Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

Perpustakaan Univeristas Multimedia Nusantara

BAB III

METODOLOGI DAN PERANCANGAN SISTEM

3.1 Metodologi Penelitian

Alur kerja penelitian ini dilakukan berdasarkan langkah-langkah berikut.

1. Studi Literatur

   Dalam tahap ini dilakukan pembelajaran mengenai teori dan literatur yang berkaitan dengan metode *n-gram*. Beberapa sumber yang digunakan adalah buku, jurnal, skripsi, dan artikel *online*.

2. Analisis Kebutuhan Sistem

   Pada tahap ini dilakukan analisis terhadap kebutuhan sistem. Analisis berupa perencanaan *training corpus* yang akan digunakan, perancangan *database*, serta perancangan fitur apa saja yang dibutuhkan dalam proses pemrograman dan *debug.*

3. Desain Sistem

   Pada tahap ini, aplikasi yang dirancang memanfaatkan konsep *n-gram* yang digunakan untuk memecah informasi berupa kumpulan kalimat yang biasa disebut *corpus* menjadi kombinasi pasangan 4 kata unik yang disebut *quadgram*. Kombinasi pasangan kata yang sudah dipecah tersebut kemudian dimasukkan kedalam *database* dan digunakan sebagai informasi dalam mengoreksi kalimat pada sistem yang dibuat. Agar dapat memahami alur kerja dari sistem, rancangan dibuat meliputi *Use Case* Diagram, *Activity* Diagram, *Sequence* Diagram, *Class* Diagram serta desain rancangan antarmuka.

4. Implementasi dan Pengujian Sistem

   Setelah dilakukan desain untuk sistem yang dibuat, pemrograman sistem untuk mengimplementasikan metode dilakukan menggunakan bahasa pemrograman PHP dan dibantu dengan menggunakan *framework CodeIgniter*. Setelah pemrograman sistem selesai, uji coba dilakukan dengan memasukkan 100 kalimat dengan banyak kata dalam kalimat yang dimasukkan paling sedikit 4 kata. Pengujian aplikasi dilakukan dengan metode *testing*.

5. Evaluasi Sistem

   Pada tahap ini, evaluasi terhadap keluaran dari sistem akan diserahkan kepada pakar dalam bidang pendidikan sekolah dasar, yaitu Alfiani Syukrina Yusut, S.Pd, untuk memeriksa berapa banyak kalimat yang bersifat *grammatical correct* atau benar secara tata bahasa.

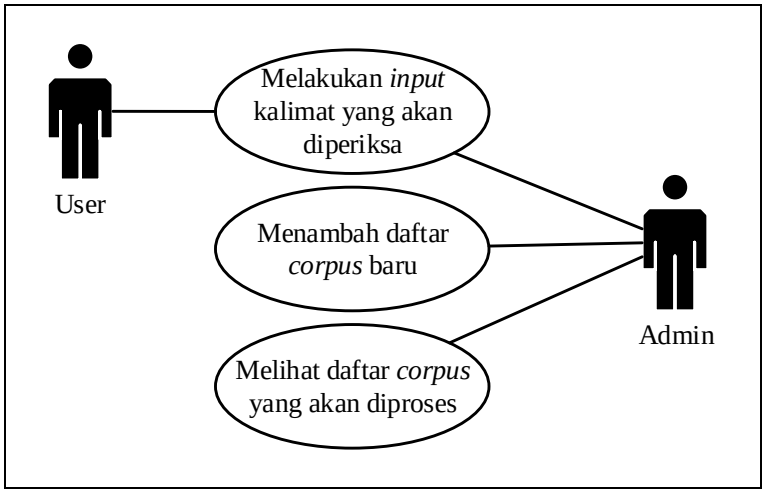
3.2 Perancangan Sistem

Perancangan sistem dalam penelitian ini dilakukan perancangan awal terlebih dahulu untuk memahami alur kerja dari sistem. Perancangan yang dibuat adalah sebagai berikut.

3.2.1 Use Case Diagram

Use case diagram digunakan untuk menggambarkan fungsionalitas suatu sistem atau bagian dari suatu sistem. *Use case* banyak digunakan untuk menggambarkan persyaratan fungsional sistem dan interaksinya dengan aktor. Pada dasarnya, *use case* adalah diagram yang mewakili berbagai skenario dimana sistem dapat digunakan sehingga dapat memberi kita pandangan tingkat tinggi tentang apa yang dilakukan oleh sistem atau bagian dari sistem tanpa masuk ke detail

implementasi (geeksforgeeks.org, 2019). Gambar 3.1 di bawah ini menggambarkan relasi antar aktor terhadap sistem yang dibuat.



