

## **BAB 3**

### **PELAKSANAAN KERJA MAGANG**

#### **3.1 Kedudukan dan Organisasi**

Magang ini dilakukan dibawah supervisi Dr.Ir. Winarno, M.Kom yang membantu memberi arahan dan bimbingan untuk proyek ini. Bimbingan diberi dalam bentuk menjawab pertanyaan mengenai pekerjaan magang, memeriksa hasil kerja yang dilakukan, dan mendiskusikan tahap selanjutnya melalui pertemuan rutin secara *online*. Pekerjaan dilakukan sebagai seorang *Unity Developer* yang bertanggung jawab atas pengembangan aplikasi yang dibuat. Tugas-tugas yang dijalankan melibatkan proses desain, implementasi, dan pengujian aplikasi yang dikembangkan selama pekerjaan magang.

#### **3.2 Tugas yang Dilakukan**

Tugas yang dilakukan meliputi pembangunan aplikasi berbasis *Virtual Reality* yang akan dihasilkan oleh proses magang. Dengan demikian, penyelesaian aplikasi merupakan tujuan pekerjaan ini yang dilaksanakan sebagai seorang *Unity Developer*. Berikut merupakan tugas-tugas yang dilakukan dalam proses pembangunan aplikasi ini:

1. Rapat dengan supervisor untuk membahas aplikasinya
2. Merancang sistem dan fitur aplikasi
3. Membangun dan mengembangkan aplikasi
4. Melakukan pengujian pada aplikasi

#### **3.3 Uraian Pelaksanaan Magang**

Pelaksanaan kerja magang diuraikan seperti pada Tabel 3.1.

Tabel 3.1. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang

Minggu Ke -	Pekerjaan yang dilakukan
1	Melakukan Integrasi API Google Gemini dalam aplikasi
2	Mengembangkan fitur <i>AI Chatbot</i>
3	Mengembangkan fitur <i>AI Chatbot</i>
4	Melakukan desain ulang ruangan <i>Main Gate</i>
5	Mengembangkan fitur <i>Settings</i>
6	Mengembangkan fitur <i>Guard NPC</i>
7	Mendesain ulang area luar gedung pameran
8	Mengembangkan fitur <i>Car Interior</i>
9	Mengimplementasi API Gemini untuk <i>Car Stand</i>
10	Desain ulang UI dan mengembangkan fitur <i>text to speech</i>
11	Mengembangkan fitur <i>text to speech</i>
12	Mengembangkan NPC <i>salesperson</i>
13	Mengembangkan fitur tutorial
14	Melanjutkan pengembangan fitur tutorial dan melakukan <i>bug fixing</i>
15	Melakukan <i>bug fixing</i>
16	Melakukan <i>user testing and evaluation</i>

Pekerjaan magang ini dilaksanakan menggunakan perangkat keras laptop pribadi, komputer lab milik Universitas Multimedia Nusantara, dan perangkat *virtual reality*. Perangkat *virtual reality* yang digunakan termasuk *headset* dan juga dua buah *controllers*. Ada dua perangkat *virtual reality* berbeda yang digunakan selama proses magang yaitu Oculus Rift dan Meta Quest 3.

Laptop pribadi yang digunakan merupakan laptop Acer Aspire E 14 dengan spesifikasi berikut:

1. CPU: Intel(R) Core(TM) i3-6006U
2. RAM: 12 GB
3. GPU: NVIDIA GeForce 940MX

Sedangkan, komputer lab Universitas Multimedia Nusantara yang digunakan memiliki spesifikasi berikut:

1. CPU: Intel(R) Core(TM) i5-4690K

2. RAM: 8 GB

3. GPU: NVIDIA GeForce 1080

### **3.3.1 Perancangan Aplikasi**

Kegiatan ini meliputi proses desain dibalik aplikasi yang sedang dikembangkan. Perancangan tampilan, fitur, dan juga alur penggunaan pada aplikasi termasuk hal-hal yang dikerjakan pada kegiatan ini. Bagian tampilan mengacu kepada aspek visual pada aplikasi seperti UI dan lingkungan 3D yang ditempati pengguna. Untuk lingkungan 3D, perhatian diberikan kepada tata letak dan penampilan objek 3D yang digunakan seperti bangku, pohon, dll. Perancangan fitur dilakukan untuk menentukan fitur-fitur yang akan dimasukkan dalam aplikasi. Menentukan bagaimana sebuah fitur akan diimplementasikan dan mempengaruhi fitur lain juga dilakukan. Alur penggunaan mengacu kepada bagaimana seorang pengguna akan menggunakan dan berinteraksi dengan aplikasinya. Perancangan pada bagian ini dilaksanakan untuk membuat pengalaman pengguna yang baik.

Perancangan aplikasi dilakukan secara kolaboratif dengan supervisor magang. Diskusi mengenai perincian desain aplikasi dilakukan setiap minggu pada sebuah pertemuan yang dilakukan secara *online*. Pembahasan pada pertemuan mencakupi desain yang akan dibuat dan diterapkan. Pembahasan untuk hasil implementasi desain dan perubahan yang mungkin diperlukan juga dilakukan. Dengan ini, supervisor memberi arahan dan masukan pada proses perancangan aplikasi.

Proses perancangan pada kerja magang ini menambahkan fitur serta mendesain ulang beberapa aspek dari aplikasi yang sudah ada. Perbedaan antara aplikasi sebelumnya dengan aplikasi yang telah dikembangkan dapat dilihat pada tabel 3.2.

U N I V E R S I T A S  
M U L T I M E D I A  
N U S A N T A R A

Tabel 3.2. Pekerjaan yang dilakukan tiap minggu selama pelaksanaan kerja magang

Aplikasi Sebelumnya	Aplikasi Sekarang
Belum ada fitur-fitur AI	Ada fitur-fitur AI
Belum ada fitur <i>text to speech</i>	Ada fitur <i>text to speech</i>
Belum ada fitur <i>settings</i>	Ada fitur <i>settings</i>
Belum ada fitur <i>car interior</i>	Ada fitur <i>car interior</i>
Hanya ada NPC pengunjung	Ada NPC pengunjung, <i>guard</i> , dan <i>salesperson</i>
Belum ada fitur <i>tutorial</i>	Ada fitur <i>tutorial</i>

### 3.3.2 Pengembangan Aplikasi

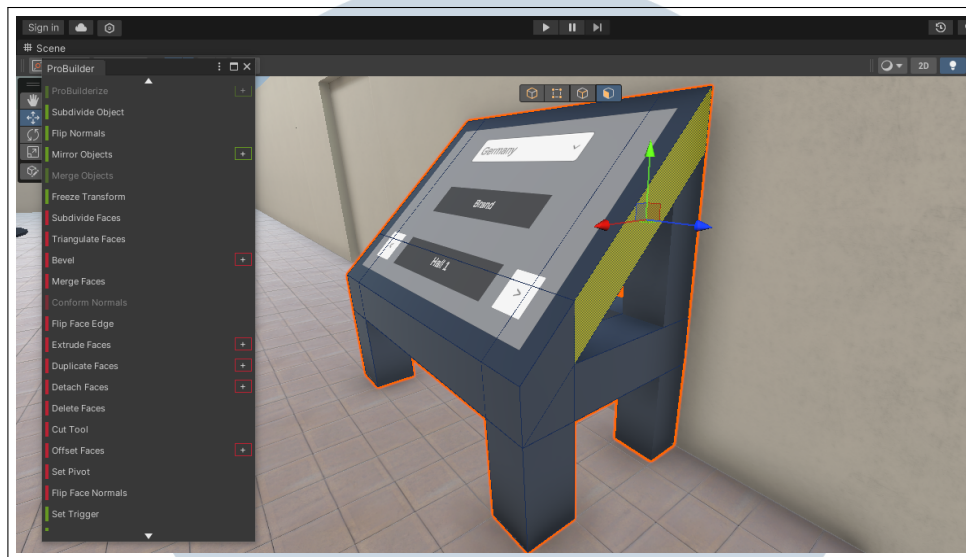
Kegiatan pengembangan aplikasi yang dilakukan melibatkan pengembangan sebuah aplikasi yang mengimplementasikan rancangan aplikasi yang telah dibuat. Hal tersebut dilakukan menggunakan *game engine* Unity. Proses ini melewati beberapa iterasi berdasarkan pergantian pada desain aplikasi yang dibuat. Pekerjaan dilaksanakan pada beberapa bagian berbeda pada aplikasi selama berjalannya magang ini.

