ISSN : 2614-0985 (Cetak)
ISSN : 2614-0977 (Online)
KOMPUTEK
url : http://studentjournal.umpo.ac.id/index.php/komputek

Optimalisasi Duplikasi Komponen *Cardview* dengan *Prototype Design Pattern* dalam Pengembangan Antarmuka Pengguna Android

Risma Auliya Salsabilla*, Aditya Daffa Syahputra, Carudin

Fakultas Ilmu Komputer, Universitas Singaperbangsa Karawang

E-mail Korespondensi: 2210631170100@student.unsika.ac.id

History Artikel

Diterima : 5 Mei 2025 Disetujui : 10 September 2025 Dipublikasikan : 25 Oktober 2025

Abstract

The rapid advancement of information technology encourages application developers to create products that are not only functional, but also efficient and maintainable. One key challenge in Android user interface (UI) development is the repeated duplication of similar components, such as CardView, which share common structures, but differ in content. This study aims to explore the effectiveness of implementing the Prototype Design Pattern to optimize the duplication of CardView components in Android UI development. An implementative case study approach and the prototype software development model were used. A simple application was developed using Kotlin and XML, featuring multiple CardView instances created from a single base object via a cloning mechanism. The results indicate that applying the Prototype Design Pattern significantly reduces code redundancy, accelerates component instantiation, and improves design maintainability. The pattern also supports the principles of code reuse and single-responsibility, which are essential in large-scale application development. These findings suggest that the Prototype Design Pattern provides an effective solution for building modular, efficient, and adaptable Android user interfaces.

Keywords: *Prototype Design Pattern, Android, CardView, User Interface, Kotlin*

Abstrak

Pesatnya perkembangan teknologi informasi mendorong pengembang aplikasi untuk menghasilkan produk yang tidak hanya fungsional, tetapi juga efisien dan mudah dipelihara. Salah satu tantangan dalam pengembangan antarmuka pengguna (*User Interface*) Android adalah duplikasi komponen yang berulang, seperti *CardView*, dengan struktur yang serupa, tetapi isi kontennya berbeda. Penelitian ini bertujuan untuk mengeksplorasi efektivitas penerapan *Prototype Design Pattern* dalam mengoptimalkan proses duplikasi komponen *CardView* pada pengembangan *User Interface* Android. Penelitian menggunakan pendekatan studi kasus implementatif dan metode pengembangan perangkat lunak model *prototype*. Aplikasi sederhana dikembangkan menggunakan bahasa pemrograman Kotlin dan XML, menampilkan beberapa *CardView* yang dibangun dari satu objek dasar melalui proses *cloning*. Hasil penelitian menunjukkan bahwa penggunaan *Prototype Design Pattern* berhasil mengurangi redundansi kode, mempercepat proses instansiasi komponen, serta meningkatkan efisiensi pemeliharaan desain antarmuka. Pola ini juga mendukung prinsip *code reuse* dan *single-responsibility*, yang penting dalam pengembangan aplikasi berskala besar. Penelitian ini menunjukkan bahwa *Prototype Design Pattern* dapat menjadi solusi yang efektif untuk membangun *User Interface* Android yang modular, efisien, dan mudah untuk dilakukan penyesuaian.

Kata Kunci: *Prototype Design Pattern, Android, CardView, Antarmuka Pengguna, Kotlin*

How to Cite: Risma Auliya Salsabilla (2025). Optimalisasi Duplikasi Komponen Cardview dengan Prototype Design Pattern dalam Pengembangan Antarmuka Pengguna Android. KOMPUTEK : Jurnal Universitas Muhammadiyah Ponorogo, Vol 9(2): Halaman 76-85

ISSN 2614-0985 (Print)
ISSN 2614-0977 (Online)

PENDAHULUAN

Perkembangan teknologi informasi sekarang ini menunjukkan peningkatan yang sangat pesat dan telah memberikan pengaruh besar terhadap berbagai aspek kehidupan manusia (Ashari et al. 2022). Teknologi dimanfaatkan untuk mendukung aktivitas sehari-hari agar lebih efektif dan efisien, termasuk dalam bidang pengembangan perangkat lunak. Para pengembang aplikasi dituntut untuk mampu mengikuti perkembangan teknologi terkini serta menghasilkan produk yang tidak hanya fungsional, tetapi juga memiliki antarmuka pengguna yang menarik dan mudah digunakan (Murti & Sujarwo 2021). Teknologi informasi juga menjadi sarana utama dalam pertukaran informasi yang cepat dan akurat (Ashari 2020), ditandai dengan meningkatnya kepemilikan perangkat elektronik, seperti *smartphone* dan komputer, dari berbagai kalangan (Ivanka & Firdaus 2024).

