

BAB 2

TINJAUAN PUSTAKA

2.1 Studi Litelatur

Adapun studi literatur yang dijadikan sebagai referensi pada penelitian ini dapat dilihat pada tabel 2.1.

Tabel 2.1 Studi Litelatur

Review Literatur Pertama	
Judul Artikel	PENERAPAN TEKNOLOGI AUGMENTED REALITY PADA APLIKASI KATALOG RUMAH BERBASIS ANDROID
Penulis	Muhammad Rifa'i, Tri Listyorini, Anastasya Latubessy
Judul Jurnal/Proceeding	Studi Informatika
Tahun Penerbitan	2014
Masalah Utama yang diangkat	Bagaimana cara menarik minat konsumen terhadap property rumah dan ruko
Kontribusi Penulis	Membuat gambar tiga dimensi untuk media promosi dengan menggunakan <i>Augmented Reality</i> .
Ikhtisar Artikel	Pada artikel dikatakan bahwa pembangunan aplikasi dengan menggunakan <i>Augmented Reality</i> digunakan sebagai media promosi untuk memperlihatkan properti secara detail dan nyata.
Hasil Penelitian, Kesimpulan dan Saran	a. Hasil: Media promosi katalog rumah menggunakan <i>Augmented Reality</i> . b. Kesimpulan: Aplikasi sangat efisien karna mempermudah untuk pembeli serta penjual. c. Saran: -
Persamaan dan Perbedaan dengan Penelitian	a. Persamaan: Menerapkan konsep media promosi dengan menggunakan teknologi <i>Augmented Reality</i> . b. Perbedaan: Kasus penelitian.

Review Literatur Kedua	
Judul Artikel	A Survey of Augmented Reality
Penulis	Ronald T. Azuma
Judul Jurnal/Proceeding	Hughes Research Laboratories
Tahun Penerbitan	1997
Masalah Utama yang diangkat	Menyeselaikan masalah social dengan menggunakan teknologi <i>Augmented Reality</i> .
Kontribusi Penulis	Memberikan gambaran bagaimana cara penyampaian teknologi <i>Augmented Reality</i> pada pekerjaan sosial.
Ikhtisar Artikel	Pada artikel dapat disimpulkan bahwa penyajian teknologi <i>Augmented Reality</i> dapat disajikan untuk membantu pekerjaan-pekerjaan yang penting seperti pekerjaan medis.
Hasil Penelitian, Kesimpulan dan Saran	<p>a. Hasil: Teknologi <i>Augmented Reality</i> dapat membantu dalam pekerjaan.</p> <p>b. Kesimpulan: Teknologi <i>Augmented Reality</i> dapat dikembangkan untuk membantu lebih banyak pekerjaan.</p> <p>c. Saran: -</p>
Persamaan dan Perbedaan dengan Penelitian	<p>a. Persamaan: Penelitian teknologi <i>Augmented Reality</i>.</p> <p>b. Perbedaan: Kasus penelitian.</p>
Review Literatur Ketiga	
Judul Artikel	PENGGUNAAN AUGMENTED REALITY UNTUK MENSIMULASIKAN DEKORASI RUANGAN SECARA REAL TIME.
Penulis	Ulva Erida Nur Rochmah, Aris Rakhmadi
Judul Jurnal/Proceeding	Techno.COM
Tahun Penerbitan	2016
Masalah Utama yang diangkat	Objek yang dipasang terlihat tidak sesuai dengan keinginan, objek itu harus dipindah lagi ke tempat lain yang lebih cocok. Hal ini akan memerlukan banyak waktu dan tenaga, apalagi jika objek yang digunakan besar dan berat.
Kontribusi Penulis	Memberikan simulasi objek.

Ikhtisar Artikel	Pada artikel menyampaikan dalam memudahkan penggunaan objek dengan <i>Augmented Reality</i> untuk mencocokkan dengan ruang.
Hasil Penelitian, Kesimpulan dan Saran	<p>a. Hasil: Objek dapat disimulasikan menggunakan <i>Augmented Reality</i>.</p> <p>b. Kesimpulan: Sistem pekerjaan lebih efektif untuk masalah waktu dan tenaga.</p> <p>c. Saran: -</p>
Persamaan dan Perbedaan dengan Penelitian	<p>a. Persamaan: Menggunakan <i>Augmented Reality</i> sebagai media informasi.</p> <p>b. Perbedaan: Cara menyampaikan simulasi objek.</p>
Review Literatur Keempat	
Judul Artikel	Pengembangan Media Edukatif Berbasis <i>Augmented Reality</i> untuk Desain Interior dan Eksterior
Penulis	Urip Muhayat Wiji Wahyudi, Hari Wibawanto & Wahyu Hardyanto
Judul Jurnal/Proceeding	Innovative Journal of Curriculum and Educational Technology
Tahun Penerbitan	2017
Masalah Utama yang diangkat	Cara agar dapat merangsang pikiran, perasaan, minat dan perhatian siswa sehingga proses interaksi komunikasi edukasi antara guru (atau pembuat media) dan siswa dapat berlangsung secara tepat guna dan berdaya guna.
Kontribusi Penulis	Membangun sistem edukatif untuk pembelajaran
Ikhtisar Artikel	Pada penelitian tersebut dibuat sistem pembelajaran media dengan menggunakan <i>Augmented Reality</i> untuk meningkatkan pemahaman siswa.
Hasil Penelitian, Kesimpulan dan Saran	<p>a. Hasil: Memberikan objek berupa desain interior</p> <p>b. Kesimpulan: Media edukatif desain interior dan eksterior berbasis <i>Augmented Reality</i> efektif untuk digunakan pembelajaran.</p> <p>c. Saran: -</p>

Persamaan dan Perbedaan dengan Penelitian	<p>a. Persamaan: Membuat desain interior menggunakan <i>Augmented Reality</i>.</p> <p>b. Perbedaan: Konsep dan fitur penelitian.</p>
Review Literatur Kelima	
Judul Artikel	PENERAPAN TEKNOLOGI AUGMENTED REALITY PADA APLIKASI MEDIA PEMBELAJARAN MIKROKONTROLER BERBASIS ANDROID DENGAN PLATFORM ARCORE
Penulis	Irkham Huda, Yusron Fuadi
Judul Jurnal/Proceeding	Jurnal Informatika Komputer, Bisnis dan Manajemen
Tahun Penerbitan	2019
Masalah Utama yang diangkat	Bagaimana sistem pembelajaran mikrokontroler dapat memudahka siswa
Kontribusi Penulis	Membangun sistem untuk media pembelajaran.
Ikhtisar Artikel	Pada artikel dapat disimpulkan bahwa teknologi <i>Augmented Reality</i> dibuat tidak hanya untuk meningkatkan minat dan daya pikir namun juga dapat mengurangi biaya yang dibutuhkan.
Hasil Penelitian, Kesimpulan dan Saran	<p>a. Hasil: Sistem media pembelajaran dengan memanfaatkan teknologi <i>Augmented Reality</i>.</p> <p>b. Kesimpulan: Sistem lebih efesien dikarenakan menggunakan teknologi <i>Augmented Reality</i> dengan menggunakan metode <i>Markerless</i>.</p> <p>c. Saran: -</p>
Persamaan dan Perbedaan dengan Penelitian	<p>a. Persamaan: Metode dan Platform <i>Augmented Reality</i>.</p> <p>b. Perbedaan: Kasus dan konsep penelitian.</p>
Review Literatur Keenam	
Judul Artikel	Pembangunan Aplikasi Augmented Reality Untuk Mensimulasikan Desain Interior Furniture Set Di CV. Boxes System
Penulis	Kusmaya Rahman
Judul Jurnal/Proceeding	-

Tahun Penerbitan	2017
Masalah Utama yang diangkat	Bagaimana pemasaran katalog dapat lebih efisien.
Kontribusi Penulis	Membangun sistem untuk media pemasaran.
Ikhtisar Artikel	Pada artikel dapat disimpulkan bahwa teknologi <i>Augmented Reality</i> dibuat untuk memudahkan pembeli dalam memilih produk.
Hasil Penelitian, Kesimpulan dan Saran	<p>a. Hasil: Sistem media pemasaran dengan memanfaatkan teknologi <i>Augmented Reality</i>.</p> <p>b. Kesimpulan: Sistem lebih efisien dikarenakan menggunakan teknologi <i>Augmented Reality</i> untuk memperlihatkan objek 3D pada produk yang dipasarkan.</p> <p>c. Saran: -</p>
Persamaan dan Perbedaan dengan Penelitian	<p>a. Persamaan: Penggunaan <i>Augmented Reality</i> untuk menampilkan objek furniture.</p> <p>b. Perbedaan: Metode dan konsep penelitian.</p>
Review Literatur Ketujuh	
Judul Artikel	Interior Design in Augmented Reality Environment
Penulis	Viet Toan Phan, Seung Yeon Choo
Judul Jurnal/Proceeding	International Journal of Computer Applications
Tahun Penerbitan	2010
Masalah Utama yang diangkat	Bagaimana sistem <i>Augmented Reality</i> dapat memudahkan kebutuhan lingkungan hidup.
Kontribusi Penulis	Membangun sistem media interaktif.
Ikhtisar Artikel	Pada artikel dapat disimpulkan bahwa teknologi <i>Augmented Reality</i> dibuat pada lingkungan desain arsitek untuk media sebagai berdiskusi dalam pekerjaan desain interior
Hasil Penelitian, Kesimpulan dan Saran	<p>a. Hasil: Sistem media interaktif dengan memanfaatkan teknologi <i>Augmented Reality</i>.</p> <p>b. Kesimpulan: Sistem lebih efisien dikarenakan menggunakan teknologi <i>Augmented Reality</i> yang dapat digunakan oleh banya orang dengan secara Real Time.</p>

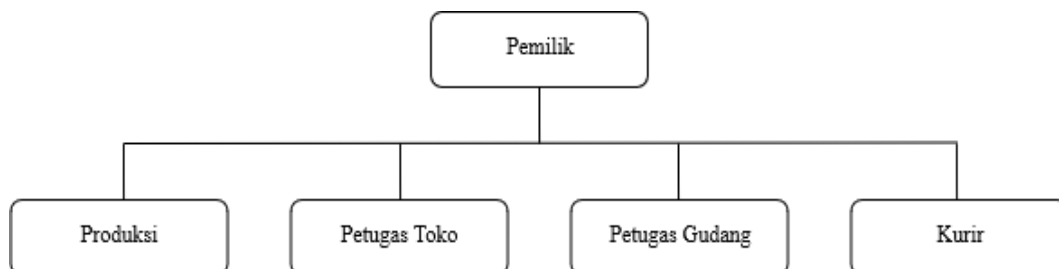
	c. Saran: -
Persamaan dan Perbedaan dengan Penelitian	a. Persamaan: Pembangunan teknologi <i>Augmented Reality</i> . b. Perbedaan: Metode dan konsep penelitian.