#### A. Integrasi Aset-Aset 3D

Bagian ini mengacu kepada proses pengumpulan aset-aset 3D yang digunakan dalam aplikasi serta proses integrasi aset-aset tersebut dengan sistem *virtual reality* pada aplikasi. Aset 3D dipakai dalam pembangunan lingkungan *virtual* seperti membuat gedung pameran. Mereka juga digunakan untuk objek-objek yang pengguna dapat berinteraksi dengan seperti mobil-mobil pada pameran. Aset-aset yang digunakan merupakan campuran antara aset buatan sendiri dan aset yang diunduh dari internet.

Pembuatan aset 3D terutama dilakukan dalam *game engine* Unity. Ada dua alat yang disediakan oleh Unity yang digunakan untuk melakukan 3D modelling yaitu ProBuilder dan Terrain Tools. ProBuilder berfungsi untuk memodelkan bentuk-bentuk sederhana seperti kubus atau silinder. Alat ini digunakan untuk memodelkan gedung pameran dengan membuat dinding, pintu, lantai, dan jendela yang digunakan. Probuilder juga digunakan untuk membuat hal lain pada aplikasi seperti meja atau pajangan mobil. Gambar 3.1 menunjukkan tampilan probuilder saat sedang digunakan dalam Unity. Terrain Tools berfungsi untuk memodelkan

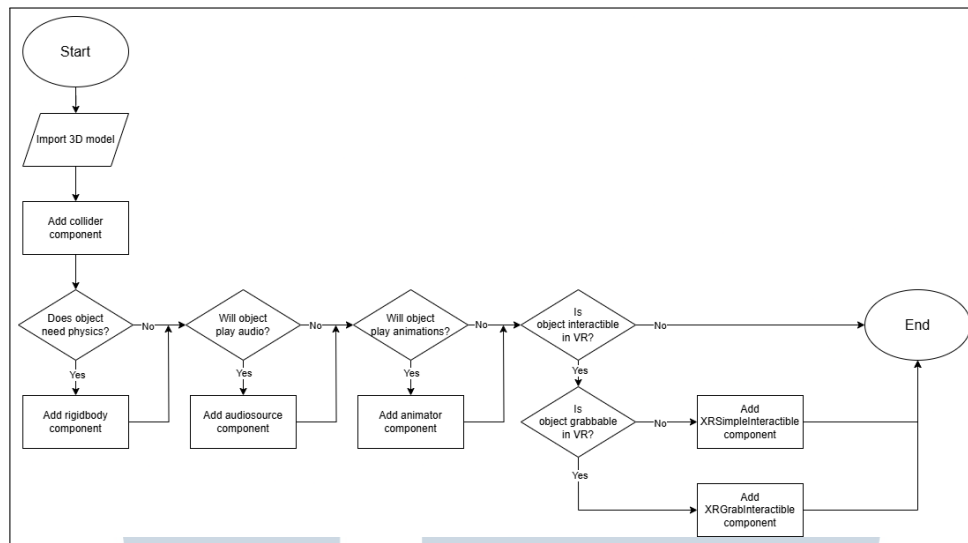
sebuah lahan atau medan. Alat ini digunakan untuk membuat daerah di luar gedung pameran.



Gambar 3.1. Penggunaan ProBuilder dalam Unity

Pengunduhan aset-aset 3D dari internet dilakukan melalui beberapa situs web berbeda. TurboSquid, Sketchfab, dan CGTrader merupakan contoh situs web yang digunakan. Mereka merupakan beberapa situs web dimana orang dapat membagi karya 3D model mereka dengan orang lain baik secara gratis atau berbayar. Kebanyakan model yang diunduh merupakan model 3D mobil yang digunakan untuk pameran mobil pada aplikasi. Juga ada model-model lain seperti bangku dan tanaman yang digunakan sebagai dekorasi.

Proses integrasi dilakukan lalu dilakukan supaya pengguna dapat berinteraksi dengan aset-aset 3D dalam aplikasi. Hal ini melibatkan penggunaan komponen-komponen yang telah disediakan dalam *game engine* Unity. Komponen ditambahkan kepada sebuah objek 3D untuk menambahkan perilaku kepada objek tersebut. Ada beberapa komponen-komponen yang digunakan seperti Collider untuk menambahkan *collision*, Rigidbody untuk menambahkan fisika, AudioSource untuk memainkan audio, dan Animator untuk memainkan animasi. Alur dari proses integrasi aset-aset 3D pada aplikasi dapat dilihat pada gambar 3.2.



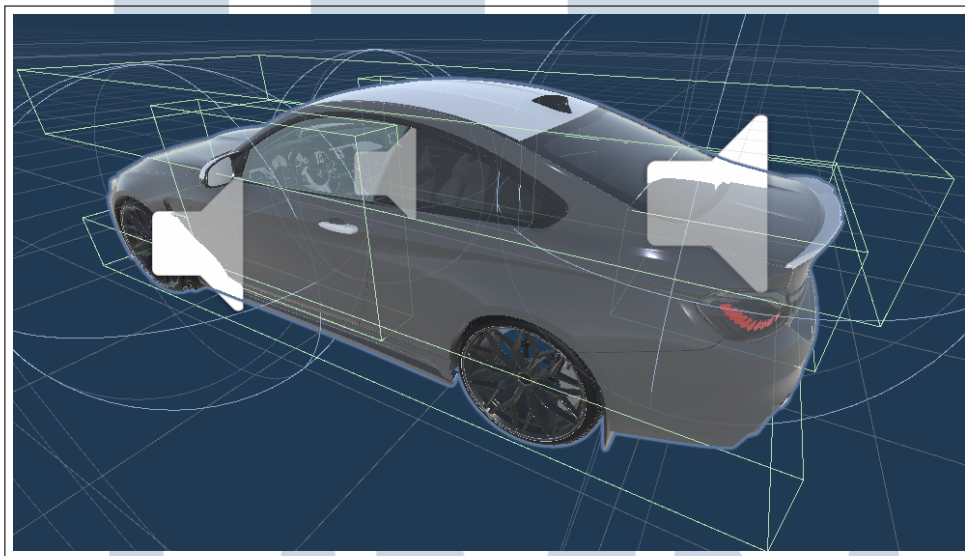
Gambar 3.2. Flowchart Integrasi Aset-Aset 3D

Integrasi dengan perangkat *virtual reality* menggunakan komponen-komponen yang lebih spesifik untuk pengembangan aplikasi berbasis *virtual reality*. Komponen-komponen ini didapatkan dari package XRInteractionToolkit yang telah disediakan oleh Unity. Terdapat dua komponen dari package tersebut yang digunakan dalam integrasi aset-aset 3D dengan perangkat *textitvirtual reality*. Yang pertama adalah komponen XRSimpleInteractible. Komponen ini memungkinkan pengguna untuk berinteraksi dengan sebuah objek menggunakan sebuah *controller virtual reality*. Saat pengguna berinteraksi dengan objek, komponen akan menjalankan sebuah aksi yang telah ditentukan melalui *game engine* Unity atau kode yang telah ditulis. Komponen kedua merupakan komponen XRGrabInteractable. Komponen ini memungkinkan pengguna untuk mengambil sebuah objek 3D menggunakan *controller virtual reality* seperti mengambil barang menggunakan tangan.

