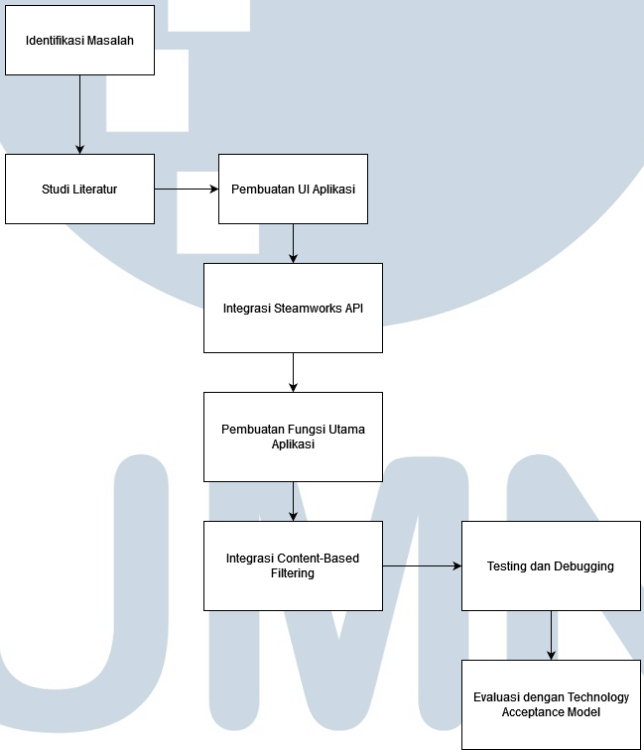
BAB 3
METODOLOGI PENELITIAN

Dalam penelitian ini, pengembangan aplikasi dilakukan dengan cara pengembangan bertahap, Beberapa fitur utama seperti integrasi API Steamworks dan sistem UI utama akan dikerjakan terlebih dahulu agar fungsi-fungsi lainnya dapat bekerja sesuai rencana.

3.1 Kerangka Penelitian



Gambar 3.1. Diagram Kerangka Penelitian

Berdasarkan diagram kerangka penelitian 3.1, berikut ini adalah penjelasan dari setiap bagian tersebut:

1. Identifikasi Masalah

   Tahapan pertama yang dilakukan dalam penelitian ini adalah mengidentifikasi masalah yang ada, yaitu banyaknya pilihan *game* dalam layanan *Steam* yang dapat menyebabkan pengguna menjadi kurang tertarik dalam membeli *game*

menurut penelitian yang dilakukan oleh Chernev, Böckenholt, dan Goodman pada tahun 2014[5].

2. Studi Literatur
   Dilakukanlah studi literatur untuk mencari berbagai penelitian serupa dan mencari sebuah cara untuk menyelesaikan masalah dengan metode yang dipilih. Selain itu, studi literatur juga dilakukan untuk lebih memperdalam pengetahuan tentang metode yang digunakan.

3. Pembuatan UI Aplikasi
   Pertama, UI aplikasi dibuat dengan *prototype* yang dibuat dengan layanan *Figma*. Prototipe ini lalu akan digunakan sebagai referensi dalam membuat tampilan aplikasi.

4. Integrasi Steamworks Web API
   Pada bagian ini, *Steamworks Web API* akan diintegrasikan ke dalam aplikasi melalui sistem login, dan fungsi yang akan mengambil data seperti *SteamID*, data *game*, informasi tambahan sebuah *game*, jumlah *game* yang dimiliki pengguna, dan *game* yang dimainkan dalam dua minggu terakhir.

5. Pembuatan Fungsi Utama Aplikasi
   Pada bagian ini, fungsi utama aplikasi seperti halaman profil, halaman rekomendasi, dan hasil rekomendasi akan dibuat dan dikembangkan.

6. Integrasi Content-Based Filtering
   Di bagian ini, *content-based filtering* akan diintegrasikan ke dalam aplikasi sebagai sistem yang memberikan rekomendasi *game* berdasarkan *game* yang dimiliki oleh pengguna.

7. Testing dan Debugging
   Setelah aplikasi selesai dibuat dan dikembangkan, aplikasi akan lalu diuji coba dan dilakukan proses *debugging* untuk memastikan bahwa aplikasi akan berjalan secara lancar sebelum disebarkan ke partisipan uji coba untuk diuji coba dan dievaluasi.

8. Evaluasi dengan Technology Acceptance Model
   Setelah proses uji coba dan *debugging* selesai, maka aplikasi akan disebarkan ke partisipan uji coba aplikasi dengan survey yang harus diisi oleh partisipan uji coba. Hasil dari survey tersebut akan digunakan untuk mengkalkulasikan seberapa berguna dan seberapa mudah digunakan aplikasi yang telah dibuat.

3.2 Identifikasi Masalah

Sebelum melakukan perancangan dan pengembangan aplikasi yang akan dibuat, hal utama yang perlu dilakukan adalah mengidentifikasi masalah yang ada. Identifikasi masalah dilakukan dengan melakukan riset pasar serta informasi dari penelitian tentang sistem rekomendasi *game* sebelumnya.

