BAB II
TINJUAN PUSTAKA

1.1. Game

Permainan elektronik, atau *game*, adalah bentuk aktivitas bermain yang menghadirkan konteks yang bukan bagian dari realitas, mengajak pemainnya untuk mencapai setidaknya satu tujuan atau misi dengan berperan sesuai aturan dalam permainan. Permainan juga termasuk dalam kategori media yang bisa digunakan untuk melatih dan mengembangkan kemampuan berpikir, seperti meningkatkan fokus, memperkuat daya ingat, serta melatih logika dalam menyelesaikan masalah. Selain itu, permainan juga dapat digunakan sebagai alat untuk menyampaikan informasi kepada pemain dengan cara yang menarik [8]. Meskipun awalnya lebih populer di kalangan anak-anak, saat ini orang dewasa juga aktif bermain game dan mengikuti perkembangan game yang ada. Jenis game yang tersedia bervariasi sesuai dengan perkembangan teknologi zaman. Jika dilihat dari segi grafis, aplikasi permainan dapat dikategorikan menjadi dua jenis, yaitu *game* 2D yang menggunakan grafis dua dimensi dan *game* 3D yang menggunakan grafis tiga dimensi.

Kemajuan teknologi dan komputer telah membuat *game* menjadi lebih realistis dan kompleks dalam segi grafis, *gameplay*, dan cerita. *Game* memiliki potensi untuk memberikan pengalaman yang berbeda dan mendalam, seperti pengalaman belajar, meningkatkan keterampilan sosial, dan merangsang kreativitas [9]. Untuk merancang game, diperlukan pemahaman yang kuat tentang pengembangan game dan elemen-elemennya, seperti grafis, suara, interaksi, dan alur cerita. Ada banyak platform dan engine game yang tersedia, termasuk Unity 3D dan Blender, yang memungkinkan pengembang game untuk membuat game yang unik dan menarik.

Game edukasi merupakan salah satu jenis media yang digunakan untuk memberikan pengajaran dan meningkatkan pengetahuan penggunanya melalui penggunaan media yang unik dan menarik. *Game* edukasi diciptakan dengan tujuan

spesifik sebagai alat pendidikan, untuk membantu pengguna belajar mengenal warna, huruf, dan bahasa asing. *Game* dengan tujuan edukasi seperti ini dapat menjadi salah satu media pembelajaran yang menerapkan pola "belajar dengan melakukan" (learning by doing). Dalam pola ini, pemain diharapkan untuk belajar melalui pengalaman bermain *game* dan menyelesaikan tantangan yang ada. Melalui status game, instruksi, dan alat yang disediakan, *game* tersebut secara aktif membimbing pemain dalam mencari informasi dan mengembangkan pengetahuan serta strategi saat bermain.

2.1.1. Perspective dan Viewpoints Game

Dari kategori permainan digital yang ada seperti *first person perspective* dan *third person perspective*, kategori permainan yang akan dibuat yaitu *third person perspective.* Sudut pandang orang ketiga memiliki arti yaitu perspektif gambar yang akan ditampilkan adalah sudut pandang ketiga / orang lain dari Tokoh yang dimainkan oleh pemain[10], [11]. Sehingga kategori ini membutuhkan game yang memiliki bentuk seperti dunia nyata atau simulasi yang terdapat hukum fisika, gerak karakter, kecepatan karakter dan lain-lain. Jenis permainan ini cukup banyak disukai, dikarenakan ketangkasan bergerak dan sesuatu yang tidak bisa pemain lakukan di dunia nyata.

2.1.2. Genre Game

Berikut ini beberapa pengelompokan jenis game berdasarkan genre-nya diantaranya adalah :

1. *Action*, Mekanik utama game action berkisar pada akurasi, pergerakan, kecepatan mengambil keputusan, reflek dan timing.
2. *Adventure*, *Game* petualangan menekankan pengalaman cerita melalui dialog dan pemecahan teka-teki. Mekanisme *gameplay* menekankan pada pengambilan keputusan atas tindakan. Pemecahan teka-teki biasanya berkisar pada menggabungkan atau memanipulasi item untuk menjalankan cerita.
3. *Role-Playing (RPG)*, *Game Role-Playing (RPG)* termasuk dalam genre *game* yang bervariasi yang berfokus pada pengembangan karakter.