Berikut adalah proses pencarian dan integrasi model-model 3D mobil yang digunakan dalam pameran *virtual* pada aplikasi. Tahap pertama adalah untuk mencari model 3D yang sesuai untuk digunakan. Model 3D yang dicari adalah model mobil yang memiliki sebuah interior yang dapat dilihat pengguna. Model juga perlu terbagi menjadi model-model lebih kecil supaya pengguna dapat berinteraksi dengan bagian spesifik pada mobil. Setelah sebuah model 3D telah ditemukan dan diunduh, model 3D tersebut akan diimpor ke dalam Unity. Lalu, beberapa komponen akan ditambahkan ke model 3D. Komponen Collider ditambahkan ke bagian bodi, pintu, dan pintu bagasi mobil untuk menambahkan *collision*. Komponen XRSimpleInteractable, AudioSource, dan Animator lalu



ditambahkan ke pintu dan pintu bagasi mobil untuk memungkinkan pengguna untuk berinteraksi dengan bagian tersebut pada mobil. Interaksi yang diinginkan adalah untuk pengguna dapat membuka dan menutup pintu pada mobil. Komponen Animator digunakan untuk memainkan animasi buka dan tutup pada pintu. Komponen Audiosource digunakan untuk memainkan audio buka dan tutup pintu. Dan komponen XRSimpleInteractable digunakan untuk memainkan animasi dan audio pintu saat pengguna berinteraksi dengan aksi *select* menggunakan *controller virtual reality* pada pintu. Hasil integrasi model 3D mobil dalam Unity dapat dilihat pada gambar 3.3.



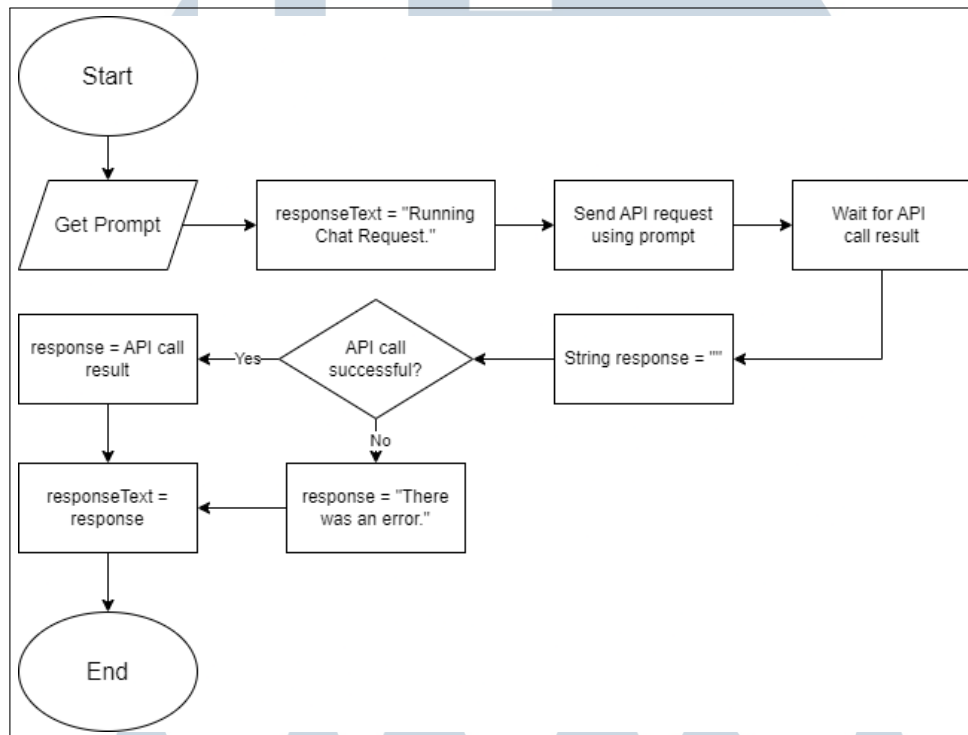
Gambar 3.3. Hasil Integrasi Model 3D Mobil

## B. API Google Gemini

Google Gemini merupakan sebuah *generative artificial intelligence* yang dikembangkan oleh Google. Google Gemini dapat digunakan untuk menghasilkan berbagai macam konten seperti teks dan gambar. Pada aplikasi, Google Gemini diimplementasikan untuk menggenerasikan beberapa konten yang akan ditampilkan ke pengguna. Hal ini dilakukan menggunakan API Google Gemini yang telah disediakan oleh Google.

Alur kerja penggunaan API Google Gemini pada aplikasi dapat dilihat pada gambar 3.4. Aplikasi akan pertama mendapatkan sebuah *prompt* dari pengguna. Teks respons lalu diganti menjadi "Running Chat Request" untuk menginformasikan kepada pengguna bahwa *prompt* mereka sedang diproses. Sebuah API call dilakukan menggunakan *prompt* tersebut dan aplikasi akan

menunggu hasil dari API call. Sebuah variabel string *response* juga dibuat dengan nilai kosong. Aplikasi lalu akan cek apakah hasil dari API call sukses atau tidak. Jika API call sukses, variabel *response* diberi nilai hasil API call. Jika API call tidak sukses, variabel *response* diberi nilai "There was an error.". Teks respons lalu diganti menjadi nilai dari variabel *response*.



Gambar 3.4. Flowchar API Google Gemini

Fitur *AI Chatbot* dikembangkan menggunakan API Google Gemini. *AI Chatbot* yang dihasilkan berupa sebuah meja dimana pengguna dapat mengetik *prompt* yang diinginkan dan mendapatkan jawaban berdasarkan apa yang dimasukan. Tampilan *AI Chatbot* dapat dilihat pada gambar 3.5. Untuk memasukan sebuah *prompt*, pengguna pertama perlu *select* kolom input menggunakan *controller virtual reality* yang sedang digunakan. Setelah itu pengguna dapat mengetik *prompt* yang diinginkan menggunakan *keyboard virtual* yang disediakan pada *AI Chatbot*. Saat pengguna menekan tombol *Enter*, *prompt* yang telah diketik akan dikirim ke API Google Gemini. Google Gemini lalu akan menghasilkan sebuah respons menggunakan *prompt* yang disediakan oleh pengguna. Respons ini akan dikirim kembali ke aplikasi dan ditampilkan oleh meja *AI Chatbot*.





