**TUGAS AKHIR**

RANCANG BANGUN ALAT PENGUBAH KOTLIN DATA CLASS Ke PROTOCOL BUFFERS MESSAGE

****

**DISUSUN OLEH:**

**FATKHI NUR AKHSAN**

**NIM. 20106050026**

**PROGRAM STUDI INFORMATIKA  
FAKULTAS SAINS DAN TEKNOLOGI  
UNIVERSITAS ISLAM NEGERI SUNAN KALIJAGA  
YOGYAKARTA   
2024**

INTISARI

ABSTRACT

KATA PENGANTAR

DAFTAR ISI

[BAB I PENDAHULUAN 1](#_Toc170908820)

[1.1 LATAR BELAKANG 1](#_Toc170908821)

[1.2 RUMUSAN MASALAH 1](#_Toc170908822)

[1.3 TUJUAN 1](#_Toc170908823)

[1.4 MANFAAT 1](#_Toc170908824)

[BAB II KAJIAN PUSTAKA 2](#_Toc170908825)

[2.1 Kotlin 2](#_Toc170908826)

[2.2 Kotlin Data Class 2](#_Toc170908827)

[2.3 Protocol Buffers 2](#_Toc170908828)

[2.4 Protocol Buffers Message 2](#_Toc170908829)

[2.5 Plugin 2](#_Toc170908830)

[BAB III METODE PENGEMBANGAN SISTEM 3](#_Toc170908831)

[3.1 Bahan dan Alat 3](#_Toc170908832)

[3.2 Metode Pengembangan 3](#_Toc170908833)

[3.3 Tahapan Pengembangan 3](#_Toc170908834)

[BAB IV PERANCANGAN DAN IMPLEMENTASI SISTEM 4](#_Toc170908835)

[BAB V KESIMPULAN DAN SARAN 5](#_Toc170908836)

[5.1 Kesimpulan 5](#_Toc170908837)

[5.2 Saran 5](#_Toc170908838)

DAFTAR GAMBAR

# PENDAHULUAN

## LATAR BELAKANG

Perkembangan teknologi dalam beberapa dekade terakhir sangatlah pesat, salah satu teknologi yang berkembang pesat adalah perangkat komputasi mobile dan smartphone dalam 3 dekade terakhir. Saat ini perangkat genggam sudah memiliki daya komputasi yang sama besarnya dengan ruangan yang penuh dengan komputer pada tahun 1970-an. Perkembangan daya komputasi tersebut tidak terlepas dari peningkatan tajam jumlah transistor yang dapat ditampung dalam sebuah *chip* akan berlipat ganda setiap dua tahun, sesuai dengan hukum Moore (Moore’s law) [1].

Meskipun memiliki kecepatan proses dan konektivitas yang masih lebih lambat dibandingkan dengan perangkat dekstop, penggunaan perangkat mobile dan aplikasi mobile lebih dipilih oleh pengguna dibandingkan dengan perangkat dekstop terutama untuk menyelesaikan pekerjaan yang mudah [2]. Salah satunya yaitu aplikasi *mobile-commerce* yang marak digunakan ketika masa pandemi COVID-19, penggunaan aplikasi *mobile-commerce* juga diprediksi masih berlanjut saat periode endemi [2]. Senada dengan hal itu, laporan dari Cisco menunjukkan bahwa hampir 300 miliar aplikasi seluler akan diunduh pada tahun 2023, dengan aplikasi berjenis media sosial, game, dan bisnis akan menjadi unduhan yang paling populer [3].

Diantara penggunaan masif perangkat mobile terdapat dua sistem operasi yang sudah mendominasi pasar dengan cakupan 90% dari market, yaitu Android (Google, Mountain View, CA) and iOS (Apple) sejak tahun 2008 [4]. Untuk mendukung produktifitas developer, Google mengumumkan dukungannya pada Google I/O 2017 untuk menjadikan bahasa pemrograman Kotlin sebagai bahasa resmi untuk pengembangan aplikasi android [5]. Setelah dua tahun berselang, tepatnya saat Google I/O 2019, pengembangan aplikasi mobile android akan menjadi *kotlin-first* setelah Google mengumumkannya, hal ini berarti Kotlin akan menjadi pilihan pertama Google dalam mengembangkan alat dan konten pengembangan aplikasi android [6].

