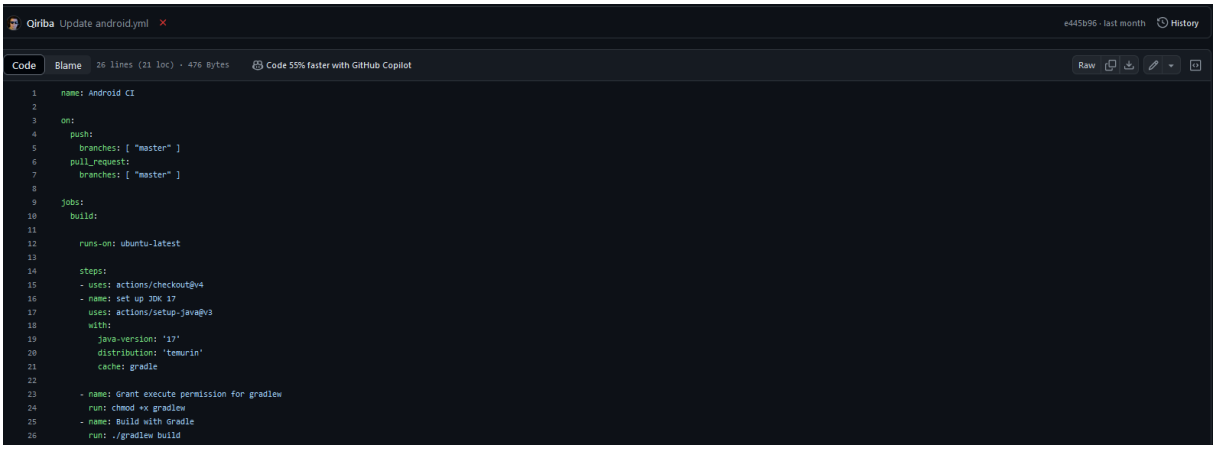
CI/CD Setup

Um unsere Unittests zu verbessern haben wir nun 3 Pipelines.

Die Tests sind unterteilt in:

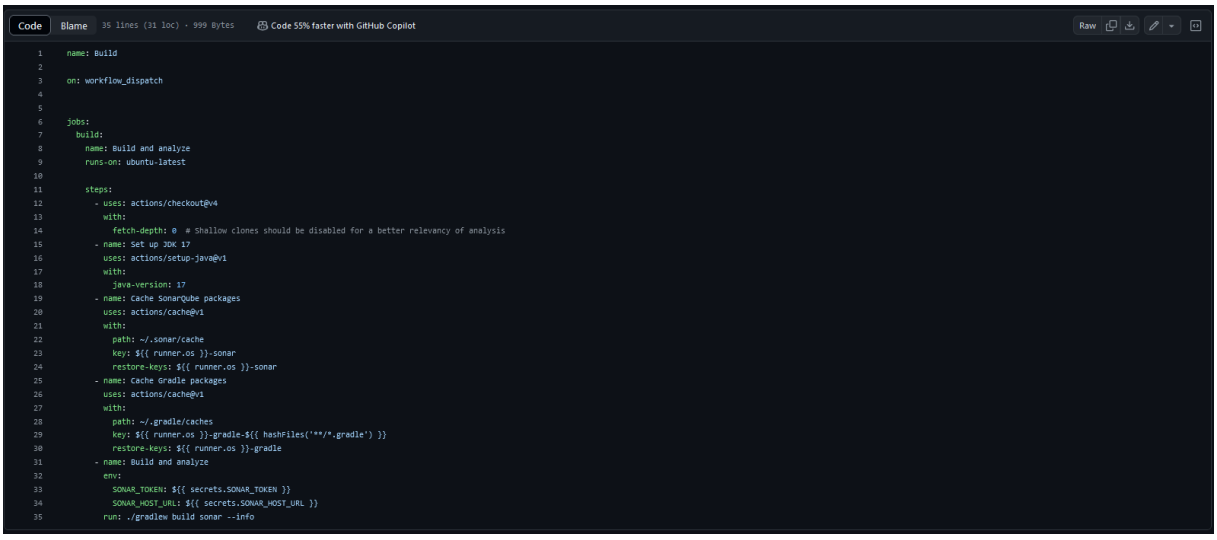
Android.yml

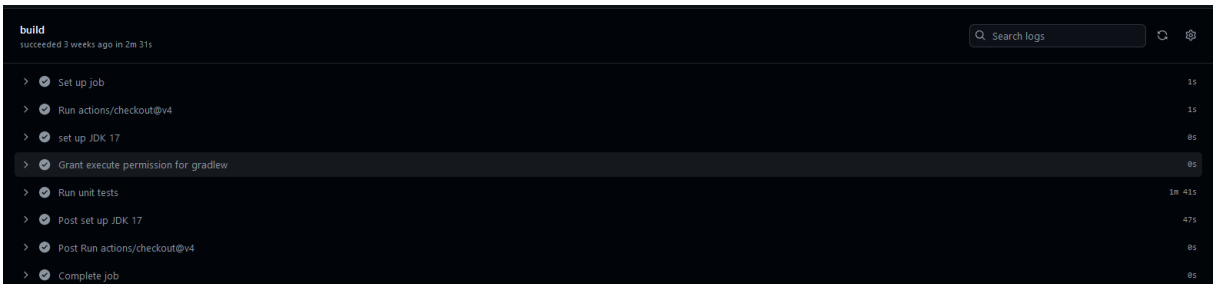
Datei konfiguriert eine GitHub Actions CI/CD-Pipeline, die bei Pushes und Pull Requests auf den master-Branch eine Android-App mit JDK 17 auf Ubuntu baut und testet



