

BAB I

PENDAHULUAN

1.1 Latar Belakang

Permintaan terhadap aplikasi web yang menarik, interaktif, dan fungsional terus meningkat seiring dengan perkembangan era digital (Putra et al., 2020). Desainer UI/UX menghadapi tantangan dalam menciptakan tampilan antarmuka yang estetis sekaligus efisien untuk diimplementasikan dalam bentuk kode. Berdasarkan data dari (Rokis et al., 2023), adopsi alat low-code dan no-code mengalami peningkatan signifikan, dengan proyeksi lebih dari 65% pengguna perangkat lunak akan menggunakan pendekatan ini dalam beberapa tahun ke depan. Tren ini memperlihatkan meningkatnya kebutuhan terhadap alat desain otomatis seperti plugin Figma, yang dapat menghasilkan kode secara langsung dari desain visual.

Kesenjangan antara proses desain dan pengembangan perangkat lunak masih menjadi hambatan nyata dalam praktik profesional. Pengembang kerap mengalami kesulitan dalam menginterpretasikan hasil desain menjadi kode, yang berdampak pada keterlambatan proyek dan meningkatnya biaya implementasi (Tazkiyah et al., 2022). Ketidaksesuaian antara desain dan kode dapat menyebabkan revisi yang memakan waktu dan berdampak pada performa serta pemeliharaan aplikasi (Pratama et al., 2022). Kondisi ini semakin kompleks karena sebagian besar desainer tidak memiliki latar belakang teknis, sehingga komunikasi antara desainer dan pengembang menjadi tidak efektif (Senubekti, 2024; Suriyanto, 2023).

Penggunaan plugin Figma dalam proses desain dinilai mampu mempersingkat waktu pengembangan dengan mengotomatiskan konversi dari desain ke kode (Mulyadi et al., 2020). Meskipun demikian, tantangan baru muncul dalam memastikan bahwa kode yang dihasilkan oleh plugin tersebut memiliki kualitas yang layak secara teknis. Sebagian besar penelitian sebelumnya lebih berfokus pada aspek visual dan estetika dari desain, tanpa mengevaluasi secara sistematis kualitas kode yang dihasilkan.

Dalam penelitian ini code yang di hasilkan berformat statis code, sementara itu Kajian terdahulu menunjukkan bahwa alat analisis statis seperti SonarQube memiliki potensi besar untuk mendeteksi permasalahan dalam kode secara otomatis, seperti bug dan code smells (Di Biase et al., 2019; Vassallo et al., 2020). Meskipun demikian, belum banyak penelitian yang secara eksplisit mengkaji hubungan antara hasil desain menggunakan plugin Figma dan kualitas kode yang dihasilkannya. Pemetaan literatur yang dilakukan dalam penelitian ini menunjukkan bahwa keterkaitan antara topik seperti code generation, plugin Figma, dan static code analysis masih jarang dibahas secara terpadu dalam satu kajian. Temuan ini semakin menguatkan urgensi penelitian untuk mengisi kekosongan tersebut dengan pendekatan yang menggabungkan evaluasi visual dan teknis secara bersamaan..

Selain itu, sebelum dilakukan analisis teknis terhadap kode, penelitian ini juga melakukan evaluasi visual dan fungsionalitas terhadap hasil generate yang ditampilkan di browser. Tujuan dari evaluasi ini adalah untuk menilai apakah tampilan dan fungsi dasar antarmuka seperti layout, warna, dan tombol telah sesuai

dengan desain awal di Figma. Langkah ini penting sebagai bentuk verifikasi awal sebelum analisis kualitas kode dilakukan, guna memastikan bahwa hasil konversi desain dapat diimplementasikan secara nyata dan berfungsi sebagaimana mestinya.

Penelitian ini diarahkan untuk mengevaluasi kualitas kode hasil generate dari plugin Dualite di Figma, dengan menggunakan SonarQube sebagai alat analisis statis. metode penelitian yang digunakan dalam studi ini adalah metode eksperimen. Metode ini memungkinkan peneliti untuk menguji pengaruh dari suatu perlakuan terhadap hasil tertentu dalam kondisi yang dapat dikendalikan (Sheraz et al., 2022). Dalam konteks penelitian ini, eksperimen memberikan keleluasaan untuk mengamati perubahan atau dampak dari konfigurasi desain tertentu terhadap kualitas hasil generate kode yang dihasilkan secara otomatis (Sugiyono, 2020; Wulan, 2020).

Pendekatan eksperimen juga relevan digunakan karena memberikan landasan kuantitatif dalam mengevaluasi kualitas teknis kode, seperti banyaknya jumlah *bugs* dan *code smells* yang dapat dideteksi dengan alat bantu analisis statis seperti SonarQube (Etemadi et al., 2021). Dengan membandingkan hasil generate berdasarkan konfigurasi desain yang berbeda, penelitian ini dapat memberikan gambaran objektif terhadap seberapa besar pengaruh aspek desain awal terhadap hasil akhir implementasi kode.

