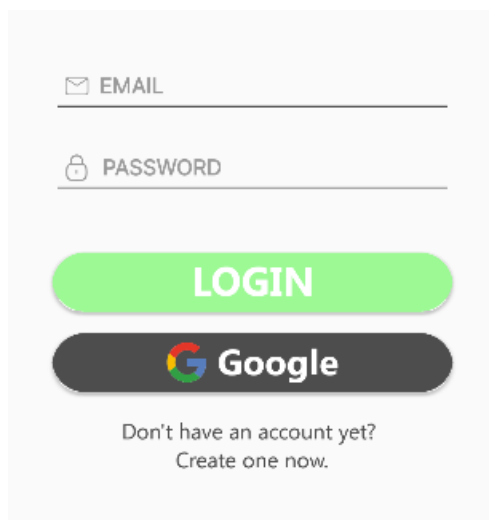
FAB



Settings Button

## User Authentication:


We utilized Firebase Authentication for secure login, offering both email registration and Single Sign-On (SSO) via Google for convenience and Biometrics for secure logging back in (Android). User details are stored in Firestore for easy retrieval and display in the settings.



EMAIL

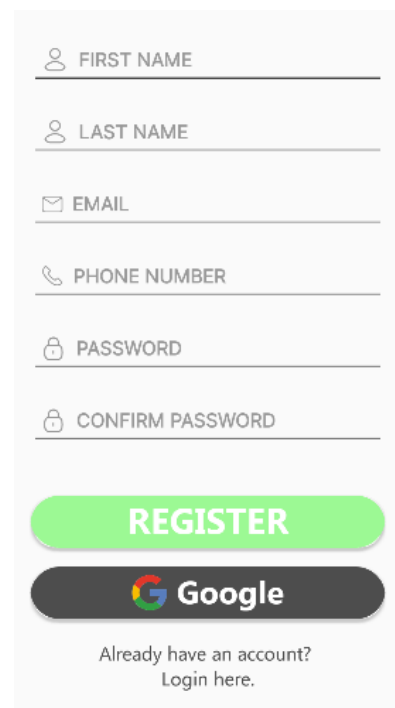
PASSWORD

**LOGIN**

 **Google**

Don't have an account yet?  
Create one now.

Email and Password Login



FIRST NAME

LAST NAME


EMAIL

PHONE NUMBER

PASSWORD

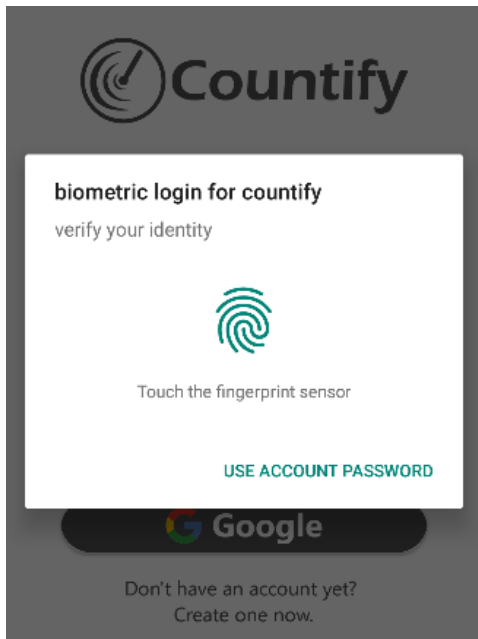
CONFIRM PASSWORD

**REGISTER**

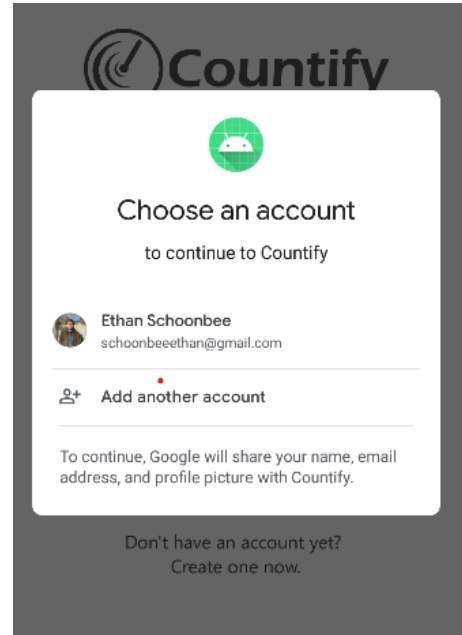
 **Google**

Already have an account?  
Login here.

Email and Password Login



Biometric Login



Google SSO Register and Login

## Firestore Database:

All counters and user preferences are stored in Firestore for scalability, real-time updates, and secure storage.