4. *Simulation*, Terdapat banyak jenis *game* simulasi. Kesamaan dari semua simulasi adalah *game* simulasi dimodelkan secara realistis dengan situasi dan/atau variabel kehidupan nyata daripada genre *game* yang lainnya.

Strategy, *Game* strategi berkisar pada penggunaan sumber daya strategis dan/atau taktis sering kali dalam skenario pertempuran atau manajerial [12].

2.1.3. Non-Player Character

Non-Player Character (NPC) adalah karakter dalam permainan yang dikendalikan oleh komputer atau sistem, bukan oleh pemain. Dalam game edukasi Sangkuriang yang Anda rancang, NPC dapat mewakili karakter-karakter dalam cerita rakyat Sangkuriang yang berinteraksi dengan pemain. NPC ini dapat berperan sebagai karakter seperti Sangkuriang, Dayang Sumbi, Tumang, atau karakter lain yang relevan dalam cerita rakyat tersebut. NPC dapat memberikan informasi tentang cerita rakyat, memberikan petunjuk atau hint, atau berinteraksi dengan pemain dalam dialog atau percakapan. Karakter ini dibutuhkan untuk menambahkan prilaku yang dapat diimplementasikan seperti bergerak pada titik tertentu, berinteraksi dengan pemain, dan lain-lain [13].

1.2. Cerita Rakyat

Cerita rakyat merujuk pada cerita yang diturunkan dari generasi ke generasi melalui lisan atau tulisan dan sering mengandung pesan moral dan nilai-nilai budaya yang dianggap penting oleh masyarakat. Cerita rakyat merupakan sarana yang efektif untuk memperkenalkan nilai-nilai budaya kepada generasi muda, dan juga menarik karena kerap mengisahkan peristiwa yang magis dan fantasi. Karena itu, cerita rakyat sering digunakan sebagai dasar untuk membuat buku anak-anak, film, atau game edukasi. Namun, dalam merancang game edukasi berbasis cerita rakyat, diperlukan perhatian yang mendalam terhadap bagaimana mengintegrasikan cerita rakyat dengan gameplay yang menarik dan mendidik, sehingga dapat meningkatkan minat dan pemahaman generasi muda tentang budaya dan cerita rakyat [14].

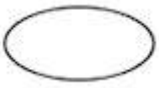
1.3. UML

UML adalah sebuah bahasa visual yang digunakan untuk memodelkan, mendokumentasikan, dan merancang perangkat lunak. Sub bab ini akan menjelaskan konsep dasar UML, tujuan penggunaannya, dan komponen-komponen utama yang ada dalam UML. Anda akan mempelajari bahwa UML terdiri dari berbagai jenis diagram yang dapat digunakan untuk merepresentasikan berbagai aspek perangkat lunak [15].