Gambar 3.1 *Use Case* Diagram

Sedangkan Tabel 3.1 di bawah ini akan memberikan informasi mengenai fungsionalitas sistem terhadap aktor yang ada.

Tabel 3.1 Tabel Kegiatan *Use Case*

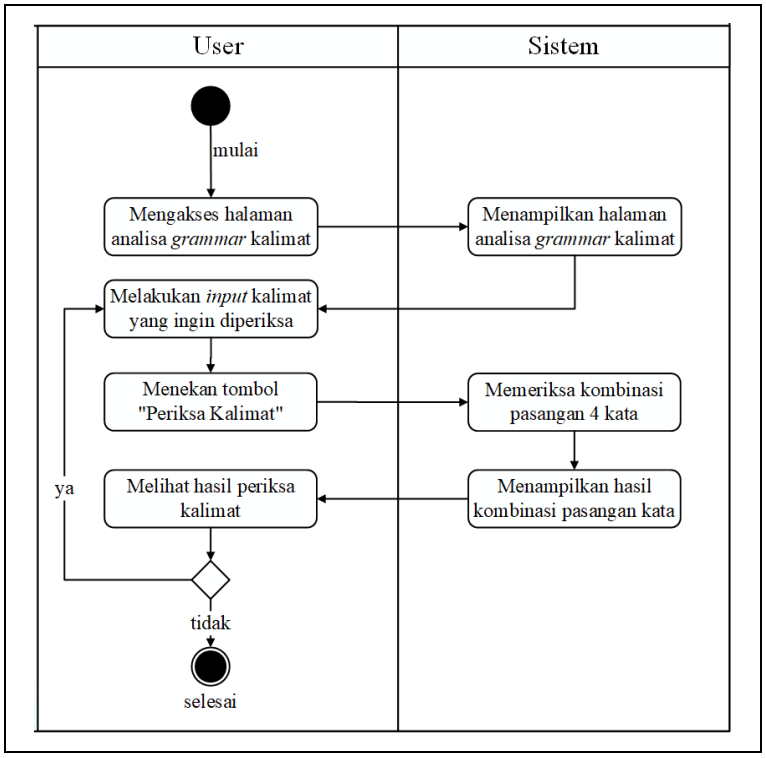
No.	Aktor	Use Case	Keterangan
1.	User	Melakukan *input* kalimat yang akan diperiksa	*User* dapat melakukan *input* kalimat apa saja yang akan diperiksa oleh aplikasi.
2.	Admin	Menambah daftar *corpus* baru	*Admin* dapat menambah daftar *corpus* baru yang akan digunakan sebagai regerensi dalam pengoreksian kalimat
3.	Admin	Melihat daftar *corpus* yang belum diproses	*Admin* dapat melihat daftar *corpus* apa saja yang belum diproses oleh sistem.

3.2.2 Activity Diagram

Activity diagram digunakan untuk menggambarkan aliran kontrol dalam suatu sistem. *Activity* diagram juga merujuk ke langkah-langkah yang terlibat dalam *use case* (uml-diagrams.org, 2019).

A. Activity Melakukan Input Kalimat yang akan diperiksa - User

Gambar 3.2 di bawah ini menggambarkan *activity* melakukan *input* kalimat yang akan diperiksa.



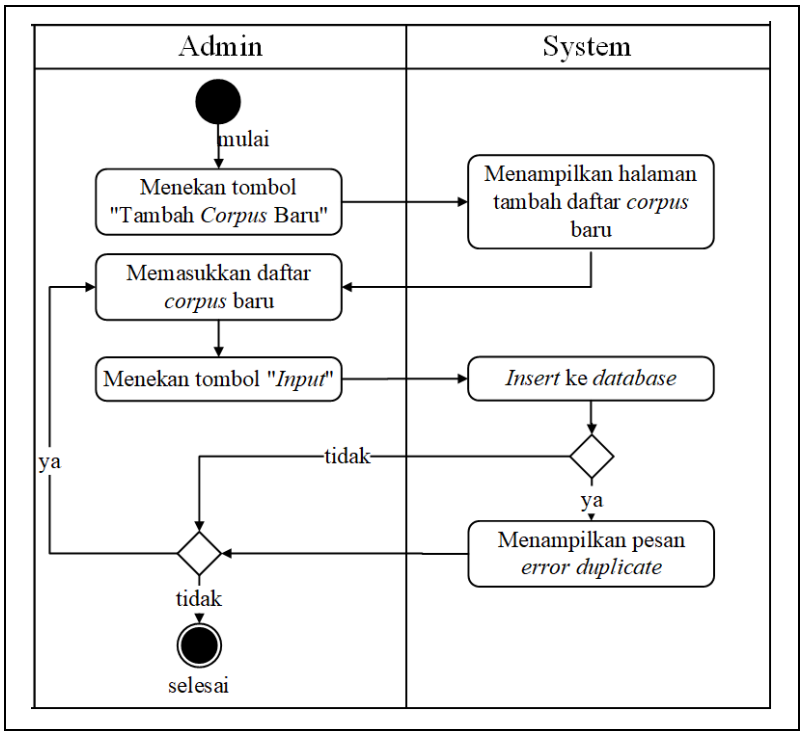
Gambar 3.2 *Activity* Melakukan *Input* Kalimat yang akan diperiksa - *User*