Gambar 3.5. AI Chatbot

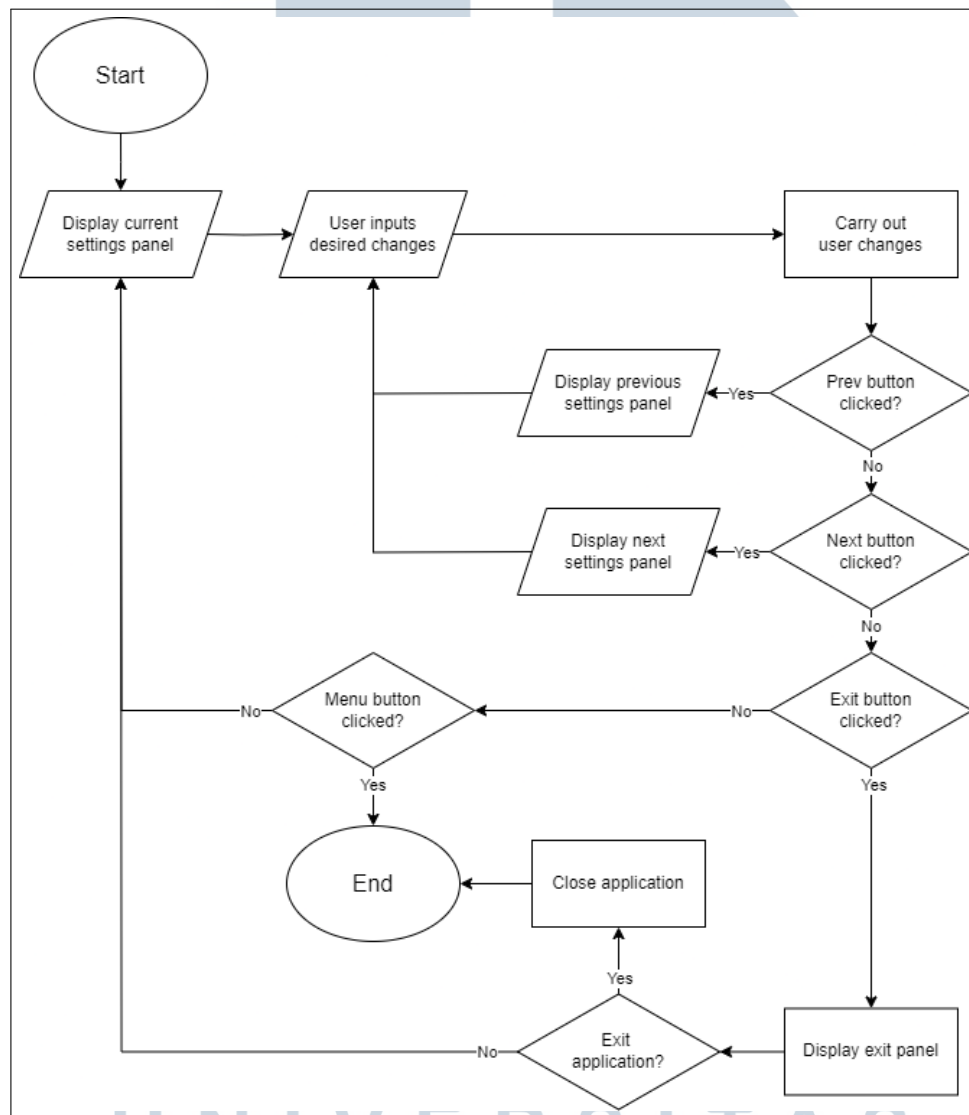
API Google Gemini juga digunakan untuk menampilkan informasi mengenai mobil-mobil yang ada pada pameran. Setiap mobil disimpan pada aplikasi dalam bentuk *scriptable object*. *Scriptable object* ini mengandung data mengenai sebuah mobil seperti nama, merek dan 3D model mobilnya. Data tersebut digunakan untuk mengirim sebuah *prompt* mengenai mobilnya ke API Google Gemini. Aplikasi akan melakukan *prompt* seputar deskripsi dan fitur-fitur untuk setiap mobil. Teks yang dihasilkan oleh Google Gemini lalu akan ditampilkan pada setiap stand mobil masing-masing.

### C. Settings

Fitur *Settings* dibuat untuk memberikan pengguna cara untuk mengganti beberapa aspek dari aplikasi. *Settings* dapat diakses dalam bentuk *gameobject* dengan panel-panel yang mengandung opsi-opsi bagi pengguna. Panel *settings* dimunculkan dengan menekan tombol menu pada *controller virtual reality*. Setelah dimunculkan, panel *Settings* akan berotasi secara otomatis untuk selalu menghadap pengguna. Panel *settings* juga akan menghilang secara otomatis jika pengguna bergerak terlalu jauh darinya.

Alur penggunaan fitur *settings* dapat dilihat pada gambar 3.6. Aplikasi akan mulai dengan menampilkan sebuah panel dimana pengguna memiliki opsi untuk mengganti beberapa aspek dari aplikasi. Ada beberapa hal yang dapat dilakukan setelah pengguna memasukkan apa yang ingin diganti. Pengguna dapat menekan tombol *prev* untuk menampilkan panel sebelumnya. Pengguna dapat menekan

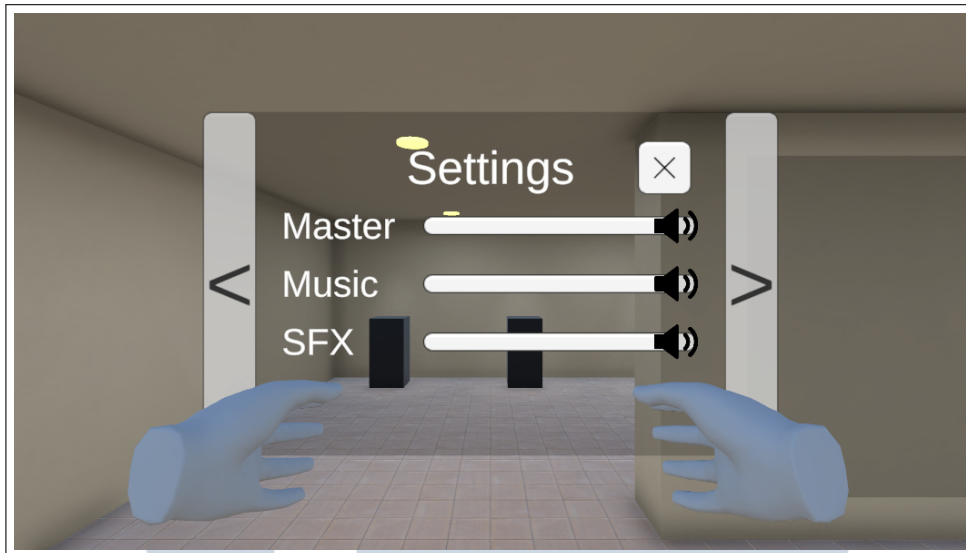
tombol *next* untuk menampilkan panel selanjutnya. Dan, pengguna dapat menekan tombol *exit* untuk menampilkan panel *exit*. Pengguna dapat memilih untuk keluar dari aplikasi pada panel *exit*. Selain itu, fitur *settings* akan ditutup saat pengguna menekan tombol menu pada *controller* yang digunakan.



Gambar 3.6. Flowchart *Settings*

Ada beberapa hal yang pengguna dapat ganti melalui fitur *settings*. Pengguna dapat mengatur volume audio aplikasi seperti yang dapat dilihat pada gambar 3.7. Audio pada aplikasi dibagi menjadi *master*, *music*, dan *SFX* yang dapat diatur secara terpisah. Selain itu pengguna dapat mengatur sistem navigasi pada aplikasi. Disini pengguna dapat mengaktifkan beberapa metode navigasi seperti teleportsasi, *snap turn*, dan penggunaan keyboard. Pengguna juga dapat mengatur

