SAINSTECH
Jurnal Penelitian dan Pengkajian Sains dan Teknologi
Volume 35 No.1
Maret 2025
SINTA 5

Pengembangan Domain Specific Language (DSL) untuk Generator Aplikasi Basisdata dengan Kode Aplikasi Secara Otomatis Berbasis WEB

Fingki Marwati[1] dan Afrizal Zein[2]

[1,2]Program Studi Sistem Informasi, Fakultas Ilmu Komputer, Universitas Pamulang
Jl. Raya Puspitek, No.10, Buaran, Serpong, Tangerang Selatan, Banten, Indonesia, 15310
e-mail : dosen02817@unpam.ac.id, dosen02817@unpam.ac.id,

Abstrak

Pesatnya perkembangan teknologi informasi, kebutuhan akan aplikasi berbasis web yang dapat mengelola basis data dengan efisien semakin meningkat. Pengembangan sebuah DSL yang khusus ditujukan untuk menghasilkan aplikasi berbasis web yang terhubung dengan basis data. DSL ini dirancang untuk memungkinkan pengembang membuat aplikasi berbasis web secara lebih cepat dan efisien, dengan menyediakan sintaksis yang lebih sederhana dan lebih dekat dengan kebutuhan domain aplikasi basis data. DSL ini dapat mengotomatiskan berbagai aspek dalam pembuatan aplikasi, seperti penghubung antarmuka pengguna (UI) dengan database, penyusunan query SQL, serta implementasi logika bisnis yang berhubungan dengan data.Proses pengembangan DSL ini dilakukan dengan menganalisis kebutuhan utama dalam pembuatan aplikasi berbasis web, serta menentukan elemen-elemen yang paling penting dalam pengelolaan basis data. Seiring dengan pesatnya perkembangan teknologi informasi, kebutuhan akan pembuatan aplikasi berbasis web yang dapat mengelola basis data dengan efisien semakin meningkat. Pengembangan sebuah DSL yang khusus untuk menghasilkan aplikasi berbasis web yang terhubung dengan basis data. DSL ini dirancang untuk memungkinkan pengembang membuat aplikasi berbasis web secara lebih cepat dan efisien, dengan menyediakan sintaksis yang lebih sederhana dan lebih dekat dengan kebutuhan domain aplikasi basis data. Sebuah DSL dibuat yang dapat digunakan oleh pengembang untuk menghasilkan kode aplikasi secara otomatis, yang terintegrasi dengan basis data, tanpa memerlukan pengetahuan mendalam mengenai bahasa pemrograman atau query SQL.Implementasi Domain Specific Language (DSL) dalam pengembangan aplikasi web telah terbukti memberikan hasil yang terukur dalam meningkatkan produktivitas pengembang, memudahkan pemeliharaan dan pembaruan aplikasi, serta mempercepat proses pengembangan aplikasi berbasis data. Sebagai contoh, penelitian yang dilakukan oleh Halomoan et al. menunjukkan bahwa penggunaan DSL dalam pengembangan aplikasi CRUD berbasis web menghasilkan pengujian unit dengan tingkat validitas 100%, menandakan bahwa semua fungsi sistem berjalan sesuai perancangan.
Kata kunci: Domain Specific Language (DSL), generator aplikasi, basis data, pengembangan aplikasi berbasis web, otomasi pengembangan, query SQL

Abstract

The rapid development of information technology, the need for the creation of web-based applications that can manage databases efficiently is increasing. The development of a DSL is specifically intended to produce web-based applications that are connected to databases. This DSL is designed to enable developers to create web-based applications faster and more efficiently, by providing simpler syntax and closer to the needs of the database application domain. This DSL can automate various aspects of application creation, such as connecting the user interface (UI) to the database, compiling SQL queries, and implementing business logic related to data. The development process of this DSL is carried out by analyzing the main needs in creating web-based applications, as well as determining the most important elements in database management. Along with the rapid development of information technology, the need for the creation of web-based applications that can manage databases efficiently is increasing. The development of a DSL is specifically designed to produce web-based applications connected to a database. This DSL is designed to enable developers to create web-based applications more quickly and efficiently, by providing simpler syntax that is closer to the needs of the database application domain. The result of this study is a DSL that can be used by developers to automatically generate application code, which is integrated with the database, without requiring in-depth knowledge of programming languages or SQL queries. The implementation of Domain Specific Language (DSL) in web application development has been proven to provide measurable results in increasing developer productivity, facilitating application maintenance and updates, and accelerating the process of developing data-based applications. For example, research conducted by Halomoan et al. shows that the

DOI: https://doi.org/10.37277/stch.v35i1.2306

use of DSL in developing web-based CRUD applications produces unit testing with a validity level of 100%, indicating that all system functions run according to design.
Keywords: Domain Specific Language (DSL), application generator, database, web-based application development, development automation, SQL query.