Proses dimulai ketika *user* mengakses halaman analisa *grammar* kalimat, kemudian sistem akan memproses untuk menampilkan halaman tersebut. Setelah halaman berhasil diakses, *user* akan memasukkan kalimat yang ingin diperiksa. Setelah selesai memasukkan kalimat, *user* akan menekan tombol “Periksa Kalimat” untuk memproses kalimat agar diperiksa oleh sistem. Sistem akan memeriksa rangkaian kombinasi kata yang dimasukkan oleh *user* ke *database*. Setelah sistem memeriksa rangkaian kombinasi kata dan hasil sudah didaptkan, hasil tersebut akan ditampilkan pada halaman analisa *grammar* kalimat tersebut. *User* akan melihat hasil dari pemeriksaan kalimat yang sudah dimasukkan sebelumnya. Jika *user* ingin

memeriksa kalimat lain, *user* dapat memasukkan kalimat pada kolom *input* kalimat, namun jika tidak ingin melakukan *input* kalimat, maka proses akan berakhir.

B. Activity Menambah Daftar Corpus – Admin

Gambar 3.3 di bawah ini menggambarkan *activity* menambah daftar *corpus.*



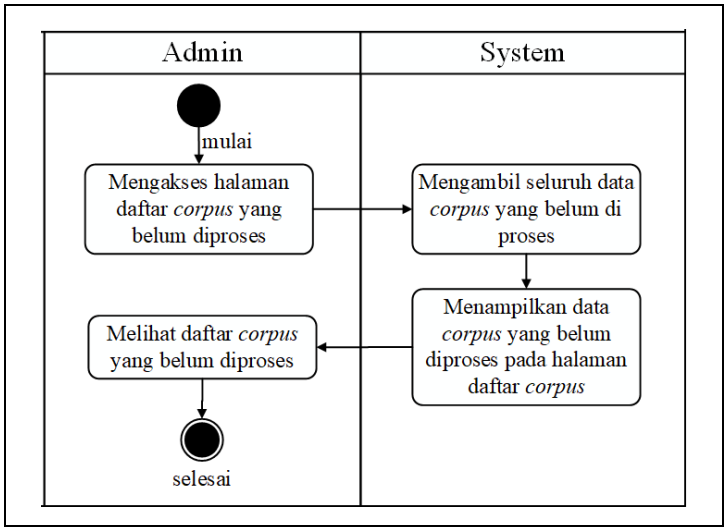
Gambar 3.3 *Activity* Menambah Daftar *Corpus* - *Admin*

Proses dimulai ketika *admin* menekan tombol “Tambah *Corpus* Baru” kemudian sistem akan memproses untuk menampilkan halaman tambah daftar *corpus* baru. Setelah halaman berhasil ditampilkan, *admin* akan memasukkan seluruh kalimat yang akan dijadikan tambahan daftar *corpus* baru. Proses memasukkan kalimat baru sebagai daftar *corpus* baru dapat dilakukan beberapa kalimat secara bersamaan. Selanjutnya, *admin* akan menekan tombol “*Input*”. Sistem akan menerima perintah untuk memproses daftar *corpus* baru yang telah dimasukkan oleh *admin* untuk dimasukkan ke dalam *database*. Jika kalimat sudah

pernah dimasukkan sebelumnya oleh *admin*, sistem akan memberikan pesan *error duplicate entry* bahwa kalimat sudah pernah dimasukkan. *Admin* dapat memilih ingin memasukkan kalimat yang lain atau tidak. Jika ingin memasukkan kalimat baru maka kalimat tersebut dapat dimasukkan pada kolom yang *input* kalimat. Jika tidak ingin memasukkan kalimat lain lagi, maka proses akan dinyatakan selesai.

C. Activity Melihat Daftar Corpus yang belum diproses - Admin

Gambar 3.4 di bawah ini menggambarkan *activity* ketika *admin* melihat daftar *corpus* yang belum diproses.