2.2 Profil Jaya Abadi Meubel Putra

Jaya Abadi Meubel Putra adalah sebuah perusahaan di bidang penjualan *furniture* dan jasa desain interior yang berdiri sejak tahun 2014. Jaya Abadi Meubel Putra menyediakan aneka jenis *furniture* yaitu seperti lemari, sofa, rak tv serta rak yang diproduksi sendiri maupun pesanan berupa warna dan bahan *furniture*. Lokasi tersebut berada di Jalan Raya Tanjungsari No. 191 Rt. 02 Rw. 05 Kecamatan Tanjungsari Kabupaten Sumedang. Semua *furniture* yang dapat dibeli tidak hanya dalam kota namun dapat memesan dari luar kota, upaya yang dilakukan oleh Jaya Abadi Meubel Putra yaitu memaksimalkan produk model *furniture* yang sesuai kepada konsumen agar tidak ada salah paham dan konsumen merasa puas atas produk model *furniture* yang telah dibeli.

2.2.1 Struktur Organisasi

Berikut ini merupakan struktur organisasi dan deskripsi jabatan pada perusahaan Jaya Abadi Meubel Putra. Adapun struktur organisasi dapat dilihat pada gambar 2.1.



Gambar 2.1 Struktur Organisasi

Adapun uraian tugas pada struktur organisasi Jaya Abadi Meubel Putra adalah sebagai berikut:

- a) **Pemilik**
Menyediakan biaya untuk pembelian bahan-bahan yang digunakan untuk memproduksi *furniture*. Bertanggung jawab atas harga, kualitas, kuantitas, dan kelanjutan produk yang diproduksi di Jaya Abadi Meubel Putra.
- b) **Produksi**
Memproduksi *furniture* serta memenuhi pesanan konsumen.
- c) **Petugas Toko**
Melayani pelanggan dalam proses penjualan dan pembelian produk *furniture*, serta menjelaskan semua model produk yang tersedia kepada konsumen.
- d) **Petugas Gudang**
Mengelola produk yang akan dijual serta mengelola produk *furniture* yang cacat produksi agar dapat diproduksi kembali oleh bagian produksi
- e) **Kurir**
Mengantarkan produk *furniture* yang sudah dibeli atau dipesan oleh pelanggan hingga tempat tujuan dengan aman.

2.3 Landasan Teori

Landasan teori yang berkaitan dengan materi atau teori yang digunakan sebagai acuan dalam melakukan penelitian. Landasan teori yang diuraikan merupakan hasil literatur dan buku-buku.

2.3.1 Pengertian Sistem

Sistem adalah kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu. Dari definisi sistem diatas maka dapat disimpulkan bahwa sistem merupakan suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan kegiatan atau menyelesaikan suatu sasaran tertentu. Sistem mempunyai karakteristik atau sifat-sifat tertentu, yaitu: [9]

- a) **Komponen Sistem** pada suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling kerjasama membentuk satu kesatuan.
- b) **Sifat-sifat Sistem** itu terdiri untuk menjalankan fungsi tertentu dan mempunyai sistem yang lain secara keseluruhan.

- c) Batasan (*Boundary*) Sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya.
- d) Lingkungan Luar Sistem (*Environment*) Adalah apapun diluar batasan sistem yang mempengaruhi operasi sistem, lingkungan luar sistem dapat bersifat menguntungkan dan juga dapat bersifat merugikan sistem tersebut.
- e) Penghubung (*Interface*) Sistem merupakan media penghubung antara suatu subsistem dengan subsistem lainnya.
- f) Masukan Sistem (*Input*) Adalah energi yang dimasukkan kedalam sistem, masukan dapat berupa masukan perawatan (*Maintenace Input*), dan masukan sinyal (*Signal Input*), maintenance input adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. Sedangkan *signal input* adalah energi yang di proses untuk mendapatkan keluaran.
- g) Keluaran Sistem (*Output*) Adalah hasil energi yang diolah dan di klasifikasikan menjadi keluaran yang berguna dan sisa pembuangan keluaran dapat merupakan masukan untuk subsistem yang lain kepada supra sistem.
- h) Sasaran Sistem Suatu sistem pasti memiliki sasaran atau tujuan (*Goal*). Sasaran dari sistem sangat menentukan sekali masukan yang dibutuhkan sistem dan keluaran yang akan di hasilkan sistem

2.3.2 Android

Android merupakan sistem operasi berbasis *Linux* yang dirancang untuk perangkat layar sentuh seperti *smartphone* dan *tablet*. Android awalnya dikembangkan oleh Android, Inc., dengan didukung finansial oleh *Google* yang kemudian dibeli pada tahun 2005. Sistem operasi ini dirilis secara resmi pada tahun 2007, bersamaan dengan didirikannya *Open Handset Alliance*, dari perusahaan-perusahaan perangkat keras, perangkat lunak dan telekomunikasi yang bertujuan untuk memajukan standar terbuka perangkat seluler. Ponsel android pertama kali mulai dijual pada bulan oktober 2008 [10].

2.3.2.1 Sejarah Android

Android, Inc. Didirikan di Palo Alto, California, pada bulan oktober 2003 oleh Andy Rubin (Pendiri *Danger*), Rich Milner (pendiri *Wildfire Cimmunication, Inc.*), Nick Sears (mantan *Vp T-Mobile*) dan Chris White (kepala desain dan

pengembangan antarmuka *WebTV*) untuk mengembangkan “Perangkat *smartphone* yang lebih sadar akan lokasi dan referensi penggunanya”. Tujuan awal pengembangan *Android* adalah untuk pengembangan sebuah sistem operasi canggih yang diperuntukan bagi kamera digital, namun kemudian *disadari* bahwa pasar untuk perangkat tersebut tidak cukup besar dan pengembangan *Android* lalu dialihkan bagi pasar *smartphone* untuk menyaingi *Symbian* dan *Windows Mobile* (*Iphone Apple* belum dirilis pada saat itu) [10].

Google mengakuisisi *Android Inc* pada tanggal 17 agustus 2005, menjadikannya sebagai anak perusahaan yang dimiliki oleh *Google*. Pendiri *Android Inc* seperti Rubin, Miner dan White tetap bekerja di perusahaan setelah diakuisisi oleh *Google*. Setelah itu tidak banyak yang diketahui tentang perkembangan *Android Inc.*, namun banyak anggapan yang menyatakan bahwa *Google* berencana untuk memasuki pasar telepon seluler dengan tindakannya ini. Di *Google*, tim yang dipimpin oleh Rubin mulai mengembangkan platform perangkat seluler dengan menggunakan *kernel linux*. *Google* memasarkan platform tersebut kepada produsen perangkat seluler dan operator nirkabel, dengan janji bahwa mereka menyediakan sistem fleksibel dan bisa diperbaharui. *Google* telah memilih beberapa mitra perusahaan perangkat lunak dan perangkat keras, serta mengisyaratkan kepada operator seluler bahwa kerja sama ini terbuka bagi siapapun yang ingin berpartisipasi [10].

Spekulasi tentang niat *Google* untuk memasuki pasar komunikasi seluler terus berkembang hingga bulan desember 2006. BBC dan Wall Street Journal melaporkan bahwa *Google* sedang bekerja keras untuk menyertakan aplikasi dan mesin pencariannya di perangkat seluler. Berbagai media cetak dan media daring mengabarkan bahwa *Google* sedang mengembangkan perangkat seluler dengan merek *Google*. Beberapa di antaranya berspekulasi bahwa *Google* telah menentukan spesifikasi teknisnya, termasuk produsen telepon seluler dan operator jaringan. Pada bulan Desember 2007, *InformationWeek* melaporkan bahwa *Google* telah mengajukan beberapa aplikasi paten di bidang telepon seluler [10].

Pada tanggal 5 November 2007, *Open Handset Alliance* (OHA) didirikan. OHA adalah konsorsium dari perusahaan-perusahaan teknologi seperti *Google*,

produsen perangkat seluler seperti HTC, Sony dan Samsung, operator nirkabel seperti *Sprint Nextel* dan *T-Mobile*, serta produsen chipset seperti *Qualcomm* dan *Texas Instruments*. OHA sendiri bertujuan untuk mengembangkan standar terbuka bagi perangkat seluler. Saat itu, *Android* diresmikan sebagai produk pertamanya; sebuah *platform* perangkat seluler yang menggunakan kernel *Linux* versi 2.6. Telepon seluler komersial pertama yang menggunakan sistem operasi *Android* adalah *HTC Dream*, yang diluncurkan pada 22 Oktober 2008 [10].