Menurut (Rusdiana & Setiawan 2018), salah satu inovasi penting dalam teknologi adalah pengembangan aplikasi seluler yang dapat diakses melalui perangkat Android. Android sendiri didefinisikan sebagai sistem operasi *mobile* berbasis Linux yang mencakup sistem operasi inti, *middleware*, dan aplikasi. Sebagai sistem operasi yang umum digunakan pada ponsel cerdas dan tablet layar sentuh, Android dibangun di atas kernel Linux (Andri et al. 2020). Android menyediakan *Application Programming Interface* (API) yang memungkinkan untuk mengakses ke perangkat keras dan data ponsel, bahkan hingga data sistem. Hal tersebut memberikan kebebasan bagi pengguna untuk menghapus atau mengganti aplikasi bawaan dengan aplikasi pihak ketiga (Kontu et al. 2015). Sifatnya yang *open source* adalah keunggulan utamanya karena memungkinkan pengembang untuk terus berinovasi dan menciptakan beragam aplikasi yang dapat dipakai di berbagai perangkat *mobile* (Kusuma 2018).

Dalam konteks tersebut, pengalaman pengguna atau *User Experience* menjadi kunci utama dalam mempertahankan keterlibatan pengguna terhadap produk dan layanan teknologi (Sodik et al. 2024). Tingkat kepuasan pengguna sangat dipengaruhi oleh pengalaman interaksinya dengan antarmuka aplikasi atau yang biasa dikenal sebagai *User Interface* (Razi et al. 2018). Antarmuka tersebut mencakup segala aspek visual dan interaktif yang dapat dilihat, diraba, serta dioperasikan oleh pengguna, seperti tata letak, desain visual, ikon, tombol, dan interaksi antarmuka, dengan fokus pada kemudahan akses, pemahaman, dan penggunaan sistem (Alaik & Sodik 2023).

Pada pengembangan aplikasi Android, salah satu komponen *User Interface* yang sangat populer dan sering diduplikasi adalah *CardView*. *CardView* berfungsi sebagai *wrapper* atau *frame layout* yang membungkus konten di dalamnya dengan desain menyerupai kartu, umumnya dijumpai pada Google Play Store dan memiliki ciri khas *rounded corner* serta *shadow* untuk efek *elevation* (Distria et al. 2017). *CardView* ideal untuk menampilkan informasi dalam format kartu dan sering digunakan dalam daftar bergulir, seperti *RecyclerView*, yaitu sebuah *ViewGroup* yang efisien untuk menampilkan koleksi data berupa tulisan, angka, serta gambar dalam format daftar atau grid (Android Developers).

Namun, ketika sebuah aplikasi memerlukan banyak *CardView* dengan desain dasar yang sama, tetapi variasi pada konten atau beberapa atribut kecil (misalnya, warna latar belakang atau *elevation*), pengembang dihadapkan pada tantangan efisiensi serius dalam instansiasi objek dan pemeliharaan kode. Pendekatan konvensional yang melibatkan instansiasi berulang dari *CardView* dan pengaturan atribut satu per satu dapat menyebabkan redundansi kode yang tinggi, *overhead* instansiasi yang memakan waktu dan sumber daya, serta kesulitan pemeliharaan karena perubahan desain dasar harus dilakukan pada banyak tempat yang dapat meningkatkan risiko kesalahan (Salman et al. 2025).

