RANCANGAN APLIKASI LATIHAN BELAJAR TENSES DENGAN METODE *OBJECT ORIENTED DESIGN*

Qoriani Widayati, Irman Effendy

[1)] Sistem Informasi Akuntansi, Ilmu Komputer
Jl. A. Yani No. 3 Plaju Palembang
email : qoriani.ubd@gmail.com
[2)] Sistem Informasi, Ilmu Komputer
Jl. A. Yani No. 3 Plaju Palembang
email : irman_effendy@mail.binadarma.ac.id

ABSTRACT

Bahasa Inggris merupakan bahasa yang digunakan secara internasional. Untuk dapat berkompetensi di dunia internasional, kita diharuskan untuk menggunakan bahasa tersebut. Tidak semua individu dapat menggunakan *grammar* yang baik. Terutama untuk pelajar dan mahasiswa. Di kala membuat tugas dalah bahasa Inggris atau jika ingin melanjutkan ke sekolah atau perguruan tinggi di luar negeri, *grammar* menjadi sesuatu yang penting dipelajari. Di era perkembangan teknologi yang semakin pesat, diperlukan sebuah perangkat lunak sebagai media yang membantu mempermudah mempelajari tenses tersebut. Sebagai media belajar itulah, maka akan dibuatkan aplikasi sebagai latihan belajar *Tenses* dalam bahasa Inggris menggunakan sistem operasi *Android.* Metode pengembangan perangkat lunak yang digunakan adalah *Object Oriented Design.*

Kata Kunci: *Grammar, Tenses, Object Oriented Design*

1. Pendahuluan

Bahasa Inggris merupakan bahasa internasional. Belajar bahasa Inggris sudah bisa lihat, baik di Sekolah Dasar (SD) sampai dengan Perguruan Tinggi. Untuk dapat bersaing baik itu di dalam negeri maupun di luar negeri, bahasa Inggris sangat berperan penting dalam hal berkomunikasi. Individu yang bisa menggunakan bahasa Inggris dapat menjadi nilai tambah bagi organisasi untuk merekrut pegawainya. Di sisi pendidikan, siswa atau mahasiswa yang mempunyai kemampuan berbahasa Inggris yang baik, dapat memperoleh beasiswa, baik dari dalam maupun dari luar negeri. Ada banyak beasiswa, jika kita bisa memiliki kemampuan tersebut.

Salah satu materi dalam pelajaran Bahasa Inggris adalah *grammar*. Penguasaan *grammar* yang baik, dapat digunakan tidak hanya untuk berkomunikasi dalam bahasa Inggris, akan tetapi sebagai modal untuk menulis dalam penulisan, baik itu secara non formal dengan berkirim surat atau pesan, maupun dalam penulisan ilmiah. Salah satu materi dalam penguasaan *grammar* adalah materi tentang *Tenses.* Materi *Tenses* merupakan pola kalimat yang disusun berdasarkan kata-kata yang akan dibuat. *Tenses* merupakan perubahan kata kerja yang dipengaruhi oleh waktu dan sifat kejadian [1]. Semua kalimat dalam bahasa Inggris tidak terlepas dari tenses karena semua kalimat pasti ada hubungannya dengan waktu dan sifat kejadiannya. Sedangkan menurut [2] *Tenses* merupakan perubahan kata kerja dalam kalimat Bahasa Inggris yang menyatakan perbedaan waktu dan sifat kegiatan atau kejadian.