3.3 Studi Literatur

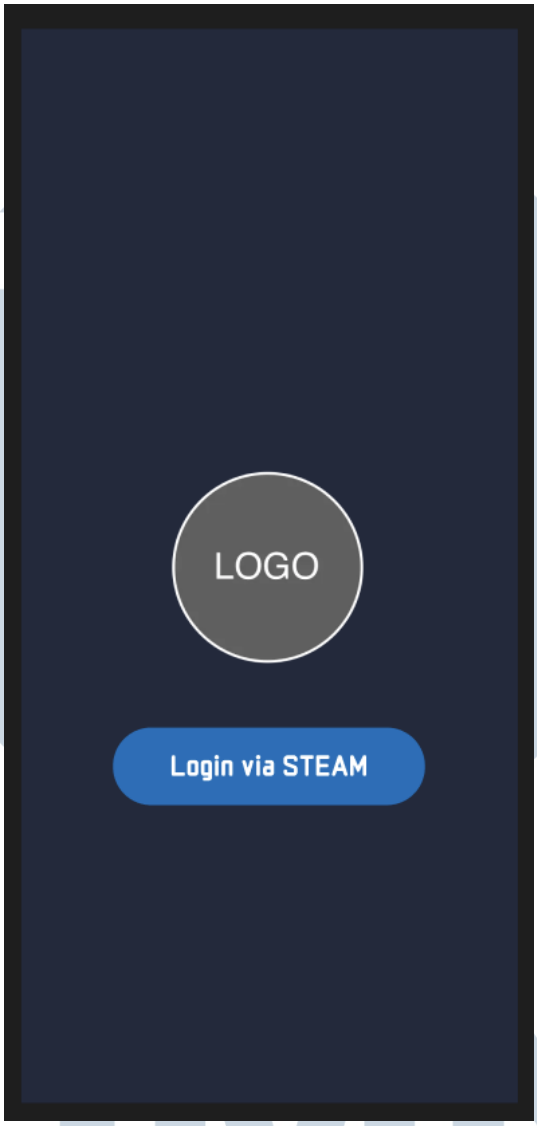
Studi literatur dilakukan dengan mencari berbagai penelitian serupa dari berbagai sumber tertulis seperti jurnal, artikel, dan laporan penelitian serupa yang telah dilakukan sebelumnya.

3.4 Prototyping UI Aplikasi

Tahapan pertama yang dilakukan dalam pengembangan aplikasi *mobile* ini adalah dengan melakukan *prototyping* untuk UI aplikasi yang akan dibuat. *Prototype* akan dibuat menggunakan aplikasi Figma dan akan digunakan sebagai basis UI untuk aplikasi yang akan dibuat. Dalam tahap ini, akan dibuat *prototype* untuk berbagai elemen yang ada di dalam aplikasi.

3.4.1 Login Screen

Pada bagian ini, aplikasi menampilkan sebuah logo dan sebuah tombol yang akan membawa pengguna ke sebuah halaman web dimana pengguna dapat melakukan *login* menggunakan akun *Steam* mereka. Berikut ini adalah *prototype* tampilan dari bagian ini.

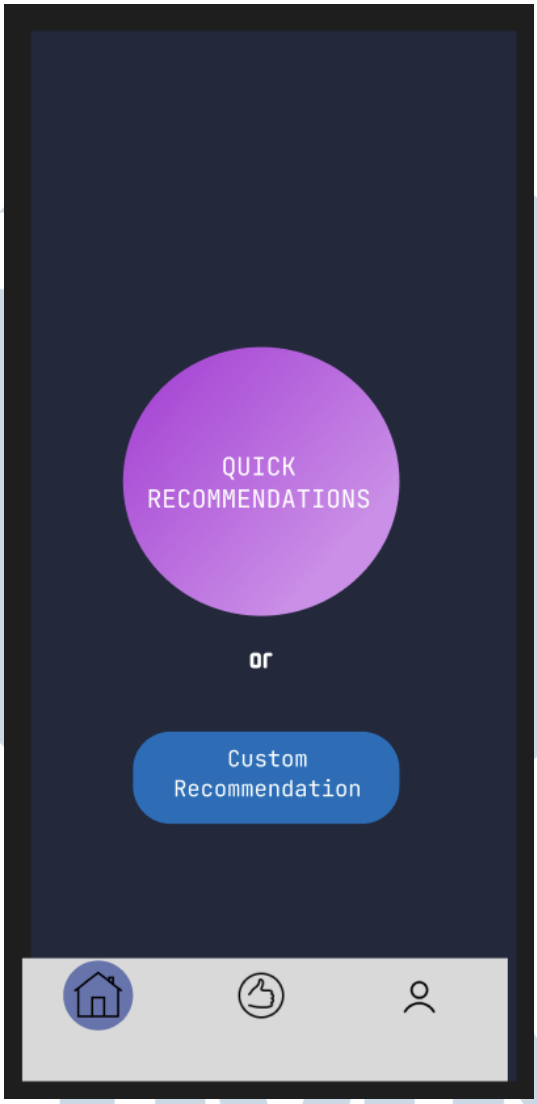


Gambar 3.2. Prototype halaman login

Pada *prototype* ini, dapat dilihat bahwa ada logo dan sebuah tombol untuk melakukan proses *login* menggunakan layanan *Steam*. Saat tombol *login* ditekan, aplikasi akan membuka halaman web *Steam* yang dimana pengguna dapat *login* dan menyambungkan akun *steam* mereka dengan aplikasi ini.

3.4.2 Home Screen

Bagian ini merupakan tampilan halaman utama aplikasi yang akan muncul setelah pengguna melakukan login menggunakan akun *Steam* mereka. Berikut ini adalah *prototype* tampilan dari bagian ini.



