PERANCANGAN *GAME VISUAL NOVEL* MENGGUNAKAN REN'PY DENGAN KUSTOMISASI *AVATAR*

TUGAS AKHIR
Diajukan untuk Memenuhi Salah Satu Syarat Kelulusan
Program Pendidikan Sarjana

Oleh :
Maria Oktaviana
2011130046

JURUSAN TEKNIK INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER LIKMI
BANDUNG
2016

PERANCANGAN *GAME VISUAL NOVEL* MENGGUNAKAN REN'PY DENGAN KUSTOMISASI *AVATAR*

Oleh:
Maria Oktaviana
2011130046

Bandung, 26 Januari 2016
Menyetujui,

Dhanny Setiawan, S.T., M.T.	Dhanny Setiawan, S.T., M.T.
Pembimbing	Ketua Jurusan

JURUSAN TEKNIK INFORMATIKA
SEKOLAH TINGGI MANAJEMEN INFORMATIKA & KOMPUTER - LIKMI
BANDUNG
2016

ABSTRAK

Perkembangan *game* saat ini merupakan hal yang tidak asing lagi dalam kehidupan. *Game* juga merupakan salah satu sarana hiburan yang banyak dilakukan diwaktu senggang. Novel juga merupakan salah satu bentuk fisik yang sering dimanfaatkan manusia untuk menghilangkan rasa jenuh. Novel dapat divisualisasikan ke dalam komputer disebut juga *visual novel*. Dengan adanya *visual novel* pembaca akan lebih mudah memvisualisasikan cerita novel yang diangkat, karena dapat membaca sebuah novel namun dengan ilustrasi gambar sambil memainkannya, sehingga pembaca akan mendapat pengalaman membaca yang lebih menyenangkan. Pembaca juga dapat menentukan akhir cerita sesuai dengan keinginannya melalui jawaban dari pertanyaan yang ada pada cerita. Berdasarkan hal tersebut, maka penulis membuat *game visual novel After Graduate.*

Metode yang digunakan dalam pengembangan *game* ini adalah studi pustaka, perancangan, analisis juga proses pembangunan. Pembuatan game dilakukan dengan menggunakan *software* Ren'Py yang merupakan *software open* source dan juga gratis sehingga bebas digunakan. Pengkodean dari Ren'Py menggunakan bahasa pemrograman Python. Bahasa pemrograman tersebut digunakan untuk mengelola alur cerita, karakter dan avatar pada *game*. Metode pengembangan yang digunakan dalam perancangan *game* ini adalah metode prototipe yang didasari pemrograman berorientasi objek dimana proses pembuatan dan pengujiannya akan terus dilakukan hingga *game* yang diinginkan dapat dimainkan.

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Allah Tritunggal Maha Kudus juga Bunda Maria, atas segala berkat, anugerah juga kasih-Nya yang diberikan sehingga dapat diselesaikan penyusunan Tugas Akhir ini, untuk memenuhi syarat kelulusan program pendidikan sarjana jurusan Teknik Informatika di STMIK LIKMI Bandung.

Penulis telah berusaha untuk melakukan yang terbaik dalam penyusunan Tugas Akhir ini, namun penulis menyadari bahwa masih terdapat banyak kekurangan dalam penyusunan Tugas Akhir ini. Untuk itu penulis memohon maaf atas segala kekurangan yang ada dalam Tugas Akhir ini. Kiranya di balik semua kekurangan yang ada, Tugas Akhir ini dapat memberikan manfaat kepada pembaca.