Gambar 3.4 *Activity* Melihat Daftar *Corpus* yang belum diproses - *Admin*

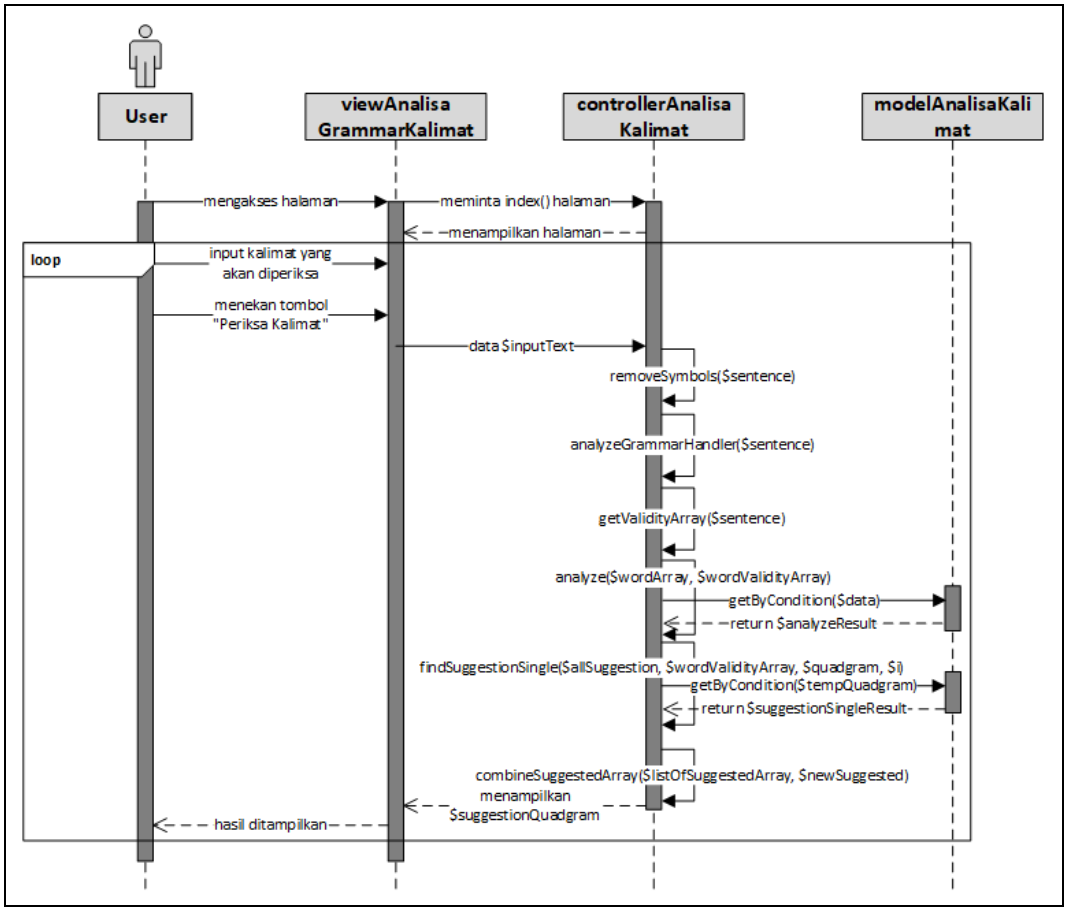
Proses dimulai ketika *admin* mengakses halaman, kemudian sistem akan memproses halaman untuk ditampilkan dan mengambil seluruh data *corpus* yang belum diproses dari *database* untuk ditampilkan pada halaman daftar *corpus* yang belum diproses.

3.2.3 Sequence Diagram

Sequence diagram merupakan jenis diagram interaksi yang paling umum, yang berfokus pada pertukaran pesan antara sejumlah *lifeline*. *Sequence* diagram umunya menggambarkan interaksi dengan berfokus pada urutan pesan yang dipertukarkan bersama dengan spesifikasi kemunculannya yang sesuai pada *lifeline* (uml-diagrams.org, 2019).

A. Sequence Melakukan Input Kalimat yang akan diperiksa – User

Gambar 3.5 di bawah ini menggambarkan *sequence* ketika *user* melakukan *input* kalimat yang akan diperiksa.