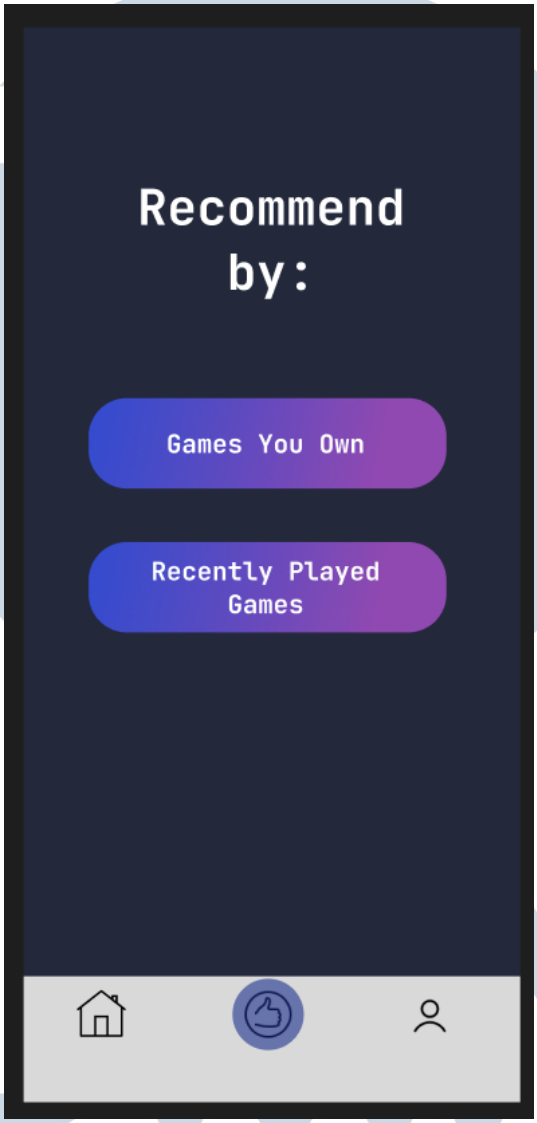
Gambar 3.3. Prototype halaman utama

Pada *prototype* ini, dapat dilihat bahwa ada dua tombol yang akan membawa pengguna ke dua halaman berbeda. Tombol ”Quick Recommendations” akan membawa pengguna ke halaman hasil rekomendasi, sementara tombol ”Custom Recommendation” akan membawa pengguna ke halaman *Recommender* dimana pengguna dapat memilih tipe rekomendasi. Selain itu, dibawah juga ada *Bottom NavBar* yang dimana pengguna dapat berpindah halaman ke halaman lain.

3.4.3 Recommender Screen

Bagian ini merupakan tampilan halaman rekomendasi aplikasi yang muncul setelah pengguna menekan tombol *Recommender* pada *Bottom Navbar* atau saat

pengguna menekan tombol ”Custom Recommendations” pada halaman utama. Berikut ini adalah *prototype* tampilan dari bagian ini.

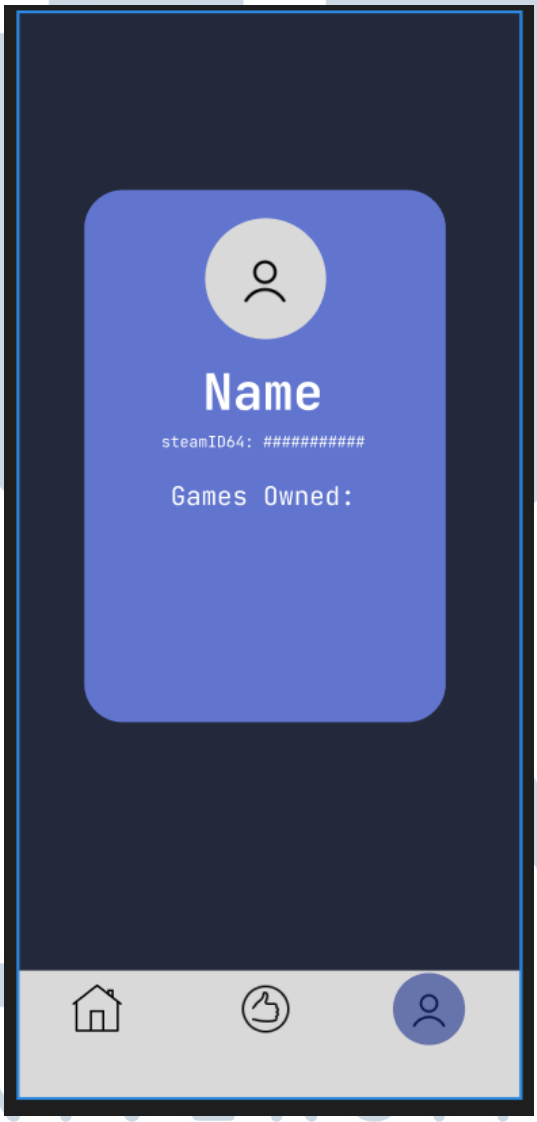


Gambar 3.4. Prototype halaman recommender

Pada *prototype* ini, dapat dilihat bahwa ada dua tombol yang akan menampilkan hasil rekomendasi yang berbeda. Saat pengguna menekan tombol ”Games You Own”, maka pengguna akan mendapatkan rekomendasi *game* berdasarkan seluruh game yang telah dimiliki oleh pengguna, sementara apabila pengguna menekan tombol ”Recently Played Games”, maka pengguna akan mendapatkan rekomendasi berdasarkan *game* apa saja yang telah dimainkan selama dua minggu terakhir.

3.4.4 Profile Screen

Bagian ini merupakan tampilan halaman profil dari aplikasi yang muncul setelah pengguna menekan tombol profil yang ada pada *Bottom Navbar*. Berikut ini adalah *prototype* tampilan dari bagian ini.