Search by email address, phone number, or user UID					Add user		
Identifier	Providers	Created ↓	Signed In	User UID			
mygoogleemail@gmail....	📧	Sep 29, 2024	Sep 29, 2024	shtFBJkb5bYUMWjPloyAb5sf...			
tester1@gmail.com	📧	Sep 29, 2024	Sep 29, 2024	Ccpf2egq1KhSjk9ZOWtJYrTnj...			
test4@gmail.com	📧	Sep 27, 2024	Sep 27, 2024	WrDwi2MyXNbkbV8iJYd5p1T...			
schoonbeeethan@gmai...	📧	Sep 25, 2024	Sep 27, 2024	6CCyy24x5kY7D9CtCvnOHh4...			
test2@gmail.com	📧	Sep 22, 2024	Sep 28, 2024	ej4fXRTydoXcEiufG0NgGYzJX...			
joesoap@gmail.com	📧	Sep 21, 2024	Sep 21, 2024	J1cgDiWNxdQ1BcoBiEFUw9v...			
test@gmail.com	📧	Sep 20, 2024	Sep 22, 2024	ITrSxiBYJHah5VOFKGv2P0nyT...			

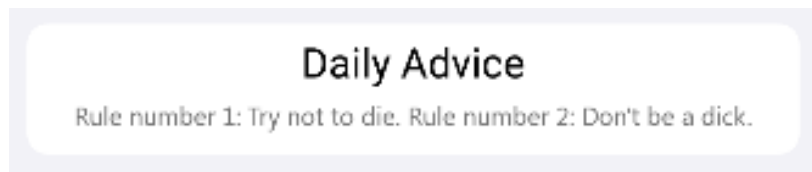
Firebase Authentication

(default)	counters	1xJYdD5LDtLojcp5Phxs
+ Start collection	+ Add document	+ Start collection
counters >	1xJYdD5LDtLojcp5Phxs >	+ Add field
counters_tests	8pK9JhQWDHCjaUSWKGIO	count: 0
users	Ms8cdIKqC5RlEmdYlpII	createdTimestamp: 1727554397302
	VrYb4PBteflcbC0garCY	incrementValue: 1
	x3iVj2BYfhSV2Nf6PCZd	name: "oui oui Mon ami"
		repetition: "Daily"
		userId: "ej4fXRTydoXcEiufG0NgGYzJXMn1"

## Firestore NoSQL Database

### Random Advice API Integration:

The settings page connects to the external Random Advice API, fetching a fresh piece of advice on every load. This adds a fun, interactive element to the app.



# Setup and Installation

## Prerequisites

- Android Studio (latest version) (Andorid, 2024)
- Kotlin configured in the Android project (JetBrains, 2024)

## Steps to Compile and Run

### 1. Clone the Repository:

- `git clone https://github.com/ST10036509/Countify.git`
- `cd countify`

### 2. Build the Project:

- Open the project in Android Studio.
- Sync Gradle files by clicking on the "Sync Now" prompt.

### 3. Run the Project:

- Click on the "Run" button in Android Studio or press Shift + F10.



# Usage

## **Register/Login:**

- Open the app and sign up using an email address or log in using Google SSO. (Google, 2024)

## **Manage Counters:**

- Once logged in, you'll see a list of counters. To add a new counter, press the FAB and enter the title, description, and increment value. The counter will be saved to Firestore. Counters can be deleted by swiping left on the counter you wish to remove.