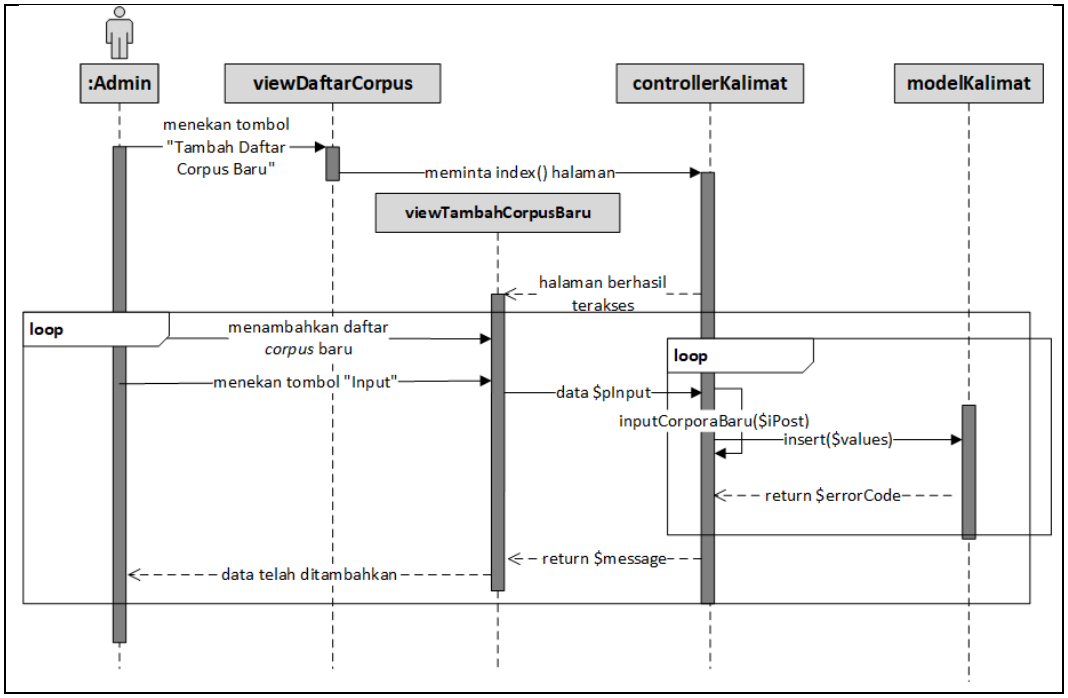
Gambar 3.5 *Sequence* Melakukan *Input* Kalimat yang akan diperiksa – *User*

Proses dimulai ketika *user* mengakses halaman, kemudian halaman akan meminta ke kontroler untuk menampilkan halaman index(). Setelah halaman

ditampilkan, *user* akan memasukkan kalimat yang ingin diperiksa. Setelah *user* memasukkan kalimat, *user* akan menekan tombol “Periksa Kalimat”. Variabel $inputText akan menyimpan hasil masukkan dari *user* untuk dikirim ke kontroler agar dapat diproses. *ControllerAnalisaKalimat* akan menjalankan fungsi removeSymbols($sentence) untuk menghapus tanda baca yang ada pada kalimat. Setelah tanda baca dihapus, kemudian fungsi analyzeGrammarHandler($sentence) akan dijalankan. Fungsi ini akan berguna untuk melakukan analisa tata bahasa pada kalimat dengan memanggil fungsi getValidityArray ($sentence) yang berguna untuk menentukan nilai *validity array* dengan membuat array dengan panjang yang sama dan berisi 0. Setelah array dibuat, fungsi analyze($wordArray, $wordValidityArray) akan dijalankan. Fungsi ini berguna untuk menganalisa tata bahasa pada kalimat. Jika *quadgram* tidak ditemukan, fungsi findSuggestionSingle ($allSuggestion, $wordValidityArray, $quadgram, $i) akan mencari saran kalimat dengan berbagai kemungkinan pada masing-masing *quadgram*, jika terdapat salah satu kata salah. Setelah semua saran ditemukan, maka saran akan dikumpulkan dan ditampilkan melalu fungsi combineSuggestedArray($listOfSuggestedArray, $newSuggested). Seluruh saran akan ditampilkan ke halaman Analisa Grammar Kalimat. *User* dapat melihat seluruh saran perbaikan yang ditampilkan.

B. Sequence Menambah Daftar Corpus Baru – Admin

Gambar 3.6 di bawah ini menggambarkan *sequence* ketika *admin* akan menambah daftar *corpus* baru.



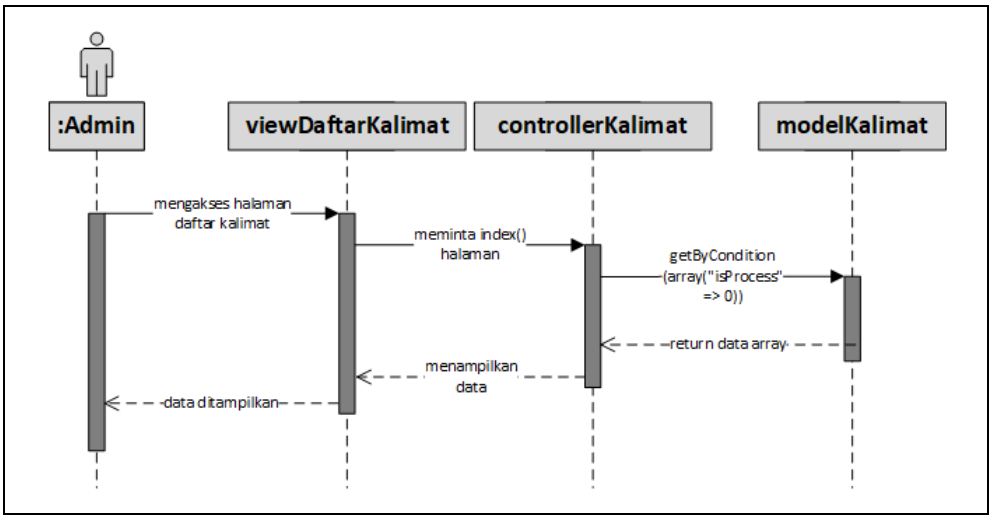
Gambar 3.6 *Sequence* Menambah Daftar *Corpus* Baru - *Admin*