Untuk mengatasi permasalahan duplikasi objek yang berulang tersebut, penerapan *design pattern* menjadi solusi yang sangat relevan untuk memastikan kinerja optimal aplikasi, termasuk kecepatan eksekusi dan efisiensi penggunaan memori. *Design pattern* sendiri didefinisikan sebagai solusi umum yang terbukti untuk masalah desain berulang yang kerap muncul selama pengembangan aplikasi (Gamma et al. 1995 dalam Pandey et al. 2025). Pola-pola tersebut layaknya *blueprint* siap pakai yang dapat disesuaikan untuk menyelesaikan persoalan desain yang berulang (Marchesi et al. 2021). Pola desain orisinal diklasifikasikan menjadi tiga kategori utama, yaitu *Creational* (berfokus pada abstraksi proses instansiasi objek untuk memisahkan kode klien dari detail pembuatan objek), *Structural* (berkonsentrasi pada interaksi hierarki kelas untuk membentuk struktur kompleks), dan *Behavioral* (solusi untuk mendelegasikan tanggung jawab antar objek) (Prokakis 2024).

xDi antara jenis-jenis pola pada *Creational Pattern* yang ada, *Prototype Design Pattern* menonjol sebagai solusi yang efektif untuk mengatasi permasalahan tersebut. *Prototype Design Pattern* memungkinkan pembuatan objek baru dengan menyalin objek yang sudah ada (*prototype*), alih-alih membangun ulang dari nol. Proses *cloning* didelegasikan ke objek itu sendiri, kemudian disediakan antarmuka umum untuk semua objek yang mendukung *cloning* tersebut sehingga pengembang dapat menduplikasi objek tanpa terikat pada kelas spesifiknya (Refactoring Guru). Pola desain ini sangat berguna ketika biaya pembuatan objek dari awal cukup tinggi dan objek-objek serupa seringkali dibutuhkan. Dengan menduplikasi *prototype*, objek baru dapat dibuat dengan cepat serta efisien (Manchana 2019).

Penerapan *Prototype Design Pattern* dilakukan menggunakan Kotlin, bahasa pemrograman modern yang telah mendapatkan dukungan resmi dari Google untuk pengembangan Android sejak tahun 2017 (Aulia & Cuhenda 2025). Bahasa pemrograman Kotlin yang digunakan dalam penelitian ini sangat relevan untuk implementasi pola desain tersebut. Kotlin adalah bahasa modern yang dirancang khusus untuk pengembangan Android, menawarkan sintaks ringkas, keamanan terhadap *null-pointer exception*, interoperabilitas penuh dengan Java, dan peningkatan produktivitas (Rhomadon & Setiawan 2025).

Berdasarkan permasalahan yang telah dijabarkan pada paragraf sebelumnya, penelitian ini bertujuan untuk

mengeksplorasi dan menganalisis efektivitas implementasi *Prototype Design Pattern* dalam optimasi duplikasi komponen *CardView* pada pengembangan *User Interface* aplikasi Android. Fokus utama dari penelitian ini ialah bagaimana pola tersebut dapat mengurangi redundansi kode, meningkatkan efisiensi instansiasi *CardView* yang serupa, dan menyederhanakan proses pemeliharaan desain komponen *User Interface* yang berulang. Hasil dari penelitian ini diharapkan dapat menjadi panduan praktis bagi pengembang Android dalam membangun *User Interface* yang lebih efisien, terstruktur, dan mudah dikelola, khususnya saat berhadapan dengan komponen yang sifatnya berulang.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan studi kasus implementatif yang difokuskan pada penerapan *Prototype Design Pattern* dalam proses pengembangan antarmuka pengguna Android, khususnya dalam optimalisasi duplikasi komponen *CardView*. Pendekatan ini dipilih karena dapat memberikan pemahaman langsung terhadap efektivitas pola desain dalam menyelesaikan permasalahan pengembangan antarmuka yang berulang. Selain itu, penelitian ini menerapkan model pengembangan perangkat lunak *prototype*, yaitu pendekatan iteratif yang memungkinkan pembuatan purwarupa antarmuka untuk diuji dan diperbaiki berdasarkan *feedback* pengguna sehingga menghasilkan sistem yang lebih optimal secara fungsional dan visual (Setiadi 2025).

