BAB III
METODE PENELITIAN

3.1 Jenis Penelitian

Jenis penelitian yang digunakan dalam tugas akhir ini adalah **penelitian eksperimen**, yang bertujuan untuk mengamati dan membandingkan hasil generate kode dari dua kondisi desain yang berbeda. Perlakuan (*treatment*) dalam penelitian ini berupa penerapan Auto Layout, penamaan frame dan tagging elemen, serta konversi ikon dari format SVG ke PNG pada desain Figma.

Penelitian ini menggunakan perbandingan, di mana data numerik diperoleh dari hasil analisis tools SonarQube terhadap kode yang dihasilkan oleh plugin Dualite. Analisis dilakukan berdasarkan jumlah *bugs*, *code smells*, dan *vulnerabilities* serta tingkat keparahannya *(Severity)*, dan disertai evaluasi fungsionalitas antarmuka dari hasil generate. Data disajikan dan dianalisis untuk membandingkan kualitas teknis dan tampilan hasil generate dari masing-masing kondisi eksperimen.

3.2 Objek Penelitian

Objek penelitian ini adalah **source code** yang dihasilkan dari proses generate desain Figma menggunakan plugin **Dualite**. Desain yang digunakan berupa satu halaman landing page statis yang dirancang dalam dua kondisi berbeda. Plugin Dualite dipilih karena memiliki kemampuan untuk mengonversi desain visual menjadi kode React JS secara otomatis dan terstruktur. Penelitian ini difokuskan pada analisis terhadap kualitas kode yang dihasilkan dari proses generate tersebut, tanpa melakukan modifikasi manual terhadap kode hasil output.

Perbedaan kondisi desain dilakukan untuk tujuan eksperimen, yaitu dengan memberikan perlakuan khusus berupa penerapan fitur Auto Layout, penamaan frame dan tagging elemen, serta konversi aset gambar dari format SVG ke PNG. Sementara pada kondisi lainnya, desain tidak diberi perlakuan tersebut, sehingga memungkinkan adanya perbedaan hasil generate kode baik dari segi struktur maupun kualitas teknis.

Analisis dilakukan secara statis menggunakan SonarQube untuk mengidentifikasi jumlah *bugs*, *code smells*, serta tingkat keparahan (*Severity*) dari setiap masalah yang ditemukan. Selain itu, dilakukan juga evaluasi dari sisi visual dan fungsionalitas untuk melihat seberapa baik tampilan hasil generate mencerminkan desain awal, serta apakah elemen interaktif seperti tombol berfungsi dengan semestinya.

3.3 Variabel Penelitian

Penelitian ini melibatkan dua jenis variabel utama, yaitu variabel bebas dan variabel terikat. Variabel bebas dalam penelitian ini adalah **perlakuan yang diberikan pada desain antarmuka sebelum proses generate kode**, berupa penerapan fitur Auto Layout, penamaan frame dan tagging elemen, serta konversi aset gambar dari format SVG ke PNG. Perlakuan ini hanya diterapkan pada satu kondisi desain, sedangkan kondisi lainnya tidak menerima perlakuan tersebut, sehingga memungkinkan untuk dilakukan perbandingan terhadap hasil generate kode yang dihasilkan oleh plugin Dualite.

Sementara itu, variabel terikat dalam penelitian ini adalah **kualitas hasil kode** yang dihasilkan dari proses generate tersebut. Kualitas kode diukur berdasarkan

data yang diperoleh dari tools analisis statis SonarQube, yang mencakup jumlah *bugs*, *code smells*, *vulnerabilities*, tingkat keparahan (*Severity*), serta total baris kode (*lines of code*). Selain itu, aspek fungsionalitas dan tampilan visual dari hasil generate juga dianalisis secara deskriptif berdasarkan pengamatan langsung terhadap struktur antarmuka saat dijalankan di browser.

3.4 Teknik Analisis Data

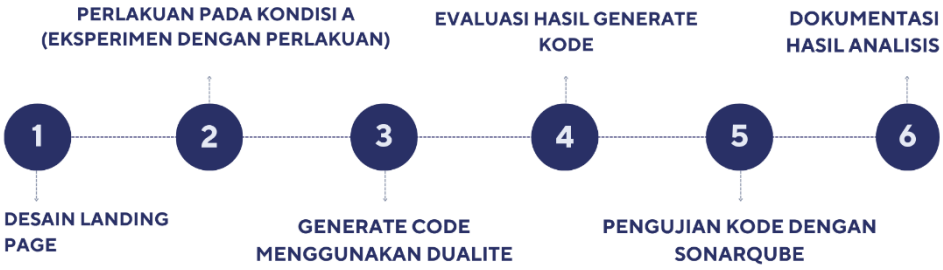
Desain eksperimen ini bertujuan untuk mengevaluasi pengaruh dari perlakuan tertentu terhadap kualitas teknis dan fungsionalitas kode yang dihasilkan. Perlakuan tersebut meliputi penerapan fitur Auto Layout, pemberian nama frame dan tagging elemen, serta konversi aset ikon dari format SVG ke PNG.