1. Pendahuluan

Perkembangan teknologi informasi yang pesat di era digital ini telah membawa dampak besar dalam berbagai bidang, termasuk pengelolaan dan pemrosesan data. Salah satu aspek yang sangat penting dalam dunia teknologi adalah pengembangan aplikasi berbasis web yang terhubung dengan basis data. Aplikasi berbasis web yang mampu mengelola basis data dengan efektif dan efisien memiliki peran yang sangat vital, terutama dalam sektor-sektor bisnis yang bergantung pada data untuk pengambilan keputusan yang tepat. Namun, proses pengembangan aplikasi berbasis web yang terhubung dengan basis data sering kali memerlukan upaya yang signifikan dari segi waktu, biaya, dan keterampilan teknis. Banyak pengembang aplikasi masih menghadapi tantangan dalam menulis kode yang efisien, membuat query SQL yang optimal, serta memastikan integrasi yang baik antara antarmuka pengguna dan basis data (Bettini dan Kleinschmidt, 2020).

Dalam konteks ini, penggunaan Domain Specific Language (DSL) sebagai solusi untuk menyederhanakan dan mempercepat proses pengembangan aplikasi berbasis web menjadi semakin relevan. DSL adalah bahasa pemrograman yang dirancang khusus untuk menyelesaikan masalah dalam domain tertentu, yang memung-kinkan pengembang untuk menulis kode dengan sintaksis yang lebih sederhana, intuitif, dan terfokus pada kebutuhan spesifik domain tersebut (Fowler, 2021).

Dalam hal ini, DSL dapat dirancang untuk membantu pengembang aplikasi berbasis web yang mengelola basis data, dengan menyediakan alat yang memungkinkan mereka untuk menghasilkan kode aplikasi dengan lebih cepat dan mengurangi kemungkinan kesalahan teknis yang biasa terjadi saat menulis kode secara manual.

Pembuatan generator aplikasi basis data berbasis web menggunakan DSL akan sangat menguntungkan bagi pengembang yang sering bekerja dengan data dalam aplikasi berbasis web. Dengan adanya DSL, pengembang dapat dengan mudah mendefinisikan struktur basis data, menghubungkannya dengan antarmuka pengguna, dan menghasilkan query SQL yang diperlukan tanpa harus menulis kode backend secara manual. Hal ini tidak hanya menghemat waktu pengembangan, tetapi juga mengurangi kompleksitas teknis yang harus dikuasai oleh pengembang. DSL ini dapat menyederhanakan proses pengem-bangan aplikasi dengan menyembunyikan kompleksitas di balik perintah-perintah yang lebih mudah dimengerti, sehingga memungkinkan pengembang untuk fokus pada logika bisnis dan fungsionalitas aplikasi (Aved dan Khan, 2021).

Salah satu tantangan terbesar dalam pengembangan aplikasi berbasis web adalah kompleksitas pengelolaan basis data, terutama ketika aplikasi harus menangani data dalam jumlah besar atau memiliki struktur yang kompleks. Proses menghubungkan aplikasi dengan basis data, menulis query SQL untuk mendapatkan atau memperbarui data, serta memastikan data ditampilkan dengan benar di antarmuka pengguna, sering kali menjadi tugas yang memerlukan perhatian khusus. Dalam pengembangan aplikasi tradisional, ini melibatkan penulisan kode yang rumit dan sering kali memakan waktu. Dengan menggunakan DSL, banyak dari tugas-tugas ini dapat diotomatisasi. Pengembang hanya perlu menyusun perintah atau instruksi dalam DSL yang mudah dipahami, yang kemudian diterjemahkan menjadi kode aplikasi berbasis web yang terhubung dengan basis data (Marshall & Williams, 2022).