Dimulai ketika *admin* menekan tombol "Tambah Daftar *Corpus* Baru". Sistem akan meminta index() halaman kepada kontroler untuk ditampilkan. Kontroler akan menampilkan halaman untuk menambah *corpus* baru tersebut. Setelah halaman ditampilkan, kemudian *admin* akan memasukkan seluruh kalimat untuk menambah daftar *corpus* baru. Setelah kalimat tersebut selesai dimasukkan, *admin* akan menekan tombol "*Input*". Sistem akan menampung seluruh masukan *admin* kedalam variabel $pInput. Kontroler akan menjalankan fungsi inputCorporaBaru($iPost). Fungsi tersebut berisi potongan kode untuk mempersiapkan data yang telah dimasukkan oleh *admin* untuk proses memasukkan kedalam *database.* Untuk memasukkan data kedalam *database*, digunakan fungsi insert($value). Setelah data dimasukkan kedalam *database,* sistem akan mengembalikan nilai $errorCode. Jika *error code* yang dikeluarkan adalah 0 artinya tidak terdapat kesalahan ketika memasukkan data *corpus* baru ke dalam *database*. Tetapi, jika *error code* tersebut bukan 0 maka sistem akan menampilkan $message

berupa pesan *error* yang terjadi pada saat proses memasukkan data *corpus*.

C. Sequence Melihat Daftar Corpus yang belum diproses - Admin

Gambar 3.7 di bawah ini menggambarkan *sequence* ketika *admin* akan melihat daftar corpus yang belum diproses.



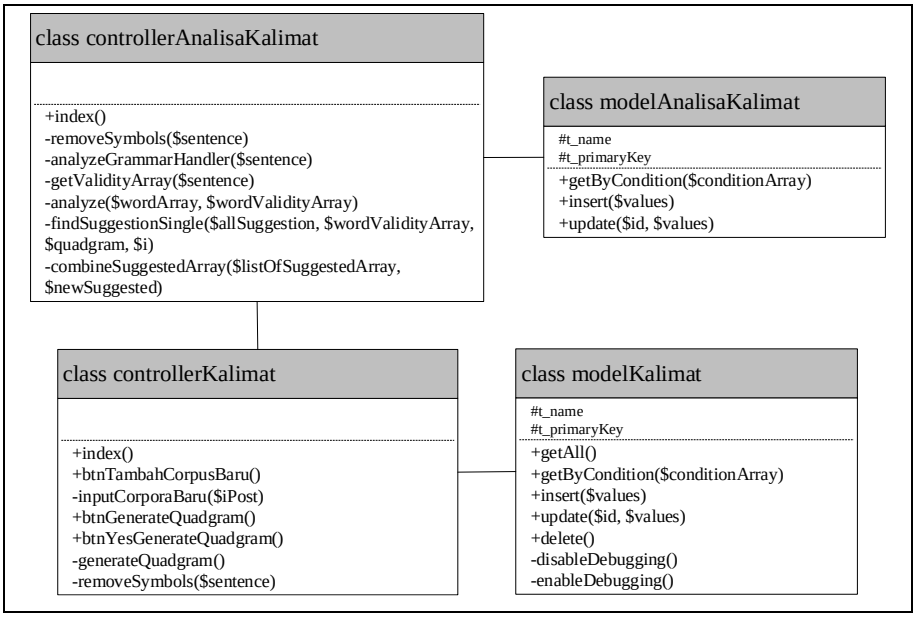
Gambar 3.7 *Sequence* Melihat Daftar *Corpus* yang belum diproses - *Admin*

Dimulai ketika *admin* mengakses halaman, kemudian kontroler akan memproses untuk menampilkan halaman dengan menampilkan data dari *database* yaitu data seluruh *corpus* yang belum diproses. Setelah seluruh data didapatkan, kemudian data tersebut akan ditampilkan pada halaman. *Admin* dapat melihat seberapa banyak data yang belum diproses.