Pada tahun 2010, *Google* merilis seri *Nexus*; perangkat telepon pintar dan tablet dengan sistem operasi *Android* yang diproduksi oleh mitra produsen telepon seluler seperti *HTC*, *LG*, dan *Samsung*. *HTC* bekerja sama dengan *Google* dalam merilis produk telepon pintar *Nexus* pertama, yakni *Nexus One*. Seri ini telah diperbarui dengan perangkat yang lebih baru, misalnya telepon pintar *Nexus 4* dan tablet *Nexus 10* yang diproduksi oleh *LG* dan *Samsung*. Pada 15 Oktober 2014, *Google* mengumumkan *Nexus 6* dan *Nexus 9* yang diproduksi oleh *Motorola* dan *HTC*. Pada 13 Maret 2013, Larry Page mengumumkan dalam postingan blognya bahwa Andy Rubin telah pindah dari divisi *Android* untuk mengerjakan proyek-proyek baru di *Google*. Ia digantikan oleh Sundar Pichai, yang sebelumnya menjabat sebagai kepala divisi *Google Chrome*, yang mengembangkan *Chrome OS* [10].

Sejak tahun 2008, *Android* secara bertahap telah melakukan sejumlah pembaruan untuk meningkatkan kinerja sistem operasi, menambahkan fitur baru, dan memperbaiki bug yang terdapat pada versi sebelumnya. Setiap versi utama yang dirilis dinamakan secara alfabetis berdasarkan nama-nama makanan pencuci mulut atau camilan bergula; misalnya, versi 1.5 bernama *Cupcake*, yang kemudian diikuti oleh versi 1.6 *Donut*. Versi terbaru adalah 5.0 *Lollipop*, yang dirilis pada 15 Oktober 2014 [10].

2.3.3 Augmented Reality

Augmented Reality (AR) adalah sebuah istilah untuk lingkungan yang menggabungkan dunia nyata dan dunia virtual yang dibuat oleh computer sehingga batas antara keduanya menjadi sangat tipis. Ronald Azuma pada tahun 1997

mendefinisikan *Augmented Reality* sebagai sistem yang memiliki karakteristik sebagai berikut:

- a) Menggabungkan lingkungan nyata dan virtual
- b) Berjalan secara interaktif dalam waktu nyata
- c) Integrasi dalam tiga dimensi (3D)

Secara sederhana *AR* bisa didefinisikan sebagai lingkungan nyata yang ditambah objek virtual. Penggabungan objek nyata dan virtual dimungkinkan dengan teknologi display yang sesuai. Interaktivitas dimungkinkan melalui perangkat-perangkat input tertentu [2].

AR merupakan variasi dari *Virtual Environments (VE)*, atau yang lebih dikenal dengan istilah *Virtual Reality (VR)*. Teknologi *VE* membuat pengguna tergabung dalam sebuah lingkungan virtual secara keseluruhan. Ketika tergabung dalam lingkungan tersebut, pengguna tidak bias melihat lingkungan nyata di sekitarnya. Sebaliknya, *AR* memungkinkan pengguna untuk melihat lingkungan nyata, dengan objek virtual yang ditambahkan atau tergabung dengan lingkungan nyata. Tidak seperti *VR* yang sepenuhnya menggantikan lingkungan nyata, *AR* sekedar menambahkan atau melengkapi lingkungan nyata [2].

Tujuan utama dari *AR* adalah untuk menciptakan lingkungan baru dengan menggabungkan interaktivitas lingkungan nyata dan virtual sehingga pengguna merasa bahwa lingkungan yang diciptakan adalah nyata. Dengan kata lain. Pengguna merasa tidak ada perbedaan yang dirasakan antara *AR* dengan apa yang mereka lihat/rasakan di lingkungan nyata. Dengan bantuan teknologi *AR* (seperti visi komputasi dan pengenalan objek) lingkungan nyata disekitar kita akan dapat berinteraksi dalam bentuk digital (virtual). Informasi tentang objek dan lingkungan disekitar kita dapat ditambahkan kedalam sistem *AR* yang kemudian informasi tersebut adalah nyata. Informasi yang ditampilkan oleh objek virtual membantu pengguna melaksanakan kegiatan-kegiatan dalam dunia nyata. *AR* banyak digunakan dalam bidang-bidang seperti kesehatan, militer, industri manufaktur dan juga telah diaplikasikan dalam perangkat-perangkat yang digunakan prang banyak, seperti pada telepon genggam [2].

2.3.3.1 Tujuan Augmented Reality

Tujuan utama dari sistem *Augmented Reality* akan berwujud sebagai sebuah kacamata atau proyektor retina yang akan menyidiakan tampilan informasi yang relevan, dipetakan kelingkungan sekitarnya secara *realtime*. Misalnya, saat melihat sebuah restoran dengan kacamata *Augmented Reality*, maka satu panggilan otomatis langsung ke *database review* atau menu dari website restoran tersebut. Seorang ilmuwan yang bekerja pada perusahaan farmasi bias menggunakan kacamata untuk menampilkan model *3D* dari berbagai molekul dan menggunakannya untuk memvisualisasikan obat yang lebih baik. Anak-anak mungkin menggunakan jaringan yang terhubung kacamata *Augmented Reality* untuk bermain *video game* kehidupan nyata yang memungkinkan menembakkan “laser” dari tangan mereka, meski kemungkinan agak terbatas [11].

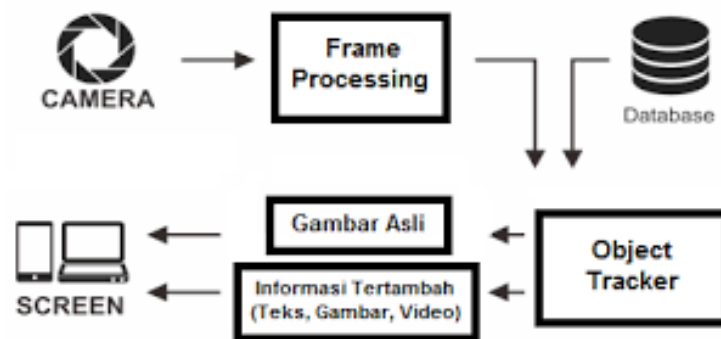
Augmented Reality bergantung pada kemajuan teknologi miniaturisasi dan komputasi bergerak (*mobile computing*). Saat ini, teknologi sistem proyeksi yang efektif dan computer kecil dan cepat masih kurang matang untuk benar-benar dapat membuat antarmuka *Augmented Reality* yang efektif, walaupun kita sudah semakin dekat pada kesempurnaan dengan riset yang dilakukan terus menerus oleh para ahli [11].

Purwarupa telah banyak dibuat selama beberapa decade, tetapi produk utama yang benar-benar layak belum ada. Salah satu teknologi yang menjanjikan adalah proyeksi retina, sehingga kacamata tidak digunakan lagi. Sistem proyeksi retina komersial sebenarnya sudah ada, namun resolusi dan *palet* warnanya masih sangat rendah [11].

Augmented Reality saat ini sebenarnya sudah ada dalam bentuk yang belum sempurna. Misalnya, komentator olahraga sering menggunakan pena cahaya untuk “menggambar” di lapangan sepak bola untuk memberikan bantuan visual untuk rekan komentatornya. Contoh lain yaitu gambar iklan yang sering tampil di lapangan bulutangkis ketika ditayangkan di TV, seakan gambar iklan ini melekat pada lapangan. Gambar iklan sponsor ini terus berganti-ganti selama pertandingan. Ini menunjukkan bukti dari konsep *Augmented Reality* meski masih terbatas [11].

2.3.3.2 Alur Kerja Augmented Reality

Alur kerja *Augmented Reality* secara umum dimulai dari pengambilan gambar marker dengan kamera atau webcam. Marker tersebut dikenali berdasarkan *feature* yang dimiliki, kemudian masuk kedalam *object tracker* (object yang dilacak) yang disediakan oleh *Software Development Kit (SDK)*. Di sisi lain, marker tersebut telah didaftarkan dan disimpan kedalam *database*. *Object tracker* selanjutnya akan melacak dan mencocokkan *marker* tersebut agar dapat menampilkan informasi yang sesuai. Hasil keluaran pelacakan *marker* segera ditampilkan kedalam layar computer dan layar ponsel cerdas. Informasi yang ditampilkan melekat pada *marker* yang bersangkutan secara *real time* [11].



Gambar 2.2 Alur Augmented Reality

Sumber: [11]

- a) *Camera*
Digunakan untuk melacak *marker/markerless*.
- b) *Frame Processing*
Proses perhitungan pixel yang digunakan untuk pendeteksian gambar atau objek.
- c) *Database*
Digunakan untuk menyimpan data *marker*.
- d) *Object Tracker*
Objek yang dilacak yang akan ditampilkan informasinya jika sesuai dengan data yang ada didalam *database*.
- e) Informasi

Menampilkan informasi yang sesuai dengan *object tracker* yang sudah diinisialisasi pada *database*.

f) *Screen*

Untuk menampilkan keluaran dari aplikasi yang berupa Teks, Gambar, Video, dan Objek *3D*.

2.3.3.3 Metode Augmented Reality

Augmented Reality memiliki berbagai jenis metode yang semakin berkembang sampai saat ini. Berikut macam-macam metode *Augmented Reality*:

[11]

a) Marker Based Tracking

Marker adalah suatu pola yang dibuat dalam bentuk gambar yang akan dikenali oleh *webcam*. *Marker* berfungsi sebagai pendefinisi dari *Augmented Reality*. Informasi *marker* akan digunakan untuk mendefinisikan dan menampilkan sebuah objek *Augmented reality*. *Marker* juga merupakan gambar yang terdiri atas *border outline* dan *pattern image* seperti terlihat pada gambar 2.3.