Rancangan aplikasi yang dihasilkan nantinya berbasis *Android.* Jadi hanya HP yang menggunakan sistem operasi *android* saja yang dapat menggunakannya. Materi yang akan dimasukkan tidak hanya materi pelajaran saja, akan tetapi juga disertai dengan soal-soal latihan bahasa Inggris menggunakan *Tenses.* Metode pengembangan perangkat lunak yang digunakan adalah *Object Oriented Design.* Perancangan berorientasi objek (OOD-*Object Oriented Design*) merupakan disiplin yang lebih kompleks dibanding perancangan konvensional. OOD mengidentifikasikan dan mendefinisikan kelas-kelas dan objek-objek tambahan yang merefleksikan implementasi. Sedangkan OOA mengidentifikasi dan mendefinisikan kelas-kelas dan objek-objek yang secara langsung merefleksikan domain masalah dan tanggungjawab sistem di dalamnya. Pada saat ini penulis hanya membatasi pada perancangan aplikasi

saja. Pada penelitian selanjutya diharapkan rancangan ini dapat di implementasikan menjadi sebuah aplikasi yang dapat dimanfaatkan.

2. Perancangan Berorientasi Objek (*OOD-Object Oriented Design*)

Perancangan berorientasi objek (*Object Oriented Design*) merupakan disiplin yang lebih kompleks dibanding perancangan konvensional. OOD mengidentifikasikan dan mendefinisikan kelas-kelas dan objek-objek tambahan yang merefleksikan implementasi. Sedangkan OOA mengidentifikasi dan mendefinisikan kelas-kelas dan objek-objek yang secara langsung merefleksikan domain masalah dan tanggungjawab sistem di dalamnya.

Ada beberapa konsep perancangan yang penting [3], antara lain: Abstraksi, Modularitas, Arsitektur perangkat lunak, Hirarki, dan Pengkapsulan dan penyembunyian informasi

Di dalam OOD terdapat beberapa metode yang popular [3]antara lain: Pertama, Metode Booch yang terdiri dari

1) Pengembangan makro. Aktivitas perencanaan arsitektur yang megnelompokkan objek-objek serupa di partisi-partisi arsitektur yang terpisah, membuat lapisan-lapisan objek berdasar level-level abstraksi, mengidentifikasi skenario-skenario yang relevan, menciptakan prototype rancangan, dan memvalidasi prototype berdasarkan skenario penggunaan.

2) Pengembangan mikro. Mendefinisikan sekumpulan ”aturan” yang menuntun penggunaan operasi dan atribut dan kebijakan spesifik domain untuk menajemen memoeri, penanganan kesalahan, dan fungsi-fungsi infrastruktur yang lain, pengembangan skenario yang mendeskripsikan semantiks dari aturan dan kebijakan, menciptakan prototype untuk masing-masing kebijakan, meperbaiki prototype, dan melakukan review atas kebijakan sehingga memperluas visi arsitektur.

Kedua adalah Metode Rumbaugh yang mempunyai beberapa langkah yaitu Perancangan sistem dan Perancangan objek.

Ketiga adalah Metode Jacobson terdiri dari: 1) Adaptasi terhadap model analisis agar dapat berada di lingkungan nyata. 2) Perancangan kelas-kelas objek utama yang disebut blok, terbagi menjadi blok antarmuka, blok entitas, dan blok kendali. 3) Pendefinisian komunikasi di antara blok-blok itu selama eksekusi. 4) Pengorganisasian blok-blok menjadi subsistem-subsistem.

Keempat, Metode Coad-Yourdan yaitu Problem domain component, Human interaction component, Data management component dan Task management component.

Kelima, Metode Wirfs-Brock mempunyai beberapa tahapan yaitu Pendefinisian protokol untuk masing-masing kelas dengan memperbaiki kondisi dari masing-masing kelas, masing-masing operasi dan protokol (antarmuka) dirancang serinci sampai dapat menjadi tuntunan bagi implementasi, pendefinisian spesifikasi untuk masing-masing kelas meliputi pendefinisian tanggung jawab private dan rincian operasi dan pengelompokkan subsistem-subsistem.