3.2.4 Class Diagram

Class diagram adalah diagram struktur UML yang menunjukkan struktur sistem yang dirancang pada tingkat *class* dan *interface*, menunjukkan fitur, kendala, dan hubungan-asosiasi, generalisasi, dependensi, dll. Jenis *class* diagram yang umum adalah, *domain model* diagram dan *class implement* diagram (uml-diagram.org, 2019). Gambar 3.8 di bawah ini menggambarkan *class* diagram pada

sistem yang dibuat.



Gambar 3.8 *Class* Diagram

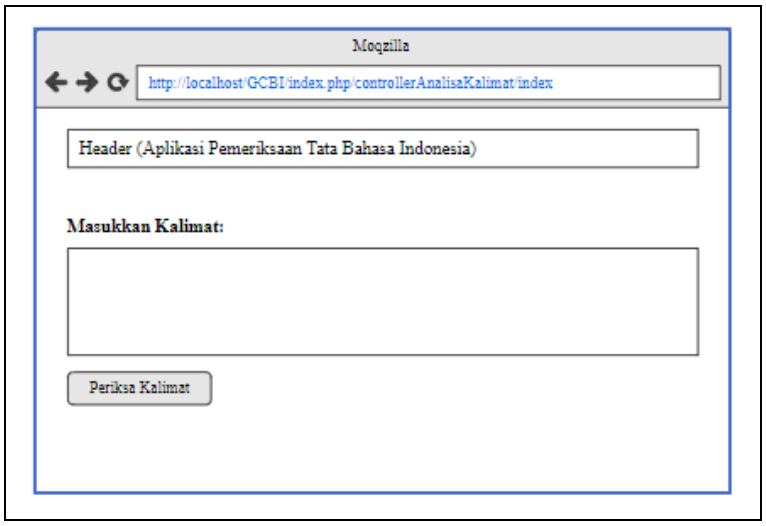
Terdapat 4 kelas diagram. *Class controllerAnalisaGrammar* berisi fungsi-fungsi yang digunakan untuk menganalisa kalimat yang dimasukkan oleh *user*. *Class modelAnalisaGrammar* berisi fungsi-fungsi yang digunakan untuk berkomunikasi antara sistem dengan *database. Class controllerQuadgram* berisi fungsi-fungsi yang digunakan untuk memasukkan dan memproses kalimat untuk menjadi *quadgram*. *ModelQuadgram* berisi fungsi-fungsi yang digunakan untuk berkomunikasi antara sistem dengan *database.*

3.3 Rancangan Antarmuka

Berikut gambar rancangan antarmuka dari aplikasi yang akan dibuat.

3.3.1 Halaman Analisa Grammar Kalimat – User

Gambar 3.9 menggambarkan rancangan antarmuka halaman analisa *grammar* kalimat yang akan digunakan oleh *user* untuk memeriksa kalimat.

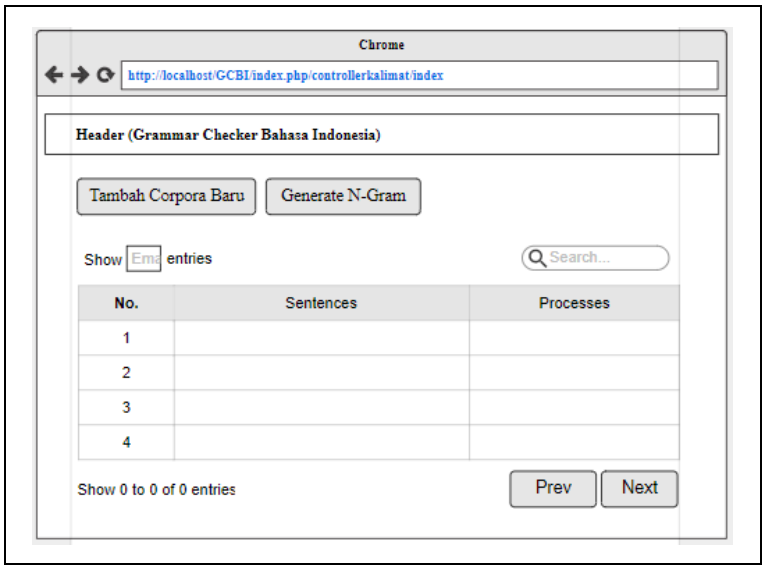


Gambar 3.9 Halaman Analisa *Grammar* Kalimat

Kotak *header* berisi nama dari aplikasi. Kolom *textarea* merupakan kolom yang digunakan untuk memasukkan kalimat yang akan dianalisa. Tombol *input* akan diberi nama "Periksa Kalimat" merupakan suatu tombol yang berfungsi untuk *submit* kalimat yang akan dianalisa oleh aplikasi.

