

Implementasi *Object-Oriented Programming* pada *Game Pembelajaran Matematika Berbasis Android* dengan *UML Class Diagram*

Reza Cahya Putra F*¹, Anang Kukuh Adi Susilo²

^{1,2} Program Studi Informatika, Fakultas Teknik, Universitas Wijaya Kusuma Surabaya
Email: ¹rcpf019@mh.s.uwks.ac.id, ²anang65@uwks.ac.id

Abstrak

Kemajuan teknologi informasi turut mempengaruhi perkembangan media pembelajaran interaktif, salah satunya melalui *game* pembelajaran berbasis Android. Dalam pengembangan *game* pembelajaran, perancangan sistem yang terstruktur menjadi aspek penting agar perangkat lunak dapat dikembangkan secara modular dan mudah dipelihara. Namun, pengembangan *game* sering kali berfokus pada implementasi tanpa perancangan perangkat lunak yang jelas sehingga struktur program menjadi kompleks dan sulit dikembangkan. Penelitian ini bertujuan menerapkan konsep *Object-Oriented Programming* (OOP) melalui perancangan *Unified Modeling Language* (UML) *Class Diagram* pada *game* pembelajaran matematika berbasis *Android*. Metode penelitian dilakukan melalui analisis kebutuhan sistem, perancangan *class diagram*, analisis relasi antar *class*, serta implementasi konsep OOP pada arsitektur *game*. *UML Class Diagram* digunakan untuk memodelkan *class*, atribut, *method*, relasi *object*, dan passing parameter dalam sistem *game* pembelajaran. Hasil penelitian menunjukkan bahwa penggunaan *UML Class Diagram* mampu membantu proses desain perangkat lunak menjadi lebih terstruktur, meningkatkan modularitas sistem, serta mempermudah implementasi konsep OOP seperti *encapsulation*, *abstraction*, *inheritance*, dan *association*. Dengan demikian, *UML Class Diagram* dapat digunakan sebagai pendekatan perancangan perangkat lunak yang efektif dalam pengembangan *game* pembelajaran berbasis Android.

Kata kunci: *Android, Game Pembelajaran, Object-Oriented Programming, Rekayasa Perangkat Lunak, UML Class Diagram.*

Abstract

Advances in information technology have influenced the development of interactive learning media, one of which is Android-based educational games. In educational game development, structured system design is an important aspect to ensure that software can be developed modularly and maintained efficiently. However, game development often focuses more on implementation without proper software design, causing program structures to become complex and difficult to maintain. This study aims to implement Object-Oriented Programming (OOP) concepts through the design of Unified Modeling Language (UML) Class Diagrams in an Android-based mathematics educational game. The research method was conducted through system requirements analysis, class diagram design, class relationship analysis, and implementation of OOP concepts in the game architecture. UML Class Diagrams were used to model classes, attributes, methods, object relationships, and parameter passing within the educational game system. The results show that the use of UML Class Diagrams helps create a more structured software design, improves system modularity, and facilitates the implementation of OOP concepts such as encapsulation, abstraction, inheritance, and association. Therefore, UML Class Diagrams can be used as an effective software design approach in the development of Android-based educational games.

Keywords: *Android, Educational Game, Object-Oriented Programming, Software Engineering, UML Class Diagram.*

I. PENDAHULUAN

Perkembangan teknologi informasi telah mendorong transformasi media pembelajaran menjadi lebih interaktif dan adaptif. Salah satu bentuk implementasinya adalah penggunaan *game* edukasi berbasis Android sebagai media pembelajaran yang mampu meningkatkan minat belajar dan interaksi pengguna dalam proses pembelajaran [1] Selain itu, penggunaan media pembelajaran berbasis *mobile* memberikan fleksibilitas bagi pengguna untuk mengakses materi pembelajaran kapan saja dan di mana saja [2]

Dalam pengembangan *game* edukasi berbasis Android, tidak hanya aspek visual dan *gameplay* yang perlu diperhatikan, tetapi juga struktur dan perancangan sistem perangkat lunak agar aplikasi dapat dikembangkan secara terorganisir dan mudah dipelihara [3] Perancangan sistem yang baik diperlukan untuk membantu pengembang dalam mengelola struktur aplikasi sehingga proses pengembangan fitur dapat dilakukan secara lebih efektif dan efisien [4]

Pendekatan *Object-Oriented Programming* (OOP) banyak digunakan dalam pengembangan perangkat lunak karena mampu membangun sistem yang modular, terstruktur, dan mudah dikembangkan kembali [[5]. Konsep OOP memungkinkan pengembang mengelompokkan data dan fungsi ke dalam objek sehingga proses pengembangan aplikasi menjadi lebih terarah dan efisien [6]