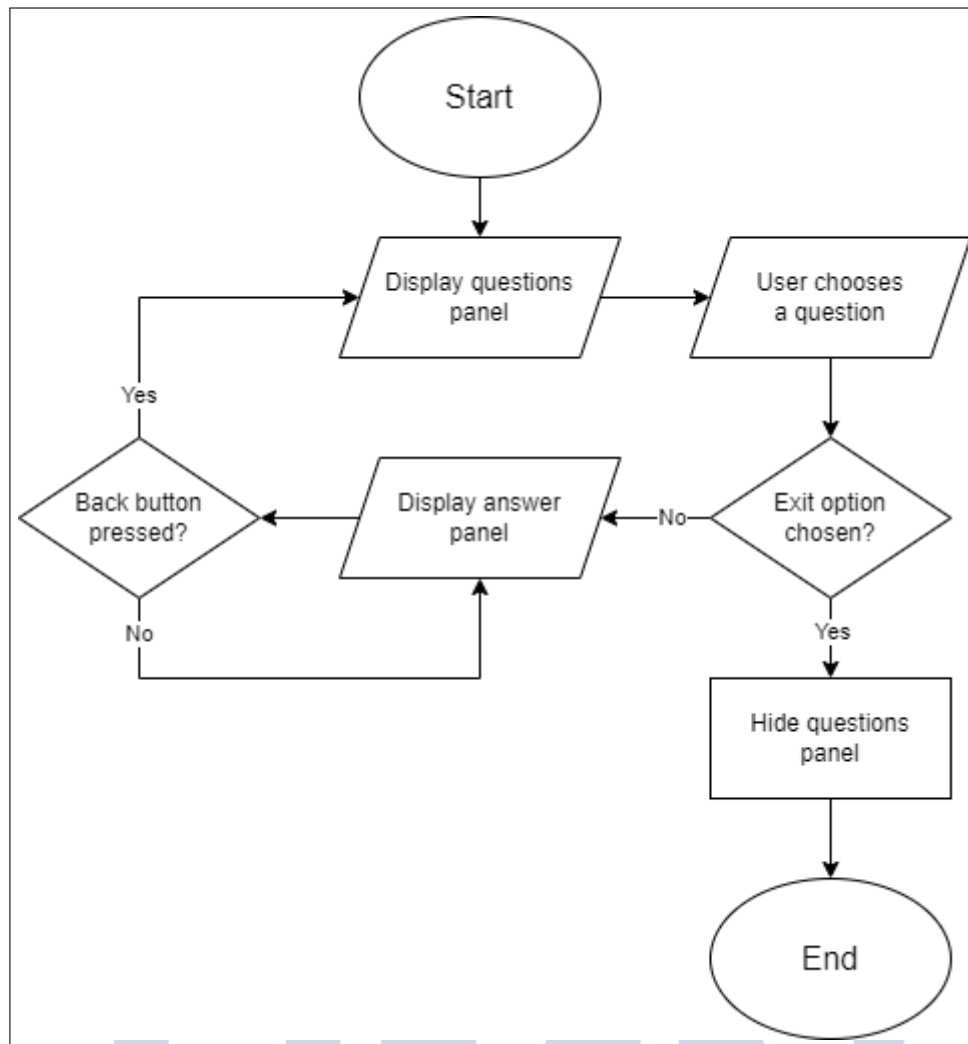
kecepatan dari pergerakan dan rotasi pada aplikasi.



Gambar 3.7. Panel *Settings*

#### D. Karakter Non-Pemain

Fitur karakter non-pemain atau NPC pada aplikasi mengacu kepada karakter-karakter yang tidak bisa dikendalikan oleh pengguna. Ada beberapa karakter non-pemain yang ditambahkan selama proses pekerjaan magang ini. Yang pertama adalah karakter non-pemain *guard*. Karakter non-pemain ini berfungsi untuk memberi informasi mengenai aplikasi kepada pengguna. Pengguna dapat berinteraksi dengan NPC *guard* dengan melakukan *select* menggunakan *controller virtual reality* yang sedang digunakan. Alur dari NPC *guard* dapat dilihat pada gambar 3.8. NPC *guard* akan mulai dengan memunculkan panel dimana pengguna dapat memilih antara beberapa pertanyaan yang sudah ditentukan. Jika pengguna memilih opsi, panel pertanyaan akan ditutup. Jika tidak, NPC *guard* akan menunjukkan panel jawaban berisi jawaban untuk pertanyaan yang dipilih oleh pengguna. NPC *guard* lalu akan balik ke panel pertanyaan saat pengguna menekan tombol *back*.



Gambar 3.8. Flowchart NPC *Guard*

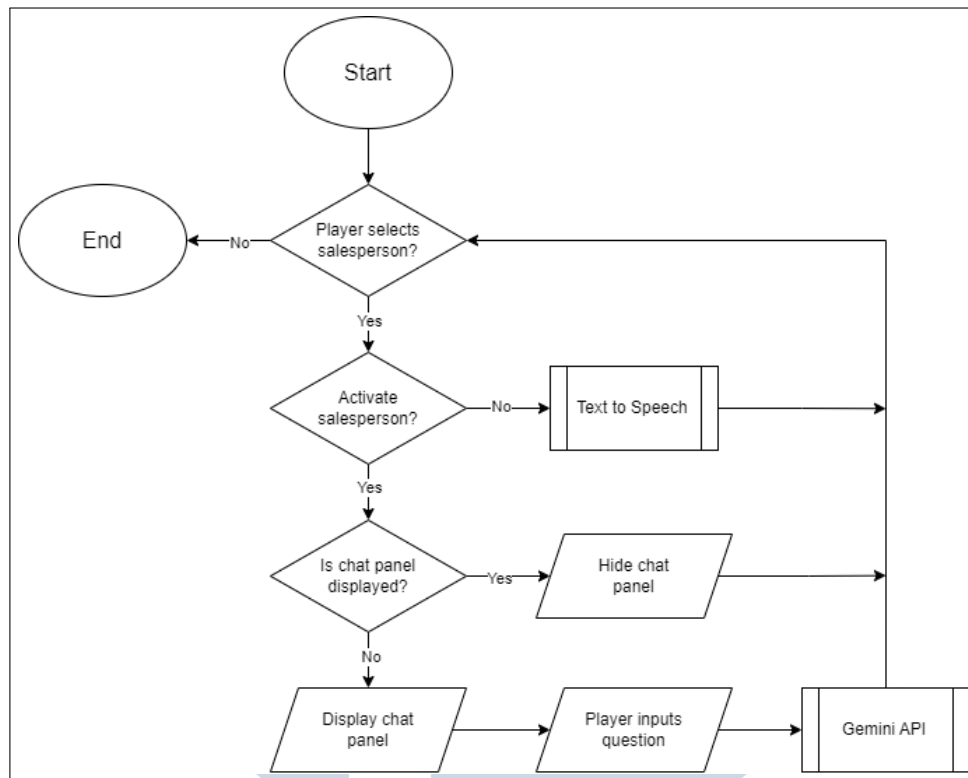
Ada dua NPC *guard* yang ditambahkan. NPC *guard* pertama berada di ruangan *main gate*. Pengguna dapat bertanya kepada NPC *guard* pertama pertanyaan seputar cara penggunaan aplikasi seperti cara interaksi dengan objek atau melakukan teleportasi. Tampilan dari NPC *guard* pertama dapat dilihat pada gambar 3.9. NPC *guard* kedua berada di ruangan pameran. Pengguna dapat bertanya kepada NPC *guard* kedua pertanyaan seputar fitur-fitur pameran seperti fitur pencarian mobil atau interaksi dengan mobil.



Gambar 3.9. NPC *Guard* pada Ruangan *Main Gate*

Karakter non-pemain kedua yang ditambahkan adalah karakter non-pemain *salesperson*. Karakter non-pemain *salesperson* terletak pada setiap stand mobil yang memiliki sebuah produk pajangan. NPC *salesperson* digunakan untuk bertanya tentang mobil yang ada pada stand. Alur dari NPC *salesperson* dapat dilihat pada gambar 3.10. NPC *salesperson* mulai dengan pengguna melakukan *select* menggunakan tombol *grip* pada *controller* mereka. NPC *salesperson* lalu akan cek jika pengguna juga menekan tombol *trigger* selama tombol *grip* sedang ditahan untuk melakukan *activate*. Jika tidak, NPC *salesperson* akan menjalankan fitur *text to speech*. Jika tombol *trigger* juga ditekan, NPC *salesperson* akan menampilkan panel *chat* jika belum ditampilkan dan menutupnya jika sudah ditampilkan. Pada panel *chat*, pengguna dapat menggunakan fitur *AI Chatbot* untuk bertanya kepada NPC *salesperson* pertanyaan mengenai mobil yang dipajang pada stand.