Gambar 2.3 Contoh Marker

Sumber: [11]

Marker biasanya merupakan ilustrasi hitam dan putih persegi dengan batas hitam tebal dan latar belakang putih. Komputer akan mengenali posisi dan orientasi marker dan menciptakan dunia *virtual 3D* yaitu titik (0,0,0) dan 3 sumbu yaitu X,Y, dan Z. Cara pembuatannya pun sederhana, tetapi harus diperhatikan ketebalan *marker* yang akan dibuat, ketebalan *marker* jangan kurang dari 25% dari panjang garis tepi agar pada saat proses deteksi Marker dapat lebih akurat. Sedangkan objek warna putih sebagai *background*, yang nantinya akan digunakan sebagai tempat objek yang akan di-*render*. Warna putih pada *marker* menunjukkan warna sebuah

objek, sedangkan warna hitam menunjukkan latar belakang. Intensitas warna pada suatu objek memiliki warna yang lebih rendah (gelap), sedangkan latar belakang mempunyai intensitas yang lebih tinggi (terang). Adapun beberapa aturan umum dalam pembuatan pola *marker*, yaitu seperti harus kotak berbingkai hitam dan ini adalah rahasia dari pelacakan sebuah *marker*, ukuran tidak lebih dari 631x634 pixel. Warna selain hitam putih juga bisa dikenali oleh sistem *marker* membantu komputer dimana letak objek akan ditampilkan. Ukuran *marker* yang digunakan dapat mempengaruhi penangkapan pola *marker* oleh kamera. Semakin besar ukuran *marker*, maka semakin jauh jarak yang dapat ditangkap oleh kamera dalam proses pendeteksian *marker*. Namun masalahnya, ketika *marker* bergerak menjauhi kamera, jumlah pixel pada layar kamera menjadi lebih sedikit dan ini bisa mengakibatkan pendeteksian tidak akurat [11].

b) Markerless Baser Tracking

Salah satu metode *Augmented Reality* yang saat ini sedang berkembang adalah metode *Markerless Augmented Reality*, dengan metode ini pengguna tidak perlu lagi menggunakan sebuah *marker* untuk menampilkan elemen-elemen digital.

Dalam perancangannya, seolah-olah *markerless* menggabungkan objek virtual dengan objek nyata, dalam hal ini objek virtual berupa objek 2D atau 3D dan objek nyatanya berupa gambar dengan pola tertentu (*markerless*). Meski dinamakan dengan *markerless* namun aplikasi tetap berjalan dengan melakukan pemindaian terhadap objek, namun ruang lingkup yang dipindai lebih luas dibanding dengan pemindaian menggunakan metode *markerbased tracking*. Berbagai macam teknik *Markerless Based Tracking* sebagai teknologi yang saat ini terus dikembangkan adalah *Face Tracking*, *3D Object Tracking*, dan *Motion Tracking* [11].

c) Face Tracking

Dengan menggunakan algoritma yang sedang dikembangkan, komputer dapat mengenali wajah manusia secara umum dengan cara mengenali posisi mata, hidung, dan mulut manusia, kemudian akan mengabaikan objek-objek lain di sekitarnya seperti pohon, rumah, dan benda-benda lainnya. Contoh *face tracking* dapat dilihat pada gambar 2.4.



Gambar 2.4 Face Tracking

Sumber: [11]

d) 3D Object Tracking

Berbeda dengan *Face Tracking* yang hanya mengenali wajah manusia, secara umum teknik *3D Object Tracking* dapat mengenali semua bentuk benda yang ada disekitar, seperti mobil, meja, televisi, dan lain-lain. Contoh *3D Object Tracking* dapat dilihat pada gambar 2.5.

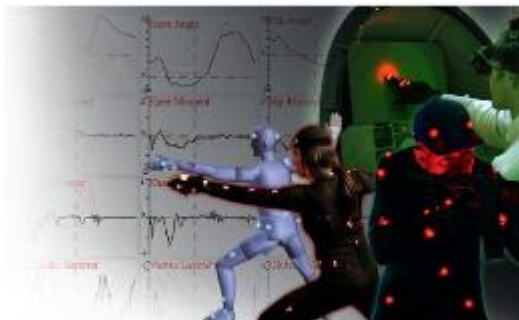


Gambar 2.5 Object Tracking

Sumber: [11]

e) Motion Tracking

Pada teknik ini komputer dapat menangkap gerakan. *Motion Tracking* mulai digunakan secara ekstensif untuk memproduksi film-film yang mencoba mensimulasikan gerakan. Contohnya pada film *Avatar The Last Airbender*, di mana James Cameron menggunakan teknik ini untuk membuat film tersebut dan menggunakannya secara *real time*. Contoh *motion tracking* dapat dilihat pada gambar 2.6.



Gambar 2.6 Motion Tracking

Sumber: [11]

2.3.4 Geometri Ruang

Geometri merupakan bagian matematika yang membahas tentang bentuk dan ukuran dari suatu obyek yang memiliki keteraturan tertentu. Geometri sudah dikenalkan sejak siswa kelas I sekolah dasar sebatas mengenal bola dan bukan bola, tabung dan bukan tabung, balok dan bukan balok, lingkaran dan bukan lingkaran, segitiga dan bukan segitiga, serta segiempat dan bukan segiempat. Di kelas-kelas berikutnya dilanjutkan dengan menggambar bangun datar, bangun ruang, menghitung panjang, luas, hingga volume pada batas-batas yang sesuai untuk tingkatan SD [12].

Di SMP pelajaran mengenai geometri (datar dan ruang) berdasar kurikulum 2004 diulang lagi dengan pendalaman dimulai dari melukis bangun datar, sudut, 2 garis sejajar, duagaris tegak lurus, membagi ruas garis atas beberapa bagian yang sama panjang, membagi sudut atas 2 bagian yang sama besar, pengenalan berpikir deduktif, dalil Pythagoras hingga terapannya dalam kehidupan sehari-hari. Sedangkan untuk geometri ruang dimulai lagi di kelas VIII akhir yaitu mengidentifikasi bangun-bangun ruang sisi lengkung (BRSL), mengidentifikasi bangun ruang sisi datar hingga menentukan besaran-besaran yang ada di dalamnya [12].

2.3.5 Teorema Pythagoras

Geometri merupakan bagian dari matematika yang menempati posisi memprihatinkan dalam pengajarannya dibandingkan dengan cabang matematika

yang lain. Pembelajaran geometri dimulai dari tingkat sekolah dasar (SD), sekolah menengah pertama (SMP), sekolah menengah atas (SMA) dan perguruan tinggi. Sehingga kemampuan matematika siswa pada jenjang Pendidikan yang lebih rendah akan mempengaruhi kemampuan matematika siswa pada tingkat di atasnya. Begitu pula pada materi jarak dua bidang diruang dimensi tiga yang merupakan satu diantara bagian dari geometri [13].

Teorema Pythagoras merupakan salah satu materi esensial dalam matematika, karena merupakan materi prasyarat dalam mempelajari materi-materi yang berkaitan dengan *Geometri*, *Teorema Pythagoras* di dalamnya berlaku bahwa pada setiap sembarang segitiga siku-siku, jika digambarkan persegi-persegi pada ketiga sisinya, maka luas persegi pada hipotenusa (sisi di depan sudut siku-siku) akan sama dengan jumlah luas daerah persegi pada dua sisi yang lain [14].

2.3.6 Object Oriented Programming

OOP (Object Oriented Programming) adalah sebuah pendekatan untuk pengembangan suatu software, dimana dalam struktur software tersebut didasarkan kepada interaksi objek dalam penyelesaian suatu proses atau tugas [15].

Pemrograman berorientasi objek (Inggris: *object-oriented programming* disingkat *OOP*) merupakan paradigma pemrograman yang berorientasikan kepada objek. Semua data dan fungsi di dalam paradigma ini dibungkus dalam kelas-kelas atau objek-objek. Bandingkan dengan logika pemrograman terstruktur. Setiap objek dapat menerima pesan, memproses data, dan mengirim pesan ke objek lainnya. *Object Oriented Programming (OOP)* adalah suatu metode dalam pembuatan program, dengan tujuan untuk menyelesaikan kompleksnya berbagai masalah program yang terus meningkat. Ada dua bagian penting dalam *Object Oriented Programming* yaitu *properties* dan *event*. *Properties* memiliki nama dan nilai, sementara *event* hanya memiliki nama dan biasanya berhubungan dengan *method*, *behavior*, *function* dan *action* [15].

Dalam melakukan pemecahan masalah *Object Oriented Programming* tidak memandang bagaimana cara memecahkan suatu masalah secara terstruktur, melainkan bagaimana suatu masalah itu dapat diselesaikan dengan menentukan objek-objek apa saja yang dapat memecahkan masalah tersebut [15].

Model data berorientasi objek dikatakan dapat memberi fleksibilitas yang lebih, kemudahan mengubah program, dan digunakan luas dalam teknik piranti lunak skala besar. Lebih jauh lagi, pendukung *OOP* mengklaim bahwa *OOP* lebih mudah dipelajari bagi pemula dibanding dengan pendekatan sebelumnya, dan pendekatan *OOP* lebih mudah dikembangkan dan dirawat [15].

2.3.7 Unified Modelling Language

Unified Modelling Language (UML) adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. *UML* menawarkan sebuah standar untuk merancang model sebuah sistem [15].

Dengan menggunakan *UML* kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena *UML* juga menggunakan *class* dan *operation* dalam konsep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa bahasa berorientasi objek seperti *C++*, *Java*, *C#* atau *VB.NET*. Walaupun demikian, *UML* tetap dapat digunakan untuk modeling aplikasi prosedural dalam *VB* atau *C* [15].