## **Settings:**

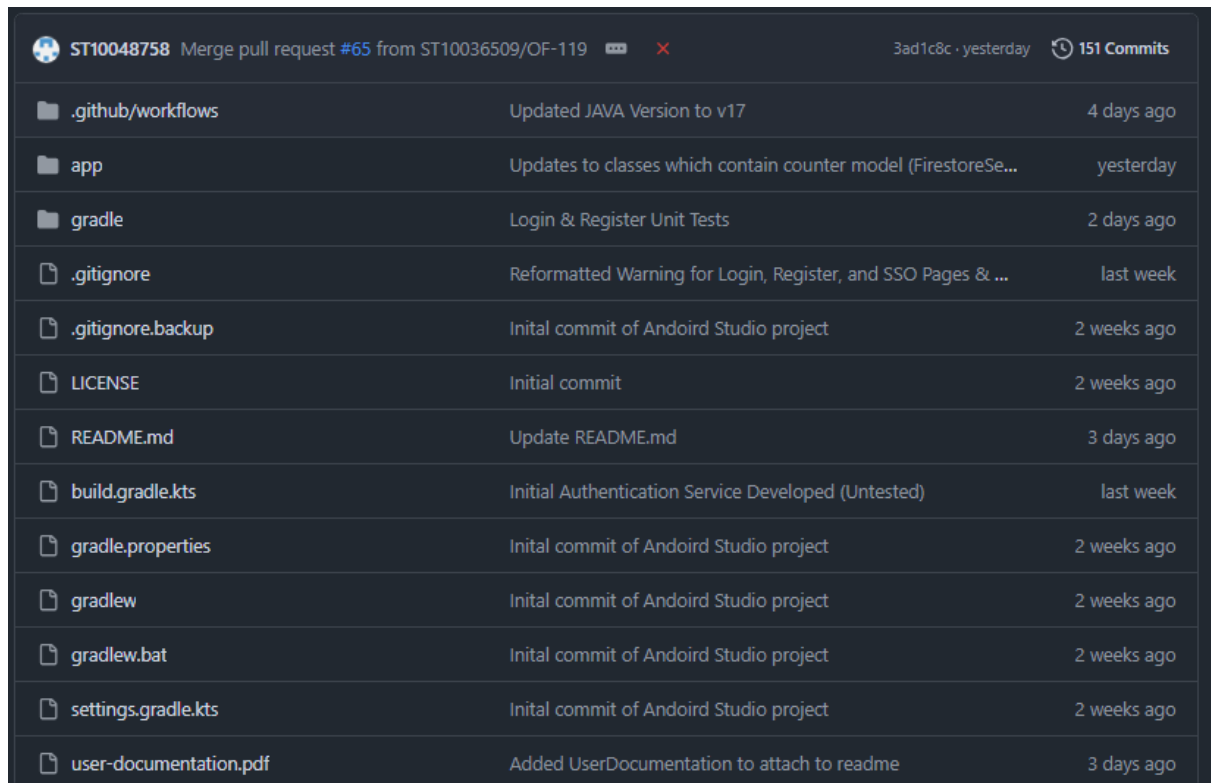
- Navigate to the settings page to manage preferences. You can toggle notifications, change the theme, and select the language. These preferences are saved in the Firestore and applied on app restart.
- You can also view your account details (name, surname, and email) on this page.
- A random piece of advice will be fetched from the Random Advice API each time the settings page is opened.

# GitHub

## GitHub Usage:

We used GitHub for source control, allowing our team to push and pull code updates seamlessly. Branching was used to isolate features and bug fixes, ensuring that the main codebase remained stable. (GitHub , 2024)

- **Branches:** Each feature was developed on its own branch and merged into the main branch via pull requests.
- **Commits:** We maintained meaningful commit messages, documenting changes clearly.
- **Pull Requests:** Code reviews were done on pull requests to ensure high code quality and collaboration.



The screenshot shows a GitHub commit history table for repository ST10048758. The table lists 15 commits, including folders like .github/workflows, app, and gradle, and files like .gitignore, LICENSE, README.md, build.gradle.kts, gradle.properties, gradlew, gradlew.bat, settings.gradle.kts, and user-documentation.pdf. Each row shows the file name, the commit message, and the time since the commit was made.

File	Commit Message	Time
.github/workflows	Updated JAVA Version to v17	4 days ago
app	Updates to classes which contain counter model (FirestoreSe...	yesterday
gradle	Login & Register Unit Tests	2 days ago
.gitignore	Reformatted Warning for Login, Register, and SSO Pages & ...	last week
.gitignore.backup	Initial commit of Andoird Studio project	2 weeks ago
LICENSE	Initial commit	2 weeks ago
README.md	Update README.md	3 days ago
build.gradle.kts	Initial Authentication Service Developed (Untested)	last week
gradle.properties	Initial commit of Andoird Studio project	2 weeks ago
gradlew	Initial commit of Andoird Studio project	2 weeks ago
gradlew.bat	Initial commit of Andoird Studio project	2 weeks ago
settings.gradle.kts	Initial commit of Andoird Studio project	2 weeks ago
user-documentation.pdf	Added UserDocumentation to attach to readme	3 days ago

## GitHub Actions:

We implemented GitHub Actions to automate testing and ensure code consistency. Whenever a new branch was pushed or a pull request was opened, tests would automatically run, helping to catch bugs early.

- **Automated Testing:** Unit tests were written for critical functions, and GitHub Actions would trigger these tests on every push to a branch. (GitHub, 2024)

- **CI/CD Pipeline:** The pipeline ensured that every merged change passed tests before being integrated into the main branch. (GitHub, 2024)
- As of 26/09/2024 the **Mockito** (Mockito , 2023) package used for mocking **AndroidX** fragments and classes had a depreciated package (**Byte Buddy**) (Winterhalter, 2024) which it relied on for simulating android environment with **Maven** so none of our tests could be run or pass.