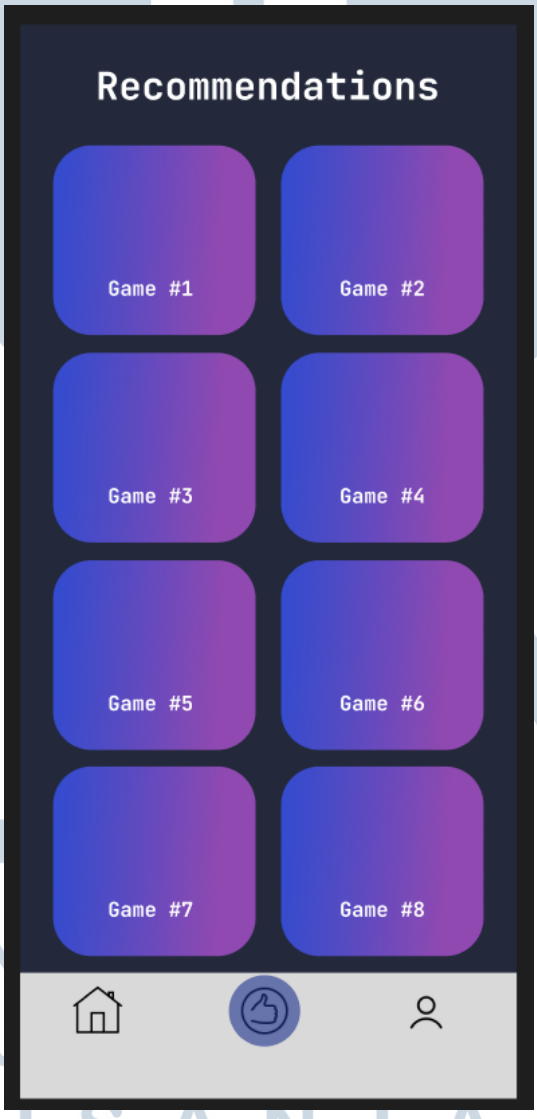
Gambar 3.5. Prototype halaman profile

Pada *prototype* ini, dapat dilihat bahwa ada dua tombol yang akan menampilkan hasil rekomendasi yang berbeda. Saat pengguna menekan tombol ”Games You Own”, maka pengguna akan mendapatkan rekomendasi *game* berdasarkan seluruh game yang telah dimiliki oleh pengguna, sementara apabila pengguna menekan tombol ”Recently Played Games”, maka pengguna akan

mendapatkan rekomendasi berdasarkan *game* apa saja yang telah dimainkan selama dua minggu terakhir.

3.4.5 Recommendation Screen

Bagian ini merupakan tampilan halaman profil dari aplikasi yang muncul setelah pengguna meminta rekomendasi game dari aplikasi. Berikut ini adalah *prototype* tampilan dari bagian ini.



Gambar 3.6. Prototype halaman recommendation

Pada *prototype* ini, dapat dilihat bahwa ada berbagai *game* yang direkomendasi oleh algoritma rekomendasi yang digunakan oleh aplikasi.

Pengguna juga dapat menekan masing-masing kartu untuk membuka *pop-up* yang akan menampilkan berbagai informasi tentang game yang dipilih.

3.5 Integrasi Steamworks Web API

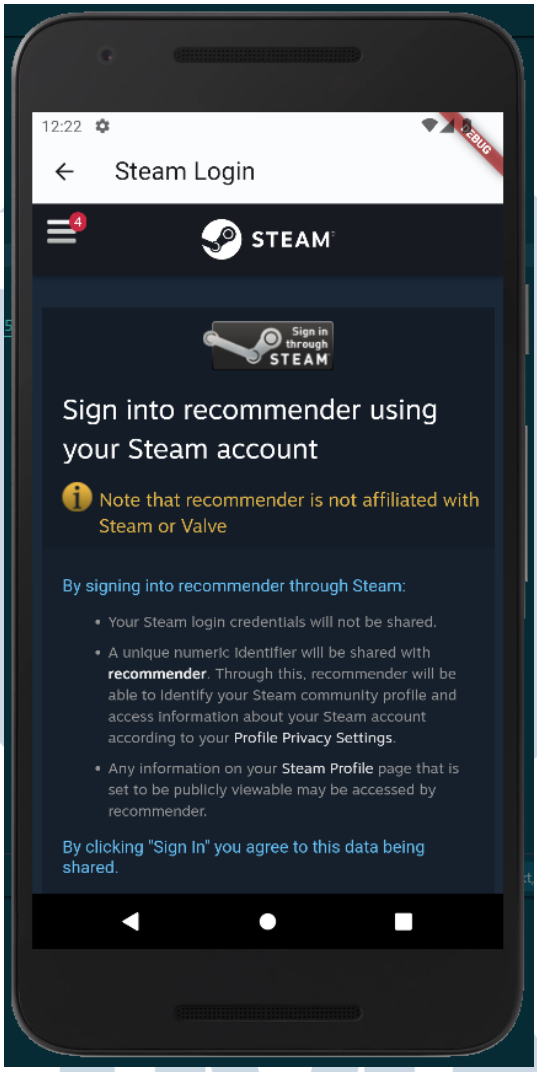
Pada tahap ini, akan dilakukan integrasi *Steamworks Web API* ke dalam aplikasi untuk menarik data publik yang penting dari layanan *Steam* untuk digunakan pada berbagai fungsi yang dilakukan oleh aplikasi ini.