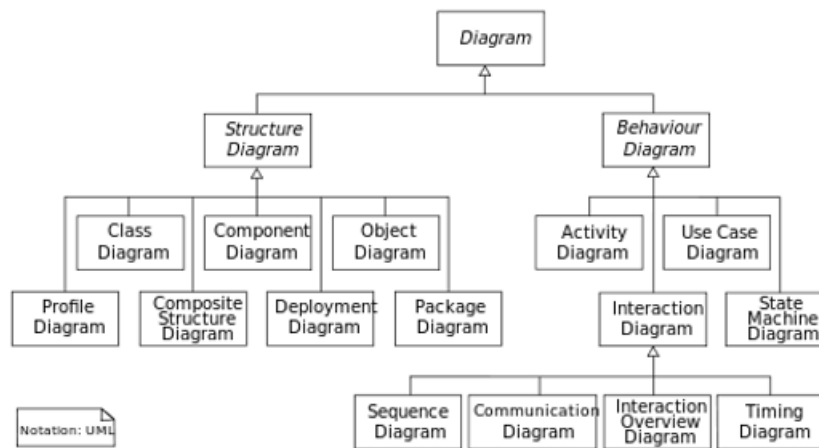
Seperti bahasa-bahasa lainnya, *UML* mendefinisikan notasi dan *syntax*/semantik. Notasi *UML* merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan *UML syntax* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi *UML* terutama diturunkan dari 3 notasi yang telah ada sebelumnya: *Grady Booch OOD (Object-Oriented Design)*, *Jim Rumbaugh OMT (Object Modeling Technique)*, dan *Ivar Jacobson OOSE (Object-Oriented Software Engineering)* [15].

Sejarah *UML* sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: metodologi *booch*, metodologi *coad*, metodologi *OOSE*, metodologi *OMT*, metodologi *shlaer-mellor*, metodologi *wirfs-brock*, dsb. Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi

sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan group/perusahaan lain yang menggunakan metodologi yang berlainan [16].

2.3.7.1 Diagram UML

Berikut penjelasan singkat dari pembagian kategori tersebut adalah:



Gambar 2.7 Diagram UML

- Structure Diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari suatu sistem yang dimodelkan
- Behavior Diagram* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
- Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi antar subsistem pada suatu sistem.

2.3.7.2 Class Diagram

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi [16].

- Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.
- Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem sehingga pembuat perangkat lunak dapat membuat kelas - kelas di dalam program perangkat lunak sesuai dengan

perancangan diagram kelas. Susunan struktur kelas yang baik pada diagram kelas sebaiknya memiliki jenis-jenis kelas sebagai berikut:

- a) Kelas main yaitu kelas yang memiliki fungsi awal dieksekusi ketika sistem dijalankan.
- b) Kelas yang menangani tampilan sistem (*view*) yaitu kelas yang mendefinisikan dan mengatur tampilan ke pemakai.
- c) *Controller* yaitu kelas yang menangani fungsi-fungsi yang harus ada diambil dari pendefinisian *use case*, kelas ini biasa disebut kelas proses.
- d) *Model* yaitu kelas yang digunakan untuk memegang atau membungkus data menjadi sebuah kesatuan yang diambil maupun akan disimpan ke basis data.

2.3.7.3 Objek Diagram

Diagram objek menggambarkan struktur sistem dari segi penamaan objek dan jalannya objek dalam sistem. Pada diagram objek harus dipastikan semua kelas sudah didefinisikan pada diagram kelas harus dipakai objeknya, karena jika tidak, pendefinisian kelas itu tidak dapat dipertanggungjawabkan [16].

2.3.7.4 Use Case Diagram

Diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu [16].

Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case*.

- a) Aktor merupakan orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor belum tentu orang.
- b) *Use case* merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.

2.3.7.5 Activity Diagram

Diagram aktivitas suatu *activity* diagram menggambarkan *workflow* (aliran kerja) atau aktifitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan disini adalah bahwa diagram aktifitas menggambarkan aktifitas bukan apa yang dilakukan aktor. Diagram aktifitas juga banyak digunakan untuk mendefinisikan hal-hak berikut: [16]

- a) Rancangan proses bisnis dimana setiap urutan aktifitas yang digambarkan merupakan proses bisnis sistem yang didefinisikan.
- b) Urutan atau pengelompokan tampilan dari sistem / user interface dimana setiap aktifitas dianggap memiliki sebuah rancangan antarmuka tampilan
- c) Rancangan pengujian dimana setiap aktifitas dianggap memerlukan sebuah pengujian yang perlu didefinisikan kasus ujiannya
- d) Rancangan menu yang ditampilkan pada diagram aktifitas

2.3.7.6 Sequence Diagram

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirim dan diterima antar objek. Oleh karena itu untuk menggambarkan diagram sekuen maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu [16].

2.3.8 Pengujian

Pengujian yang digunakan adalah *black box*, *black box* fokus pada keperluan penelusuran kesalahan fungsional dari software. Ujicoba *black box* berusaha untuk menentukan kesalahan dalam beberapa kategori, diantaranya:

- a) Fungsi-fungsi yang salah atau hilang.
- b) Kesalahan *interface*.
- c) Kesalahan dalam struktur data atau akses database eksternal.
- d) Kesalahan performa.
- e) Kesalahan inisialisasi dan terminasi.

Teknik pengujian black box terdiri dari 10 jenis diantaranya *Equivalence Partitioning*, *Boundary Value Analysis/Limit Testing*, *Comparison Testing*, *Sample Testing*, *Robustness Testing*, *Behavior Testing*, *Requirement Testing*, *Performance*

Testing, Endurance Testing, Cause-Effect Relationship Testing. Salah satunya yang akan dibahas adalah *Equivalence partitioning* [17].

Equivalence partitioning merupakan metode ujicoba *black box* yang membagi domain *input* dari program menjadi beberapa kelas data dari kasus ujicoba yang dihasilkan. Kasus uji penanganan *single* yang ideal menemukan sejumlah kesalahan (misalnya: kesalahan pemrosesan dari seluruh data karakter) yang merupakan syarat lain dari suatu kasus yang dieksekusi sebelum kesalahan umum diamati [17].

2.3.9 Software

Software merupakan peralatan-peralatan yang dibutuhkan dalam membangun perangkat lunak. Adapun *Software* yang digunakan untuk membuat aplikasi Penerapan Teknologi *Augmented Reality* di Jaya Abadi Meubel Putra ini adalah *Android Studio, ARCore API, dan blender*.

2.3.9.1 Android Studio

Android Studio adalah *IDE* resmi untuk pengembangan aplikasi *Android*, berdasarkan *IntelliJ IDEA*. Selain kemampuan yang diharapkan dari *IntelliJ*, *Android Studio* menawarkan: [18]

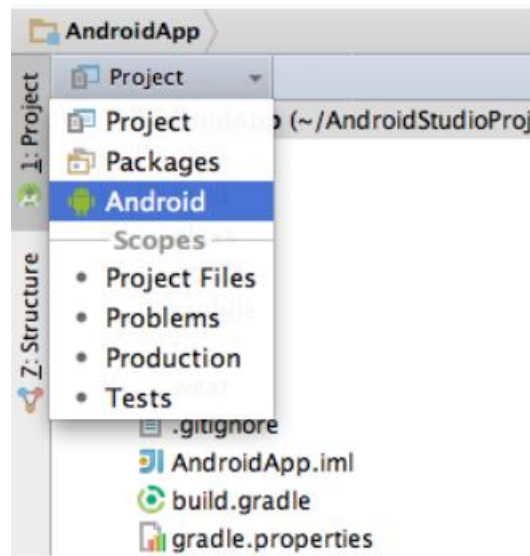
- a) Sistem pembangunan berbasis *Gradle* fleksibel.
- b) Buat varian dan beberapa pembuatan file *apk*.
- c) Templat kode untuk membantu dalam membangun fitur aplikasi umum.
- d) Editor tata letak kaya dengan dukungan untuk mengedit tema *drag and drop*.
- e) Alat serat untuk menangkap kinerja, kegunaan, kompatibilitas versi, dan masalah lainnya.
- f) Kemampuan *ProGuard* dan penandatanganan aplikasi.
- g) Dukungan bawaan untuk *Google Cloud Platform*, membuatnya mudah untuk mengintegrasikan *Google Cloud Messaging* dan *App Engine*.

2.3.9.1.1 Struktur Proyek

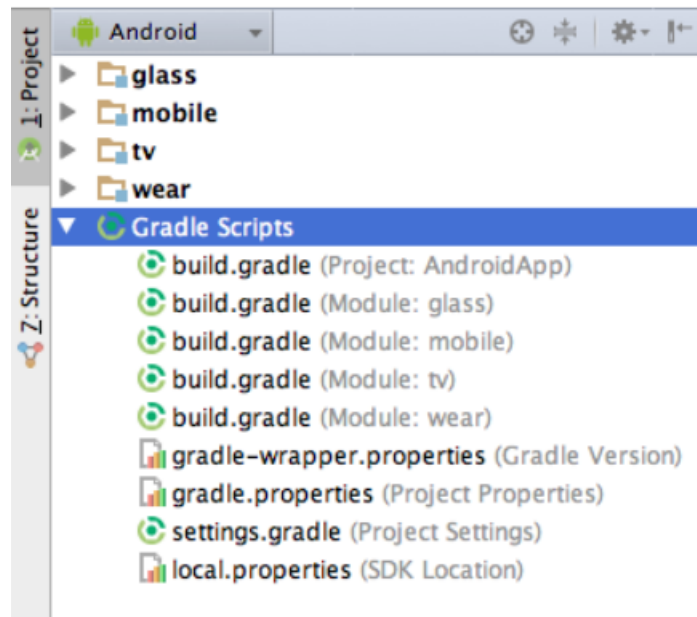
Secara default, *Android Studio* menampilkan file profil dalam tampilan proyek *Android*. Tampilan ini menunjukkan versi rata dari struktur proyek yang menyediakan akses cepat ke file sumber utama proyek *Android* dan membantu

bekerja dengan sistem build berbasis Gradle yang baru. Tampilan proyek Android:
[18]

- a) Grup file *build* untuk semua modul di tingkat atas hirarki proyek.
- b) Memperlihatkan direktori sumber paling penting di tingkat atas hirarki modul.
- c) Grup semua file *manifes* untuk setiap modul.
- d) Memperlihatkan file sumber daya dari semua set sumber *Gradle*.
- e) Grup file sumber daya untuk berbagai lokal, orientasi, dan jenis layar dalam satu grup per jenis sumber daya.