U N I V E R S I T A S  
M U L T I M E D I A  
N U S A N T A R A

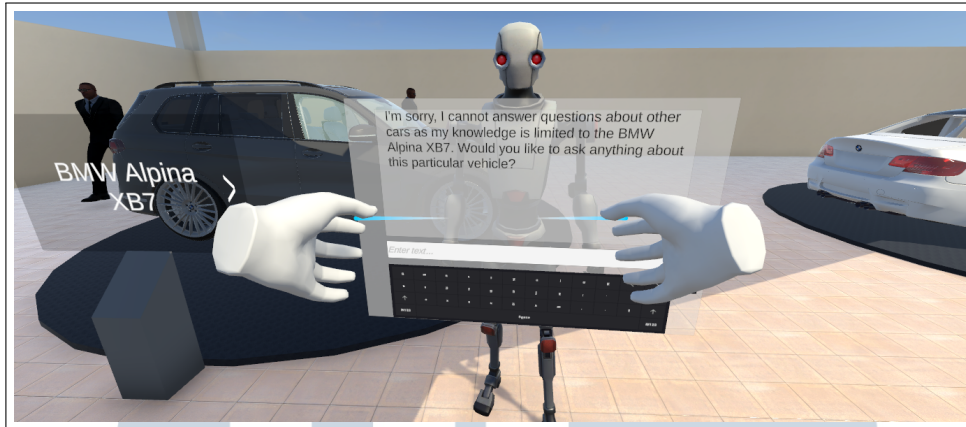


Gambar 3.10. Flowchart NPC *Salesperson*

Hasil interaksi dengan NPC *salesperson* tergantung apakah panel *chat* sedang dimunculkan atau tidak. Saat panel *chat* sedang dimunculkan seperti pada gambar 3.11, NPC *salesperson* akan menghasilkan audio *text to speech* untuk teks respons yang dihasilkan sebagai jawaban untuk pertanyaan pengguna. Melakukan aksi *activate* juga akan membuat panel *chat* menghilang. Sebaliknya, NPC *salesperson* akan menghasilkan audio *text to speech* untuk teks pada panel info stand mobil jika panel *chat* sedang tidak dimunculkan. Aksi *activate* juga akan memunculkan panel *chat*.

U N I V E R S I T A S  
M U L T I M E D I A  
N U S A N T A R A



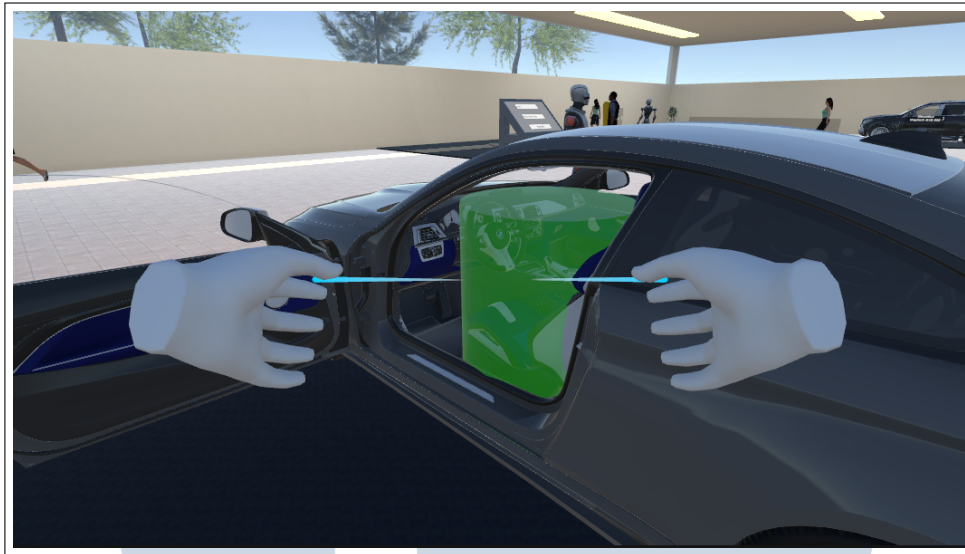


Gambar 3.11. NPC Salesperson dengan Panel Chatbot

### E. Car Interior

Fitur *car interior* mengacu kepada fitur pameran yang dikembangkan untuk pameran mobil pada aplikasi dimana pengguna dapat masuk ke dalam 3D model sebuah mobil. Dengan ini, pengguna dapat merasa seperti sedang duduk dalam mobil tersebut dan dapat melihat bagian dalam mobil dengan lebih dekat. Pengguna dapat memasuki bagian dalam mobil dengan melakukan *select* menggunakan *controller virtual reality* pada *teleportation anchor* yang muncul dalam mobil. Setiap mobil pada pameran akan mempunyai *teleportation anchor* pada setiap tempat duduk mobil. *Teleportation anchor* hanya akan muncul saat pengguna melakukan *hover* padanya. Tampilan *teleportation anchor* dapat dilihat pada gambar 3.12.

U N I V E R S I T A S  
M U L T I M E D I A  
N U S A N T A R A

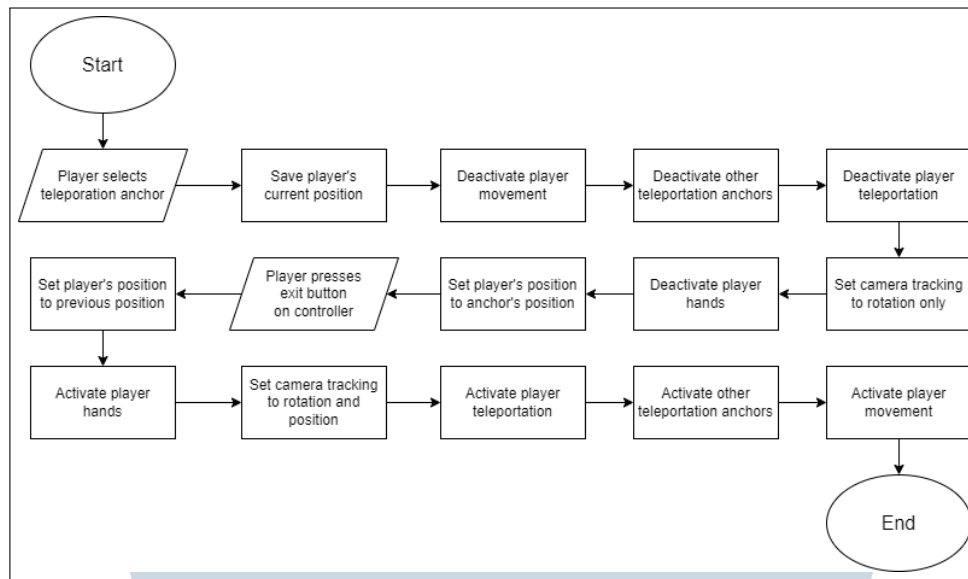


Gambar 3.12. *Teleportation Anchor* pada Mobil

Saat pengguna melakukan *select* pada sebuah *teleportation anchor*, pengguna akan diletakan ke lokasi *teleportation anchor* tersebut. Hal ini dapat dilihat pada gambar 3.13. Disini pengguna tidak dapat bergerak namun tetap dapat berotasi dan menggerakkan kepala mereka. Untuk keluar dari bagian dalam mobil, pengguna dapat menekan tombol trigger pada *controller virtual reality* yang digunakan. Pengguna akan dikembalikan ke lokasi sebelum masuk bagian dalam mobil dan akan dapat bergerak kembali seperti sebelumnya. Alu dari fitur *car interior* ini dapat dilihat pada gambar 3.14.