Penelitian ini dilakukan dengan membangun aplikasi sederhana yang menampilkan beberapa *CardView* dengan struktur dan gaya tampilan yang serupa, tetapi dengan konten yang berbeda. Dengan menerapkan *Prototype Design Pattern*, proses duplikasi *CardView* dilakukan melalui mekanisme *cloning* terhadap objek antarmuka yang telah dibuat sebelumnya sehingga pengembangan menjadi lebih efisien tanpa perlu membuat ulang strukturnya dari awal. Implementasi dilakukan menggunakan bahasa pemrograman Kotlin dan XML sebagai komponen dasar pembangunan antarmuka pengguna.

Jenis data yang digunakan dalam penelitian ini merupakan data kualitatif yang diperoleh melalui dokumentasi proses pengembangan antarmuka, hasil pengamatan terhadap implementasi duplikasi *CardView*, serta kode program yang diterapkan. Sumber data primer berasal dari pengembangan langsung aplikasi Android menggunakan bahasa pemrograman Kotlin dan XML, sementara sumber data sekunder diperoleh melalui kajian literatur yang relevan, seperti dokumentasi resmi dari Android Developer, referensi mengenai *design pattern*, serta penelitian terdahulu yang menggunakan metode *prototype*.

Terdapat beberapa tahapan dalam proses penelitian, yaitu pencarian studi kasus yang relevan dengan penerapan *Prototype Design Pattern*, menganalisis bagaimana pola desain tersebut dapat diimplementasikan dalam proses duplikasi antarmuka, serta melakukan pengembangan aplikasi Android dengan memanfaatkan prinsip-prinsip dari *Prototype Design Pattern*. Setiap tahapan ditujukan untuk memperlihatkan efektivitas pola desain tersebut dalam menyederhanakan pengulangan komponen *User Interface* dan meningkatkan efisiensi pengembangan antarmuka yang bersifat modular serta konsisten.

HASIL DAN PEMBAHASAN

Dalam penelitian ini, tahapan awal metode penelitian difokuskan pada pencarian studi kasus relevan yang secara spesifik menunjukkan permasalahan duplikasi komponen *User Interface* Android dan bagaimana *Prototype Design Pattern* dapat menjadi solusinya. Kami mengidentifikasi *CardView* sebagai komponen antarmuka utama dalam studi kasus ini. Pemilihan *CardView* bukan tanpa alasan, tetapi popularitasnya yang meluas dalam desain aplikasi Android menjadikannya sesuatu yang cocok untuk diteliti. Komponen ini sering kali diduplikasi secara masif dalam berbagai skenario, seperti pada daftar berita, katalog produk maupun *feed* media sosial, di mana setiap kartu memiliki konfigurasi dasar yang serupa, tetapi konten atau beberapa atribut kecilnya berbeda.

Setelah studi kasus diidentifikasi, tahapan selanjutnya adalah menganalisis secara mendalam bagaimana *Prototype Design Pattern* dapat diimplementasikan dalam proses duplikasi antarmuka *CardView* tersebut. Analisis ini tidak hanya mencakup identifikasi titik-titik di mana duplikasi terjadi, tetapi juga mengeksplorasi bagaimana struktur *Prototype Design Pattern* dapat diterapkan untuk menghasilkan objek *CardView* baru secara efisien. Fokus utama adalah menguraikan mekanisme *cloning* objek *CardView* dasar (*prototype*), kemudian memodifikasi hanya atribut-atribut yang bervariasi untuk setiap instansinya. Analisis ini bertujuan untuk membuktikan bahwa *Prototype Design Pattern* mampu mengatasi masalah efisiensi dan pemeliharaan kode yang sering muncul pada pengembangan *User Interface* Android berskala besar.

Penelitian ini menghasilkan sebuah aplikasi sederhana yang memiliki tampilan utama berisi beberapa *CardView* yang dibangun dengan prinsip *Prototype Design Pattern* untuk duplikasi *CardView* yang tampil pada halaman utama. Semua komponen ditampilkan menggunakan *RecyclerView* yang dikendalikan oleh *adapter*. Semua aset gambar dan beberapa teks statis di-*input* secara manual dan dimasukkan ke dalam *resource drawable* pada direktori *project.* Gambar 1 menunjukkan desain awal dari *CardView* yang dibuat.