Gambar 2.8 File proyek Android



Gambar 2.9 File pembuatan proyek Android

Setiap proyek di *Android Studio* berisi satu atau beberapa modul dengan file kode sumber dan file sumber daya. Jenis-jenis modul mencakup:

- a) Modul aplikasi *Android*.
- b) Modul Pustaka
- c) Modul *Google App Engine*.

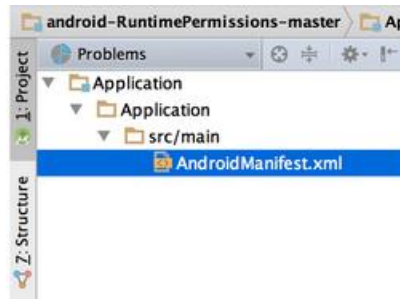
Secara default, *Android Studio* akan menampilkan file proyek dalam tampilan proyek *Android*, seperti yang ditampilkan dalam gambar 2.9. Tampilan disusun berdasarkan modul untuk memberikan akses cepat ke file sumber utama proyek [18].

Semua file versi terlihat di bagian atas di bawah **Gradle Scripts** dan masing-masing modul aplikasi berisi folder berikut:

- a) manifests: Berisi file *AndroidManifest.xml*.
- b) java: Berisi file kode sumber Java, termasuk kode pengujian *JUnit*.
- c) res: Berisi semua sumber daya bukan kode, seperti tata letak *XML*, *string UI*, dan gambar bitmap.

Struktur proyek *Android* pada *disk* berbeda dari representasi rata ini. Untuk melihat struktur file sebenarnya dari proyek ini, pilih **Project** dari menu tarik turun **Project** (dalam gambar 2.8, struktur ditampilkan sebagai **Android**) [18].

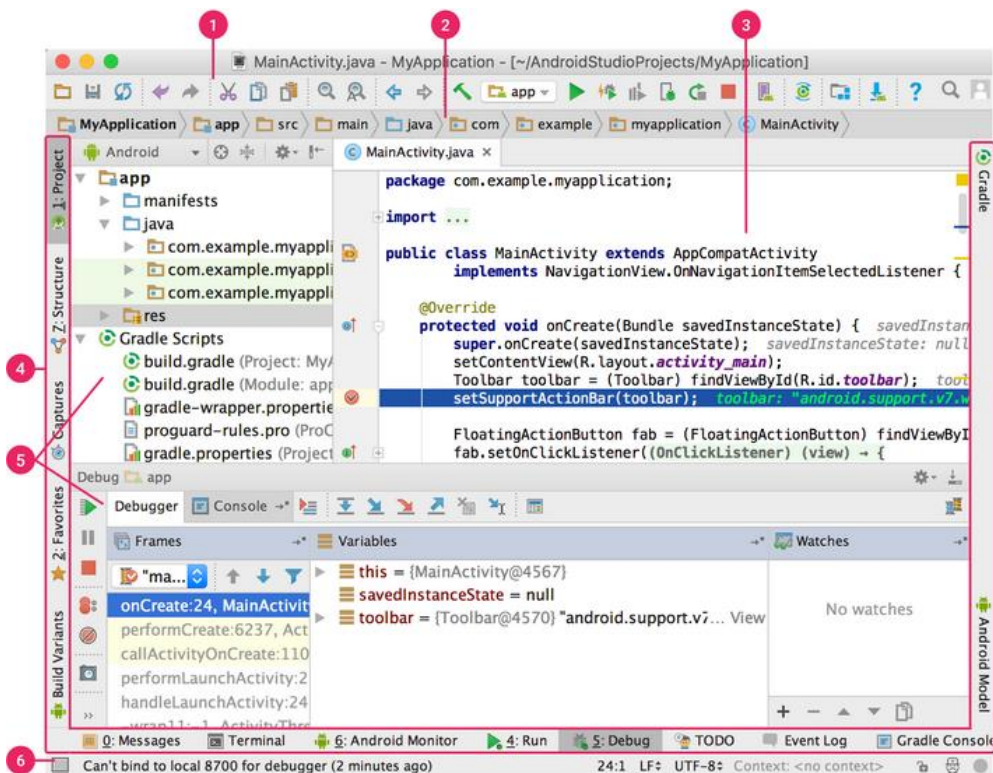
Pengguna juga bisa menyesuaikan tampilan file proyek untuk berfokus pada aspek tertentu dari pengembangan aplikasi. Misalnya, memilih tampilan **Problems** dari tampilan proyek akan menampilkan tautan ke file sumber yang berisi kesalahan pengkodean dan sintaks yang dikenal, misalnya tag penutup elemen XML tidak ada dalam file tata letak [18].



Gambar 2.10 File proyek di tampilan Problem

2.3.9.1.2 Antarmuka Pengguna

Jendela utama *Android Studio* terdiri dari beberapa bidang logika yang diidentifikasi dalam gambar berikut [18].



Gambar 2.11 Jendela utama Android Studio

- a) Bilah alat memungkinkan untuk melakukan berbagai jenis tindakan, termasuk menjalankan aplikasi dan meluncurkan alat *Android*.
- b) Bilah navigasi membantu bernavigasi di antara proyek dan membuka file untuk diedit. Bilah ini memberikan tampilan struktur yang terlihat lebih ringkas dalam jendela *Project*.
- c) Jendela editor adalah tempat membuat dan memodifikasi kode. Bergantung pada jenis file saat ini, editor dapat berubah. Misalnya, ketika melihat file tata letak, editor menampilkan *Layout Editor*.
- d) Bilah jendela alat muncul di luar jendela *IDE* dan berisi tombol yang memungkinkan pengguna meluaskan atau menciutkan jendela alat individual.
- e) Jendela alat memberi akses ke tugas tertentu seperti pengelolaan proyek, penelusuran, kontrol versi, dan banyak lagi. Pengguna bisa meluaskan dan juga menciutkannya.
- f) Bilah status menampilkan status proyek dan *IDE* itu sendiri, serta setiap peringatan atau pesan.

Pengguna bisa menata jendela utama untuk memberi ruang layar yang lebih luas dengan menyembunyikan atau memindahkan bilah alat dan jendela alat. Pengguna juga bisa menggunakan pintasan *keyboard* untuk mengakses sebagian besar fitur *IDE* [18].

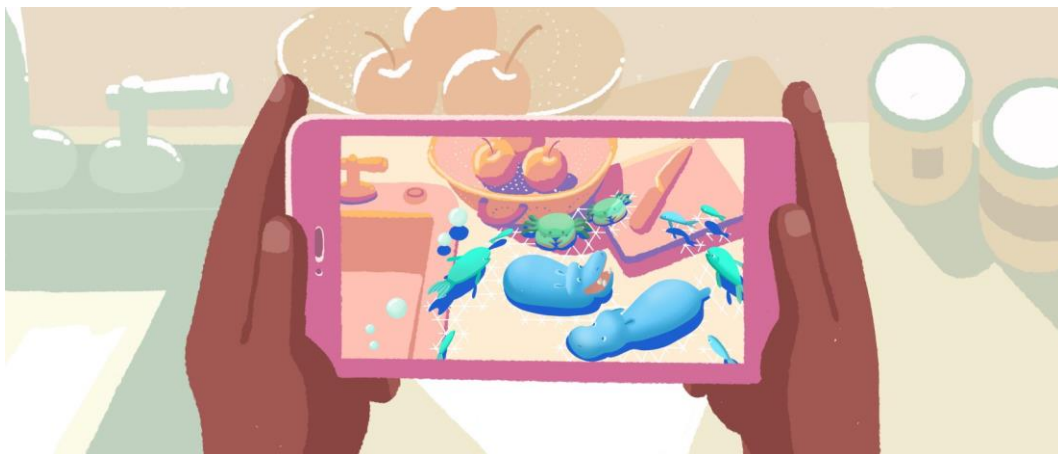
Pengguna dapat menelusuri seluruh kode sumber, basis data, tindakan, elemen antarmuka pengguna, dan seterusnya setiap saat dengan menekan tombol *Shift* dua kali, atau mengeklik kaca pembesar di sudut kanan atas dari jendela *Android Studio*. Ini akan sangat berguna misalnya saat mencoba menemukan tindakan *IDE* tertentu yang pengguna lupakan cara memicunya [18].

2.3.9.2 ARCore API

ARCore merupakan *Software Development Kit (SDK)* untuk *Android* dan *iOS* yang pertama rilis Maret 2018 (versi stabil September 2018). Sebelum merilis *SDK* ini, *Google* sempat merilis *Tango* pada 2014, yang diharapkan akan *booming GPS*, mampu memetakan dunia berbekal sensor dan kamera *motion-tracking*. Namun, *Google* ‘berubah pikiran’ setelah *ARKit* milik *Apple* rilis.

Google mengembangkan *SDK* berbasis *AR* yang beroperasi melalui 2 cara: *tracking* dan *building environment*. Posisi perangkat *mobile* pengguna selagi bergerak. Lalu, sistem akan menerjemahkan lingkungan hidup pengguna agar tampak serealistis mungkin. Pada dasarnya, teknologi *motion tracking* menggunakan kamera *handphone* untuk mengidentifikasi poin-poin menarik alias ‘fitur’ kemudian melacak pergerakannya seiring waktu. Dengan mengkombinasikan pergerakan poin dan membaca sensor inersia, ditentukan posisi dan orientasi *handphone* selagi pengguna bergerak berpindah tempat [19].