Untuk membantu proses perancangan sistem berbasis objek, digunakan Unified Modeling Language (UML) sebagai bahasa pemodelan standar dalam menggambarkan struktur dan perilaku sistem perangkat lunak [7] UML digunakan untuk memvisualisasikan hubungan antar komponen sistem sebelum proses implementasi dilakukan sehingga pengembang dapat memahami struktur sistem dengan lebih baik [8]

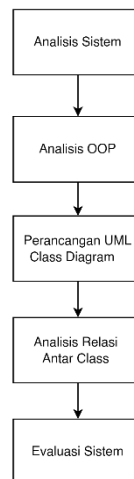
Salah satu diagram UML yang paling penting dalam pendekatan OOP adalah *class diagram*. *Class diagram* digunakan untuk merepresentasikan struktur class, atribut, *method*, serta hubungan antar *object* dalam sistem perangkat lunak [9] Dengan menggunakan *class diagram*, pengembang dapat memvisualisasikan struktur perangkat lunak secara lebih jelas sehingga mempermudah proses implementasi dan pengembangan aplikasi [10]

Namun, pada proses pengembangan *game* pembelajaran sering kali implementasi aplikasi lebih difokuskan pada hasil akhir tampilan dan *gameplay* dibandingkan perancangan struktur perangkat lunaknya. Akibatnya, dokumentasi sistem menjadi kurang terstruktur dan menyulitkan proses pengembangan lanjutan. Oleh karena itu, analisis UML *Class Diagram* menjadi penting untuk memahami implementasi *Object-Oriented Programming* pada *game* pembelajaran berbasis Android.

II. METODE PENELITIAN

Penelitian ini menerapkan pendekatan Object-Oriented Analysis and Design (OOAD) untuk menganalisis dan merancang struktur perangkat lunak pada *game* pembelajaran matematika berbasis Android. Pendekatan tersebut dipilih karena mampu memodelkan sistem berbasis objek secara modular melalui UML Class Diagram. Proses pengembangan dilakukan menggunakan Unity Engine dengan bahasa pemrograman C# serta integrasi Firebase Authentication dan Firebase Realtime Database.

Tahapan penelitian meliputi analisis sistem, analisis penerapan Object-Oriented Programming (OOP), perancangan UML Class Diagram, analisis relasi antar class, dan evaluasi sinkronisasi diagram terhadap source code implementasi.



Gambar 1. Metode Penelitian OOAD

2.1. Analisis Sistem

Analisis sistem dilakukan untuk mengidentifikasi modul utama pada *game* pembelajaran matematika yang dikembangkan. Sistem terdiri dari *Authentication System*, *Gameplay System*, *Result System*, *Level System*, *Profile System*, *Leaderboard System*, dan *Firebase System*. *Authentication System* menangani proses login, registrasi, reset password, dan logout pengguna. *Gameplay System* mengatur mekanisme permainan, interaksi object, validasi jawaban, dan timer permainan. *Result System* menangani evaluasi hasil permainan dan penyimpanan progres pemain. Selain itu, *Firebase System* digunakan untuk

otentikasi pengguna dan sinkronisasi data permainan secara *realtime*. Hasil analisis sistem digunakan sebagai dasar dalam proses perancangan struktur perangkat lunak berbasis *OOP*.

2.2. Analisis *OOP*

Analisis *OOP* dilakukan untuk mengidentifikasi penerapan konsep pemrograman berorientasi objek pada sistem permainan. Konsep *encapsulation* diterapkan menggunakan atribut *private* dan *method public* untuk membatasi akses data internal *class*. *Abstraction* diterapkan melalui pemisahan fungsi *gameplay*, autentikasi, dan *service* ke dalam *class* yang berbeda. Konsep *inheritance* diterapkan menggunakan *MonoBehaviour* sebagai *parent class* utama pada *Unity*, sedangkan *singleton pattern* digunakan pada beberapa *class* utama untuk memastikan *object* hanya memiliki satu *instance global* selama aplikasi berjalan. Penerapan *OOP* menghasilkan struktur sistem yang modular, fleksibel, dan mempermudah proses pengembangan perangkat lunak.