Dalam penulisan dan pengerjaan Tugas Akhir ini, banyak pihak yang telah memberi dukungan, bantuan, serta semangat kepada penulis. Oleh karena itu, penulis mengucapkan banyak terima kasih kepada:

1. Bapak Dhanny Setiawan, S.T., M.T. selaku dosen pembimbing dan ketua jurusan Teknik Informatika yang tidak pernah lelah dalam membimbing dan memberi semangat kepada penulis, serta banyak membantu proses akademis dan memberi banyak ilmu juga saran kepada penulis selama menempuh kuliah di STMIK LIKMI.
2. Dosen-dosen STMIK LIKMI yang telah memberikan ilmu dan pengajaran kepada penulis, juga staff-staff kampus yang telah memberi banyak bantuan kepada penulis selama menempuh pendidikan di STMIK LIKMI.
3. Orang tua tercinta, mama Caecilia Elly dan ayah Abdul Kharis, juga koko Andre dan dede Rafa atas doa, motivasi, semangat dan segala dukungannya dalam bentuk apapun yang membuat penulis bisa menyelesaikan penyusunan tugas akhir ini.
4. Stevanus Luvin Tiusana Gutomo yang selalu menemani, mendampingi, mendukung dan memberi semangat kepada penulis dalam penyusunan Tugas Akhir.
5. Sahabat-sahabat terbaik Sandy Cahya, S.T., Evi Hariyanti, Sianly Sugiani, S.T., Michael Alvin, S.T., Asbian Wangsadijaya, cici Kezia Stefani, S.T., M.Kom., cici

Jenisa Felisa, S.T., M.Kom., dan sahabat-sahabat yang selalu menyemangati penulis dan memberikan dukungan tanpa henti.

6. Guru-guru SDS Makedonia, terutama Ibu Christina Marleen Tanamal, S.S., Ibu Emmawati H. Lenggu, S.Si., M.Div., juga Kepala Sekolah yang sudah memberi semangat, masukan dan mendukung penulis dalam penyusunan Tugas akhir ini.
7. Anggota Capella Cantorum Choir juga warga lingkungan yang sudah memberi dukungan, mengingatkan, dan mendoakan penulis dalam penyusunan Tugas Akhir ini.
8. Teman-teman jurusan Teknik Informatika angkatan 2011 yang selama kurang lebih empat tahun ini berjuang bersama-sama dalam mencapai kelulusan.
9. Teman-teman mahasiswa/i STMIK LIKMI baik yang berbeda jurusan maupun angkatan yang selalu memberikan dukungan dan penghiburan.
10. Pihak lain yang membantu yang tidak dapat disebutkan namanya satu persatu.

Penulis selalu membuka diri untuk setiap saran maupun kritik membangun yang ditujukan baik penulis maupun isi dari Tugas Akhir ini. Sehingga di lain kesempatan penulis dapat melakukan yang lebih baik lagi.

Bandung, Januari 2016

Penulis

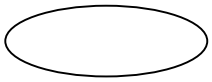
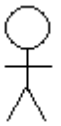
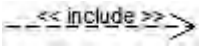
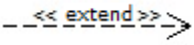
DAFTAR ISI

DAFTAR GAMBAR

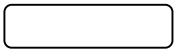
DAFTAR TABEL

DAFTAR SIMBOL

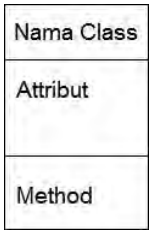
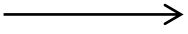
Use case diagram

Nama Simbol	Simbol	Keterangan
Use Case		Sebuah kebiasaan atau fungsi utama (*key behaviour*) yang dilakukan oleh sistem perangkat lunak
Asosiasi		Hubungan antara aktor dengan *use case* yang menunjukan adanya interaksi aktor dengan sistem atau campur tangan aktor dalam suatu *use case*
Actor		Entitas eksternal yang berhubungan dengan sistem
Include		Menspesifikasikan bahwa *use case* sumber secara eksplisit.
Extend		Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan

Activity diagram

Nama Simbol	Simbol	Keterangan
Kondisi awal		Menunjukkan kondisi awal kegiatan sistem
Kondisi akhir		Menunjukkan kondisi berakhirnya kegiatan sistem
Action State		Kegiatan yang sistem lakukan
Desicion		Percabangan dimana sistem harus melakukan salah satu cabang keputusan
Transition		Menunjukkan arah sebuah *action state* ke *action state* selanjutnya

Class diagram

Nama Simbol	Simbol	Keterangan
Class		Menunjukan nama kelas yang terdapat pada perangkat lunak beserta dengan attribute dan method yang ada di dalamnya.
Directed Association		Relasi antar *class* dengan makna *class* yang satu digunakan oleh *class* yang lain.

