BAB V
KESIMPULAN DAN SARAN

5.1. Kesimpulan

Berdasarkan uraian penelitian yang telah penulis paparkan pada bab-bab sebelumnya, dapat ditarik kesimpulan sebagai berikut:

1. Untuk membuat aplikasi visualisasi diagram alir yang lebih mudah diakses, perlu dimanfaatkan teknologi berbasis web untuk bagian *frontend*-nya seperti HTML 5 Canvas, JavaScript, dan React.js. Sedangkan jika hendak melengkapi aplikasi tersebut dengan fitur *cloud save*, untuk bagian *backend* dapat digunakan teknologi di sisi server yang dalam penelitian ini penulis menggunakan Nest.js dan PostgreSQL untuk menyimpan datanya. Data diagram alir dapat direpresentasikan sebagai larik JSON yang masing-masing objek di dalamnya memiliki atribut berikut: id, jenis node, posisi x, posisi y, panjang, tinggi, konten, id dari node selanjutnya. Penggunaan tipe data JSON akan memudahkan pertukaran data antara *frontend* dan *backend* dikarenakan tipe data ini telah didukung secara internal oleh banyak bahasa pemrograman. Selain itu, aplikasi visualisasi diagram alir memerlukan fungsionalitas input ekspresi pemrograman, untuk itu diperlukan *interpreter* sendiri seperti halnya ketika membuat bahasa pemrograman. *Interpreter* ini terdiri atas beberapa bagian, seperti Tokenizer dan Parser. Algoritma *Recursive Descent Parsing* dapat dimanfaatkan pada bagian Parser untuk membangun Abstract Syntax Tree yang dapat dievaluasi sebagai ekspresi pemrograman ketika *flowchart* dijalankan.

2. Aplikasi Dializer berhasil memenuhi tujuannya sebagai alternatif RAPTOR yang lebih mudah diakses dengan 82% responden menyatakan bahwa Dializer lebih mudah diakses dibandingkan RAPTOR, 18% responden memilih netral, serta tidak ada responden yang menyatakan Dializer lebih sulit diakses. Kemudahan akses Dializer didukung dengan fungsionalitas yang sudah memadai yang dapat dilihat dari nilai rata-rata Dializer secara keseluruhan yang berada di angka 4,37. Dengan nilai

tersebut, Dializer sudah tergolong sangat baik dalam berbagai aspek seperti desain antarmuka, kemampuan memvisualisasikan diagram alir, kemampuan memfasilitasi input ekspresi pemrograman, serta kemudahan akses dan penggunaan. Selain itu, Dializer mampu bekerja dengan baik lintas-platform. Tercatat meskipun responden menggunakan peramban internet dan sistem operasi yang berbeda-beda, 100% responden menyatakan bahwa Dializer dapat diakses pada perangkat mereka dengan 94% responden menyatakan bahwa Dializer dapat dimuat dengan cepat dan 85% responden menyatakan Dializer dapat memvisualisasikan diagram alir dengan akurat. Yang terakhir, data hasil kuesioner juga menunjukkan bahwa 80% responden memberikan respon positif terkait kemungkinan mereka lebih memilih menggunakan Dializer dibandingkan RAPTOR jika situasi tidak memaksa mereka untuk menggunakan RAPTOR yang makin menegaskan posisi Dializer sebagai alternatif RAPTOR yang lebih mudah diakses dengan fungsionalitas yang memadai.

5.2. Saran

Kendati dapat memenuhi kebutuhan responden dengan cukup baik, Dializer masih dapat dikembangkan lebih lanjut karena masih memiliki beberapa kekurangan. Berikut adalah beberapa kekurangan tersebut yang dikonfirmasi oleh data masukan dari responden:

1. Dializer belum memfasilitasi drag-and-drop untuk mengubah stuktur diagram alir.
2. Belum adanya fungsionalitas *undo* dan *redo*.
3. Belum adanya fungsionalitas *auto-save* ketika pengguna membuat perubahan pada diagram alir.
4. Fungsionalitas pembagian *workspace* ke pengguna lain dapat dibuat lebih detail (seperti Google Docs dan Figma).
5. Belum adanya fungsionalitas kolaborasi secara *real-time* dengan pengguna lain.