Gambar 3.13. Bagian Dalam Mobil

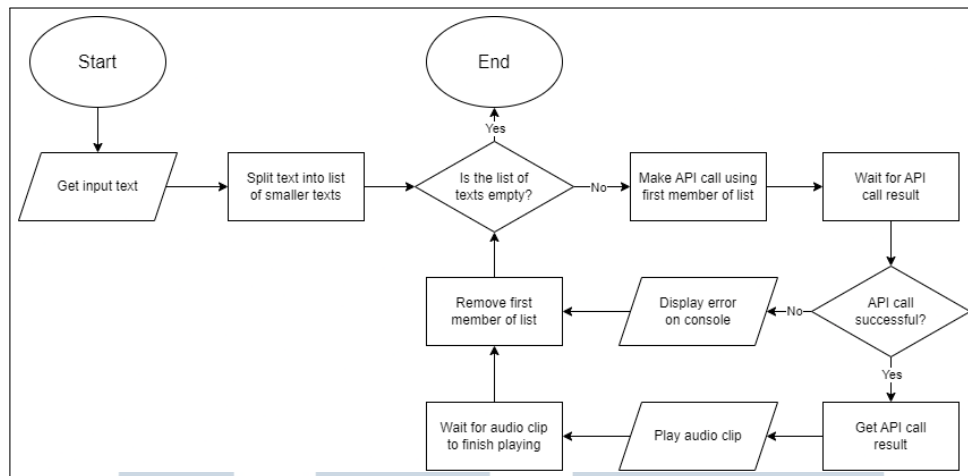


Gambar 3.14. Flowchart *Car Interior*

## F. Text to Speech

Fitur *text to speech* pada aplikasi dibuat untuk menghasilkan sebuah transkripsi audio berdasarkan sebuah teks. Aplikasi menggunakan *text to speech* milik Google Translate untuk menghasilkan audionya. Hal ini dilakukan dengan mengirim teks yang ingin digunakan ke Google Translate. Google Translate lalu akan menghasilkan audio yang diinginkan dan mengirimnya balik ke aplikasi. Audio tersebut lalu dapat dimainkan melalui sebuah komponen *audiosource* dalam Unity.

Alur dari fitur *text to speech* dapat dilihat pada gambar 3.15. Tahap pertama adalah untuk mendapatkan teks yang ingin digunakan untuk menghasilkan audio *text to speech*. Teks tersebut lalu akan dibagi menjadi bagian dan disimpan dalam sebuah lis. Hal ini dilakukan karena API Google Translate tidak dapat menghasilkan audio untuk teks yang terlalu panjang. API call akan dilakukan untuk setiap teks pada lis yang dihasilkan satu per satu. Aplikasi akan menunggu hasil dari API call yang dilakukan dan cek apakah hasil tersebut sukses. Jika hasil tidak sukses, aplikasi akan menampilkan pesan error pada *console*. Jika hasilnya sukses, aplikasi akan mengakses audio yang telah dihasilkan. Aplikasi lalu akan memainkan audio tersebut dan menunggu hingga audionya telah selesai sebelum lanjut ke audio berikutnya.



Gambar 3.15. Flowchart *Text to Speech*

## G. Tutorial

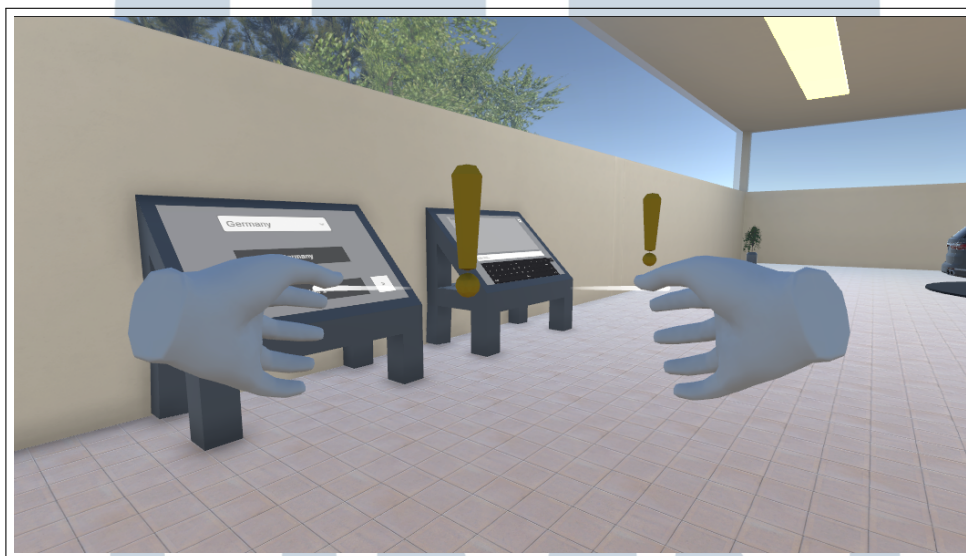
Fitur *tutorial* dibuat pada aplikasi untuk membantu mengarahkan pengguna dalam penggunaan aplikasi. Pada awal penggunaan aplikasi, teks *tutorial* akan dimunculkan pada layar pengguna untuk menjelaskan sistem pergerakan pada aplikasi. Sistem *tutorial* akan menjelaskan cara bergerak, berputar, dan teleportasi kepada pengguna. Hal ini dilakukan dengan menampilkan teks untuk menjelaskan input yang diperlukan untuk masing-masing metode pergerakan satu per satu seperti yang dapat dilihat pada gambar 3.16. Teks tersebut hanya akan lanjut ke teks selanjutnya setelah pengguna sudah melakukan input yang diminta dan mencoba metode pergerakannya.



Gambar 3.16. Flowchart *Text to Speech*

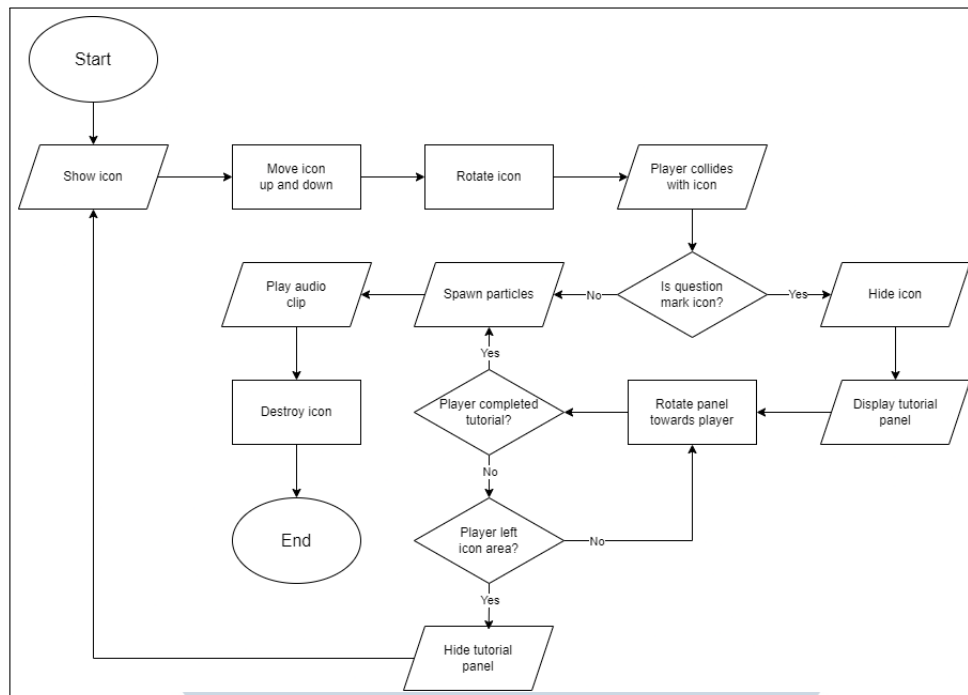


Sistem *tutorial* juga ditambahkan dalam aplikasi dalam bentuk ikon 3D pada lingkungan *virtual* aplikasi. Ada dua tipe ikon yang digunakan. Ikon pertama berupa sebuah tanda tanya. Saat pengguna mendekati ikon tersebut, sebuah teks akan muncul yang akan menjelaskan suatu aspek dari aplikasi yang telah dibuat. Teks serta ikonnya akan menghilang setelah pengguna melakukan aksi yang dijelaskan oleh teksnya. Ikon kedua berupa sebuah tanda seru seperti yang dapat terlihat pada gambar 3.17. Ikon ini digunakan untuk mendapatkan perhatian pengguna dan mengarahkan mereka kepada fitur-fitur pameran yang ada. Ikon tanda seru akan menghilang saat pengguna mendekatinya.