Tahap pertama dalam mengintegrasikan *Steamworks Web API* adalah dengan mendapatkan *API Key* dari layanan *Steam* itu sendiri. *API Key* ini didapatkan melalui sebuah *form* yang ada pada website *Steam* yang akan meminta koneksi ke akun *Steam* yang akan digunakan untuk mengakses portal *Developer* untuk mengakses berbagai dokumentasi dan hal-hal lain yang berhubungan dengan API itu sendiri.

Setelah mendapatkan API key, aplikasi ini dapat memanggil fungsi dari API yang memberikan data ke aplikasi melalui *API call* yang menggunakan fungsi GET atau POST untuk menjalankan fungsi tertentu dari *Steamworks Web API* melalui metode Uri.parse(). Apabila aplikasi memanggil fungsi GET dari API, maka hasil dari data yang didapat akan dikonversikan menjadi JSON yang akan dibaca dan digunakan oleh aplikasi untuk menjalankan fungsi utama aplikasi seperti fungsi rekomendasi dan halaman profil.

3.5.1 Login using STEAM

Dalam mendapatkan data user, aplikasi ini mengharuskan pengguna untuk menyambungkan aplikasi ini dengan layanan *Steam* dengan melakukan *login* ke *Steam* menggunakan akun *Steam* mereka. Sistem *login* ini tidak menggunakan sistem autentikasi bawaan aplikasi, tetapi menggunakan sistem autentikasi yang digunakan oleh *Steam*, yaitu *OpenID*. Berikut ini adalah tampilan dari halaman login tersebut.



Gambar 3.7. Tampilan halaman login melalui Steam

Dengan melakukan *login* melalui *Steam*, aplikasi akan mengambil ”steamid” dari *Steam Web API*, yaitu data pengenal pengguna yang unik yang berbentuk sebuah *String* yang berisikan kombinasi huruf dan angka. Data ini lalu akan digunakan dalam berbagai fitur aplikasi seperti menampilkan informasi pengguna di halaman profil, dan mengakses daftar *game* yang dimiliki pengguna untuk digunakan dalam memberikan rekomendasi yang akurat.

3.6 Pembuatan Fungsi Utama Aplikasi

Pada tahapan ini, fungsi-fungsi utama yang awalnya dirancang sebagai *prototype* akan mulai dibuat dan digunakan di dalam aplikasi sesuai dengan rancangan yang telah dibuat. Fungsi-fungsi ini akan dibuat dengan menggunakan

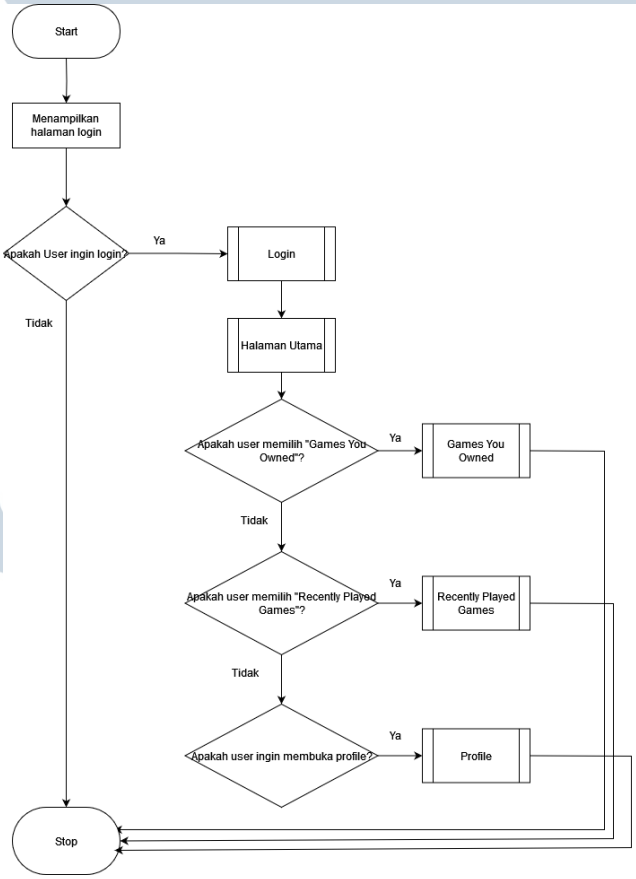
alat-alat yang disediakan oleh framework Flutter yang digunakan dalam pembuatan aplikasi.

Selain itu, dengan adanya integrasi dengan *Steamworks Web API*, aplikasi dapat sekarang menggunakan fungsi API dari *Steamworks Web API* untuk mendapatkan data yang akan digunakan dalam menjalankan fungsi rekomendasi pada aplikasi.

3.6.1 Flowchart Aplikasi

Berikut ini adalah *flowchart* yang memberikan gambaran tentang cara kerja dan alur dari aplikasi yang dibuat.