Build.yml

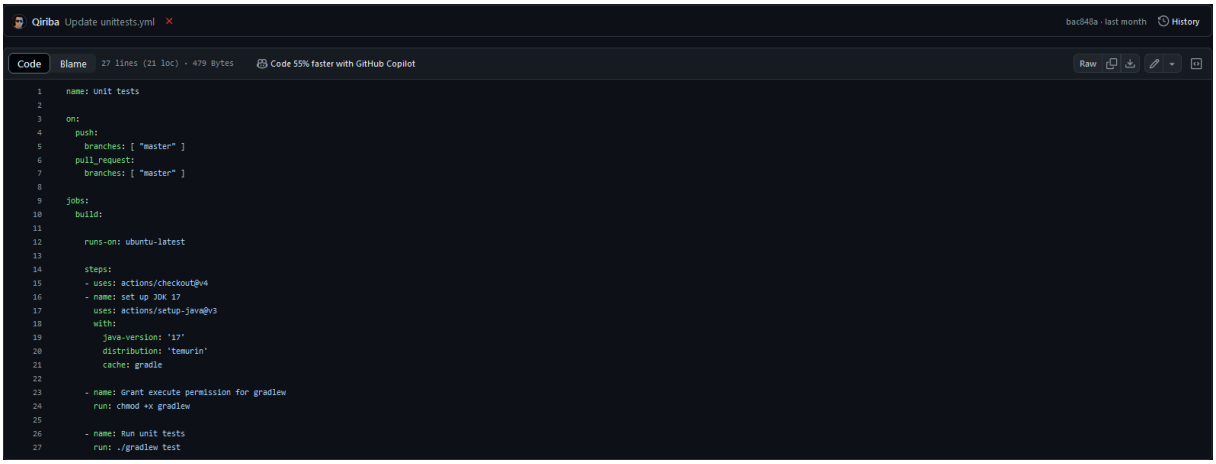
Datei konfiguriert eine GitHub Actions CI/CD-Pipeline, die manuell ausgelöst wird, um eine Android-App mit JDK 17 zu bauen, SonarQube-Analysen durchzuführen und Pakete zu cachen.



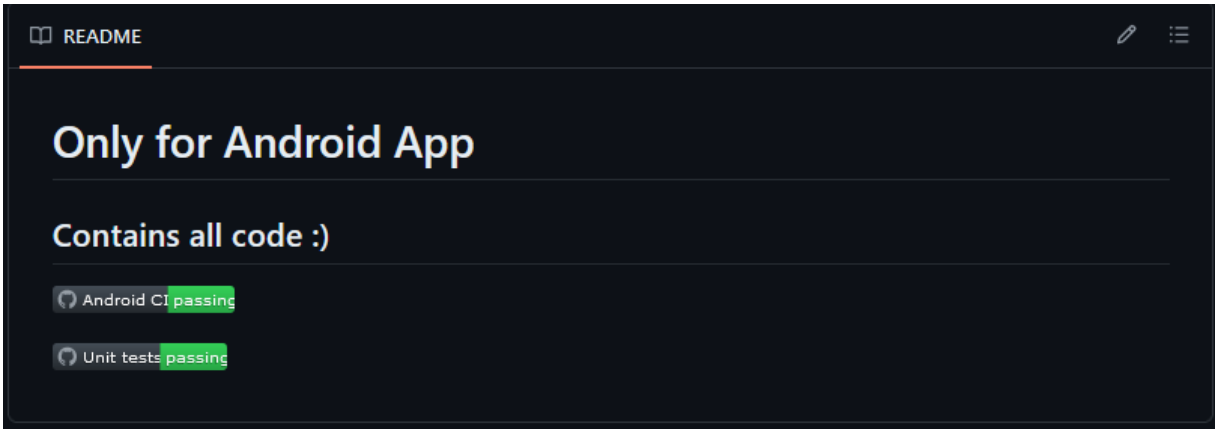


Unittests.yml

Datei konfiguriert eine GitHub Actions CI/CD-Pipeline, die bei Pushes und Pull Requests auf den master-Branch Unit-Tests für eine Android-App mit JDK 17 auf Ubuntu ausführt.



Wir haben uns außerdem dafür entschieden, weil CI/CD die Zusammenarbeit innerhalb des Entwicklungsteams fördert. Dadurch wird die Kommunikation verbessert und das Risiko von Konflikten minimiert.



Insgesamt bieten CI/CD Pipelines eine effiziente und zuverlässige Möglichkeit, Software zu entwickeln, zu testen und bereitzustellen, was zu einer verbesserten Produktqualität und einer erhöhten Entwicklerproduktivität führt.