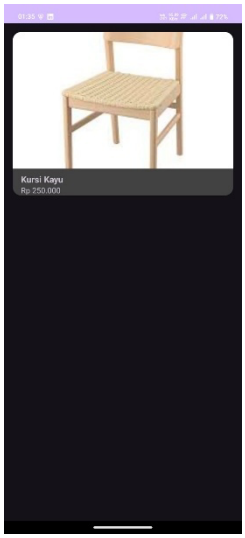
Gambar 1. Desain Awal *CardView*

Desain *CardView* yang dibuat memiliki tiga komponen inti, yaitu *ImageView* sebagai tempat untuk gambar serta dua *TextView* yang masing-masing memiliki fungsi untuk menampilkan ‘Title’ dan ‘Price’. Tiga komponen tersebut dapat dilihat dengan jelas pada Gambar 2.

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main)

    recyclerView = findViewById(R.id.recyclerView)
    recyclerView.layoutManager = LinearLayoutManager( context: this)

    // Original product
    val originalCard = ProductCard(
        title = "Kursi Kayu",
        price = "Rp 250.000",
        imageResId = R.drawable.kursi
    )

    productList.add(originalCard)
    productAdapter = ProductAdapter(productList)
    recyclerView.adapter = productAdapter

}
}
```

Gambar 2. Kode pada MainActivity.kt untuk Membangun Objek *CardView* Utama

Pada kode di atas, tertulis pembuatan variabel 'originalCard' yang digunakan sebagai penyimpan *value* yang akan ditampilkan pada *CardView*. Kode di atas akan menghasilkan *output* dari suatu *CardView* sesuai dengan yang telah dibuat sebelumnya menggunakan *value* yang tertulis. Di bawah ini ditunjukkan implementasi dari kode dasar tersebut pada Gambar 3.



Gambar 3. Hasil Antarmuka *CardView* Dasar

Dalam penerapan *Prototype Design Pattern*, kata kunci yang sering dijumpai dalam penulisan kode adalah 'clone()' yang digunakan untuk melakukan *cloning* pada objek agar tidak perlu membuat objek baru lagi apabila objek tersebut memiliki atribut yang sama.

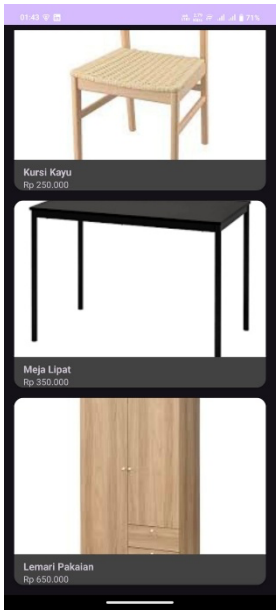
Tahapan selanjutnya untuk membuat objek baru ialah dengan menambah variabel lain dan langsung melakukan *clone* pada objek dasar yang telah dibuat sebelumnya. Pembuatan objek lain dapat disesuaikan dengan atribut dan berapa *CardView* yang dibutuhkan seperti yang dijabarkan pada kode di Gambar 4.

```
val meja = originalCard.clone().apply {
    title = "Meja Lipat"
    price = "Rp 350.000"
    imageResId = R.drawable.meja
}

val lemari = originalCard.clone().apply {
    title = "Lemari Pakaian"
    price = "Rp 650.000"
    imageResId = R.drawable.lemari
}

productList.add(originalCard)
productList.add(meja)
productList.add(lemari)

productAdapter = ProductAdapter(productList)
recyclerView.adapter = productAdapter
```

Gambar 4. Penambahan Kode Inti untuk Menambah Dua Objek *CardView*

Kode di atas merupakan pola penerapan *Prototype Design Pattern* dengan melakukan *clone* pada objek dasar *CardView.* Penerapan fungsi *clone* ditujukan kepada variabel 'originalCard', yang merupakan variabel dasar untuk menampung objek *CardView* saat pertama kali dibuat. Penerapan *Prototype Design Pattern* membuat penulisan kode dalam aplikasi lebih mudah terbaca dan dipelihara.

Berikut ini adalah antarmuka dari penerapan *Prototype Design Pattern* yang telah diduplikasi menjadi beberapa objek lainnya yang ditunjukkan pada Gambar 5.



Gambar 5. Hasil Penambahan Objek *CardView* Baru Menggunakan *Prototype Design Pattern*

Adapun sebagai perbandingan dengan hasil yang sama pada Gambar 5, penulisan kode dapat ditulis secara manual seperti pada Gambar 6 di bawah ini.