ARCore mampu mendeteksi permukaan datar, seperti meja dan lantai, sekaligus memperkirakan pencahayaan rata-rata di area sekitar titik utama. Detail tentang ini akan kami berikan di bawah. Seketika memahami lingkungan pengguna, Pengguna bebas meletakkan objek visual, membaca anotasi, atau informasi lain seakan-akan terjadi langsung di dunia nyata. *ARCore* terbentuk menggunakan beberapa set *API* yang berbeda-beda, *framework*, dan tools lainnya. Beberapa *API* itu sudah bisa diterapkan menggunakan *Android* dan *iOS* tertentu [19].



Gambar 2.12 ARCore

ARCore adalah *platform Google* untuk membangun pengalaman *augmented reality*. Menggunakan *API* yang berbeda, *ARCore* memungkinkan ponsel merasakan lingkungannya, memahami dunia, dan berinteraksi dengan informasi. Beberapa *API* tersedia di *Android* dan *iOS* untuk memungkinkan pengalaman *AR* yang dibagikan [19].

2.3.9.2.1 Cara Kerja ARCore

Pada dasarnya, *ARCore* melakukan dua hal: melacak posisi perangkat seluler saat bergerak, dan membangun pemahamannya sendiri tentang dunia nyata. Teknologi pelacakan gerak *ARCore* menggunakan kamera ponsel untuk mengidentifikasi titik-titik menarik, yang disebut fitur, dan melacak bagaimana titik-titik itu bergerak seiring waktu. Dengan kombinasi pergerakan titik-titik ini dan pembacaan dari sensor inersia telepon, *ARCore* menentukan posisi dan orientasi telepon saat bergerak melalui ruang [19].

Selain mengidentifikasi poin-poin penting, *ARCore* dapat mendeteksi permukaan datar, seperti meja atau lantai, dan juga dapat memperkirakan pencahayaan rata-rata di area di sekitarnya. Kemampuan ini bergabung untuk memungkinkan *ARCore* membangun pemahamannya sendiri tentang dunia di sekitarnya [19].

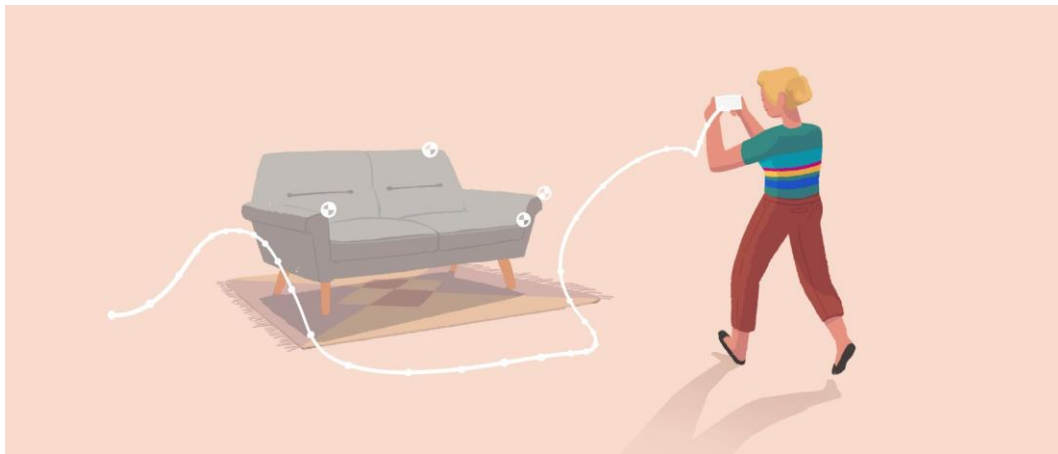
Pemahaman *ARCore* tentang dunia nyata memungkinkan pengguna menempatkan objek, anotasi, atau informasi lainnya dengan cara yang diintegrasikan dengan dunia nyata. Pengguna dapat menempatkan anak kucing yang sedang tidur siang di sudut meja kopi, atau membuat anotasi lukisan dengan informasi biografis tentang artis tersebut. Pelacakan gerak berarti dapat bergerak dan melihat benda-benda ini dari sudut mana pun, dan bahkan jika berbalik dan meninggalkan ruangan, ketika kembali, anak kucing atau anotasi akan berada tepat di tempat pengguna meninggalkannya. *ARCore* menggunakan tiga kemampuan utama untuk mengintegrasikan konten virtual dengan dunia nyata seperti yang terlihat melalui kamera ponsel pengguna: [19]

a) *Motion tracking*

Saat ponsel bergerak di seluruh dunia, *ARCore* menggunakan proses yang disebut odometry dan pemetaan bersamaan, untuk memahami di mana ponsel relatif terhadap dunia di sekitarnya. *ARCore* mendeteksi fitur yang berbeda secara visual dalam gambar kamera yang ditangkap yang disebut *feature point* dan menggunakan titik-titik ini untuk menghitung perubahan lokasinya. Informasi visual dikombinasikan dengan pengukuran inersia dari IMU perangkat untuk

memperkirakan *pose* (posisi dan orientasi) kamera relatif terhadap dunia dari waktu ke waktu [19].

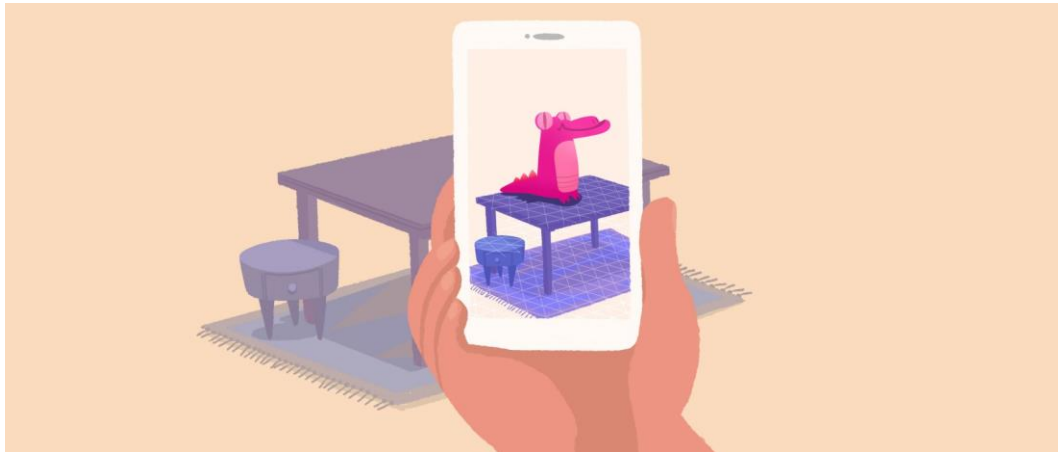
Dengan menyelaraskan pose kamera virtual yang membuat konten 3D dengan pose kamera perangkat yang disediakan oleh *ARCore*, pengembang dapat membuat konten virtual dari perspektif yang benar. Gambar virtual yang diberikan dapat overlay di atas gambar yang diperoleh dari kamera perangkat, menjadikannya seolah-olah konten virtual adalah bagian dari dunia nyata [19].



Gambar 2.13 Motion Tracking

b) *Environmental understanding*

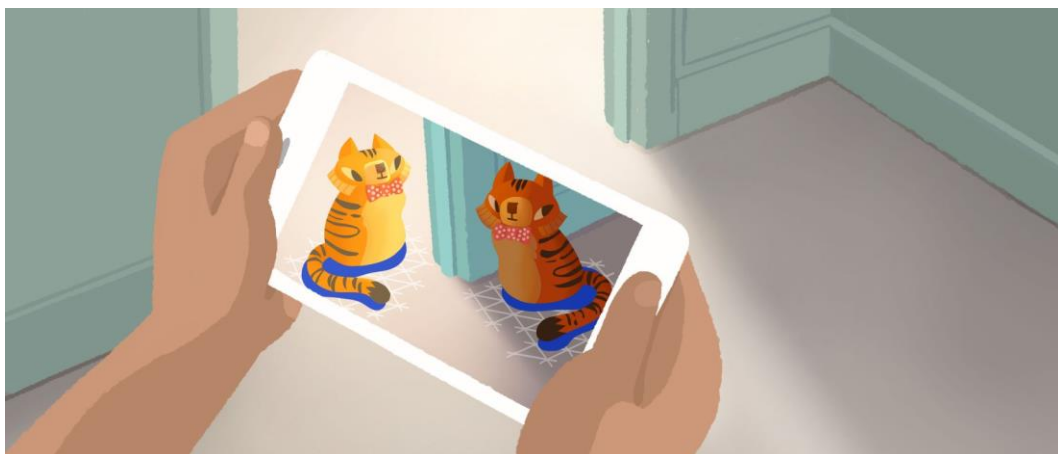
ARCore terus meningkatkan pemahamannya tentang lingkungan dunia nyata dengan mendeteksi *feature point* dan *plane*. *ARCore* mencari kelompok titik fitur yang tampaknya terletak pada permukaan horizontal atau vertikal yang umum, seperti meja atau dinding, dan menjadikan permukaan ini tersedia untuk aplikasi sebagai *plane*. *ARCore* juga dapat menentukan batas setiap pesawat dan membuat informasi itu tersedia untuk aplikasi. Pengguna dapat menggunakan informasi ini untuk menempatkan benda-benda virtual di permukaan yang rata. Karena *ARCore* menggunakan titik fitur untuk mendeteksi bidang, permukaan datar tanpa tekstur, seperti dinding putih, mungkin tidak terdeteksi dengan benar [19].