2.3. Perancangan *UML Class Diagram*

Perancangan *UML Class Diagram* dilakukan berdasarkan hasil analisis *source code Unity* menggunakan bahasa pemrograman *C#*. Diagram digunakan untuk merepresentasikan struktur *class*, *atribut*, *method*, *visibility*, serta relasi antar *class* pada sistem permainan. *Class* utama pada sistem terdiri dari *AuthManager*, *AuthService*, *FirebaseService*, *SaveService*, *ResultManager*, *LevelManager*, *LevelButton*, *ProfileManager*, *LeaderboardManager*, *PlayerLeaderboardData*, *Level1Manager*, *FenceSlot*, *FenceDrag*, dan *TimeManager*. Perancangan diagram juga memvisualisasikan relasi *dependency* dan *association* antar *class*. *Dependency* digunakan pada komunikasi antar *manager* dan *service* melalui pemanggilan *method*, sedangkan *association* digunakan pada *class* yang memiliki referensi *object* secara langsung. Hasil perancangan *UML Class Diagram* digunakan sebagai acuan sinkronisasi antara desain sistem dan implementasi *source code Unity*.

2.4. Analisis Relasi Antar *Class*

Analisis relasi dilakukan untuk mengidentifikasi komunikasi antar *object* pada sistem permainan. Relasi *dependency* diterapkan pada komunikasi antar *manager* dan *service*, sedangkan *association* digunakan pada *class* yang memiliki referensi *object* secara langsung. Selain itu, sistem juga menerapkan *singleton pattern* pada beberapa *class* utama untuk mempermudah akses global terhadap *object* penting dalam sistem permainan. Struktur relasi tersebut menghasilkan komunikasi *object* yang modular dan terorganisir.

2.5. Evaluasi Sistem

Evaluasi sistem dilakukan dengan membandingkan hasil perancangan *UML Class Diagram* terhadap implementasi *source code Unity* berbasis *C#*. Evaluasi mencakup kesesuaian struktur *class*, *atribut*, *method*, *visibility*, *dependency*, *association*, dan *singleton pattern*. Hasil evaluasi digunakan untuk memastikan bahwa desain *UML* telah sinkron dengan implementasi sistem sehingga struktur perangkat lunak yang dihasilkan tetap modular, terorganisir, dan sesuai dengan konsep *Object-Oriented Programming (OOP)*.

III. HASIL DAN PEMBAHASAN

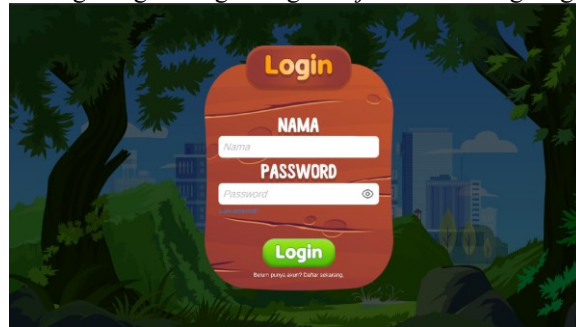
3.1 Implementasi *Object-Oriented Programming (OOP)*

Sistem *game* pembelajaran matematika dikembangkan menggunakan pendekatan modular berbasis *Object-Oriented Programming (OOP)* dengan pemisahan fungsi *gameplay*, *authentication*, *result system*, *level system*, dan *Firebase service* ke dalam *class* yang berbeda. Pendekatan tersebut membuat komunikasi antar *object* menjadi lebih terstruktur serta mempermudah proses pengembangan dan maintenance perangkat lunak.

Struktur sistem dibagi menjadi beberapa *manager* dan *service* utama. *Class manager* digunakan untuk mengatur alur permainan dan antarmuka pengguna, sedangkan *class service* menangani autentikasi pengguna, penyimpanan data, dan komunikasi *Firebase*. Implementasi tersebut terlihat pada penggunaan *AuthManager*, *Level1Manager*, *ResultManager*, dan *LevelManager* sebagai pengontrol sistem utama, sedangkan *AuthService*, *SaveService*, dan *FirebaseService* digunakan sebagai *service layer* pada aplikasi.

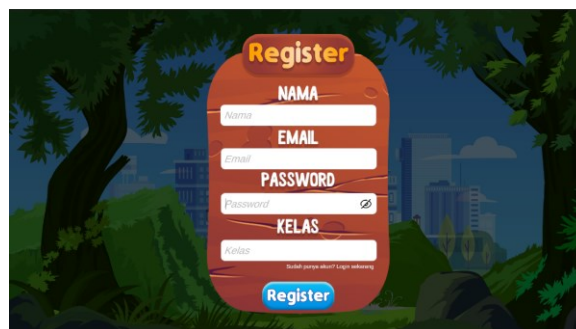
Selain itu, sistem juga menerapkan *singleton pattern* pada beberapa *class* utama seperti *FirebaseService*, *AuthService*, *SaveService*, *Level1Manager*, dan *TimeManager* untuk memastikan *object*

utama hanya memiliki satu instance global selama aplikasi berjalan. Penerapan *singleton* mempermudah komunikasi antar *class* dan mengurangi ketergantungan *object* secara langsung.