```
val kursi = ProductCard(
    title = "Kursi Kayu",
    price = "Rp 250.000",
    imageResId = R.drawable.kursi
)
val meja = ProductCard(
    title = "Meja Lipat",
    price = "Rp 350.000",
    imageResId = R.drawable.meja
)
val lemari = ProductCard(
    title = "Lemari Pakaian",
    price = "Rp 650.000",
    imageResId = R.drawable.lemari
)

productList.add(kursi)
productList.add(meja)
productList.add(lemari)

productAdapter = ProductAdapter(productList)
recyclerView.adapter = productAdapter
```

Gambar 6. Potongan Kode Inti Pembuatan *CardView* Tanpa *Prototype Design Pattern*

Dari perbandingan tersebut, terlihat bahwa pendekatan tanpa pola desain akan mengarah pada redundansi kode yang tidak efisien, terutama jika jumlah produk sangat banyak. Sebaliknya, *Prototype Design Pattern* memungkinkan pengembangan dilakukan lebih modular dan hemat penulisan kode karena struktur dasar produk hanya ditulis sekali serta dapat diduplikasi secara fleksibel dengan konten berbeda.

Penerapan pola desain ini juga mendukung prinsip *single-responsibility* dan *code reuse*, di mana struktur antarmuka dipisahkan dari data serta memungkinkan skalabilitas antarmuka yang lebih baik. Dalam konteks pengembangan Android, hal tersebut berdampak positif terhadap pemeliharaan kode dan konsistensi antarmuka pengguna.

KESIMPULAN

Berdasarkan hasil dan pembahasan yang telah dijelaskan sebelumnya, penelitian ini menunjukkan bahwa penerapan *Prototype Design Pattern* dalam pengembangan *User Interface* Android memberikan kontribusi signifikan dalam mengoptimalkan proses duplikasi komponen yang memiliki struktur dan tampilan serupa. Dengan memanfaatkan mekanisme *cloning* pada objek *prototype*, pengembang dapat menghindari penulisan kode yang berulang, meningkatkan efisiensi waktu pengembangan, serta menjaga konsistensi desain antarmuka.

Melalui studi kasus pembuatan daftar produk dengan tampilan *CardView*, pendekatan ini terbukti mampu menyederhanakan pembuatan komponen antarmuka pengguna secara modular tanpa mengorbankan fleksibilitas konten yang ditampilkan. Selain itu, implementasi *Prototype Design Pattern* juga mendukung prinsip pemrograman berorientasi objek yang baik, seperti *code reuse* dan *maintainability*, yang menjadi aspek penting dalam pengembangan aplikasi berskala lebih besar. Dengan demikian, *Prototype Design Pattern* dapat menjadi salah satu solusi desain yang efektif dalam pengembangan *User*

Interface Android, terutama pada konteks aplikasi yang membutuhkan banyak elemen serupa, tetapi memiliki isi yang berbeda.

DAFTAR PUSTAKA

Alaik, L. M., & Sodik, A. (2023). Perancangan User Interface Dan User Experience Pada Website Paid Newsletter XYZ Dengan Model User Centered Design. Pada *Prosiding Seminar Nasional Sains dan Teknologi Terapan*.

Andri, R., Saputri, N. A. O., & Akbar, M. (2020). Sistem Notifikasi Tugas Akhir Universitas Bina Darma Berbasis Mobile. *Sistemasi: Jurnal Sistem Informasi*, *9*(1), 155-165.

Android Developers. *Create dynamic lists with RecyclerView*. Tersedia di: https://developer.android.com/develop/ui/views/layout/recyclerview. Html. Diakses pada 07 Juni 2025.

Ashari, I. F. (2020). Implementation of cyber-physical-social system based on service-oriented architecture in smart tourism. *Journal of Applied Informatics and Computing*, *4*(1), 66-73.