Gambar 2.14 Enviromental Understanding

c) *Light estimation*

ARCore dapat mendeteksi informasi tentang pencahayaan lingkungannya dan memberi intensitas dan koreksi warna rata-rata dari gambar kamera yang diberikan. Informasi ini memungkinkan menyalakan objek virtual pengguna dalam kondisi yang sama seperti lingkungan di sekitar mereka, meningkatkan rasa realisme [19].



Gambar 2.15 Light Estimation

d) *User interaction*

ARCore menggunakan uji coba untuk mengambil (x, y) koordinat yang sesuai dengan layar ponsel (disediakan oleh keran atau interaksi apa pun yang ingin didukung aplikasi) dan memproyeksikan sinar ke dalam pandangan kamera tentang dunia, mengembalikan pesawat atau fitur poin bahwa sinar berpotongan, bersama dengan pose persimpangan itu di ruang dunia. Ini memungkinkan pengguna untuk memilih atau berinteraksi dengan objek di lingkungan [19].

e) *Oriented points*

Poin yang berorientasi memungkinkan menempatkan objek virtual pada permukaan miring. Saat melakukan uji tekan yang mengembalikan titik fitur, *ARCore* akan melihat titik fitur terdekat dan menggunakannya untuk mencoba memperkirakan sudut permukaan pada *feature point* yang diberikan. *ARCore* kemudian akan mengembalikan pose yang memperhitungkan sudut itu. Karena *ARCore* menggunakan kelompok *feature point* untuk mendeteksi sudut permukaan, permukaan tanpa tekstur, seperti dinding putih, mungkin tidak terdeteksi dengan benar [19].

f) *Anchors and trackables*

Poses dapat berubah saat *ARCore* meningkatkan pemahamannya tentang posisinya sendiri dan lingkungannya. Ketika pengguna ingin menempatkan objek virtual, pengguna perlu mendefinisikan jangkar untuk memastikan bahwa *ARCore* melacak posisi objek dari waktu ke waktu. Sering kali membuat *anchor* berdasarkan *pose* yang dikembalikan oleh *test hit*, seperti yang dijelaskan dalam *User Interaction* [19].

Fakta bahwa *pose* dapat berubah berarti bahwa *ARCore* dapat memperbarui posisi objek lingkungan seperti *plane* dan *feature point* dari waktu ke waktu. *Plane* dan *point* adalah jenis objek khusus yang disebut dapat dilacak. Seperti namanya, ini adalah objek yang akan dilacak *ARCore* dari waktu ke waktu. Pengguna dapat menambatkan objek virtual ke *trackable* spesifik untuk memastikan bahwa hubungan antara objek virtual dan yang dapat dilacak tetap stabil meskipun perangkat bergerak. Ini berarti bahwa jika menempatkan *figurine* Android virtual di atas meja, jika *ARCore* kemudian menyesuaikan pose pesawat yang terkait dengan desk tersebut, *figurine* Android akan tetap tampak berada di atas meja [19].

Jika Pengguna baru menggunakan *ARCore*, akan sangat membantu untuk meninjau *world space dan poses*.

a) *World space*

b) Koordinasikan ruang di mana kamera dan objek diposisikan

c) Posisi kamera dan objek diperbarui di ruang dunia dari bingkai ke bingkai

d) *Poses*

e) Merupakan posisi dan orientasi objek dalam ruang dunia

Saat membuat *ARCore*, menggunakan pose yang menggambarkan posisi dan orientasi relatif terhadap perkiraan *world space* untuk *frame* saat ini. Pengguna melampirkan satu atau lebih objek ke *ARCore* ini. *ARCore* dan benda-benda yang melekat padanya tampaknya tetap berada di tempat mereka ditempatkan di *world*. Karena pose *ARCore* beradaptasi dengan pembaruan *world space* di setiap *frame*, *ARCore* memperbarui pose objek sesuai. Pengguna dapat melampirkan banyak objek ke *ARCore* yang sama untuk memastikan bahwa objek-objek ini mempertahankan posisi dan orientasi relatifnya, bahkan ketika pose jangkak menyesuaikan [19].

2.3.9.3 Firebase

2.3.9.2.1 Firebasse Realtime Database

Firestore Realtime Database adalah database yang di-host di *cloud*. Data disimpan sebagai *JSON* dan disinkronkan secara *realtime* ke setiap klien yang terhubung. Ketika membuat aplikasi lintas-platform dengan *SDK Android*, *iOS*, dan *JavaScript*, semua klien akan berbagi sebuah *instance Realtime Database* dan menerima *update* data terbaru secara otomatis [20].

Kemampuan utama:

a) Realtime

Sebagai ganti permintaan *HTTP* biasa, *Firestore Realtime Database* menggunakan sinkronisasi data setiap kali data berubah, semua perangkat yang terhubung akan menerima *update* dalam waktu milidetik. Memberikan pengalaman yang kolaboratif dan imersif tanpa perlu memikirkan kode jaringan.

b) Offline

Aplikasi *Firestore* tetap responsif bahkan saat offline karena *SDK Firestore Realtime Database* menyimpan data ke *disk*. Setelah konektivitas pulih, perangkat klien akan menerima setiap perubahan yang terlewat dan melakukan sinkronisasi dengan status server saat ini.

c) Dapat Diakses dari Perangkat Klien

Firestore Realtime Database dapat diakses secara langsung dari perangkat seluler atau browser web server aplikasi tidak diperlukan. Keamanan dan validasi data dapat diakses melalui aturan Keamanan *Firestore Realtime Database* yang merupakan kumpulan aturan berbasis ekspresi dan dijalankan ketika data dibaca atau ditulis.

d) Menskalakan di beberapa database

Dengan *Firestore Realtime Database* pada paket harga *Blaze*, Pengguna dapat mendukung kebutuhan data aplikasi pada skala tertentu dengan membagi data di beberapa *instance database* di *project Firestore* yang sama. Menyederhanakan autentikasi dengan *Firestore Authentication* pada project Anda dan mengautentikasi pengguna di *instance database* pengguna. Mengontrol akses ke data di tiap database dengan Aturan *Firestore Realtime Database* khusus untuk tiap *instance database*.

2.3.9.2.2 Firestore Storage

Cloud Storage for Firestore adalah layanan penyimpanan objek yang andal, sederhana, dan hemat biaya yang dibuat untuk skala *Google*. *Firestore SDK* untuk *Cloud Storage* menambahkan keamanan *Google* pada *upload* dan *download* file untuk aplikasi *Firestore* yang dibuat, bagaimanapun kualitas jaringannya. Pengguna dapat menggunakan *SDK* kami untuk menyimpan gambar, audio, video, atau konten buatan pengguna lainnya. Di server, Pengguna dapat menggunakan *Google Cloud Storage* untuk mengakses file yang sama [20].

Kemampuan utama:

a) Operasi yang stabil

Firestore SDK untuk *Cloud Storage* melakukan *upload* dan *download*, bagaimanapun kualitas jaringannya. *Upload* dan *download* bersifat stabil, artinya proses ini akan dilanjutkan dari posisi berhenti, sehingga menghemat waktu dan *bandwidth* pengguna.

b) Keamanan yang kuat

Firestore SDK untuk *Cloud Storage* terintegrasi dengan *Firestore Authentication* untuk menyediakan autentikasi yang mudah dan intuitif bagi

developer. Pengguna dapat menggunakan model keamanan deklaratif kami untuk mengizinkan akses berdasarkan nama file, ukuran, jenis konten, dan metadata lainnya.

c) Skalabilitas tinggi

Cloud Storage for Firebase dibuat untuk skala *exabyte* ketika aplikasi menjadi viral. Berkembang dari prototipe menjadi produksi dengan mudah menggunakan infrastruktur yang juga memberdayakan *Spotify* dan *Google Foto*.

2.3.9.4 Blender

Blender adalah program pembuat konteks 3D yang bersifat sumber terbuka (*open source*), yakni bebas untuk dikembangkan oleh penggunanya dan dapat didistribusikan kembali secara legal [21].

Blender biasa digunakan untuk kompositor video dan pembuat permainan yang terintegrasi. Karya yang dihasilkan tidak dikenakan royalti kepada pengembang, dan dapat dipublikasikan secara gratis maupun untuk dikomersilkan [21].

Orang-orang biasa memandang perangkat-perangkat lunak gratis atau *freeware* dengan negatif, fiturnya terbatas, atau hanya sekedar demo. *Blender* sepenuhnya sangat fungsional. *Blender* berada dibawah bangunan komunitas sumber terbuka sehingga orang-orang dari seluruh dunia memberikan kontribusi untuk keberhasilannya. Sampai saat ini *Blender* terus berkembang menjadi perangkat lunak pengembang 3D yang kompeten dan beberapa produk 3D profesional pun berhasil diciptakan dengan perangkat lunak sumber terbuka ini [21].

Blender adalah program *rendering*, pengembangan animasi dan permainan 3D yang dipelihara oleh *Blender Foundation* dan dapat diunduh gratis dari <http://www.blender.org>. *Blender Foundation* merupakan sebuah organisasi independen asal Belanda, bertindak sebagai perusahaan nirlaba untuk kepentingan publik. *Blender* merupakan salah satu program pemodelan dan animasi 3D yang mempunyai kelebihan dibanding program pemodelan 3D lainnya, kelebihan yang dimiliki *Blender* adalah dapat membuat permainan tanpa menggunakan program

tambahan lainnya. Blender menggunakan *OpenGL* untuk render grafiknya yang dapat digunakan pada berbagai sistem operasi. Kekurangannya terletak pada *GUI* yang agak sulit dimengerti sehingga membutuhkan waktu untuk membiasakan dan membuat permainan dengan lancar pada *Blender* [21].