BAB I

PENDAHULUAN

1.1 Latar Belakang

Teknologi Informasi saat ini merupakan hal yang tidak asing lagi bagi kehidupan manusia. Terutama dalam kehidupan sehari-hari, manusia banyak memanfaatkan teknologi. Pada dasarnya teknologi memberikan pengaruh yang positif. Manusia membutuhkan teknologi dalam memenuhi kebutuhannya setiap hari, ataupun untuk menghilangkan kejenuhan yang dialami.

Salah satu teknologi yang banyak dimanfaatkan adalah teknologi komputer. Teknologi komputer terus berkembang dengan pesat, dan salah satu bagiannya adalah teknologi multimedia. Teknologi multimedia yang juga ikut berkembang telah membuat penyampaian suatu informasi dapat disampaikan dengan lebih interaktif dan efektif karena dapat menjangkau indera manusia. Perkembangan teknologi dapat mensimulasikan perangkat-perangkat diluar komputer, dan disimulasikan ke dalam komputer dalam bentuk virtual.

Game atau permainan merupakan salah satu simulasi dari bentuk-bentuk nyata kehidupan manusia. Perkembangan *game* saat ini merupakan hal yang tidak asing lagi dalam kehidupan. *Game* juga merupakan salah satu sarana hiburan yang banyak dilakukan diwaktu senggang. *Game* juga banyak diimplementasikan ke dalam komputer. Banyak juga *game* yang dapat dimainkan di dalam komputer, dari *game* sederhana sampai *game* yang cukup rumit dalam pembuatannya.

Novel merupakan salah satu bentuk fisik yang sering dimanfaatkan manusia untuk menghilangkan rasa jenuh. Cerita pada novel ada berbagai macam, antara lain cerita anak-anak, kisah remaja, hingga kisah yang menginspirasi pun ada. Oleh karena itu novel dapat dibaca oleh siapa saja, baik anak-anak maupun dewasa.

Novel juga dapat divisualisasikan ke dalam komputer. *Game* dan novel dapat digabungkan ke dalam sebuah game statik, permainan ini sering disebut dengan *Visual Novel*. Dengan adanya *visual novel* pembaca akan lebih mudah memvisualisasikan cerita

novel yang diangkat, karena dapat membaca sebuah novel namun dengan ilustrasi gambar sambil memainkannya, sehingga pembaca akan mendapat pengalaman membaca yang lebih menyenangkan. Pembaca juga dapat menentukan akhir cerita sesuai dengan keinginannya melalui jawaban dari pertanyaan yang ada pada cerita.

Berdasarkan latar belakang tersebut, disimpulkan bahwa proyek Tugas Akhir yang penulis susun adalah **"PERANCANGAN *GAME VISUAL NOVEL* MENGGUNAKAN REN'PY DENGAN KUSTOMISASI *AVATAR* ".**

1.2 Rumusan Masalah

Mengacu pada latar belakang di atas, penulis menyimpulkan ada beberapa rumusan masalah. Rumusan masalah tersebut adalah :

1. Bagaimana cara membuat *game visual novel* dengan menggunakan Ren'Py?
2. Bagaimana cara menyimpan pertanyaan dan jawaban yang ada dalam *game*?
3. Bagaimana cara menampilkan *avatar* sesuai dengan keinginan pemain?

1.3 Tujuan Penulisan

Tujuan dari pembuatan perangkat lunak *visual novel* ini yaitu :

1. Untuk memenuhi salah satu syarat kelulusan program pendidikan sarjana di Sekolah Tinggi Manajemen Informatika dan Komputer LIKMI.
2. Untuk mempelajari cara memvisualisasikan sebuah novel menggunakan *Ren'Py*.

1.4 Batasan Masalah

Mengingat begitu luasnya ruang permasalahan yang ada, maka batasan masalah yang diambil oleh penulis antara lain :

1. Pembuatan *game visual novel* menggunakan *Ren'Py*.
2. *Game visual novel ini* dimainkan secara *offline*.
3. Hanya terdapat satu *background music* pada *game,* dan tidak memiliki *sound efect*.