Gambar 3.17. Flowchart *Text to Speech*

Alur dari fitur ikon *tutorial* dapat dilihat pada gambar 3.18. Aplikasi mulai dengan menampilkan ikon *tutorial*. Ikon *tutorial* lalu akan bergerak dan berotasi di tempat. Tahap selanjutnya terjadi saat pengguna mendekati ikon *tutorial*. Jika ikonnya berupa sebuah tanda seru, ikon tersebut langsung akan menghilang. Hal ini disertai dengan munculnya partikel dan audio. Jika ikonnya berupa sebuah tanda tanya, ikonnya akan disembunyikan dan sebuah panel *tutorial* akan dimunculkan. Panel tersebut dibuat agar selalu berotasi untuk menghadapi pengguna. Jika pengguna melakukan aksi yang tertulis pada panel tutorial, maka objek ikonnya akan menghilang disertai munculnya partikel dan audio. Jika tidak, aplikasi akan cek apakah pengguna masih berada dekat dengan ikonnya. Jika pengguna masih dekat, panel *tutorial* akan terus dimunculkan. Jika pengguna sudah menjauh, panel *tutorial* akan disembunyikan dan ikon akan dimunculkan kembali.



Gambar 3.18. Flowchart Ikon Tutorial

### 3.3.3 Testing dan Evaluasi Aplikasi

Kegiatan testing dilakukan untuk menguji apakah hasil implementasi perancangan aplikasi dapat berjalan dengan benar. Setiap fitur yang ditambahkan akan diuji untuk melihat fungsionalitasnya. Hal ini dilakukan dengan melakukan *white box testing* untuk fitur tersebut. Dengan ini, logika dan alur dari kode dan fitur yang dikembangkan diuji dengan menjalankan *test cases* untuk melihat kinerjanya. Jika sebuah error atau *bug* ditemukan maka proses *debugging* akan dilakukan untuk memperbaiki fiturnya. Testing juga dilakukan dengan meminta pengguna untuk menjalankan aplikasinya. Proses ini membantu untuk melihat bagaimana pengguna menggunakan aplikasinya dan menemukan error yang terlewatkan.

Pada tahap evaluasi, fitur-fitur yang telah diimplementasi akan diuji untuk menilai kualitas dan kegunaannya. Evaluasi dilakukan bersama dengan supervisor pada pertemuan mingguan yang dilakukan secara *online*. Hal ini dilakukan dengan menunjukan sebuah demonstrasi dari fitur yang telah dikembangkan. Jika sebuah fitur tidak sebaik yang diharapkan maka perancangan ulang akan dilakukan. Sebuah iterasi baru untuk fitur tersebut lalu akan dikembangkan. Namun jika sebuah fitur sudah dirasa bagus maka fitur tersebut akan dianggap selesai dan pekerjaan akan dimulai untuk fitur selanjutnya.



Evaluasi juga dilakukan dengan pengguna serta supervisor menggunakan sebuah *user acceptance test* untuk mengukur. Dengan ini pengguna mengisi sebuah kuesioner setelah mencoba aplikasinya untuk memberi penilaian. *User acceptance test* yang dilakukan terdiri dari sembilan pertanyaan. Pengguna diminta untuk memberikan penilaian untuk berbagai aspek aplikasi seperti penampilan aplikasi, konten pada aplikasi, UI aplikasi, dan lainnya. Jawaban diberi dalam bentuk Likert Scale dimana nilai satu merupakan nilai terendah dan nilai lima merupakan nilai tertinggi. Hasil *user acceptance test* dengan semua pengguna dapat dilihat pada tabel 3.3. Evaluasi yang diberi oleh supervisor dapat dilihat pada tabel 3.4.

Tabel 3.3. Hasil dari User Acceptance Test dengan supervisor

Pertanyaan	Nilai				
	1	2	3	4	5
Apakah kualitas penampilan dari aplikasi baik?	0	0	0	2	3
Apakah kualitas sistem pergerakan pada aplikasi (berjalan, berputar, teleportasi) baik?	0	0	0	1	4
Apakah kualitas interaksi dengan objek-objek dan UI pada aplikasi baik?	0	0	1	3	1
Apakah kualitas audio pada aplikasi baik?	0	0	1	3	1
Apakah kualitas konten yang ditampilkan pada pameran baik?	0	0	0	0	5
Apakah informasi yang diberikan mengenai produk bersifat informatif?	0	0	0	0	5
Apakah aplikasi intuitif dan mudah untuk digunai?	0	0	1	2	2
Apakah pengalaman menggunakan aplikasi bersifat imersif?	0	0	0	2	3
Secara keseluruhan, apakah anda puas dengan aplikasi Virtual Reality Expo?	0	0	0	1	4
Rata-rata	4.58				

Tabel 3.4. Hasil dari User Acceptance Test dengan supervisor

Pertanyaan	Nilai				
	1	2	3	4	5
Apakah kualitas penampilan dari aplikasi baik?					✓
Apakah kualitas sistem pergerakan pada aplikasi (berjalan, berputar, teleportasi) baik?					✓
Apakah kualitas interaksi dengan objek-objek dan UI pada aplikasi baik?					✓
Apakah kualitas audio pada aplikasi baik?					✓
Apakah kualitas konten yang ditampilkan pada pameran baik?					✓
Apakah informasi yang diberikan mengenai produk bersifat informatif?					✓
Apakah aplikasi intuitif dan mudah untuk digunai?					✓
Apakah pengalaman menggunakan aplikasi bersifat imersif?					✓
Secara keseluruhan, apakah anda puas dengan aplikasi Virtual Reality Expo?					✓
Rata-rata	5				

Rata-rata dari nilai yang diberi pengguna adalah 4.58. Nilai ini menunjukkan bahwa pengguna yang telah mencoba aplikasi merasa sangat puas dengan pengalamannya secara keseluruhan. Pengguna memberi nilai tertinggi kepada kualitas konten serta informasi produk pada aplikasi dengan nilai rata-rata 5. Sedangkan, nilai terendah diberi ke kualitas audio serta interaksi pengguna dengan objek dan UI pada aplikasi. Dapat disimpulkan bahwa kualitas konten dalam bentuk produk pameran dan implementasi API Google Gemini untuk menampilkan informasi produk sudah cukup baik. Namun, kualitas sistem interaksi, tampilan UI, dan audio masih dapat ditingkatkan.

### 3.4 Kendala dan Solusi yang Ditemukan

Berikut adalah kendala-kendala yang dialami selama menjalankan magang:

1. Hardware yang kurang optimal untuk pengembangan aplikasi *Virtual Reality*. Laptop pribadi yang digunakan tidak mempunyai *graphics card* yang dapat

mendukung penggunaan aplikasi *virtual reality*. Sedangkan, komputer pada *lab game development* memiliki spesifikasi yang rendah dan *headset virtual reality* pada lab berada dalam kondisi yang buruk.

2. Tidak adanya pendanaan untuk penggunaan API pada aplikasi.

Berikut adalah solusi yang ditemukan untuk menyelesaikan kendala yang ditemukan:

1. Pengembangan terutama dilakukan menggunakan laptop pribadi lalu *testing* dilakukan pada komputer lab. Aplikasi di *build* terlebih dahulu dibandingkan dijalankan dalam *Unity editor* untuk menghemat sumber daya komputer. *Lab game development* juga mendapatkan *headset virtual reality* baru dekat akhir periode magang yang memudahkan proses *testing*.
2. Alternatif bersifat gratis untuk API yang sering digunakan dicari untuk digunakan dalam aplikasi. API Google Gemini digunakan sebagai alternatif untuk API ChatGPT. API Google Translate digunakan sebagai alternatif untuk API *Text to Speech* berbayar lainnya.