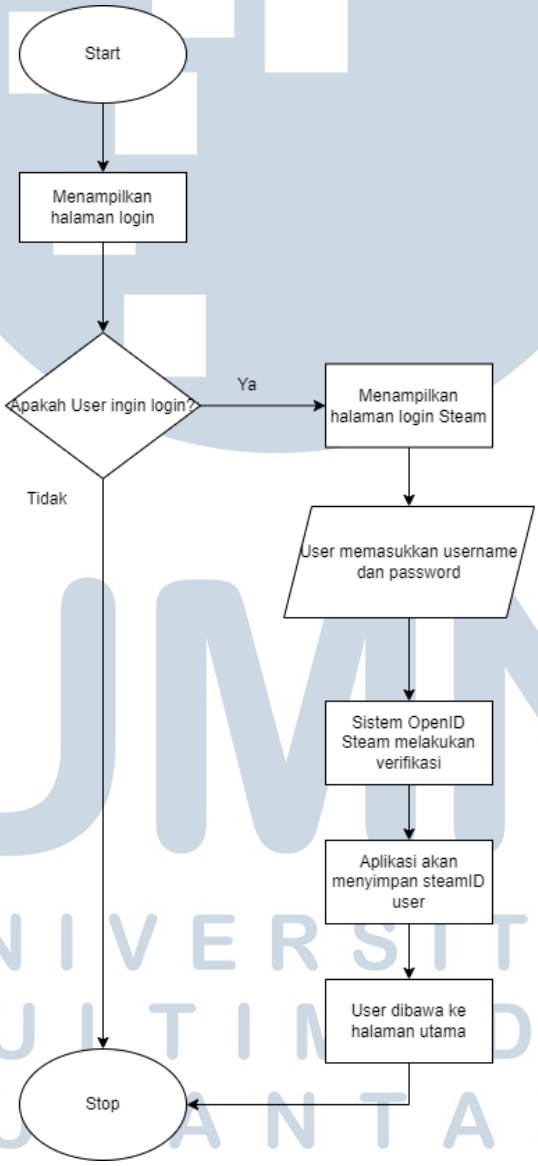
A Flowchart Keseluruhan Aplikasi



Gambar 3.8. Flowchart Proses Login

Gambar 3.8 memperlihatkan alur dari aplikasi yang dibuat. Saat aplikasi dibuka, pengguna akan diminta untuk melakukan login menggunakan fungsi login yang lalu akan menampilkan halaman utama. Setelah itu, pengguna dapat memilih untuk mendapatkan rekomendasi berdasarkan *game* yang dimiliki pengguna, *game* yang telah dimainkan selama dua minggu terakhir, atau membuka halaman profil.

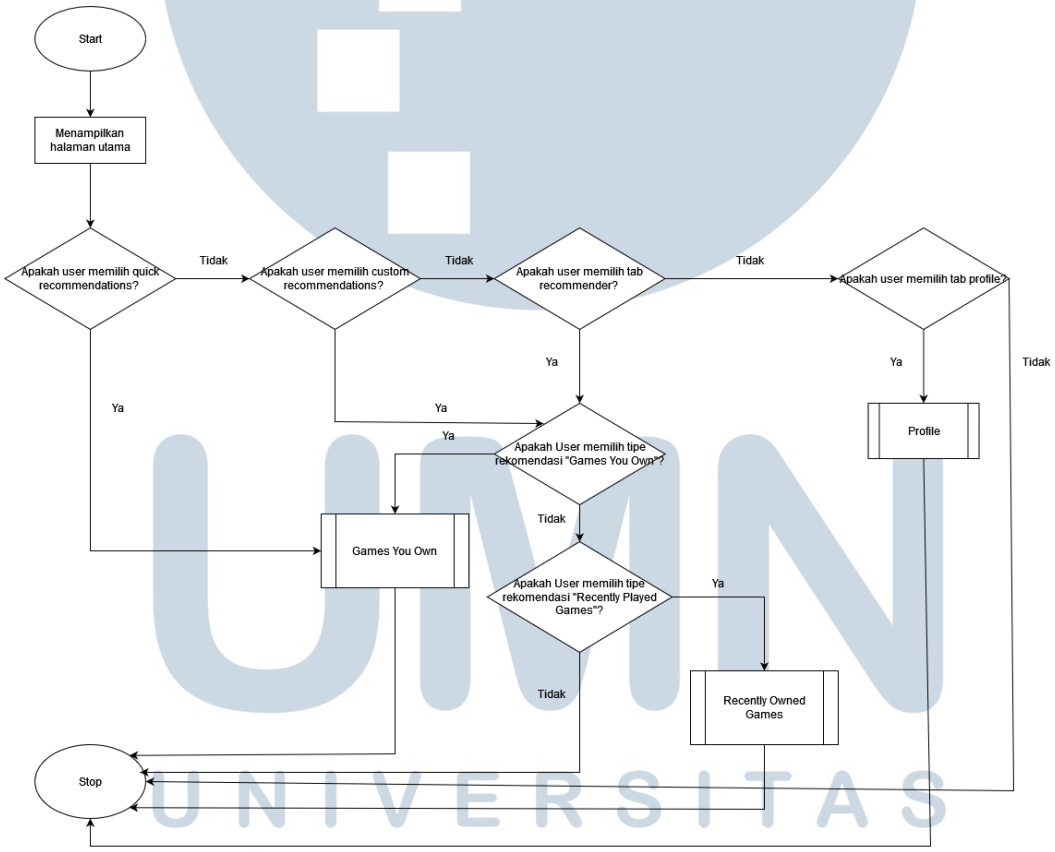
B Flowchart Bagian Login



Gambar 3.9. Flowchart Proses Login

Gambar 3.9 memperlihatkan cara kerja dari bagian *login* aplikasi yang dibuat. Saat pengguna membuka aplikasi, akan diperlihatkan halaman *login* yang akan meminta pengguna untuk melakukan *login* menggunakan *Steam*. Apabila pengguna memilih untuk login, maka aplikasi akan menampilkan halaman web dimana pengguna dapat masuk menggunakan akun *Steam* mereka. Setelah memasukkan data diri dan menyambungkan aplikasi ini dengan *Steam*, sistem akan melakukan verifikasi dan aplikasi akan menyimpan *steamID* milik pengguna. Setelah itu, pengguna akan dibawa ke halaman utama aplikasi.