Pada kondisi pertama, desain dibuat dengan menerapkan seluruh perlakuan tersebut. Tujuannya adalah untuk menguji bagaimana hasil generate kode ketika desain sudah disusun secara terstruktur dan disesuaikan dengan praktik yang direkomendasikan dalam dokumentasi plugin Dualite. Sementara pada kondisi kedua, desain dibuat tanpa menerapkan perlakuan apa pun, sehingga mewakili kondisi standar atau dasar dari proses generate kode tanpa optimasi.

3.5 Tahapan penelitian

Penelitian ini menggunakan metode eksperimen dengan membandingkan dua kondisi desain Figma, yaitu desain dengan perlakuan (penerapan Auto Layout, penamaan frame otomatis, tagging elemen, dan konversi gambar SVG ke PNG) dan desain tanpa perlakuan. Tujuan eksperimen ini adalah untuk mengetahui pengaruh konfigurasi desain terhadap kualitas hasil generate kode menggunakan plugin Dualite, baik dari segi visual–fungsionalitas maupun kualitas teknis yang dianalisis

menggunakan SonarQube. Secara umum, tahapan penelitian meliputi pembuatan desain antarmuka, pemberian perlakuan pada salah satu kondisi, generate kode dari kedua desain menggunakan plugin Dualite, evaluasi hasil di sisi tampilan dan teknis, serta dokumentasi hasil analisis. Alur proses ini disajikan pada Gambar 3.1.



Gambar 3. 1 Tahapan Penelitian

Gambar 3.1 menggambarkan alur eksperimen yang dilakukan dalam penelitian ini. Penelitian diawali dengan pembuatan desain landing page sebagai objek dasar eksperimen (Nelli, 2024). Desain tersebut kemudian dibagi ke dalam dua kondisi: satu diberikan perlakuan tertentu sesuai rekomendasi plugin Dualite, dan satu tanpa perlakuan sebagai pembanding. Selanjutnya, kedua desain dikonversi menjadi kode menggunakan plugin Dualite, lalu diuji dari dua sisi: tampilan visual di browser dan kualitas teknis kode melalui analisis SonarQube. Hasil dari kedua kondisi didokumentasikan dan dianalisis untuk menarik kesimpulan secara objektif. Penjelasan detail dari setiap tahapannya yaitu sebagai berikut:

1. *Desain Landing Page*

Penelitian diawali dengan pembuatan dua desain landing page di Figma yang memiliki struktur visual dan konten serupa, namun dengan perbedaan konfigurasi. Desain pertama, yaitu **Kondisi A**, diberi perlakuan berupa penerapan

Auto Layout, tagging elemen, penamaan frame otomatis, dan konversi aset gambar dari SVG ke PNG. Desain kedua, yaitu **Kondisi B**, tidak diberi perlakuan tersebut. Kedua desain mencakup elemen visual utama seperti header, banner, informasi utama, fitur layanan, dan footer, dengan aset visual yang konsisten secara warna dan gaya (Nelli, 2024).

2. *Pemberian Perlakuan pada Kondisi A*

Khusus untuk Kondisi A, dilakukan konfigurasi desain sesuai rekomendasi dokumentasi plugin Dualite. Auto Layout diterapkan untuk memastikan struktur responsif dan konsisten. Pengaturan yang digunakan antara lain Wrap Layout, Align Center, Fill Container untuk lebar, dan Hug Content untuk tinggi. Seluruh aset vektor juga dikonversi menjadi format PNG untuk menghindari kerumitan struktur kode SVG. Selain itu, digunakan plugin Auto Name untuk menamai frame secara otomatis, dan dilakukan tagging manual dengan menambahkan #link# untuk hyperlink dan #button# untuk tombol interaktif, agar Dualite dapat mengenali elemen tersebut saat proses generate.

3. *Generate Code Menggunakan Plugin Dualite*

Setelah desain siap, dilakukan proses generate kode dari kedua kondisi menggunakan plugin Dualite. Proses ini menghasilkan proyek berbasis React JS yang secara otomatis terbuka di platform CodeSandbox. Proyek kemudian diekspor ke GitHub dan di-clone ke komputer lokal untuk keperluan evaluasi lebih lanjut. Proses generate dilakukan secara terpisah untuk setiap kondisi, tanpa intervensi atau modifikasi manual terhadap hasil kode yang dihasilkan.