1.3.1. Use Case Diagram

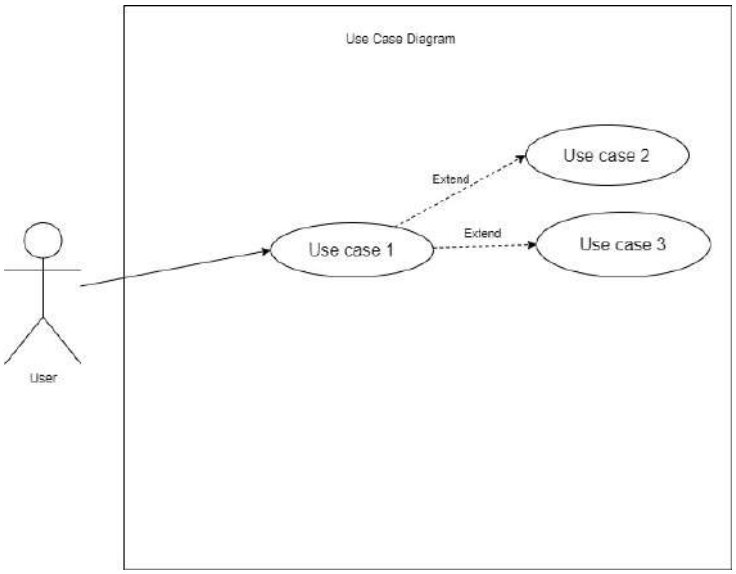
Use Case Diagram adalah jenis diagram UML yang digunakan untuk menggambarkan interaksi antara pengguna (aktor) dan sistem perangkat lunak. Sub bab ini akan menjelaskan bagaimana membuat *Use Case Diagram*, termasuk komponen-komponen yang ada dalam diagram ini, seperti aktor, use case, hubungan antara aktor dan use case, serta bagaimana menggambarkan alur dari use case.

Tabel 2.1. Simbol Use Case Diagram

No	GAMBAR	NAMA	KETERANGAN
1	*Simbol 2.1. Sistem Use Case Diagram* 	Sistem	Sistem mewakili aplikasi atau sistem yang sedang dikembangkan dan dijelaskan dalam diagram use case.
2	*Simbol 2.2. Aktor Use Case Diagram* 	Aktor	Entitas eksternal yang berinteraksi dengan sistem dan terlibat dalam satu atau lebih skenario use case
3	*Simbol 2.3. Use Case Use Case Diagram* 	Use Case	Kumpulan dari satu atau lebih langkah-langkah yang menggambarkan interaksi antara aktor dan sistem untuk mencapai tujuan tertentu.

4	*Simbol 2.4. Asosiasi Use Case Diagram*	Asosiasi	Menghubungkan aktor dengan use case yang mereka terlibat

Berikut adalah contoh gambar dari use case diagram pada Gambar 2.1.



Gambar 2.1. Use Case Diagram

1.3.2. Activity Diagram

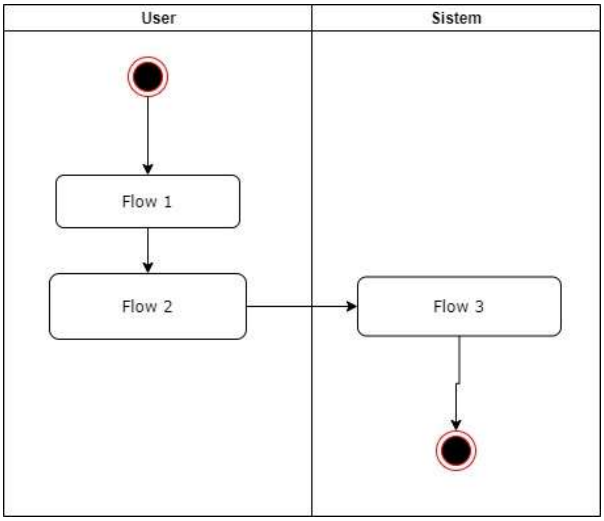
Activity Diagram adalah jenis diagram UML yang digunakan untuk menggambarkan alur atau urutan aktivitas dalam suatu proses atau fungsi dalam perangkat lunak. Sub bab ini akan menjelaskan bagaimana membuat Activity Diagram, termasuk simbol-simbol yang digunakan dalam diagram ini, seperti aktivitas, keputusan, garis aliran, dan bagaimana menggambarkan kontrol alur dalam suatu proses [16].

Tabel 2.2. Simbol Activity Diagram

No	Gambar	Nama	Keterangan
1	*Simbol 2.5. Status Awal Activity diagram* 	Status Awal	Titik awal di mana aktivitas dimulai.