C Flowchart Halaman Utama Aplikasi

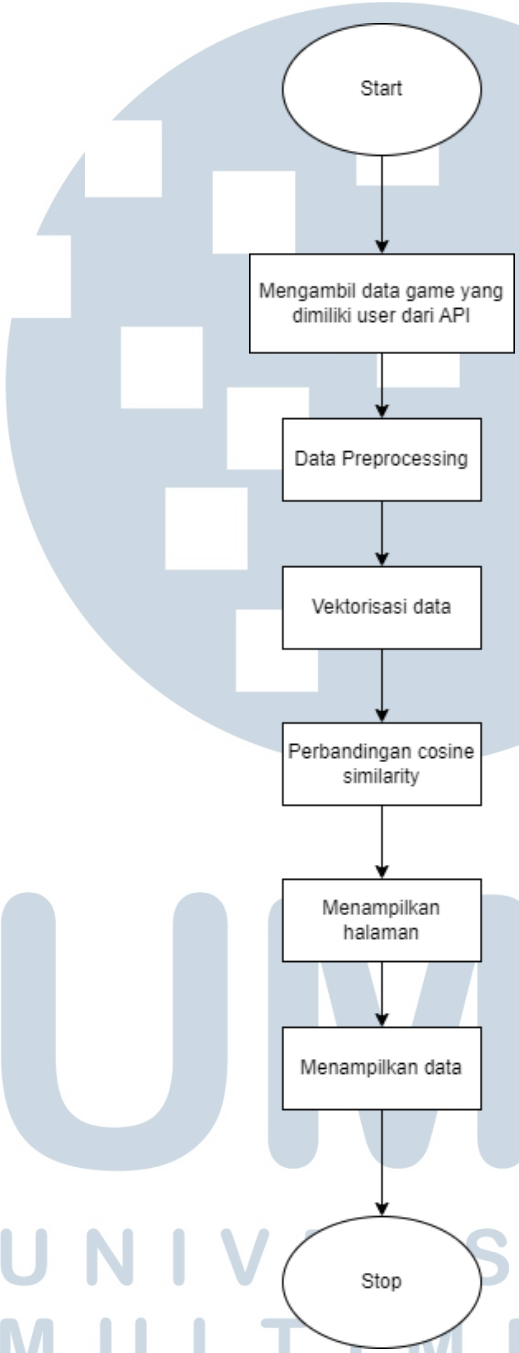


Gambar 3.10. Flowchart Halaman Utama

Gambar 3.10 memperlihatkan cara kerja dari halaman utama aplikasi ini. Setelah pengguna melakukan *login*, pengguna akan mendapat pilihan untuk menekan tombol ”Quick Recommendations”, tombol ”Custom Recommendations”, tombol halaman ”Recommender”, dan tombol ”Profile” untuk membuka profil.

Tombol ”Quick Recommendations akan langsung memberikan rekomendasi kepada pengguna berdasarkan *game* yang dimiliki pengguna. Tombol ”Custom Recommendations” dan tombol halaman ’Recommender” akan membawa pengguna ke halaman ”Recommender”. Sementara itu, tombol ”Profile” akan membawa pengguna ke halaman profil.

D Flowchart Halaman Rekomendasi ”Games You Owned”

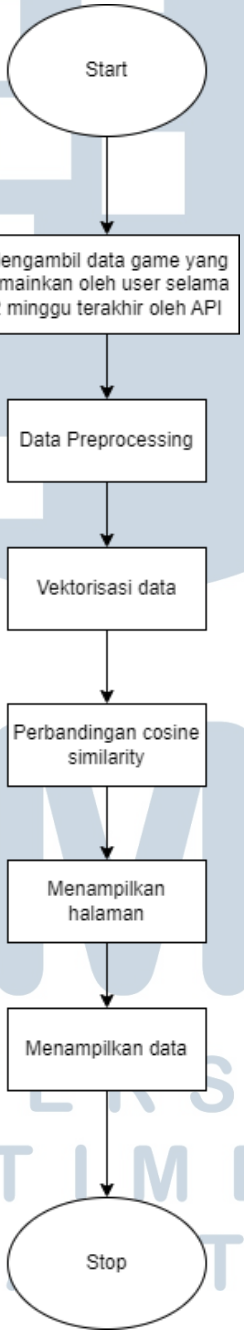


Gambar 3.11. Flowchart Halaman Rekomendasi Berdasarkan Game Yang Dimiliki

Gambar 3.11 memperlihatkan cara kerja dari sistem rekomendasi menurut jumlah *game* yang dimiliki oleh pengguna. Pertama, aplikasi akan menggunakan *Steam Web API* untuk mengambil data tentang *game* yang dimiliki oleh pengguna. Setelah itu, data tersebut akan dilakukan *pre-processing* untuk digunakan dalam

algoritma *Content-Based Filtering*. Setelah itu, rekomendasi yang diberikan akan muncul di halaman sebagai data yang ditampilkan.