Terakhir adalah Generik. Setelah melihat beberapa metode dari perancangan berorientasi objek maka dapat disimpulkan: 1) Perancangan sistem meliputi arsitektur sistem dan pendeskripsian subsistem-subsistem dan alokasinya di pemroses dan proses. 2) Pemilihan strategi perancangan untuk implementasi manajemen data, dukungan antarmuka dan manajemen proses/memori, penanganan kesalahan. 3) Perancangan mekanisme kendali yang cocok untuk sistem. 4) Perancangan rinci kelas objek dalam hal struktur data dan algoritmanya. 5) Perancangan pertukaran pesan menggunakan kolaborasi antar objek dan hubungan objek. 6) Penciptaan model pertukaran pesan dan 7) Melakukan review atas model rancangan dan melakukan iterasi bila perlu untuk perbaikan model rancangan yang sebelumnya.

2.1 Unified Modelling Language (UML)

UML merupakan kesatuan dari bahasa pemodelan yang dikembangkan oleh Booch, Object Modelling Technique (OMT) dari Raumbaugh dan Object Oriented Software Engineering (OOSE) dari Jacobson. Dengan metode UML yang merupakan gabungan dari beberapa metode dengan membuang beberapa elemen dari metode lain yang lebih efektif dan elemen-elemen baru yang belum ada pada metode terdahulu sehingga UML lebih ekspresif dan seragam daripada metode lainnya.

UML adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek [4]. UML dapat menyediakan bahasa pemodelan yang mudah dimengerti oleh pengembang dan dapat dikomunikasikan dengan pemakai.

Metode perancangan yang digunakan pada penelitian ini adalah metode berorientasi objek menggunakan UML. Adapun diagram-diagram yang dibuat pada UML menurut [3]terdiri dari:

a. Diagram Perilaku

   Diagram ini untuk memvisualisasi, menspesifikasikan, membangun dan mendokumentasikan aspek dinamis dari sistem. Diagram perilaku di UML terdiri dari :

   1) Diagram use-case (*Use Case Diagram*)

   Diagram ini menunjukkan sekumpulan kasus fungsional dan aktor (jenis kelas khusus) dan keterhubungannya.

   2) Diagram sekuen (*Sequence Diagram*)

Diagram ini menunjukkan interaksi yang terjadi antar objek. Diagram ini merupakan pandangan dinamis terhadap sistem. Diagram ini menekannkan pada basis keberurutan waktu dari esan-pesan yang terjadi.

3) Diagram kolaborasi (*Collaboration Diagram*)

Diagram ini menekankan pada organisasi struktur dari objek-objek
yang mengirim dan menerima pesan.

4) Diagram statechart (*Statechart Diagram*)

Diagram ini adalah state-machine diagram, berisi state, transisi, kejadian dan aktivitas. Diagram ini penting dalam memodelkan perilaku antarmuka, kelas, kolaborasi dan menekankan pada urutan kejadian.

5) Diagram aktivitas (*Activity Diagram*)

Diagram ini menunjukkan aliran aktivitas di sistem. Diagram ini adalah pandangan dinamis terhadap sistem. Diagram ini penting untuk memodelkan fungsi sistem dan menekankan pada aliran kendali di antara objek-objek.

b. Diagram Struktur

Diagram ini untuk memvisualisasi, menspesifikasikan, membangun dan mendokumentasikan aspek statik dari sistem. Diagram struktur di UML terdiri dari :

1) Diagram kelas (*Class Diagram*)

Diagram ini menunjukkan sekumpulan kelas, interface dan kolaborasi dan keterhubungannya.

2) Diagram objek (*Object Diagram*)

Diagram ini menunjukkan sekumpulan objek dan keterhubungannya. Diagram ini menunjukkan potongan statik dari instan-instan yang ada di diagram kelas. Diagram ini untuk memperlihatkan satu prototipe atau kasus tertentu yang mungkin terjadi.

3) Diagram komponen (*Component Diagram*)

Diagram ini menunjukkan oganisasi dan kebergantungan di antara sekumpulan komponen. Diagram ini merupakan pandangan statik terhadap implementasi sistem.