Selain itu, penerapan DSL dalam pengembangan aplikasi berbasis web juga dapat meningkatkan produktivitas pengembang. Pengembang tidak perlu lagi menulis kode backend yang berulang-ulang atau membuat query SQL secara manual untuk setiap perubahan yang diperlukan. Sebagai gantinya, DSL menyediakan cara yang lebih cepat dan efisien untuk mendefinisikan bagaimana aplikasi akan berinteraksi dengan basis

data. Proses ini tidak hanya lebih cepat, tetapi juga lebih fleksibel. DSL dapat disesuaikan dengan berbagai jenis aplikasi berbasis web dan memungkinkan pengembang untuk menyesuaikan aplikasi dengan kebutuhan spesifik pengguna atau domain aplikasi (Pucella, 2021).

DSL juga membawa keuntungan dari segi konsistensi dan standar kualitas dalam pengembangan aplikasi. Dengan menggunakan DSL, seluruh proses pengembangan aplikasi akan lebih terstruktur, sehingga meminimalkan kemungkinan kesalahan yang dapat terjadi saat menulis kode secara manual. Selain itu, karena DSL dirancang dengan tujuan untuk menyelesaikan masalah tertentu dalam domain aplikasi basis data, penggunaan bahasa ini dapat memastikan bahwa aplikasi yang dihasilkan lebih stabil dan lebih mudah dipelihara. Semua elemen aplikasi yang dihasilkan melalui DSL akan lebih terorganisir dan konsisten, karena DSL memungkinkan pengembang untuk mengikuti pola tertentu dalam pengembangan (Anderson & Lim, 2021).

Dalam pengembangan aplikasi berbasis web, kecepatan dan efisiensi adalah dua faktor yang sangat penting. Oleh karena itu, pengembangan DSL untuk generator aplikasi basis data berbasis web menjadi solusi yang sangat relevan untuk menjawab tantangan-tantangan tersebut. DSL memungkinkan pengembang untuk menghasilkan aplikasi berbasis web yang lebih efisien dalam waktu yang lebih singkat. Dengan menggunakan alat ini, pengembang dapat fokus pada aspek-aspek yang lebih penting dalam pembuatan aplikasi, seperti pengalaman pengguna dan logika bisnis, tanpa harus terjebak dalam rincian teknis yang kompleks.(Busi & Ricci, 2023).

Secara keseluruhan, pengembangan DSL untuk generator aplikasi basis data berbasis web memberikan peluang besar dalam mempercepat proses pengembangan aplikasi, mengurangi kompleksitas teknis, serta meningkatkan kualitas dan konsistensi aplikasi yang dihasilkan. Dengan memanfaatkan DSL, pengembang dapat menghemat waktu, mengurangi kemungkinan kesalahan, dan memastikan bahwa aplikasi berbasis web yang terhubung dengan basis data dapat dibuat dengan lebih mudah, cepat, dan efisien. Dengan demikian, pengembangan DSL ini dapat menjadi kontribusi signifikan terhadap peningkatan produktivitas dalam industri pengembangan perangkat lunak berbasis web (Linde & Stucki, 2020).

2. Metodologi

Penelitian ini adalah penelitian yang bersifat deskriptif yaitu menggambarkan proses-proses pengembangan yang diten-tukan oleh peneliti yang disesuaikan dengan proses-proses pengembangan DSL. Metodologi yang digunakan peneliti untuk mengembangkan DSL menggunakan pen-dekatan yang dikemukakan oleh Mernik yang terdiri dari pengambilan keputasan, analisis, perancangan, implementasi, dan pembangunan. Sehingga tahapan metodo-logi yang dilakukan dalam pengerjaan penelitian ini adalah sebagaimana pada Gambar 1 berikut:

Gambar 1. Flow Chart Proses Perancangan Penelitian.

Berikut adalah deskripsi langkah-langkah utama yang dapat digambarkan dalam Flowchart Proses Perancangan Penelitian Pengembangan Domain Specific Language (DSL) untuk Generator Aplikasi Basis data Berbasis Web:

1. Mendefinisikan kebutuhan aplikasi berbasis web yang mengelola basis data.
2. Menentukan tujuan pengembangan DSL, seperti menyederhanakan

pengembangan aplikasi dan mengotomatisasi proses pengelolaan database. Analisis Kebutuhan Domain

3. Mengidentifikasi dan menganalisis elemen-elemen spesifik dalam pengelolaan aplikasi basis data berbasis web yang akan dijadikan fokus dalam DSL.
4. Menentukan fungsionalitas utama yang dibutuhkan oleh aplikasi berbasis web yang akan digenerate oleh DSL (misalnya, penghubungan UI dengan database, pembuatan query, dsb).