3.3.2 Halaman Daftar Corpus

Gambar 3.10 menggambarkan rancangan antarmuka halaman daftar *corpus* yang belum diproses.

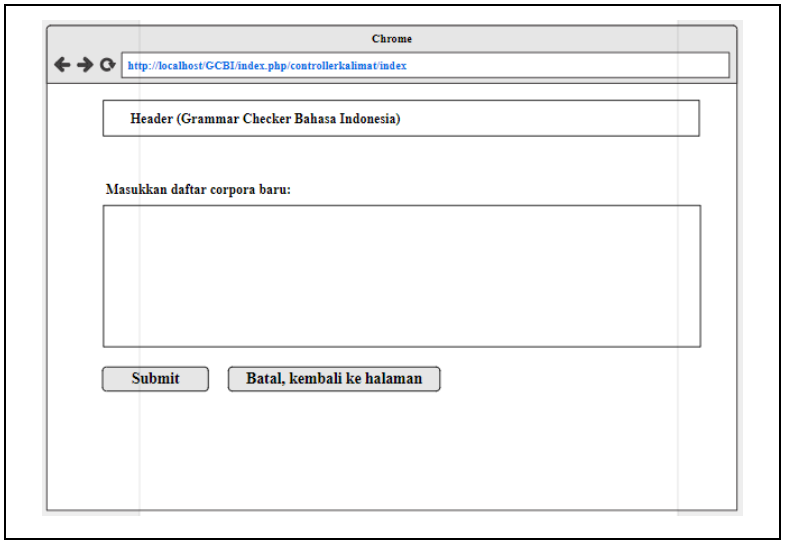


Gambar 3.10 Halaman Daftar *Corpus*

Tombol “Tambah *Corpora* Baru” akan mengarah ke rancangan antar muka halaman tambah *corpora* baru. Tombol “*Generate N-Gram*” merupakan tombol yang akan digunakan untuk memproses *corpus* baru menjadi *quadgram*. Sedangkan, tabel yang berada di bawah kedua tombol adalah tabel yang digunakan untuk melihat seluruh *corpora* yang belum diproses menjadi *quadgram*.

3.3.3 Halaman Tambah Corpus Baru

Gambar 3.11 menggambarkan rancangan antarmuka halaman tambah *corpus* baru.

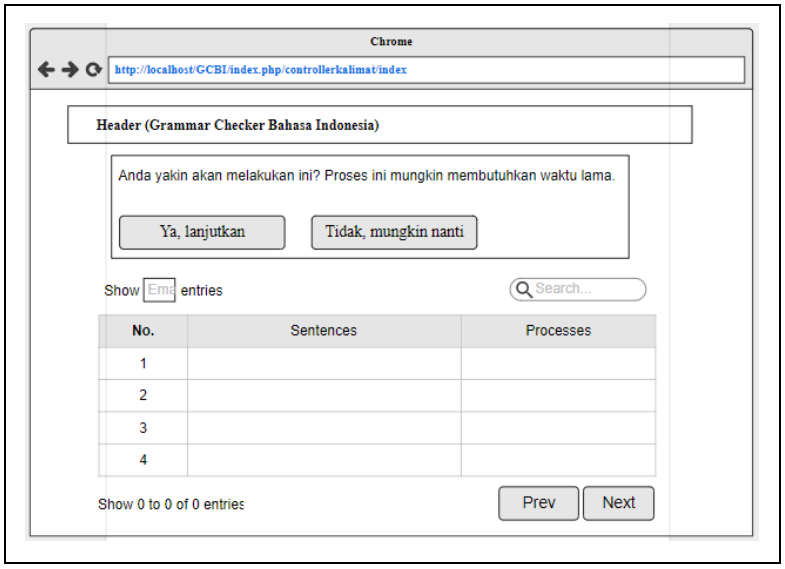


Gambar 3.11 Halaman Tambah *Corpus* Baru

Gambar 3.11 menggambarkan halaman tambah *corpus* yang dilakukan oleh admin. *Textbox* digunakan untuk memasukkan seluruh kalimat yang dimasukkan oleh admin. Terdapat dua tombol di bawah *textbox* tersebut. Tombol *submit* akan diberi nama “*Input*” yang digunakan untuk proses memasukkan kalimat ke dalam *database*. Sedangkan tombol yang lainnya digunakan untuk kembali ke halaman daftar *corpus* yang belum diproses.

3.3.4 Halaman Generate N-gram

Gambar 3.12 menggambarkan rancangan antarmuka halaman *generate n-gram* – admin.



Gambar 3.12 Halaman *Generate N-Gram*

Gambar 3.12 menggambarkan rancangan antar muka halaman *generate ngram*. Terdapat sebuah label dan dua tombol yang memberikan pilihan kepada *admin* untuk melanjutkan atau membatalkan perintah.