Penelitian ini diharapkan dapat memberikan kontribusi baik secara teoretis maupun praktis. Secara teoretis, penelitian ini memperkuat literatur terkait hubungan antara desain UI/UX dan pengembangan perangkat lunak, khususnya

dalam konteks otomatisasi kode. Secara praktis, hasil penelitian ini dapat menjadi acuan bagi desainer dan pengembang untuk memanfaatkan plugin Figma secara lebih optimal dalam meningkatkan efisiensi kerja serta kualitas hasil kode yang dihasilkan.

1.2 Rumusan Masalah

- a. Bagaimana perbandingan hasil generate kode dari plugin Dualite pada dua kondisi desain yang berbeda dilihat dari aspek visual dan fungsionalitas?
- b. Bagaimana perbandingan kualitas kode hasil generate plugin Dualite pada dua kondisi desain yang berbeda berdasarkan hasil analisis SonarQube?

1.3 Tujuan Penelitian

- a. Membandingkan kesesuaian tampilan dan fungsionalitas hasil generate plugin Dualite terhadap desain awal pada dua kondisi desain yang berbeda.
- b. membandingkan kualitas kode yang dihasilkan oleh plugin Dualite pada dua kondisi desain yang berbeda menggunakan alat analisis SonarQube.

1.4 Manfaat Penelitian

- a. Manfaat Teoritis:

Penelitian ini diharapkan dapat memberikan kontribusi dalam pengembangan ilmu pengetahuan di bidang UI/UX design dan software development, khususnya terkait integrasi antara alat desain dan proses pengembangan perangkat lunak melalui plugin seperti Figma. Selain itu, penelitian ini juga memperluas pemahaman mengenai kualitas hasil generate desain ke kode, baik dari sisi tampilan antarmuka maupun aspek teknis kode yang dihasilkan.

- b. Manfaat Praktis:

- Bagi desainer Plugin ini memungkinkan desainer UI /UX, untuk langsung mengubah desain mereka menjadi kode. Hal ini juga membantu mengurangi miskomunikasi saat berkolaborasi dengan programmer, karena desain yang dihasilkan lebih mudah dipahami dan diimplementasikan.
- Bagi programmer Plugin Figma dapat menghemat waktu dalam mengembangkan kode dari desain. Selain itu, dengan adanya evaluasi menggunakan SonarQube, programmer dapat mengetahui kualitas teknis dari kode yang dihasilkan, sehingga proses validasi dan pemeliharaan kode menjadi lebih efisien.

1.5 Batasan Masalah

- a. Penelitian hanya menggunakan satu plugin Figma, yaitu Dualite.dev, sebagai alat untuk menghasilkan kode dari desain antarmuka berbasis React JS.
- b. Evaluasi tampilan dan fungsionalitas dilakukan secara manual, tanpa melibatkan uji pengguna atau wawancara dengan pihak ketiga.
- c. Elemen desain kompleks seperti animasi, interaksi dinamis, tidak menjadi fokus utama dalam penelitian ini. Evaluasi dilakukan hingga tingkat tampilan.
- d. Penelitian menggunakan SonarQube untuk melakukan analisis kualitas kode dengan fokus pada identifikasi bugs dan code smells, tanpa mengevaluasi aspek performa runtime atau keamanan sistem secara mendalam.

1.6 Sistematik Penulisan

1. BAB I Pendahuluan

Pada BAB I, berisi tentang pembahasan latar belakang penelitian, rumusan masalah, tujuan, manfaat, dan batasan masalah. Pembahasan pada BAB I ini bertujuan untuk menjelaskan konteks kesenjangan antara desain dan pengembangan, serta pentingnya mengevaluasi plugin Figma dalam menghasilkan kode berkualitas. Selain itu, BAB I juga mencakup sistematika penulisan penelitian yang dilakukan.

2. BAB II Landasan Teori

BAB II memaparkan teori-teori yang mendukung penelitian, termasuk konsep desain UI /UX, prinsip kode generasi, serta metode profiling untuk mengevaluasi kualitas kode. Kajian terhadap penelitian terdahulu yang relevan juga dibahas untuk memberikan landasan ilmiah dalam menganalisis efektivitas plugin Figma.

3. BAB III Metodologi Penelitian

Pada BAB III, dijelaskan metodologi penelitian yang meliputi langkah-langkah pengumpulan data, proses pengujian menggunakan plugin Figma, dan penerapan uji profiling. Alur penelitian ini bertujuan untuk memastikan bahwa penelitian dilakukan secara sistematis dan terukur.

4. BAB IV Hasil dan Pembahasan

BAB IV memaparkan hasil dari penelitian yang telah dilakukan berdasarkan metode yang dijelaskan pada BAB III. Pembahasan meliputi analisis kualitas kode

yang dihasilkan oleh plugin Figma, validitas hasil berdasarkan uji profiling, serta interpretasi data yang mengacu pada tujuan penelitian.

5. BAB V Kesimpulan dan Saran

Pada BAB V, disampaikan kesimpulan dari penelitian berdasarkan hasil analisis dan pembahasan. Selain itu, diberikan saran untuk pengembangan lebih lanjut, baik dalam penggunaan plugin Figma maupun dalam konteks penelitian yang relevan.