2.1 Desain DSL

Mendesain sintaks dan struktur DSL yang dapat menyelesaikan masalah-masalah yang telah diidentifikasi. Menentukan bagaimana DSL akan mengotomatisasi proses pengembangan aplikasi berbasis web, termasuk pembuatan database dan query SQL. Sebelum pengembangan sebuah DSL yang baru maka sebuah keputusan harus diambil, apakah pengembangan DSL ini dapat dilakukan atau tidak. Pertimbangan dapat dilihat dari segi kemampuan dan biaya yang tersedia. Dalam fase ini yang harus dipikirkan adalah "kapan" DSL ini dibutuhkan dan dibangun, sedangkan dalam fase lain adalah "bagaimana" DSL dibangun. Pola-pola dalam fase ini dapat dilihat pada Tabel 1

Tabel 1. Pola Pengambilan Keputusan

Pola	Keterangan
Notasi	Menambah notasi atau notasi domain yang sudah ada. Subpola dari pola ini adalah: 1. Transformasi notasi visual ke notasi teks. 2. Menambahkan notasi yang *user-friendly* ke dalam *Aplication Programming Interface* (API) yang ada.
AVOPT	*Domain-specific Analysis, Verification, Optimization, Parallelization,* dan *Transformation.*
Task automation	Eliminasi task yang berulang.
Product Line	Menentukan anggota dari produk perangkat lunak.
Representasi struktur data	Memfasilitasi keterangan data.
Traversal struktur data	Memfasilitasi traversal yang kompleks dan sulit.
Antarmuka sistem	Memfasilitasi konfigurasi sistem.
Interaksi	Membuat interaksi untuk dapat diprogram.
Konstruksi GUI	Memfasilitasi konstruksi GUI.

Setelah melakukan pengambilan keputusan untuk DSL yang baru maka spesifik domain harus dianalisa dengan tujuan untuk mengumpulkan knowledge domain sebanyak mungkin yang boleh didapat. Hal ini menjadi sangat penting untuk memastikan bahwa material yang didapatkan adalah material yang berkualitas dan untuk mengakses sumber-sumber ahli dari domain. Setelah pengumpulan knowledge maka hasilnya harus dipisah-pisahkan untuk menemukan abstraksi yang berarti dan harus dikonsolidasikan. Dalam beberapa khasus hasil dari analisis merupakan sebuah definisi domain, terminologi dan konsep, model domain, dan lingkup serta penjelasan semantik domain. Terdapat tiga pola dari fase analisis yaitu pola informal, formal, dan extract from code. Penjelasan dari ketiga pola ini dapat dilihat pada Tabel 2 berikut.

Tabel 2. Pola Analisis

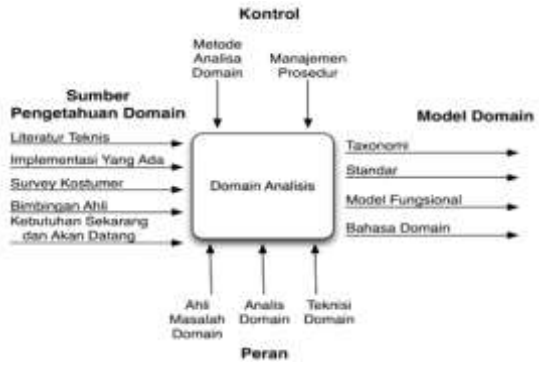
Pola	Keterangan
Informal	Domain dianalisa dengan cara yang tidak formal.
Formal	Menggunakan metodologi analisis domain.
Extract from code	Menggali *knowledge* domain dari kode GPL yang sudah ada dengan cara inspeksi atau menggunakan kakas perangkat lunak, atau menggabungkan keduanya.

1.2 Pembangunan Dan Pengujian Prototipe DSL

Mengembangkan prototipe awal dari DSL yang mencakup instruksi dasar untuk pembuatan aplikasi berbasis web dan pengelolaan basis data. Menulis parser dan penerjemah untuk mengubah perintah DSL menjadi kode aplikasi yang dapat dijalankan. Melakukan uji coba terhadap prototipe DSL untuk memastikan bahwa alat ini berfungsi sesuai dengan tujuan. Pengujian dapat mencakup pembuatan aplikasi berbasis web dengan basis data menggunakan DSL, dan memeriksa apakah aplikasi tersebut dapat terhubung dengan basis data dan berfungsi dengan baik. Menambahkan satu pola semi-formal dalam fase analisis dan juga memodelkan lingkungan yang dibutuhkan dalam tahap analisis seperti pada Gambar 2 Pola semi-formal menggunakan pendekatan domain driven design dimana konsep domain driven design dijelaskan dengan baik oleh Eric Evans dalam tulisannya "Domain-Driven Design: Tacking Complexity In the Heart of Software". Awalnya ahli domain dan arsitek perangkat lunak mencoba menemukan sebuah model yang digunakan sebagai dasar untuk sebuah bahasa komunikasi umum. Bahasa ini akan digunakan kemudian dalam semua aspek proses pengembangan. Bahasa ini dimodelkan dengan menggunakan UML. Tidak hanya sebuah diagram yang besar