4) Diagram deployment (*Deployment Diagram*)

Diagram ini menunjukkan konfigurasi pemrosesan saat jalan dan
komponen-komponen yang terdapat di dalamnya. Diagram ini merupakan pandangan statik dari arsitektur.

3. Hasil Percobaan

3.1 Analisis Skenario

1. Skenario Entry Rumus Tenses

   Skenario ini merupakan skenario di mana Admin (sebagai orang yang memasukkan data) dapat memasukkan nama-nama *Tenses* ke aplikasi

Tabel 1. Skenario Entry Nama *Tenses*

Identifikasi	
Nomor	1
Nama	Entry Nama *Tenses*
Tujuan	Memasukkan data nama-nama*Tenses*
Deskripsi	Halaman ini adalah halaman untuk memasukkan, menghapus dan mengubah data nama *Tenses.*
Tipe	-
Aktor	Admin
Skenario Utama	
Kondisi awal	Program dengan tampilan halaman kategori.
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu nama *Tenses.* 3. Memasukkan nama *Tenses.* 4. Meng-klik tombol "Simpan". 6. Jika ingin menghapus data nama *Tenses*, maka pilih icon Hapus. 8. Jika ingin mengubah data nama *Tenses*, maka pilih icon Ubah.	2. Menampilkan menu nama *Tenses.* 5. Menyimpan data nama *Tenses.* 7. Menghapus data nama *Tenses.* 9. Mengubah data nama *Tenses.*
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
-	-
Kondisi akhir	Sistem akan menampilkan data nama-nama Tenses yang sudah dimasukkan, disimpan, dihapus atau diubah.

2. Skenario Entry data Rumus *Tenses*

   Skenario ini merupakan skenario untuk memasukkan data rumus *Tenses* pada aplikasi *mobile*

Tabel 2. Skenario Entry data rumus tenses

Identifikasi	
Nomor	2
Nama	Entry rumus *tenses*
Tujuan	Memasukkan semua rumus *tenses*.
Deskripsi	Halaman ini adalah halaman untuk memasukkan data rumus tenses.
Tipe	-
Aktor	Admin
Skenario Utama	
Kondisi awal	Program dengan tampilan halaman

	entry rumus *tenses*.
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu entry rumus tenses 3. Memilih id tenses 5. Memasukkan rumus tenses dengan memilih file yang telsh disimpan. 7. Meng-klik tombol "simpan. 9. Jika ingin menghapus data nama *Tenses*, maka pilih icon Hapus. 11. Jika ingin mengubah data nama *Tenses*, maka pilih icon Ubah.	2. Menampilkan menu entry rumus tenses. 4. Menampilkan nama tenses. 6. Mencari file di komputer. 8. Menyimpan rumus tenses. 10. Menghapus data nama *Tenses.* 12. Mengubah data nama *Tenses.*
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
-	-
Kondisi akhir	Sistem akan menampilkan data-data rumus *tenses.*

3. Skenario Entry data Soal *Tenses.*
   Skenario ini merupakan skenario untuk memasukkan soal-soal yang berasal dari rumus *Tenses*.

Tabel 3. Skenario data soal tenses

Identifikasi	
Nomor	3
Nama	Entry data Soal *Tenses*
Tujuan	Memasukkan soal-soal *Tenses.*
Deskripsi	Halaman ini adalah halaman untuk memasukkan soal-soal *Tenses* yang berasal daru rumus *Tenses.*
Tipe	-
Aktor	Admin
Skenario Utama	
Kondisi awal	Program dengan tampilan halaman soal *Tenses.*
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu Soal Tenses. 3. Memasukkan soal, pilihan a, pilihan b, pilihan c, pilihan d, pilihan e, dan kunci jawaban. 4. Meng-klik tombol "simpan. 6. Jika ingin menghapus soal *Tenses*, maka pilih icon Hapus. 8. Jika ingin mengubah soal *Tenses*, maka pilih icon Ubah.	2. Menampilkan menu soal tenses. 5. Menyimpan soal tenses. 7. Menghapus soal *Tenses.* 9. Mengubah soal *Tenses.*
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
-	-
Kondisi akhir	Sistem akan menampilkan soal-soal beserta pilihannya.