Gambar 2. Halaman Login

Gambar 2 menunjukkan implementasi halaman *login* pada *game* pembelajaran matematika berbasis *Android*. Halaman ini digunakan pengguna untuk melakukan autentikasi akun menggunakan *Firebase Authentication* sebelum memasuki sistem permainan. Proses autentikasi dikelola oleh *AuthManager* sebagai *controller* antarmuka dan *AuthService* sebagai *service* autentikasi yang terhubung dengan *Firebase Database*.



Gambar 3. Halaman Register

Gambar 3 menunjukkan implementasi halaman registrasi pengguna yang digunakan untuk membuat akun baru sebelum memasuki sistem permainan. Data pengguna yang berhasil didaftarkan akan tersimpan pada *Firebase Database* sehingga dapat digunakan untuk autentikasi dan penyimpanan progres permainan secara *realtime*.

Implementasi registrasi memanfaatkan komunikasi antara *AuthManager*, *AuthService*, dan *FirebaseService*. *AuthManager* menangani *input* pengguna pada antarmuka, sedangkan *AuthService* mengelola proses registrasi akun dan komunikasi data *Firebase*.



Gambar 4. Implementasi Gameplay *Puzzle* Matematika

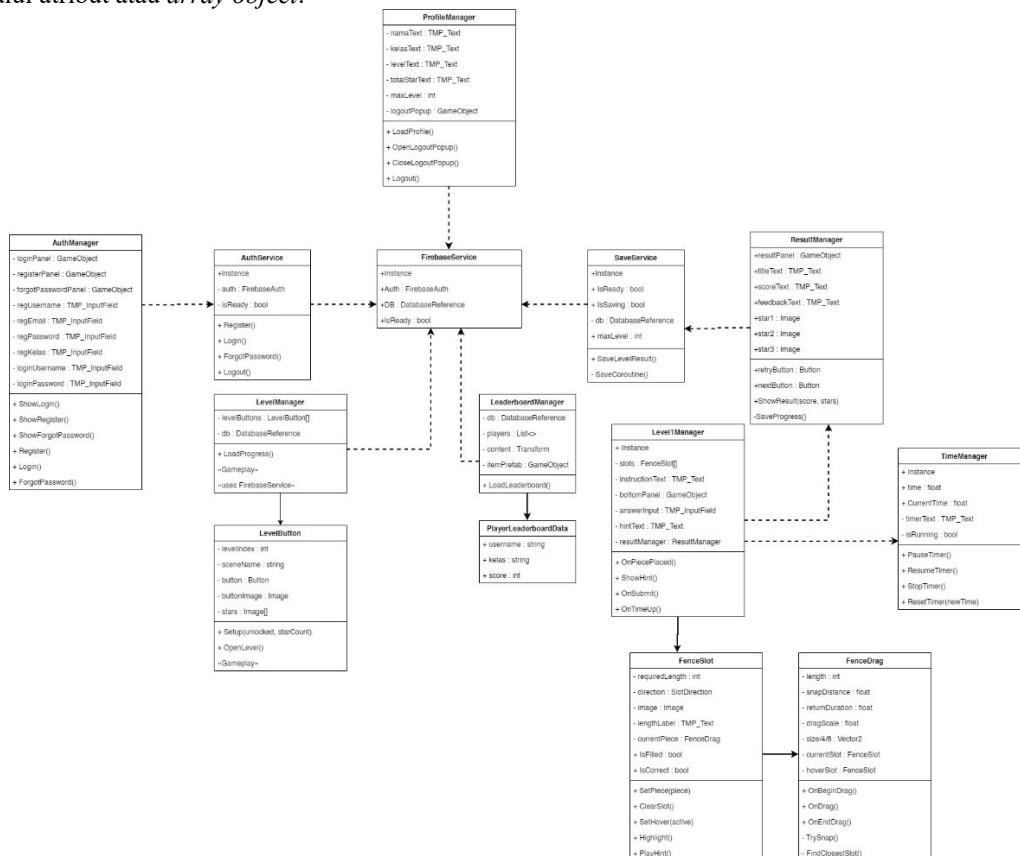
Gambar 4 menunjukkan implementasi *gameplay puzzle* matematika berbasis *drag and drop*. Sistem menampilkan area permainan, *inventory* object pagar, *timer* permainan, serta *input* jawaban yang digunakan pemain untuk menyelesaikan *puzzle* matematika.

Gameplay dikendalikan oleh *LevelManager* sebagai *controller* utama yang mengatur evaluasi jawaban, pengelolaan *score*, dan komunikasi antar *object gameplay*. Selain itu, *FenceSlot* digunakan untuk menangani area penempatan *puzzle*, *FenceDrag* mengatur interaksi *drag and drop object*, sedangkan *TimeManager* menangani *countdown* timer selama permainan berlangsung.