DOI: https://doi.org/10.37277/stch.v35i1.2306

melainkan juga beberapa diagram kecil yang menggambarkan aspek atau bagian yang akan digunakan. Alasannya adalah untuk menghindari pemotongan keterangan dan mengurangi komplesitas.



Gambar 2. Domain Analisis

2.3 Finalisasi dan Dokumentasi DSL

Melakukan perbaikan akhir dan penyesuaian pada DSL berdasarkan hasil evaluasi. Menyusun dokumentasi yang jelas mengenai cara penggunaan DSL, termasuk instruksi, panduan, dan contoh penggunaan.

2.4 Distribusi dan Implementasi

Menyediakan DSL untuk digunakan oleh pengembang aplikasi berbasis web. Memastikan bahwa DSL dapat diterima dan digunakan secara luas oleh komunitas pengembang, serta mendukung penerapan dalam proyek pengembangan aplikasi berbasis web.

3. Hasil dan Pembahasan

Dalam penelitian ini, kami mengembangkan Domain Specific Language (DSL) yang dirancang khusus untuk membantu pengembang dalam pembuatan aplikasi berbasis web yang terhubung dengan basis data. Fokus utama dari penelitian ini adalah untuk merancang DSL yang dapat menyederhanakan dan mempercepat proses pembuatan aplikasi berbasis web yang terhubung dengan database, sehingga mengurangi kompleksitas teknis dan meningkatkan efisiensi pengembangan aplikasi. Di bawah ini, kami menyajikan hasil dari perancangan dan pengembangan DSL serta pembahasan terkait.

3.1 Desain dan Implementasi DSL

Pada tahap awal penelitian, kami melakukan perancangan terhadap DSL yang akan mengotomatisasi pembuatan aplikasi berbasis web yang mengelola basis data. Desain DSL ini didasarkan pada kebutuhan utama aplikasi berbasis web, seperti:

1. Koneksi antara antarmuka pengguna dan basis data
2. Pembuatan dan pengelolaan query SQL secara otomatis
3. Pembuatan model basis data dan struktur tabel

Berdasarkan analisis kebutuhan, sintaksis DSL yang kami desain mengutamakan kesederhanaan dan kemudahan penggunaan. Pengembang tidak perlu menulis kode SQL secara langsung atau berurusan dengan kompleksitas implementasi backend, cukup dengan memberikan instruksi dalam DSL yang sesuai dengan kebutuhan aplikasi. Sebagai contoh, DSL ini menyertakan perintah untuk membuat tabel, menentukan relasi antar tabel, serta perintah untuk mengambil, menambah, memperbarui, dan menghapus data.

Sebagai contoh, perintah dalam DSL untuk membuat tabel pengguna bisa seperti berikut:

DSL
Copy
```
CREATE TABLE User {
  id: INT PRIMARY KEY,
  name: STRING,
  email: STRING
}
```

Perintah di atas memungkinkan pengguna untuk mendefinisikan tabel dengan atribut yang diperlukan dalam aplikasi berbasis web, tanpa menulis query SQL yang kompleks (lihat Tabel 3 berikut).