2	*Simbol 2.6. Status Akhir Activity diagram* 	Status Akhir	Titik akhir di mana aktivitas selesai.
3	*Simbol 2.7. Aktivitas Activity diagram* 	Aktivitas	Langkah atau tindakan yang dilakukan dalam proses.
4	*Simbol 2.8. Percabangan Activity Diagram* 	Percabangan	Keputusan yang diambil berdasarkan kondisi tertentu, mengarahkan aktivitas ke jalur berbeda.
5	*Simbol 2.9. Transition Activity Diagram* 	Transition	ubungan antara status atau aktivitas yang menunjukkan aliran perpindahan dari satu langkah ke langkah berikutnya dalam diagram.

Berikut adalah contoh gambar dari Activity Diagram pada Gambar 2.2.

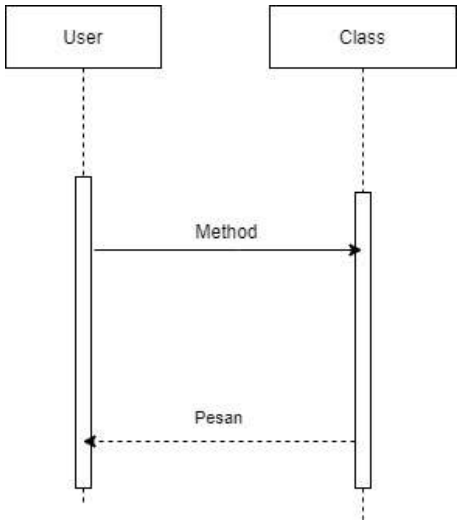


Gambar 2.2. Activity Diagram

1.3.3. Sequence Diagram

Sequence Diagram adalah jenis diagram UML yang digunakan untuk menggambarkan urutan pesan yang dikirim antara objek dalam sistem perangkat lunak. Sub bab ini akan menjelaskan bagaimana membuat

Sequence Diagram, termasuk simbol-simbol yang digunakan dalam diagram ini, seperti objek, pesan, urutan eksekusi, dan bagaimana menggambarkan interaksi antara objek dalam suatu skenario.

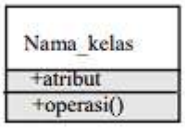


Gambar 2.3. Sequence Diagram

1.3.4. Class Diagram

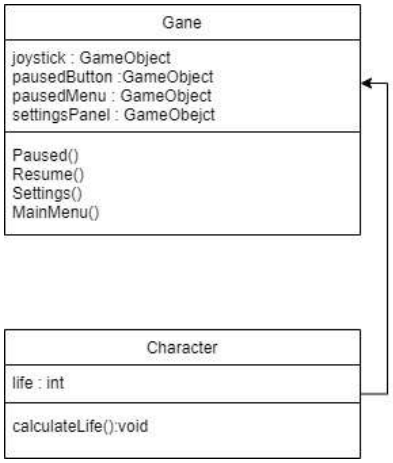
Class Diagram adalah jenis diagram UML yang digunakan untuk menggambarkan struktur kelas, atribut, dan hubungan antara kelas dalam suatu sistem perangkat lunak. Sub bab ini akan menjelaskan bagaimana membuat Class Diagram, termasuk simbol-simbol yang digunakan dalam diagram ini, seperti kelas, atribut, metode, hubungan antar kelas, dan bagaimana menggambarkan sifat-sifat kelas dalam diagram [17].