Ashari, I. F., Banjarnahor, R., Farida, D. R., Aisyah, S. P., Dewi, A. P., & Humaya, N. (2022). Application of data mining with the K-means clustering method and Davies Bouldin index for grouping IMDB movies. *Journal of Applied Informatics and Computing*, *6*(1), 07-15.

Aulia, R., & Cuhenda, C. (2025). Perancangan Aplikasi Customer Relationship Management (CRM) Untuk PT. Wahana Satu Propertindo. *Jurnal Informatika dan Teknologi Pendidikan*, *5*(1).

Distria, E., Rumani, R., & Setianingsih, C. (2017). Perancangan Aplikasi Mobile Berbasis Android Pemantau Kepadatan Lalu Lintas Menggunakan Ip Camera. *eProceedings of Engineering*, *4*(3).

Ivanka, A. A., & Firdaus, M. E. B. (2024). Fico Words: Perancangan Game Antarmuka Tebak Kata Interaktif Berbasis Android untuk Anak Usia Dini Mengenal Huruf dan Kata. *KOMPUTEK*, *8*(1), 51-60.

Kontu, R. H., Sompie, S. R., & Sinsuw, A. A. (2015). Perancangan Sistem Pembaca Surat Tanda Nomor Kendaraan Dengan Teknologi NFC. *Jurnal Teknik Elektro dan Komputer*, *4*(3), 79-85.

Kusuma, W. A. (2018). Sistem Informasi Geografis Pemetaan Lokasi Bird Contest Kota Malang Berbasis Android. *SISTEMASI: Jurnal Sistem Informasi*, *7*(3), 212-219.

Manchana, R. (2019). Exploring Creational Design Patterns: Building Flexible and Reusable Software Solutions. *International Journal of Science Engineering and Technology*, *7*, 1-10.

Marchesi, L., Marchesi, M., Destefanis, G., Barabino, G., & Tigano, D. (2020). Design patterns for gas optimization in ethereum. In *2020 IEEE International Workshop on Blockchain Oriented Software Engineering (IWBOSE)* (pp. 9-15). IEEE.

Murti, S. K., & Sujarwo, A. (2021). Membangun Antarmuka Pengguna Menggunakan ReactJs untuk Modul Manajemen Pengguna. *AUTOMATA*, *2*(2).

Pandey, S. K., Chand, S., Horkoff, J., Staron, M., Ochodek, M., & Durisic,

D. (2025). Design pattern recognition: a study of large language models. *Empirical Software Engineering*, *30*(3), 69.

Prokakis, E. (2024). *Benchmarking software design patterns in CRM systems* (Master's thesis, Πανεπιστήμιο Πειραιώς).

Razi, A. A., Mutiaz, I. R., & Setiawan, P. (2018). Penerapan metode design thinking pada model perancangan ui/ux aplikasi penanganan laporan kehilangan dan temuan barang tercecer. *Demandia: Jurnal Desain Komunikasi Visual, Manajemen Desain, dan Periklanan*, *3*(02), 219-237.

RefactoringGuru. Design Pattern. Tersedia di: https://refactoring.guru/design-patterns. Html. Diakses pada 07 Juni 2025.

Rhomadon, A. Z., & Setiawan, R. (2025). Implementasi Aplikasi Monitoring Kos Pondok Ina 2. *Jurnal Janitra Informatika dan Sistem Informasi*, *5*(1), 43-53.

Rusdiana, L., & Setiawan, H. (2018). Perancangan Aplikasi Monitoring Kesehatan Ibu Hamil Berbasis Mobile Android. *Sistemasi: Jurnal Sistem Informasi*, *7*(3), 197-203.

Salman, R. M., Akbar, M. A., & Afirianto, T. (2025). Analisis Perbedaan Perfoma Penggunaan Lazygrid Dan Recyclerview Dalam Menampilkan Koleksi Data. *Jurnal Teknologi Informasi dan Ilmu Komputer*, *12*(1), 87-92.

Setiadi, T., Yasin, V., & Yulianto, A. B. (2025). Perancangan Dashboard Monitoring Stok Berbasis. *Jurnal Riset Sistem Informasi Dan Teknik Informatika (JURASIK)*, 131-137.

Sodik, A., Noviyanti, D. A., & Antoko, N. A. (2024). Penerapan Metode Design Thinking dalam Pengembangan Antarmuka Pengguna dan Pengalaman Pengguna pada Website Learning Management System (LMS). *INTEGER: Journal of Information Technology*, *9*(1).