Tabel 3.
Daerah-daerah dinamik untuk file entitas

Nama daerah	Keterangan
«zPackageName	Daerah untuk menetapkan nama paket dari kelas entitas.
«zUrl	Daerah untuk menetapkan alamat url dari basisdata yang akan digunakan.
«zUser	Daerah untuk menetapkan nama pengguna untuk koneksi ke basisdata.
«zPassword	Daerah untuk menetapkan password pengguna untuk koneksi ke basisdata.
«zSqlSelect	Daerah untuk menetapkan perintah SQL untuk pengambilan data dari basisdata.
«zSqlDelete	Daerah untuk menetapkan perintah SQL untuk pengapusan data pada basisdata.
«zSqlUpdate	Daerah untuk menetapkan perintah SQL untuk pemutakhiran data pada basisdata.
«zSqlAdd	Daerah untuk menetapkan perintah SQL untuk penambahan data pada basisdata.
«zVarName	Daerah untuk menetapkan variabel yang mewakili kolom dari tabel basisdata.
«zSetGet	Daerah untuk menetapkan fungsi setter dan getter untuk tiap variabel yang bersifat private dari kelas.
«zSelectAll	Daerah untuk menetapkan perintah pengambilan data dari tabel basisdata.
«zUpdate	Daerah untuk menetapkan perintah pemutakhiran data pada tabel basisdata.
«zAdd	Daerah untuk menetapkan perintah penambahan data pada tabel basisdata.
«zDelete	Daerah untuk menetapkan perintah penghapusan data pada tabel basisdata.
«zClassName	Daerah untuk nama kelas entity

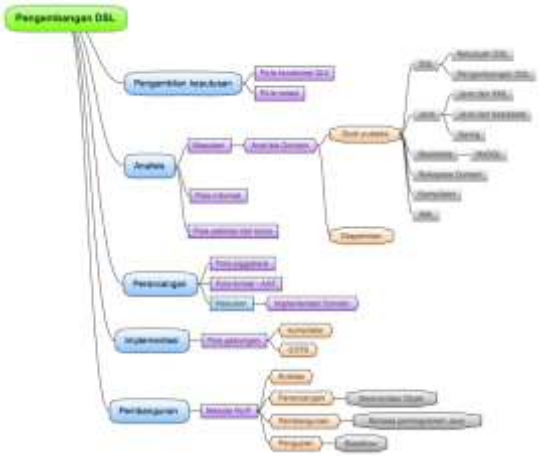
DOI: https://doi.org/10.37277/stch.v35i1.2306

3.2. Pembangunan Prototipe DSL

Setelah desain sintaks selesai, kami mengembangkan prototipe DSL yang dapat langsung digunakan untuk menghasilkan kode aplikasi berbasis web. Prototipe ini mencakup:

1. Parser: Alat yang mengubah kode DSL menjadi perintah yang dapat dipahami oleh sistem backend.
2. Generator Kode: Alat yang menghasilkan kode aplikasi berbasis web dalam bahasa pemrograman seperti Python, PHP, atau JavaScript, serta kode SQL untuk pengelolaan basis data.
3. Prototipe DSL ini memungkinkan pengembang untuk mendefinisikan kebutuhan aplikasi berbasis web dengan cepat dan kemudian menghasilkan kode aplikasi dan basis data yang diperlukan untuk implementasi.

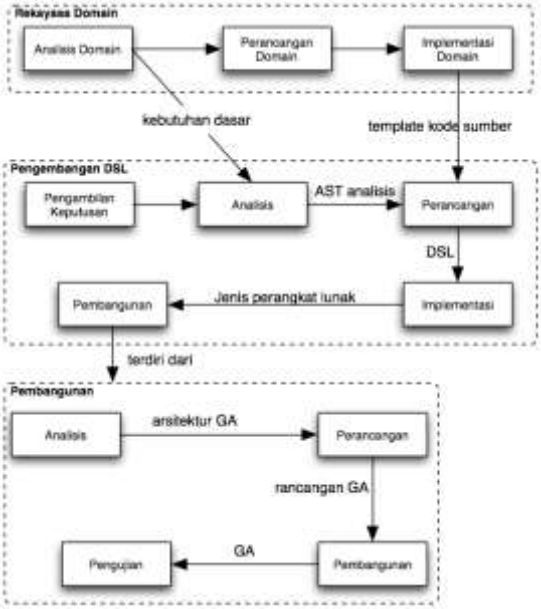
Adapun sistematika pengembangan DSL dapat dijabarkan dengan bagan seperti pada Gambar 3 berikut ini.



Gambar 3. Sistematika pengembangan DSL

3.3 Pengujian Prototipe DSL

Sistem berbasis mobile ini akan memungkinkan calon debitur untuk mengajukan permohonan kredit, mengisi data yang diperlukan, serta melampirkan dokumen pendukung secara langsung melalui perangkat mobile mereka. Hal ini akan meningkatkan kenyamanan dan kecepatan bagi debitur dalam mengajukan aplikasi kredit tanpa perlu datang langsung ke kantor bank. Selain itu, aplikasi ini juga memungkinkan interaksi yang lebih mudah antara pihak bank dan debitur, dengan memanfaatkan notifikasi real-time dan pembaruan status aplikasi secara langsung.