Hasil implementasi menunjukkan bahwa penerapan *OOP* berhasil menghasilkan struktur sistem yang modular dan terorganisir. Pemisahan fungsi antar *manager* dan *service* membuat komunikasi *object* menjadi lebih fleksibel sehingga proses pengembangan *gameplay*, autentikasi pengguna, dan penyimpanan progres permainan dapat berjalan secara terintegrasi.

3.2 Implementasi UML Class Diagram

Implementasi *UML Class Diagram* dilakukan berdasarkan struktur *source code Unity* yang telah dikembangkan menggunakan bahasa pemrograman *C#*. Diagram digunakan untuk merepresentasikan struktur *class*, *atribut*, *method*, *visibility*, serta relasi antar *class* pada sistem permainan. *Class* utama pada sistem terdiri dari *AuthManager*, *AuthService*, *FirebaseService*, *SaveService*, *ResultManager*, *LevelManager*, *LevelButton*, *ProfileManager*, *LeaderboardManager*, *PlayerLeaderboardData*, *Level1Manager*, *FenceSlot*, *FenceDrag*, dan *TimeManager*. Relasi antar *class* terdiri dari *dependency* dan *association*. *Dependency* digunakan pada komunikasi antar *manager* dan *service* melalui pemanggilan *method*, sedangkan *association* digunakan pada *class* yang memiliki referensi *object* secara langsung melalui atribut atau *array object*.



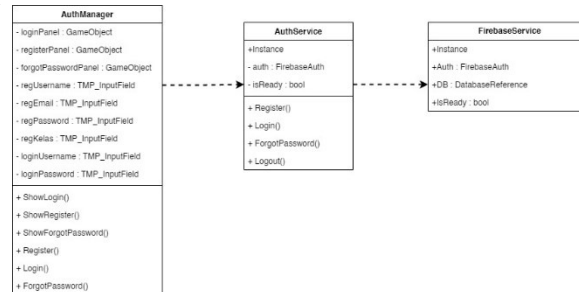
Gambar 5. UML Class Diagram Keseluruhan Sistem

Gambar 5 menunjukkan struktur keseluruhan sistem *game* pembelajaran matematika berbasis *Android*. Diagram memvisualisasikan hubungan antar *manager*, *gameplay object*, *service layer*, serta *integrasi Firebase* pada sistem permainan. Berdasarkan hasil implementasi, struktur perangkat lunak berhasil dibagi menjadi beberapa *modul* utama yaitu *Authentication Module*, *Gameplay Module*, *Level*

Module, serta Profile dan Leaderboard Module. Pemisahan modul dilakukan untuk mempermudah komunikasi *object*, pengembangan fitur, dan *maintenance* sistem.

3.2.1 Authentication Module

Authentication Module digunakan untuk menangani proses autentikasi pengguna seperti *login*, *registrasi*, *reset password*, dan *logout*. Modul ini terdiri dari *AuthManager*, *AuthService*, dan *FirebaseService*. *AuthManager* berfungsi sebagai *controller* antarmuka pengguna, sedangkan *AuthService* menangani proses autentikasi dan komunikasi *Firebase Authentication*. Selain itu, *FirebaseService* menyediakan akses global terhadap *Firebase Authentication* dan *Firebase Realtime Database* menggunakan *singleton pattern*.

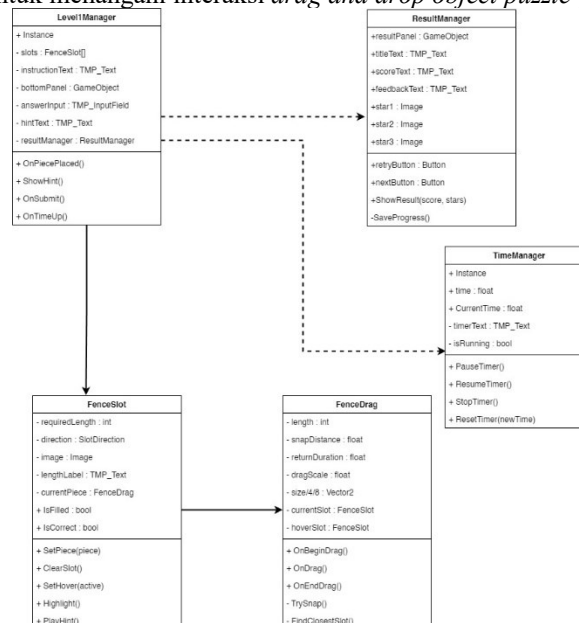


Gambar 6. Authentication Module

Gambar 6 menunjukkan relasi antar class pada *Authentication Module*. Relasi *dependency* digunakan pada komunikasi antara *AuthManager* dan *AuthService*, sedangkan *AuthService* berkomunikasi dengan *FirebaseService* untuk proses autentikasi pengguna dan sinkronisasi data *Firebase*. Hasil implementasi menunjukkan bahwa pemisahan manager dan *service* membuat proses autentikasi lebih modular dan mempermudah pengelolaan komunikasi data pada sistem.