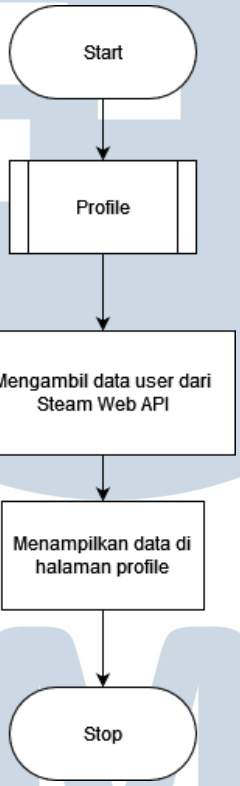
E Flowchart Halaman Rekomendasi ”Recently Played Games”



Gambar 3.12. Flowchart Halaman Rekomendasi Berdasarkan Game Yang Dimainkan Selama 2 Minggu Terakhir

Gambar 3.12 memperlihatkan cara kerja dari sistem rekomendasi menurut jumlah *game* yang dimainkan selama 2 minggu terakhir oleh pengguna. Sistem yang digunakan juga tidak jauh berbeda dengan gambar 3.11. Perbedaan utama dari kedua sistem ini adalah data yang diambil dari API.

F Flowchart Halaman Profile



Gambar 3.13. Flowchart Halaman Profile

Gambar 3.13 memperlihatkan cara kerja dari halaman profil dari aplikasi ini. Saat halaman profil dibuka, aplikasi akan mengambil data pengguna dari *Steam Web API* dan menampilkan beberapa dari data tersebut di halaman profil.

3.7 Integrasi Content-Based Filtering

Algoritma Content-Based Filtering digunakan untuk menghasilkan rekomendasi yang sesuai dengan pengguna berdasarkan kesamaan antara *game* yang pernah dimainkan sebelumnya dan *game* yang tersedia di layanan *Steam* yang memiliki kesamaan dengan *game* yang telah dimainkan. Proses ini dilakukan

dengan membandingkan profil *tag* pengguna berdasarkan *tag* dari *game* yang dimiliki pengguna atau *game* yang dimainkan pengguna pada dua minggu terakhir dengan *tag* yang digunakan untuk mendeskripsikan *game* yang dibandingkan. Pertama, akan dilakukan *data preprocessing* untuk membersihkan dan memproses data menjadi lebih terstruktur. Setelah itu, data tersebut akan digunakan dalam sistem rekomendasi untuk mendapatkan rekomendasi yang akurat terhadap user menggunakan teknik *cosine similarity*. Setelah itu, aplikasi akan menampilkan hasil yang didapatkan dalam sebuah *list Top N Recommendation* yang akan menampilkan hasil 10 *game* yang paling sesuai dengan pengguna. Berikut ini adalah tahapan sistem *Content-Based Filtering* dalam aplikasi yang dibuat:

1. Tahapan pertama, aplikasi akan mengambil data dari sumber yang digunakan yaitu *Steam Web API*, *SteamSpy API*, dan dataset berupa JSON berisi data 55 ribu *game* di *Steam*.

2. Tahapan kedua, data tersebut akan dilakukan *data preprocessing* dengan menghilangkan simbol dan menjadikan *tags* yang ada menjadi huruf kecil (*lowercase*).

3. Tahapan ketiga, data *tags* yang telah dilakukan *preprocessing* akan dilakukan vektorisasi. Vektor biner yang dihasilkan dari data ini akan digunakan dalam kalkulasi *cosine similarity*.

4. Tahapan keempat, aplikasi akan memulai kalkulasi *cosine similarity* dengan cara mengkalkulasikan produk dot dari vektor A yang merupakan *user profile* berisi *tags* yang berasal dari *game* dari akun pengguna, dan vektor B yang merupakan *tag* dari *game* tertentu yang dibandingkan dengan *user profile* pengguna.

5. Tahapan kelima, setelah produk dot dikalkulasi, akan dilakukan kalkulasi magnitudo untuk kedua vektor dengan mengkalikan pangkat dua untuk vektor masing-masing.

6. Tahapan keenam, setelah produk dot dan kedua magnitudo ditemukan, produk dot akan dibagi dengan akar dari magnitudo A dan B untuk mendapatkan *cosine similarity*.

7. Tahapan ketujuh, setelah melakukan proses tahapan empat sampai enam berulang kali sesuai dengan jumlah *game* yang dilakukan perbandingan, maka

sistem akan mengsortir *game* berdasarkan besarnya nilai *cosine similarity* dan menampilkan 10 *game* dengan nilai *cosine similarity* terbesar.

3.8 Testing dan Debugging

Setelah pengembangan aplikasi selesai, langkah selanjutnya adalah melakukan serangkaian pengujian (*testing*) untuk memastikan bahwa aplikasi berfungsi sesuai dengan yang diharapkan. Aplikasi akan diuji untuk mencari masalah dan menilai fungsionalitas, keandalan, kinerja, dan keamanan aplikasi.

Setelah masalah (*bug*) dalam aplikasi ditemukan, maka akan dilakukan proses debugging. Akan dilakukan berbagai cara untuk mencari solusi dalam memperbaiki kesalahan tersebut, seperti mencari akar masalah dengan analisis kode, reproduksi kesalahan, dan pengujian berkala.

3.9 Evaluasi Dengan Technology Acceptance Model

Setelah aplikasi dibuat, aplikasi akan disebarkan ke beberapa orang yang akan menjadi sampel dalam penelitian ini. Aplikasi akan disebarkan kepada partisipan penelitian dengan sebuah formulir yang dibuat dengan *Google Forms* yang dimana partisipan dapat menilai beberapa aspek dari aplikasi berdasarkan faktor kemudahan dan kebergunaan yang dibuat menggunakan pertanyaan yang sesuai dengan model *Technology Acceptance Model* yang dikemukakan oleh Fred Davis dan Richard Bagozzi[20]. Hasil yang didapatkan akan diambil dan dihitung untuk mendapatkan nilai penerimaan dan penggunaan sebuah sistem dan teknologi yang digunakan oleh pengguna.