✓	Merge pull request #51 from ST10036509/ST10032014-ReadMe	dev	3 days ago 4m 31s	...
	Android CI #16: Commit c3c99f4 pushed by ST10032014			
✓	Update README.md	ST10032014-ReadMe	3 days ago 4m 17s	...
	Android CI #15: Pull request #51 opened by ST10032014			
✓	Merge pull request #50 from ST10036509/ST10032014-patch-1	dev	3 days ago 4m 14s	...
	Android CI #14: Commit 9051d75 pushed by ST10032014			
✓	Added UserDocumentation to attach to readme	ST10032014-patch-1	3 days ago 4m 53s	...
	Android CI #13: Pull request #50 opened by ST10032014			
✓	Merge pull request #47 from ST10036509/OF-89	dev	4 days ago 4m 11s	...
	Android CI #12: Commit 1d3b8bf pushed by ST10026396			
✓	OF-89 cleaned up counter creation ui	OF-89	4 days ago 4m 35s	...
	Android CI #11: Pull request #47 opened by ST10026396			
✓	Merge pull request #46 from ST10036509/UPDATE_JAVA_17	dev	4 days ago 4m 28s	...
	Android CI #10: Commit 24477c1 pushed by ST10036509			
✓	Updated JAVA Version to v17	UPDATE_JAVA_17	4 days ago 5m 10s	...
	Android CI #9: Pull request #46 opened by ST10036509			

✗	Merge pull request #65 from ST10036509/OF-119	dev	yesterday 4m 48s	...
	Android CI #46: Commit 3ad1c8c pushed by ST10048758			
✗	Updates to classes which contain counter model (FirestoreSer...	OF-119	yesterday 5m 4s	...
	Android CI #45: Pull request #65 opened by ST10048758			
✗	Merge pull request #64 from ST10036509/OF-114	dev	yesterday 3m 30s	...
	Android CI #44: Commit 9fabf05 pushed by ST10048758			
✗	Changed counter model and changed CounterAdapter and C...	OF-114	yesterday 3m 57s	...
	Android CI #43: Pull request #64 opened by ST10048758			
✗	Merge pull request #63 from ST10036509/OF-118	dev	2 days ago 4m 11s	...
	Android CI #42: Commit 02aa7e4 pushed by ST10026396			
✗	OF-118	OF-118	2 days ago 4m 3s	...
	Android CI #41: Pull request #63 opened by ST10026396			
✗	Merge pull request #62 from ST10036509/OF-116	dev	2 days ago 4m 0s	...
	Android CI #40: Commit 58f5ab3 pushed by ST10026396			
✗	OF-116	OF-116	2 days ago 4m 6s	...
	Android CI #39: Pull request #62 opened by ST10026396			

Byte Buddy could not instrument all classes within the mock's type hierarchy

This problem should never occur for javac-compiled classes. This problem has been observed for classes that are:

- Compiled by older versions of scalac
- Classes that are part of the Android distribution
  - at org.mockito.internal.creation.bytebuddy.InlineBytecodeGenerator.triggerRetransformation(InlineBytecodeGenerator.java:280)
  - at org.mockito.internal.creation.bytebuddy.InlineBytecodeGenerator.mockClass(InlineBytecodeGenerator.java:213)
  - at org.mockito.internal.creation.bytebuddy.TypeCachingBytecodeGenerator.lambda\$mockClass\$0(TypeCachingBytecodeGenerator.java:47)
  - at net.bytebuddy.TypeCache.findOrInsert(TypeCache.java:168)
  - at net.bytebuddy.TypeCache\$WithInlineExpunction.findOrInsert(TypeCache.java:399)
  - at net.bytebuddy.TypeCache.findOrInsert(TypeCache.java:190)
  - at net.bytebuddy.TypeCache\$WithInlineExpunction.findOrInsert(TypeCache.java:410)
  - at org.mockito.internal.creation.bytebuddy.TypeCachingBytecodeGenerator.mockClass(TypeCachingBytecodeGenerator.java:40)
  - at org.mockito.internal.creation.bytebuddy.InlineDelegateByteBuddyMockMaker.createMockType(InlineDelegateByteBuddyMockMaker.java:391)

## Logs showing Byte Buddy failing

## Further Details

### Firestore Integration:

- Firestore was used to store both the user counters and their settings/preferences. Each user has their own document, storing counters (with titles, descriptions, and increment values) and a separate collection for their settings (notifications, theme, language). This separation makes it easy to scale and manage data.

### Firebase Authentication:

- Firebase Authentication was used to handle email-based login and Google SSO. Upon successful login, user details are pulled from Firebase to display in the settings page.

### API Integration:

- We implemented two of Firebase's REST APIs utilising their **Firebase Authentication** service accepting either **Firebase Tokens** or **OAuth 2.0 Tokens** and their NoSQL database service **Firestore** for handling RESTful remote cloud data storage. (Google, 2024) (Google, 2024)
- The Random Advice API is a public REST API that returns a random piece of advice. We integrated it into the settings page using a simple HTTP GET request, which is triggered every time the settings page is loaded. The response is parsed and displayed in a TextView. (Kiss, 2024)
- Access the API [here](#).