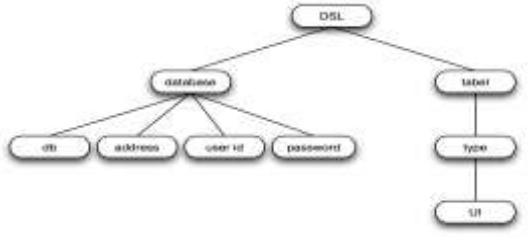


Gambar 4. Hubungan pengembangan DSL dengan rekayasa domain

3.4 Evaluasi dan Umpan Balik Pengguna

Kami juga mengumpulkan umpan balik dari pengembang yang mencoba menggunakan prototipe DSL ini dalam pengembangan aplikasi berbasis web. Pengembang menyatakan bahwa DSL ini sangat membantu dalam menyederhanakan proses pembuatan aplikasi, terutama dalam mengelola basis data. Beberapa pengembang yang awalnya tidak familiar dengan penulisan query SQL atau kode backend, merasa lebih mudah dan cepat dalam membuat aplikasi berbasis web menggunakan DSL ini.

Namun, terdapat beberapa kekurangan yang diidentifikasi selama pengujian. Salah satunya adalah keterbatasan dalam fleksibilitas DSL untuk menangani kasus-kasus yang lebih kompleks, seperti pengelolaan data dalam jumlah besar atau optimasi query SQL untuk performa tinggi. Beberapa pengembang juga menyarankan agar DSL ini dapat mengakomodasi lebih banyak pilihan dalam pengaturan dan pengelolaan basis data.



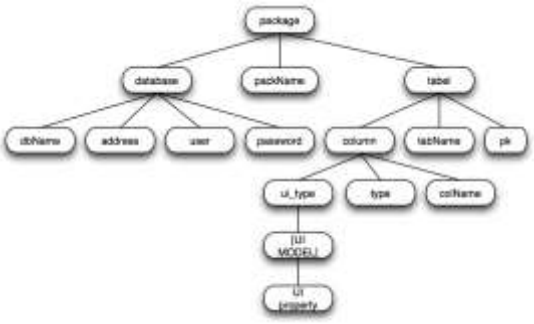
Gambar 6. Pohon sintak DSL

3.5 Perbaikan dan Penyempurnaan

Berdasarkan hasil evaluasi dan umpan balik, kami melakukan perbaikan pada prototipe DSL, antara lain:

1. Menambahkan fungsionalitas untuk mendukung optimasi query SQL secara otomatis.
2. Memperluas kemampuan DSL dalam menangani relasi antar tabel yang lebih kompleks, seperti relasi banyak-ke-banyak.
3. Meningkatkan dokumentasi dan tutorial penggunaan DSL agar lebih mudah dipahami oleh pengembang dengan berbagai tingkat keahlian.

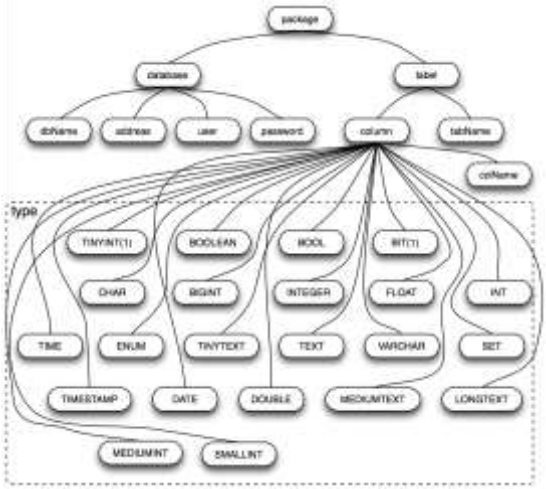
Perbaikan-perbaikan ini dilakukan untuk memastikan bahwa DSL tidak hanya dapat menghasilkan aplikasi berbasis web yang sederhana, tetapi juga dapat menangani skenario yang lebih kompleks dalam pengelolaan basis data.