4. skenario *View* Rumus *Tenses.*
   Skenario ini merupakan skenario bagi pengunjung untuk melihat rumus tenses

Tabel 4. Skenario View Rumus Tense

Identifikasi	
Nomor	4
Nama	View Rumus Tenses
Tujuan	Melihat rumus-rumus tenses.
Deskripsi	Halaman ini merupakan halaman untuk menampilkan rumus-rumus tenses.
Tipe	-
Aktor	Pengunjung
Skenario Utama	
Kondisi awal	Program dengan tampilan menu Rumus Tenses.
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu "Rumus Tenses". 3. Memilih Tenses. 5. Klik home	2. Menampilkan menu "Rumus Tenses". 4. Menampilkan materi tentang tenses. 6. Kembali ke menu utama
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
-	-
Kondisi akhir	Sistem akan kembali ke menu utama.

5. Skenario Latihan Soal.
   Skenario ini merupakan skenario bagi pengunjung untuk melakukan latihan dengan menjawab soal-soal yang berhubungan dengan materi tenses

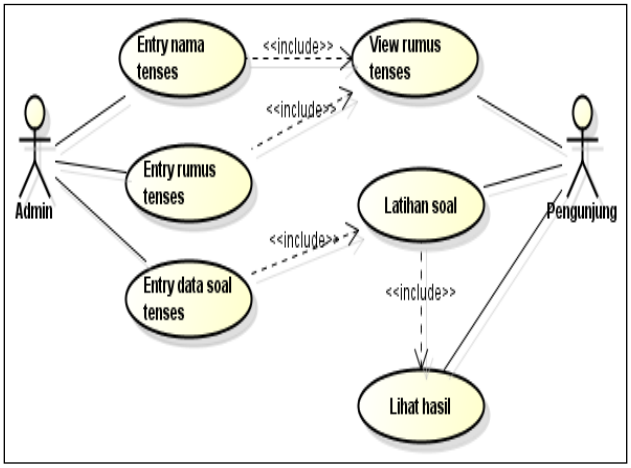
Tabel 5. Skenario Latihan Soal

Identifikasi	
Nomor	5
Nama	Latihan Soal
Tujuan	Menjawab pertanyaan dengan tipe pilihan berganda.
Deskripsi	Halaman ini merupakan halaman bagi pengunjung untuk menjawab pertanyaan-pertanyaan dalam bentuk plihan berganda.
Tipe	-
Aktor	Pengunjung

Skenario Utama	
Kondisi awal	Program dengan tampilan menu latihan soal.
Aksi Aktor	**Reaksi Sistem**
1. Memilih menu “Latihan Soal”. 3. Meng-klik jawaban yang dipilih. 4. Meng-klik tombol “Lanjut” 6. Klik tombol “selesai”	2. Menampilkan menu “Latihan Soal”. 5. Menampilkan soal berikutnya. 7. Menampilkan hasil benar dan salah.
Skenario Alternatif – Autentikasi Gagal	
Aksi Aktor	**Reaksi Sistem**
-	-
Kondisi akhir	Sistem akan kembali ke menu utama.