3.2.2 Gameplay Module

Gameplay Module merupakan modul utama pada sistem permainan yang menangani mekanisme *gameplay*, interaksi *puzzle*, evaluasi jawaban, dan *timer* permainan. Modul ini terdiri dari *LevelManager*, *FenceSlot*, *FenceDrag*, *ResultManager*, dan *TimeManager*. *LevelManager* berfungsi sebagai *controller* utama yang mengatur *gameplay* dan komunikasi antar *object* permainan. Selain itu, *FenceSlot* dan *FenceDrag* digunakan untuk menangani interaksi *drag and drop object puzzle* pada area permainan.

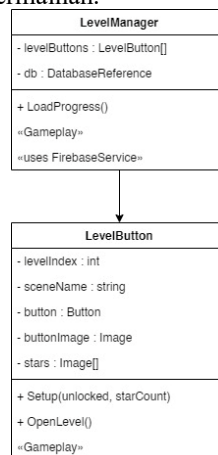


Gambar 7. Gameplay Module

Gambar 7 menunjukkan struktur *class* dan relasi pada *Gameplay Module*. *Association* digunakan pada relasi *LevelManager* terhadap *FenceSlot*, serta *FenceSlot* terhadap *FenceDrag*. Selain itu, *dependency* diterapkan pada komunikasi *LevelManager* terhadap *ResultManager* dan *TimeManager* untuk proses evaluasi permainan dan pengelolaan timer. Berdasarkan hasil implementasi, *Gameplay Module* berhasil menerapkan struktur modular berbasis *object-oriented* sehingga komunikasi antar *object gameplay* menjadi lebih terorganisir dan fleksibel.

3.2.3 Level Module

Level Module digunakan untuk mengatur sistem pembukaan level berdasarkan progres permainan pemain. Modul ini terdiri dari *LevelManager* dan *LevelButton*. *LevelManager* mengelola proses pembacaan progres permainan dari *Firestore Database*, sedangkan *LevelButton* digunakan untuk mengatur status level seperti *unlock* dan *lock* pada antarmuka permainan.

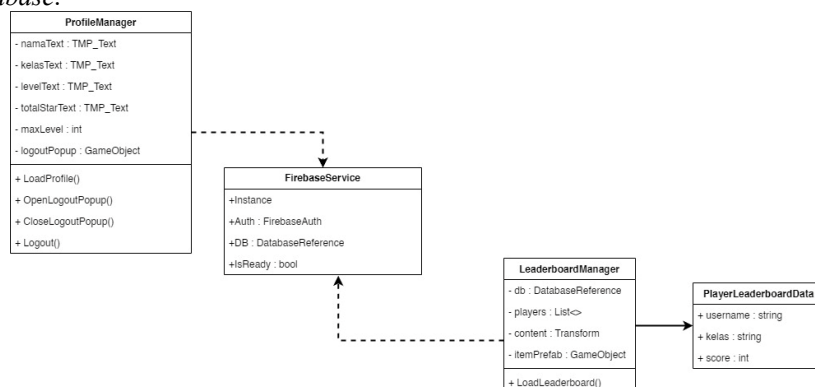


Gambar 8. Level Module

Gambar 8 menunjukkan relasi *association* antara *LevelManager* dan *LevelButton[]*. Relasi tersebut digunakan untuk mengelola *object* level pada sistem permainan berdasarkan progres pemain. Implementasi *Level Module* membantu proses pengelolaan level menjadi lebih modular dan mempermudah pengembangan fitur level pada sistem permainan.

3.2.4 Profile dan Leaderboard Module

Profile dan *Leaderboard Module* digunakan untuk mengelola data profil pengguna dan menampilkan peringkat pemain berdasarkan score permainan. Modul ini terdiri dari *ProfileManager*, *LeaderboardManager*, *PlayerLeaderboardData*, dan *FirestoreService*. *ProfileManager* digunakan untuk menampilkan informasi profil pemain seperti nama, kelas, level, dan total bintang. Selain itu, *LeaderboardManager* digunakan untuk mengambil dan menampilkan data *leaderboard* pemain dari *Firestore Database*.