1.5 Kegunaan Hasil

Kegunaan hasil dari tugas akhir ini diharapkan dapat bermanfaat bagi orang banyak. Beberapa kegunaan bagi pembaca antara lain :

1. Dapat memahami isi dari cerita dengan lebih mudah.
2. Agar membaca novel menjadi lebih menyenangkan, sehingga membuat minat membaca di masyarakat meningkat.

1.6 Metodologi Penelitian

Untuk lebih jelas dalam penyusunan Tugas Akhir ini, penulis menggunakan metode-metode yang sesuai dengan masalah tersebut, adapun metode yang digunakan dalam pengembangannya adalah sebagai berikut:

1. Studi Pustaka

   Untuk melengkapi data yang diperlukan, maka penulis melakukan studi pustaka, yaitu dengan cara membaca sumber data lain yaitu buku-buku, *website,* ataupun artikel sumber yang dinilai bersangkutan dengan masalah yang diteliti dan dapat membantu penelitian. Serta untuk memperoleh data-data sekunder yang digunakan sebagai landasan teori dan sebagai pedoman dalam pemecahan masalah.

2. Perancangan

   Dimulai dengan menganalisa komponen-komponen dan alur cerita pada game. Setelah gambaran secara umum lengkap, maka mulai dilakukan desain atas objek-objek yang sudah dianalisa. Proses perancangan terus berkembang sampai *game* yang dirancang sudah memenuhi kriteria selesainya *game.*

1.7 Sistematika Penulisan

Penulisan tugas akhir ini terdiri dari 5 (lima) bab, uraian singkat mengenai isi dari masing-masing bab antara lain :

1. BAB I : PENDAHULUAN

   Bab ini membahas mengenai latar belakang pertimbangan penulis untuk mengambil topik pembuatan *visual novel* dengan menggunakan Ren'Py. Dari latar belakang

penulisan dibahas pula mengenai permasalahan yang dihadapi, tujuan dari penulisan, batasan masalah, kegunaan hasil tugas akhir, metode penelitian, dan sistematika dalam penulisan karya tulis ini.

2. BAB II : LANDASAN TEORI

   Bab ini berisi teori yang dipakai penulis sebagai landasan dalam penulisan Tugas Akhir. Bab ini terdiri atas landasan teori yang mulai dari teori yang bersifat umum seperti Pengertian rekayasa perangkat lunak, Pemrograman berbasis objek, UML, pengertian *game*, sampai teori tentang *visual novel* juga perangkat lunak yang mendukung perancangan dan sistematika penulisan seperti Ren'py juga bahasa Python.

3. BAB III : ANALISIS DAN PERANCANGAN

   Bab ini menjelaskan tentang analisa masalah, serta perancangan perangkat lunak yang dirancang penulis berdasarkan pendekatan berorientasi objek yang meliputi diagram *use case diagram*, skenario, *activity diagram*, *class diagram* dan perancangan antar muka.

4. BAB IV : IMPLEMENTASI DAN PENGUJIAN

   Dalam bab ini menjelaskan implementasi *game* yang dikembangkan serta merinci setiap pengujian atas bagian-bagian yang ada di dalamnya dengan tujuan agar dapat membuat kesimpulan mengenai *game* yang dibuat.

5. BAB V : KESIMPULAN DAN SARAN

   Bab terakhir ini berisi mengenai kesimpulan dan saran-saran apa saja yang dapat mendukung pengembangan *game* sejenis dan perbaikanya.