3.2 Use Case Diagram

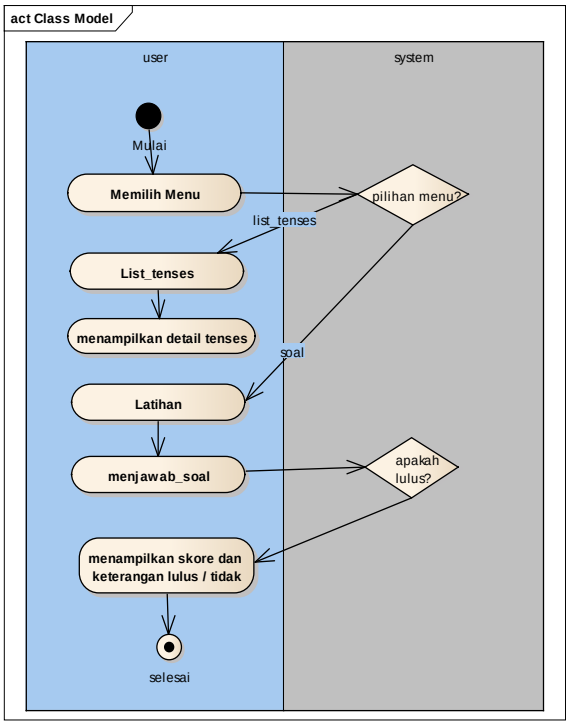
Use case diagram merupakan tampilan antarmuka antara pengguna dengan aplikasi yang dibuat. *Use case* berasal dari skenario yang telah dibuat sebelumnya. Aktor yang terlibat ada dua yaitu Admin dan Pengunjung. Admin merupakan orang yang akan memasukkan data master ke aplikasi



Gambar 1. Use Case Diagram

3.3 Activity Diagram

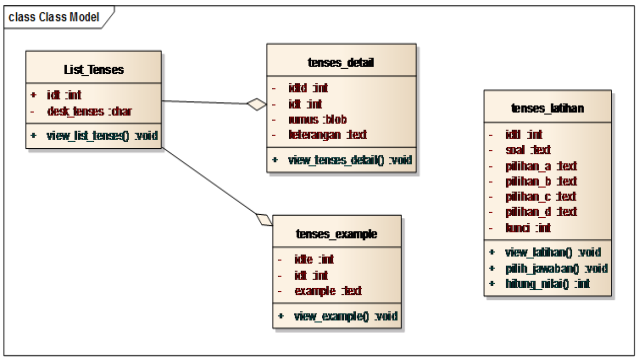
Diagram ini menunjukkan aliran aktivitas di sistem. Diagram ini adalah pandangan dinamis terhadap sistem.



Gambar 2. Activity Diagram

3.4 Class Diagram

Diagram ini menunjukkan sekumpulan kelas, interface dan kolaborasi dan keterhubungannya



Gambar 3. Class Diagram

3.5 Desain Antarmuka Pemakai

Desain antar muka pemakai merupakan desain yang menunjukkan antar muka aplikasi dengan pengguna. desain antarmuka pemakai berasal dari skenario.

1. Desain Antarmuka Menu Utama

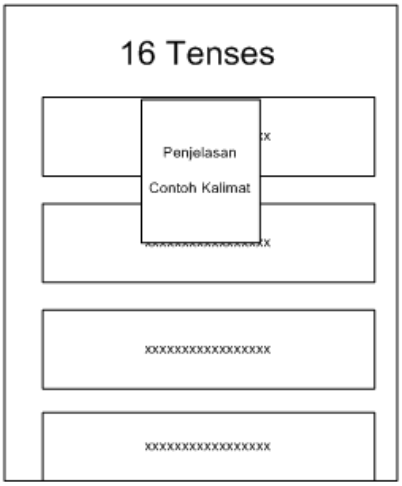
Desain yang pertama adalah desain antarmuka menu utama yang digunakan untuk masuk ke dalam aplikasi. Desain ini merupakan desain

yang nantinya akan dipakai oleh pengunjung sebagai tampilan awal dari aplikasi

Gambar 4. Desain antarmuka menu utama

2. Desain Antarmuka Tenses

Desain antarmuka kedua adalah desain tenses. Menu ini nantinya digunakan pengunjung untuk melihat 16 tenses dalam bahasa Inggris