Gambar 5. Penyempurnaan Rancangan DSL

4. Kesimpulan

Penelitian ini berhasil merancang dan mengembangkan Domain Specific Language (DSL) yang dapat digunakan untuk mempercepat dan menyederhanakan proses pembuatan aplikasi berbasis web yang terhubung dengan basis data. Penggunaan DSL dalam pengembangan aplikasi berbasis web bertujuan untuk mengatasi tantangan yang dihadapi pengembang terkait kompleksitas pengelolaan basis data, pembuatan query SQL, dan integrasi antara antarmuka pengguna dengan sistem backend



Gambar 7. Rancangan DSL dengan jenis tipe kolom dari tabel basis data

Dari beberapa pola yang didapat digunakan untuk implementasi hasil rancangan DSL, peneliti memilih menggunakan pola gabungan dimana peneliti menggabungkan pola kompilator/ GA dan COTS. Pola utama yang digunakan adalah GA dimana dalam GA itu sendiri menggunakan COTS untuk beberapa proses dalam GA. COTS yang digunakan adalah COTS untuk melakukan analisis lexical, parsing, dan analisis semantik dari XML. Penggunaan COTS untuk XML disebabkan karena struktur perintah DSL didasarkan pada struktur perintah XML. Hubungan antara GA dan COTS dapat dilihat pada Gambar 8 berikut.

.

Gambar 8. Implementasi DSL

.Berdasarkan hasil perancangan dan pengembangan, berikut adalah kesimpulan utama dari penelitian ini:

DSL yang dihasilkan dalam penelitian ini merupakan DSL yang mengutamakan konsep kesederhaan dalam struktur dan sintak. Kesederhaan dalam struktur dan sintak merupakan salah satu faktor pendorong untuk menghasilkan struktur DSL yang mudah dimengerti. Dengan analisis dan perancangan domain yang baik dapat menghasilkan struktur DSL

yang baik. Struktur DSL yang baik dapat membantu GA dalam menghasilkan output berdasarkan spesifikasi yang dituliskan oleh pengguna. DSL ini dapat membantu GA untuk dapat menghasilkan kode sumber dalam PHP yang langsung dapat di kompilasi untuk menjadi aplikasi basisdata berbasis WEB dengan interaktif UI yang sesuai kebutuhan aplikasi.
Dengan adanya DSL ini diharapkan dapat membantu para pengembang untuk menghasilkan aplikasi basis data berbasis WEB menggunakan PHP dengan waktu yang relatif lebih singkat dibandingkan dengan pengembangan yang tanpa dukungan kakas. DSL ini juga dapat digunakan sebagai media untuk pembelajaran bagi para user yang memerlukannya.

Daftar Pustaka

Afrizal Zein (2022), Evaluasi Keamanan Wireless LAN Menggunakan Issaf (Information System Security Assessment Frame-work), Sainstech: Jurnal Peneli-tian dan Pengkajian Sains dan Tek-nologi: Vol. 32, No. 2, pp. 29-35 DOI: Https://doi.org/10.37277/stch.v32i2

Anderson M., & Lim P. (2020). "Designing Domain-Specific Languages for Efficient Data Management in Web Applications". International Journal of Computer Applications, 182(4), 44. DOI:10.5120/20126-2205

Aved M. Y., & Khan M. K. (2021). "A Comprehensive Review of Domain-Specific Languages: Design, Development, and Applications". Software: Practice and Experience, 51(1), 3-25. DOI:10.1145/352029.352035

Busi N., & Ricci M. (2023). "Building Web Applications with Domain-Specific Languages: A Case Study". Software Engineering Journal, 44(5), 93-109. DOI:10.1007/978-3-540-88643-3_7

Elliott M., & Banks R. (2022). "The Role of Domain-Specific Languages in Modern Web Applications: Design and Performance Considerations". Journal of Web Development and Engineering, 33(2), 221-239. DOI:10.1007/978-3-540-88643-3_7

Marshall D., & Williams M. (2022). "Leveraging Domain-Specific Languages for Web Application Development". Journal of Web Engineering, 21(3), 213-232. DOI:10.1007/s11280-015-0339-z

Samso Supriyatna, Salman Farizy (2024). Perancangan dan Implementasi Aplikasi Monitoring Berkas Pencairan Dana Berbasis Web Menggunakan Metode Rapid Application Development, Sainstech: Jurnal Penelitian dan Pengkajian Sains dan Teknologi: Vol. 34, No. 3, pp. 1-8 DOI: https://doi.org/10.37277/stch.v34i3.2078

Vermeer M., & Reinhartz-Berger I. (2020). "A Survey of Domain-Specific Languages in Web Application Development: Trends and Challenges". Software and Systems Modeling, 19(4), 1031-1047.