BAB II

LANDASAN TEORI

2.1 Rekayasa Perangkat Lunak

2.1.1 Definisi Perangkat Lunak

Definisi perangkat lunak menurut beberapa buku *"Software Engineering"* adalah :

1. *Software is: (1) instructions (computer programs) that when executed provide desired features, function, and performance; (2) data structures that enable the programs to adequately manipulate information, and (3) descriptive information in both hard copy and virtual forms that describes the operation and use of the programs.* (Pressman, 2010:4)

2. *Perangkat lunak adalah instruksi langsung komputer untuk melakukan pekerjaan dan dapat ditemukan di setiap aspek kehidupan modern dari aplikasi yang kritis untuk hidup (life-critical), seperti perangkat pembantuan medis dan pembangkit tenaga listrik sampai perangkat hiburan, seperti video game.* (Simarmata, 2010:1)

3. *"Perangkat lunak adalah seluruh perintah yang digunakan untuk memproses informasi."* (Mulyanto, 2008:2)

4. *"Perangkat lunak (software) adalah program komputer yang terasosiasi dengan dokumentasi perangkat lunak seperti dokumentasi kebutuhan, model desain, dan cara penggunaan (user manual)"* (Sukamto, 2013:2)

Berdasarkan definisi di atas maka penulis menyimpulkan bahwa perangkat lunak (*software*) merupakan suatu perintah yang digunakan untuk memproses informasi dan diimplementasikan ke dalam komputer. Atau program komputer yang berfungsi sebagai jembatan antara pengguna dan komputer. Perangkat lunak dapat berupa program atau prosedur.

Rosa A. Sukamto dan M. Shalahuddin menjelsakan bahwa perangkat lunak memiliki beberapa karakter. Karakter-karakter perangkat lunak tersebut adalah (Sukamto, 2013:2) :

1. Perangkat lunak dibangun dengan rekayasa (*software engineering*) bukan diproduksi secara manufaktur atau pabrikan.
2. Perangkat lunak tidak pernah usang (*"wear out"*) karena kecacatan dalam perangka lunak dapat diperbaiki.

3. Barang produksi pabrikan biasanya komponen barunya akan terus diproduksi, sedangkan perangkat lunak biasaya terus diperbaiki seiring bertamahnya kebutuhan

Perangkat lunak yang dibuat oleh pengembang (*developer*) perangkat lunak terdiri dari dua jenis, yaitu ;

1. Perangkat lunak generik

   Produk perangkat lunak yang dibuat untuk dijual atau dipopulerkan tanpa ada yang memesan terlebih dahulu (*open source*). Misalnya perangkat lunak *system* perasi, perangkat lunak untuk membuat dokumen, *slide presentasi*, dan lain sebagainya.

2. Perangkat lunak pemesanan

   Produk perangkat lunak yang dibuat karena ada pelanggan yang melakukan pemesanan. Misalnya sebuah instansi memerlukan perangkat lunak untuk memenuhi proses bisnis yang ada, maka instansi tersebut akan bekerja sama dengan *developer* untuk membuat perangkat lunak yang diinginkan.

2.1.2 Definisi Rekayasa Perangkat Lunak

Menurut Janer Simarmata rekayasa adalah

"penerpan ilmu dan teknologi untuk menyelesaikan permaslahan manusia." (Simarmata, 2010:10).

Jadi rekayasa perangkat lunak sendiri memiliki arti

"disiplin rekayasa dengan perangkat lunak yang dikembangkan" (Simarmata, 2010:11).

Sedangkan menurut Ivan Marsic

"Software engineering is a discipline for solving business problems by designing and developing software-based systems." (Marsic, 2012:1).

Rosa A. Sukamto dan M. Shalahuddin menjelaskan bahwa rekayasa perangkat lunak

"merupakan pembangunan dengan menggunakan prinsip atau konsep rekayasa dengan tujuan menghasilkan perangkat lunak yang bernilai ekonomi yang dipercaya dan bekerja secara efisien menggunkan mesin." (Sukamto, 2013:4)

Rekayasa perangkat lunak dapat diartikan sebagai disiplin untuk memecahkan masalah bisnis dengan merancang dan mengembangkan sistem berbasis *software.*

Rekayasa perangkat lunak muncul karena krisis perangkat lunak selama tahun 1960 dan tahun 1970-an. Menurut Janer Simarmata rekayasa perangkat lunak merupakan penggabungan dengan empat elemen kunci, yaitu :

1. Kelayakan: menemukan solusi yang terbaik untuk suatu masalah,
2. Memaksimalisasi nilai: memaksimalkan nilai dari solusi yang disediakan,
3. Strategi yang efektif: mengadopsi suatu strategi yang efektif untuk mengebangkan solusi, dan
4. Pemodelan: perancangan suatu gambaran visual dari tahap sebelum solusi sampai implementasinya.