Gambar 5. Desain Antarmuka Tenses

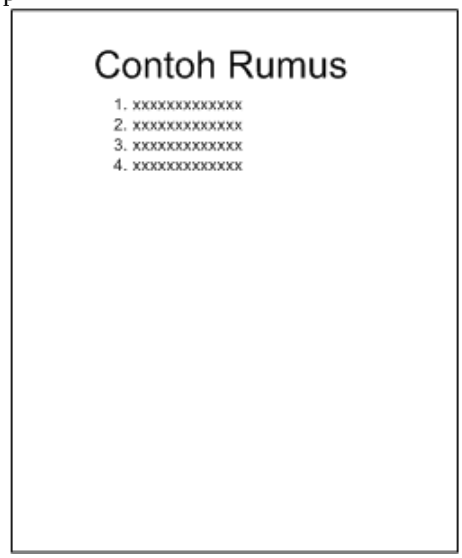
3. Desain Antarmuka Judul Rumus

Desain antarmuka ketiga adalah desain antarmuka judul Rumus. Desain ini digunakan untuk menampilkan judul rumus

Gambar 6. Desain Antarmuka Judul Rumus

4. Desain Antarmuka Contoh Rumus

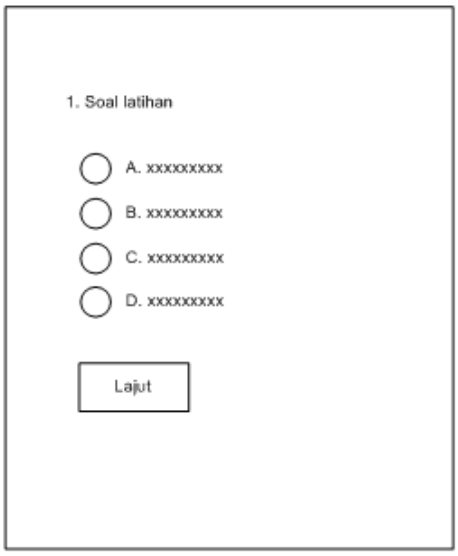
Desain antarmuka contoh rumus merupakan desain bagi pengunjung untuk menampilkan contoh-contoh yang berdasarkan tenses yang dipilih



Gambar 7. Desain Antarmuka Contoh Rumus

5. Desain Antarmuka Soal Latihan

Desain antarmuka soal altihan merupakan desain yang nantinya akan digunakan pengunjung untuk melakukan latihan soal-soal tenses dengan memilih jawaban dalam bentuk pilihan berganda.



Gambar 8. Desain Antarmuka Soal Latihan

6. Desain Antarmuka Tampilan Hasil

Desain antarmuka keempat tampilan hasil merupakan halaman untuk menampilkan hasil setelah Pengunjung menjawab latihan soal.

Gambar 9. Desain Antarmuka Tampilan Hasil

4. Kesimpulan

Kesimpulan yang didapatkan dari penelitian ini yaitu:

1. Rancangan aplikasi latihan tenses menggunakan metode OOD bertujuan sebagai media alternatif dalam belajar tenses
2. Rancangan aplikasi ini selanjutnya dapat dikembangkan oleh peneliti lain menjadi aplikasi yang lebih baik, baik dalam tampilan dan kegunaan.

REFERENSI

[1] Hari, Erwin, Kurniawan. 2010. *Basic English Grammar*. Online. (Diakses http://kartika.staff.stainsalatiga.ac.id/wp-content/uploads/sites/90/2013/10/ english-modul-for-sma.pdf, tanggal 25 April 2014).

[2] Amin, Bunyamin, Kajo. 2014. *Materi I Tenses*. Online. (Diakses http://kartika.staff.stainsalatiga.ac.id/wp-content/.../english-modul-for-sma.pdf%E2%80%8E, tanggal 25 April 2014)

[3] Haryanto, Bambang., 2004. *Rekayasa Sistem Berorientasi Objek*. Bandung: Informatika

[4] Munawar. 2005., *Pemodelan Visual*. Jakarta: Graha Ilmu.