Gambar 9. Profile dan Leaderboard Module

Gambar 9 menunjukkan implementasi relasi antar *class* pada *Profile* dan *Leaderboard Module*. *Dependency* digunakan pada komunikasi antara *ProfileManager* dan *LeaderboardManager* terhadap *FirebaseService*. Selain itu, *association* diterapkan pada relasi *LeaderboardManager* terhadap *PlayerLeaderboardData* untuk menyimpan data *score* pemain pada sistem *leaderboard*. Hasil implementasi menunjukkan bahwa pemisahan *Profile* dan *Leaderboard Module* membantu pengelolaan data pemain menjadi lebih terstruktur dan sinkron dengan *Firebase Database*.

3.2.5 Analisis

Passing	Class Pengirim	Class Penerima	Parameter	Fungsi
	LevelManager	LevelButton	unlocked, starCount	Status level pemain

Parameter Antar Class

Passing parameter antar *class* digunakan sebagai mekanisme komunikasi *object* pada sistem permainan. Implementasi *passing parameter* dilakukan melalui pemanggilan *method* dan pengiriman data antar *manager*, *gameplay object*, dan *service layer*. Pendekatan tersebut membuat komunikasi sistem menjadi lebih modular dan mengurangi ketergantungan antar *object* secara langsung. Pada *Authentication Module*, *AuthManager* mengirimkan parameter berupa *username*, *email*, *password*, dan kelas pengguna ke *AuthService* untuk proses *login* dan registrasi akun. Selanjutnya, *AuthService* meneruskan data autentikasi tersebut ke *FirebaseService* untuk diproses menggunakan *Firebase Authentication*.

Tabel 1. Menunjukkan implementasi *passing parameter* pada *Authentication Module*.

Class Pengirim	Class Penerima	Parameter	Fungsi
AuthManager	AuthService	username, password	Proses login
AuthManager	AuthService	username, email, password, kelas	Registrasi akun
AuthService	FirebaseService	user data	Autentikasi Firebase

Pada *Gameplay Module*, *LevelManager* mengirimkan *parameter gameplay* ke *FenceSlot*, *ResultManager*, dan *TimeManager*. *Passing parameter* digunakan untuk mengatur *object puzzle*, evaluasi jawaban, *score* permainan, dan *timer gameplay*.

Tabel 2. Menunjukkan implementasi *passing parameter* pada *Gameplay Module*.

Class Pengirim	Class Penerima	Parameter	Fungsi
LevelManager	FenceSlot	piece	Penempatan <i>object puzzle</i>
FenceSlot	FenceDrag	currentPiece	Referensi <i>object drag</i>
LevelManager	ResultManager	score, stars	Menampilkan hasil permainan
LevelManager	TimeManager	newTime	Reset timer permainan

Pada *Level Module*, *LevelManager* mengirimkan *parameter* status level ke *LevelButton* untuk mengatur sistem unlock level berdasarkan progres pemain.

Tabel 3. Menunjukkan *passing parameter* pada *Level Module*.

Selain itu, *passing parameter* juga diterapkan pada proses penyimpanan data permainan menggunakan *SaveService* dan *FirebaseService*. Data *score*, level, dan progres pemain dikirimkan untuk disimpan pada *Firebase Realtime Database*.

Tabel 4. Menunjukkan *passing parameter* pada *Firebase Module*.

Class Pengirim	Class Penerima	Parameter	Fungsi
ResultManager	SaveService	score, stars	Penyimpanan hasil permainan
SaveService	FirebaseService	level data	Sinkronisasi Firebase
LeaderboardManager	PlayerLeaderboardData	username, kelas, score	username, kelas, score

Berdasarkan hasil implementasi, *passing parameter* antar *class* berhasil mendukung komunikasi *object* secara modular dan terstruktur. Pendekatan tersebut juga mempermudah sinkronisasi data antar *manager*, *gameplay object*, dan *Firebase service* pada sistem permainan.

IV. KESIMPULAN

Berdasarkan hasil penelitian, implementasi *Object-Oriented Programming (OOP)* pada *game* pembelajaran matematika berbasis Android berhasil diterapkan menggunakan struktur perangkat lunak yang modular dan terorganisir. *UML Class Diagram* berhasil merepresentasikan struktur *class*, *atribut*, *method*, serta relasi antar *class* sesuai dengan implementasi *source code Unity* berbasis C#. Hasil pengujian menunjukkan bahwa relasi *dependency*, *association*, dan *singleton pattern* berhasil diimplementasikan secara sinkron pada sistem permainan. Selain itu, pembagian sistem ke dalam beberapa modul seperti *authentication*, *gameplay*, *level*, dan *leaderboard* membantu proses pengembangan dan *maintenance* perangkat lunak menjadi lebih fleksibel dan terstruktur. Dengan demikian, penerapan *OOP* menggunakan *UML Class Diagram* berhasil menghasilkan arsitektur perangkat lunak *game* pembelajaran matematika berbasis *Android* yang sesuai dengan implementasi sistem dan mendukung pengembangan perangkat lunak secara modular.