Rekayasa perangkat lunak terfokus pada beberapa kriteria perangkat lunak yang harus dipenuhi dalam pembuatannya. Kriteria tersebut adalah (Sukamto, 2013:5) :

1. Dapat terus dipelihara setelah perangkat lunak selesai dibuat seiring berkembangnya teknologi dan lingkungan (*maintainability*)
2. Dapat diandalkan dengan proses bisnis yang dijalankan dan perubahan yang terjadi (*dependapbility* dan *robust*)
3. Efisien dari segi sumber daya dan penggunaan
4. Kemampuaan untuk dipakai sesuai dengan kebutuhan (*usability*)

Dari kriteria di atas, maka perangkat lunak yang baik adalah perangkat lunak yang dapat memenuhi kebutuhan pelanggan atau *user* atau berorientasi pada pelanggan, bukan berorientasi pada pembuat atau pengembang perangkat lunak. Tujuan dari rekaysa perangkat lunak sendiri untuk meminimalkan biaya agar lebih efisien.

2.1.3 Model Pengembangan Sistem Perangkat Lunak

Dalam pengembangan perangkat lunak, ada beberapa model dalam penerapan tahapan prosesnya. Beberapa contoh model pengembangan tersebut adalah :

1. Model pengembangan Prorotipe (*Prototype*)

   Model pengembangan prototipe menurut Roger S. Pressman

Prototype. Often, a customer defines a set of general objectives for software, but does not identify detailed requirements for functions and features. In other cases, the developer may be unsure of the efficiency of an algorithm, the adaptability of an operating system, or the form that human-machine interaction should take. In these, and many other situations, a prototyping paradigm may offer the best approach. (Pressman, 2010:43)

Dari defnisi tersebut, model *prototype* adalah sebuah model yang mendefinisikan suatu tujuan umum untuk perangkat lunak dan pembuat perangkat lunak akan menyediakan *prototype* dari perangkat lunak yang digunakannya. Dalam hal ini, paradigma *prototype* mungkin menawarkan pendekatan dengan konsumen dan hasil yang lebih baik.

Janer Simarmata menjelaskan bahwa

"Sebuah prototipe adalah bagian dari produk yang mengekspresikan logika maupun fisik antarmuka eksternal yang ditampilkan." (Simarmata, 2010:62)

Model pengembangan prototipe (*prototype*) ini menyediakan sebuah bentuk *prorotype* (contoh) sebagian ataupun seluruh sistem yang sedang dikerjakan. Dengan adanya contoh dari *system* tersebut, membuat konsumen dapat menggunakan prototipe dan menyedikan masukan untuk pengembangan dari sistem sebelum pengembangan dengan skala yang lebih besar berlanjut.

Langkah-langkah dalam membuat model *prototype* yaitu :

a. *Communication*

Pada tahap ini *developer* mengkonsultasikan kebutuhan awal perangkat lunak dengan konsumen, juga mengumpulkan dan menganalisis kebutuhan untuk pembuatan sistem.

b. *Quick Plan and Modeling Quick Design*

Setelah tahap pertama, lalu *developer* membuat desain atau perancangan dari sistem dengan cepat

c. *Construction of Prototype*

Kemudian *developer* mulai membangun *prototype*, dan diserahkan kepada konsumen untuk diuji coba.

d. *Development Delivery and Feedback*

Pada tahapan ini konsumen mendapat dan mencoba *prototype system*, kemudian memberi *feedback* / evaluasi untuk memperbaiki *prototype.* Revisi berdasarkan *feedback* dibangun lalu diserahkan kembali kepada konsumen untuk diuji coba.

e. Langkah-langkah tersebut terus berulang hingga konsumen merasa perangkat lunak sudah sesuai dengan yang diharapkan.

f. Apabila konsumen sudah puas terhadap prototipe yang telah dibangun, pembuatan sistem berskala besar dapat dimulai.