Bahasa pemrograman Kotlin sendiri merupakan bahasa pemrograman open source berjenis statis yang mendukung pemrograman berorientasi objek dan fungsional [[Ringkasan Kotlin  |  Android Developers](https://developer.android.com/kotlin/overview?hl=id)]. Kotlin pada mulanya dirancang oleh JetBrains, kemudian terus dikembangkan oleh para kontributor dan dikelola oleh Kotlin Foundation yang didirikan oleh JetBrains dan Google [[Home | Kotlin Foundation – official site](https://kotlinfoundation.org/)]. Meskipun tergolong bahasa yang modern, Kotlin juga dapat dikatakan bahasa yang sudah matang. Bahasa ini ringkas, aman, dapat dioperasikan dengan Java dan bahasa lain, dan menyediakan banyak cara untuk menggunakan kembali kode di antara berbagai platform untuk pemrograman yang produktif [https://kotlinlang.org/]. Oleh karena itu, beberapa API pada Android seperti AndroidKTX yang dikhususkan untuk kotlin tetap dapat kompatibel dengan Bahasa Java yang sudah lebih dulu menjadi bahasa resmi untuk pengembangan aplikasi android. Dengan interoperabilitas yang baik dengan Java tersebutlah, Kotlin berhasil menghasilkan pengalaman pengembangan aplikasi android yang lebih nyaman [].

Salah satu library yang memiliki AndroidKTX sebagai ekstensi adalah library Jetpack DataStore. Jetpack DataStore adalah salah satu library yang menyediakan solusi penyimpanan data lokal yang terdapat pada platform android, khususnya untuk kumpulan data yang kecil dan sederhana dan tidak memerlukan pembaruan parsial atau integritas referensial. Jetpack DataStore dapat menyimpan data berupa pasangan *key*-*value* atau objek yang ditulis menggunakan protocol buffers. Oleh karena itu, Jetpack DataStore dibagi menjadi dua jenis, yaitu Preferences DataStore, yang menyimpan dan mengakses data menggunakan kunci dan Proto DataStore, yang menyimpan data berupa instansiasi objek yang sebelumnya telah didefinisikan menggunakan schema yang ditulis menggunakan protocol buffers. Jetpack DataStore menggunakan Kotlin Coroutines dan Flow untuk menyimpan data secara asinkron, konsisten, dan transaksional [].

Protocol Buffers atau yang juga dapat disingkat dengan protobuf, merupakan mekanisme yang dapat diekstensi untuk membuat serialisasi data terstruktur yang *language-netral* dan *platform-netral*. Protocol Buffers sendiri unggul dalam hal ukuran dan kecepatan dibandingkan dengan format data lain seperti JSON dan XML, dan dapat menghasilkan binding bahasa asli. Protocol Buffers adalah kombinasi bahasa definisi (dibuat dalam file .proto), kode yang dibuat oleh kompiler proto untuk berinteraksi dengan data, pustaka runtime yang *language-specific*, format serialisasi untuk data yang ditulis ke file (atau dikirim melalui koneksi jaringan), dan data yang terserialisasi [https://protobuf.dev/].

Salah satu cara untuk menampung data pada bahasa pemrograman kotlin adalah dengan menggunakan Data Class yang ditandai dengan keyword “data”. Untuk setiap Data Class, kompiler akan secara otomatis menghasilkan fungsi tambahan yang memungkinkan untuk mencetak instance ke output yang dapat dibaca, membandingkan instance, menyalin instance, dan banyak lagi. Hal ini mengurangi kode *boilerplate* yang dihasilkan dibandingkan dengan mekanisme penampungan data yang terdapat pada bahasa pemrograman lain, seperti POJO pada bahasa Java [https://kotlinlang.org/].

## RUMUSAN MASALAH

## TUJUAN

## MANFAAT

# KAJIAN PUSTAKA

## Kotlin

## Kotlin Data Class

## Protocol Buffers

## Protocol Buffers Message

## Plugin

# METODE PENGEMBANGAN SISTEM

## Bahan dan Alat

## Metode Pengembangan

## Tahapan Pengembangan

# PERANCANGAN DAN IMPLEMENTASI SISTEM

# KESIMPULAN DAN SARAN

## Kesimpulan

## Saran

DAFTAR PUSTAKA