V. UCAPAN TERIMA KASIH

Penulis mengucapkan puji syukur kepada Tuhan Yang Maha Esa atas segala rahmat dan karunia-Nya sehingga penelitian ini dapat diselesaikan dengan baik. Penulis juga mengucapkan terima kasih kepada dosen pembimbing yang telah memberikan arahan, bimbingan, serta masukan selama proses penyusunan penelitian ini. Ucapan terima kasih turut disampaikan kepada keluarga dan teman-teman yang telah memberikan dukungan, motivasi, dan bantuan selama proses pengembangan sistem dan penulisan artikel berlangsung. Selain itu, penulis juga berterima kasih kepada seluruh pihak yang telah mendukung penelitian mengenai implementasi *Object-Oriented Programming (OOP)* dan *UML Class Diagram* pada *game* pembelajaran matematika berbasis Android ini.

REFERENSI

- [1] F. Sadewa, I. Kuantan Singingi, I. K. Jl Gatot Subroto, K. Nenas, D. Jake, and K. Kuantan Singingi, "PERANCANGAN APLIKASI GAME EDUKASI UNTUK MATERI ALJABAR BERBASIS ANDROID (STUDI KASUS: SMP 3 SENTAJO RAYA)," vol. 4, no. 1, pp. 103–113, 2021, doi: <https://doi.org/10.36378/jupersatek.v4i1.1520>.
- [2] K. Aisah and H. Yanto, "PERANCANGAN SISTEM INFORMASI APLIKASI E-LEARNING BERBASIS WEB DI SMA N 9 PADANG," *Jurnal KomtekInfo*, vol. 8, no. 1, 2021, doi: 10.35134/komtekinfo.v7i4.
- [3] A. Gilang Prakoso and F. Bambang Sukoco, "Jurnal Manajemen Teknologi dan Sistem Informasi (JMS) Perancangan Game Edukasi Pramuka 2D Berbasis Android (Studi Kasus SDS Adhyaksa 1 Kota Jambi)," doi: 10.33998/jms.v4i2.
- [4] Indah Nurlita and Reni Anggraini, "Analysis and Design of Incoming and Outgoing Cash Accounting Information Systems at Kilometer 28 Laundry using the Pieces and Waterfall Methods with Unified Modeling Language (Uml) Tools," *Formosa Journal of Applied Sciences*, vol. 2, no. 6, pp. 1065–1090, Jun. 2023, doi: 10.55927/fjas.v2i6.4411.
- [5] M. Rahmawati, A. Kusniawati, and R. K. Lesmana, "Design and Development of Ceramic Information System Based on Object Oriented Programming," *JOURNAL OF INFORMATICS AND TELECOMMUNICATION ENGINEERING*, vol. 4, no. 1, pp. 105–111, Jul. 2020, doi: 10.31289/jite.v4i1.3819.
- [6] A. Samsudin and H. Raif Hakim, "Sistem Informasi Manajemen Masjid Menggunakan Metode Rapid Application Development Berbasis Website," 2024.
- [7] Siska Narulita, Ahmad Nugroho, and M. Zakki Abdillah, "Diagram Unified Modelling Language (UML) untuk Perancangan Sistem Informasi Manajemen Penelitian dan Pengabdian Masyarakat (SIMLITABMAS)," *Bridge : Jurnal publikasi Sistem Informasi dan Telekomunikasi*, vol. 2, no. 3, pp. 244–256, Aug. 2024, doi: 10.62951/bridge.v2i3.174.
- [8] A. U. Dewi and A. Voutama, "IMPLEMENTASI UML DALAM PERANCANGAN SISTEM INFORMASI KEUANGAN MASJID AL-AMANAH BERBASIS WEB," 2024.
- [9] V. Aulia, M. Adriyani, M. Wahyuda, G. Langi, and S. Ria, "Implementasi Diagram UML (Unified Modelling Language) Pada Perancangan Sistem Informasi Laporan Persediaan Barang," *Jurnal Ilmu Komputer dan Bisnis*, vol. 12, no. 2, pp. 91–98, Nov. 2021, doi: 10.47927/jikb.v12i2.145.
- [10] M. Iqbal and R. Lia Andharsaputri, "Implementasi UML Untuk Perancangan Sistem Informasi Pengadaan Barang Pada RSUD Kota Bogor."