4. *Evaluasi Visual dan Fungsionalitas Web*

Tahap ini bertujuan untuk membandingkan kesesuaian antara tampilan hasil generate dan desain awal. Evaluasi dilakukan dengan membuka hasil generate di browser lokal dan mencermati struktur layout, warna, posisi elemen, fungsi tombol, serta kesesuaian navigasi. Pengamatan dilakukan terhadap kedua kondisi (A dan B), untuk melihat perbedaan dampak konfigurasi desain terhadap hasil tampilan akhir..

5. *Pengujian Kode dengan SonarQube*

Seluruh file hasil generate dari masing-masing kondisi kemudian dianalisis menggunakan SonarQube versi 25.1.0.102122, dengan dukungan PostgreSQL sebagai basis data. Token autentikasi dihasilkan dan diintegrasikan ke terminal Visual Studio Code untuk memulai proses pemindaian. Fokus analisis berada pada tiga jenis masalah utama: *bugs*, *code smells*, dan *vulnerabilities*. Seluruh direktori proyek seperti src, components, dan App.js dianalisis tanpa pengecualian.

6. *Evaluasi dan Dokumentasi Hasil*

Hasil pemindaian dari masing-masing kondisi didokumentasikan dalam bentuk tabel. Analisis mencakup jumlah temuan untuk setiap kategori masalah serta tingkat keparahannya (*Severity*: Major, Minor, Critical). Selanjutnya dilakukan perhitungan rasio jumlah masalah terhadap total baris kode, serta persentase distribusi tiap jenis masalah. Dokumentasi hasil ini digunakan sebagai dasar untuk membandingkan kualitas teknis kode antara dua kondisi desain secara objektif.

3.6 Teknik Pengumpulan Data

Pengumpulan data dalam penelitian ini dilakukan melalui dua pendekatan utama, yaitu secara otomatis menggunakan tools analisis statis, serta secara manual

melalui observasi tampilan hasil generate. Sumber data berasal dari hasil generate kode React JS menggunakan plugin Dualite terhadap dua kondisi desain yang berbeda pada aplikasi Figma.

Pada masing-masing kondisi, kode yang dihasilkan dianalisis menggunakan tools **SonarQube** untuk memperoleh data kuantitatif terkait kualitas teknis kode. Data yang dikumpulkan meliputi jumlah *bugs*, *code smells*, *vulnerabilities*, tingkat keparahan (*Severity*), serta jumlah baris kode (*lines of code*). Proses analisis dilakukan dengan menghubungkan proyek hasil generate ke server SonarQube lokal menggunakan token autentikasi dari Visual Studio Code, sehingga seluruh file dalam proyek dapat dipindai dan direkam hasilnya oleh sistem.

Selain itu, data juga dikumpulkan melalui observasi langsung terhadap tampilan visual dan fungsionalitas dari masing-masing hasil generate ketika dijalankan di browser. Pengamatan ini mencakup kesesuaian layout dengan desain awal di Figma, keberfungsian elemen interaktif seperti tombol dan tautan, serta konsistensi struktur halaman secara umum.

3.7 Teknik Analisis Data

Analisis data dalam penelitian ini dilakukan dengan membandingkan hasil generate kode dari dua kondisi desain yang berbeda menggunakan plugin Dualite. Data diperoleh melalui pemindaian kode dengan bantuan tools SonarQube dan dianalisis berdasarkan jumlah temuan masalah yang teridentifikasi dalam struktur proyek.

Beberapa aspek teknis yang dianalisis meliputi jumlah *bugs*, *code smells*, dan *vulnerabilities*, serta tingkat keparahannya (*Severity*), dan total baris kode (*lines of*

code) pada masing-masing kondisi. Untuk memberikan gambaran kuantitatif terhadap hasil yang diperoleh, dilakukan perhitungan **rasio jumlah masalah terhadap total baris kode** dengan menggunakan rumus yang menrujuk pada dokumentasi sonarqube sebagai berikut (Sonarqube, 2024):

$$Rasio\ Masalah = \left(\frac{Total\ masalah}{Total\ baris\ kode}\right) x\ 100$$

Perhitungan rasio ini juga dilakukan secara terpisah untuk setiap jenis masalah berdasarkan tingkat *Severity*, seperti rasio *bugs major*, *code smells minor*, dan sebagainya. Hasil perhitungan ini disusun dalam bentuk tabel dan digunakan sebagai dasar untuk membandingkan kualitas kode antara dua kondisi.

Selain aspek teknis, hasil generate kode juga dijalankan di browser untuk mengamati tampilan visual dan fungsi interaktif dari antarmuka. Observasi dilakukan secara langsung terhadap elemen-elemen seperti tombol, tautan, layout halaman, dan konsistensi tampilan. Hasil observasi ini dikompilasi dalam bentuk uraian deskriptif sebagai pelengkap dari analisis teknis.