Tabel 2.3. Simbol Class Diagram

No	GAMBAR	NAMA	KETERANGAN
1	*Simbol 2.10. Kelas Class Diagram* 	Kelas	Representasi visual dari sebuah objek atau entitas dalam sistem yang memiliki atribut dan metode terkait
2	*Simbol 2.11. Asosiasi Class Diagram*	Asosiasi	Hubungan antara dua atau lebih kelas yang menunjukkan keterkaitan

			atau interaksi di antara mereka
3	*Simbol 2.12. Asosiasi yang diarahkan Class Diagram* 	Asosiasi yang diarahkan	Hubungan antara dua kelas di mana salah satu kelas memainkan peran yang lebih aktif dalam hubungan daripada yang lain.
4	*Simbol 2.13. Aggregation Class Diagram* 	Aggregation	Relasi antar kelas dengan makna semua- bagian (whole- part)
5	*Simbol 2.14. Generalisasi Class Diagram* 	Generalisasi	Hubungan di antara kelas di mana satu kelas merupakan jenis umum yang mewakili karakteristik bersama dari beberapa kelas khusus

Berikut adalah contoh dari Class Diagram yang dapat dilihat pada Gambar 2.4.



Gambar 2.4. Class Diagram

1.4. Unity 3D

Gambar 2.5. Unity

Unity adalah sebuah Game Engine yang telah berkembang menjadi Integrated Development Environment (IDE) atau alat pengembangan yang cepat. Sebagai contoh lain dari Game Engine, ada juga Unreal Engine dan CryENGINE 3, serta banyak lainnya seperti Torsi, Lumberyard, Ogre3D, Blender, JavaFX, dan lain sejenisnya. Jika memprogram game dari awal tanpa menggunakan Game Engine yang sudah ada, harus mengkodekan setiap detail secara manual, yang dapat memakan waktu dan sumber daya yang besar. Dalam hal ini, Unity berperan sebagai jembatan antara kode dan perangkat yang ditargetkan. Unity menangani kompilasi game dengan mengompresi aset dan mengubahnya ke format yang sesuai untuk platform distribusi. Dengan menggunakan Unity, dapat mengembangkan game dengan kompleksitas tinggi tanpa harus khawatir tentang hal-hal teknis seperti rendering, fisika, dan pencahayaan, karena Unity mengambil alih tugas-tugas tersebut. Kelebihan penggunaan Unity sebagai IDE adalah kemudahan penggunaannya, dengan antarmuka yang ramah pengguna, memungkinkan elemen-elemen dapat dipindahkan dengan mudah di sekitar layar. Unity juga mengintegrasikan berbagai alat komprehensif untuk developeran, memungkinkan untuk melihat dan memodifikasi berbagai aspek dari program, seperti tampilan scene, hierarki elemen dalam scene (GameObject), detail item yang sedang difokuskan, folder aset, dan lain sebagainya. Dalam perkembangannya, sebagian besar pengembang, termasuk studio besar, menggunakan IDE yang sudah ada seperti Unity atau Unreal, karena membuat mesin permainan khusus memerlukan waktu dan usaha yang lebih besar tanpa memberikan manfaat nyata [18].

Game engine menyediakan beragam fitur yang berguna untuk berbagai jenis permainan, memungkinkan permainan yang diimplementasikan menggunakan mesin tersebut mendapatkan fitur-fitur tersebut, serta menambahkan aset seni dan kode permainan khusus yang dibutuhkan. Unity, sebagai contoh game engine, memiliki simulasi fisika, peta normal, oklusi ambien ruang layar (SSAO), bayangan dinamis, dan banyak fitur lainnya. Dibandingkan dengan alat developeran game lainnya, Unity memiliki keunggulan dalam alur kerja visual yang sangat produktif dan dukungan lintas platform tingkat tinggi. Alur kerja visual dalam Unity menggunakan editor visual yang canggih dan unik, yang memungkinkan penataan scene dalam game dan penggabungan aset seni dan kode menjadi objek interaktif. Keunggulan editor ini mempercepat pembuatan game berkualitas profesional dengan efisiensi tinggi dan memanfaatkan teknologi terbaru dalam video game. Unity juga menonjol karena dukungannya yang kaya untuk lintas platform, memungkinkan pengembangan game untuk berbagai platform, termasuk PC, web, seluler, dan konsol, serta kompatibilitas dengan sistem operasi Windows dan Mac OS. Kemampuan platformagnostik ini berasal dari awalnya Unity dirancang untuk Mac dan kemudian di-porting ke Windows, dan terus berkembang untuk mendukung berbagai platform penyebaran, termasuk perangkat mobile dan konsol game . Selain itu, Unity menggunakan sistem komponen modular yang memungkinkan objek game dibuat melalui komposisi komponen, bukan melalui pewarisan kelas yang ketat. Dengan pendekatan ini, pembuatan prototipe cepat menjadi lebih mudah karena dapat dengan cepat mencampur dan mencocokkan komponen untuk menciptakan objek yang berbeda.

Namun, Unity juga memiliki beberapa kelemahan, seperti kompleksitas kombinasi editor visual dan pengkodean yang bisa menimbulkan kesulitan dalam mengelola scene yang kompleks. Selain itu, Unity tidak mendukung penautan di pustaka kode eksternal, memerlukan perpustakaan untuk disalin manual ke setiap proyek yang membutuhkannya, dan mengedit prefab bisa menjadi hal yang agak aneh dan rumit. Meskipun memiliki kelemahan, Unity tetap menjadi pilihan populer untuk pengembangan game karena keunggulan-keunggulannya, dan selain itu, Unity tersedia sepenuhnya gratis untuk digunakan, dengan beberapa pilihan

akun berbayar yang menawarkan fitur tambahan dan batasan berbeda bagi pengembang yang membutuhkannya [19].

1.5. Blender

Gambar 2.6. Blender

Blender adalah perangkat lunak open source untuk desain grafis 3D, animasi, dan rendering yang sering digunakan untuk membuat animasi, film, dan game. Dikenal sebagai alat serbaguna dan bertenaga, Blender menawarkan beragam fitur untuk menciptakan model 3D, animasi, simulasi fisika, dan bahkan permainan. Dalam dunia pengembangan permainan, Blender kerap digunakan untuk membuat elemen seperti objek, karakter, dan lingkungan permainan. Didukung oleh antarmuka yang intuitif, Blender memungkinkan pengguna untuk berkreasi dengan lebih mudah, khususnya dalam menciptakan lingkungan virtual yang menarik. Konsep *low poly* dan *high poly* dalam pembuatan model 3D mengacu pada tingkat kompleksitas geometri suatu objek. Model *low poly* memiliki sedikit titik atau vertex, mengonsumsi daya pemrosesan dan memori yang lebih rendah. Biasanya digunakan untuk objek-objek kecil atau latar belakang. Sebaliknya, model *high poly* memiliki detail dan titik yang lebih banyak, menawarkan realisme namun memerlukan daya pemrosesan dan memori lebih besar. Pada tahap pengembangan permainan, penting untuk memilih jenis model yang sesuai dengan kebutuhan permainan dan mempertimbangkan batasan perangkat yang mungkin digunakan oleh pemain.

Langkah *mark seam* dan *unwrap* dalam Blender berhubungan dengan menandai tepi (seam) pada model dan mengembangkannya menjadi bentuk datar (unwrap). Teknik ini berguna untuk memastikan tepatnya tekstur yang diterapkan pada objek, terutama pada objek yang rumit seperti karakter dengan pakaian atau

elemen yang menempel. Dengan menandai tepi secara akurat dan melaksanakan unwrap yang cermat, memungkinkan penerapan tekstur dengan lebih mudah dan menghasilkan tampilan visual yang realistis. Setelah proses ekspor, tahap selanjutnya adalah mengimpor aset ke Unity 3D. Unity 3D adalah platform pengembangan permainan yang populer, memungkinkan integrasi beragam aset, termasuk model 3D, suara, dan animasi, menjadi satu kesatuan yang membentuk permainan. Saat mentransfer aset dari Blender ke Unity 3D, penting untuk memastikan kesesuaian format file, biasanya melalui format FBX. Ini memastikan aset yang dihasilkan dari Blender dapat diimpor ke Unity 3D tanpa kendala, memastikan visual dan fungsionalitas yang diharapkan dalam permainan Anda [20].