Menurut Rosa A. Sukamto mengenai model *prototype* adalah

"Model pengembangan prototype cocok digunakan untuk menggali spesifikasi kebutuhan pelanggan secara lebih detail tetapi beresiko tinggi terhadap membengkaknya biaya dan waktu proyek." (Sukamto, 2013:33)

Berdasarkan penjelasan mengenai beberapa model pengembangan sistem perangkat lunak di atas, maka penulis mengambil model pengembangan *prototype* dalam mengerjakan tugas akhir ini.

Model pengembangan perangkat lunak ini menjadi salah satu model yang digunakan karena adanya contoh sistem *prototype*, sehingga konsumen dapat lebih mengenal perangkat lunak yang dibuat. Beberapa kelemahan dari model *prototype* sebagai berikut :

1. Konsumen menjadi lebih sering mengubah-ubah atau menambah spesifikasi kebutuhan karena menganggap aplikasi dengan cepat dikembangkan. Dengan adanya iterasi ini dapat menyebkan pengembang banyak mengalah kepada konsumen karena perubahan dan penambahan spesifikasi kebutuhan perangkat lunak.
2. Proses analisis dan perancangan terlalu singkat, karena konsumen hanya menambahkan spesifikasi kebutuhan setelah menggunakan aplikasi prototipe yang sudah dibuat.

3. *Prototype* yang dihasilkan tidak selamanya mudah dirubah, dikarenakan proses analisis yang singkat bersama konsumen.
4. Konsumen terkadang tidak menyadari bahwa perangkat lunak yang ada belum maksimal dan belum memiliki kualitas yang baik juga belum memikirkan kemampuan untuk pemeliharaan dalam jangka panjang.
5. *Developer* ingin lebih cepat menyelesaikan proyek. Hal ini dapat menyebabkan kualitas perangkat lunak dengan penggunaan algoritma dan bahasa pemrograman yang sederhana agar prototipe lebih cepat selesai, dan membuat sistem yang bekerja tidak efisien.
6. Hubungan antara konsumen dengan komputer yang disediakan memungkinkan tidak mencerminkan teknik perancangan yang baik.

Sedangkan keunggulan dari model pengembangan *prototype* yaitu:

1. Komunikasi *user-developer*.

   Dengan seringnya komunikasi antara konsumen dan *developer,* maka membuat *developer* dapat selalu meminta pendapat *user* dalam pengembangan perangkat lunak dan menjalin komunikasi yang baik antara konsumen dan *developer*.
2. Membantu analisis.

   Menentukan kebutuhan *user* yang sebenarnya dan meminimalkan kesalahan pada persepsi mengenai perangkat lunak.
3. Peran konsumen meningkat.

   Evaluasi yang sering dilakukan oleh *user* dan dapat memberikan masukan setiap saat.
4. Pengembangan lebih cepat.

   Perangkat lunak dapat langsung dibuat dan *user* pun dapat melihat perkembangan dari sistem setiap tahapnya.
5. Implementasi mudah.

   User sudah mengenal perangkat lunak yang dikembangkan, jadi tidak akan merasa asing. Sejak awal juga *user* sudah lebih akrab dengan sistem yang dibuat.

2.2 Metodologi Penelitian Berorientasi Objek

Metodologi berarti suatu cara atau pendekatan yang digunakan untuk mencapai tujuan tertentu. Ada beberapa pendekatan metodologi dalam pengembangan perangkat lunak, salah satunya adalah pendekatan berorientasi objek. *Object Oriented Methodology* (OOM) adalah pendekatan pengembangan sistem baru dan memfasilitasi penggunaan kembali komponen *software*.

Menurut Rosa A. Sukamto metodologi berorientasi objek banyak dipilih, karena motodologi sebelumnya banyak menimbulkan masalah seperti adanya kesulitan saat mentransformasi hasil dari satu tahap pengembangan ke tahap berikutnya, misalnya pada metode pendekatan terstruktur. Aplikasi yang dikebangkan saat ini sangat beragam, contohnya aplikasi bisnis, *real-time, utility*, dan lain sebagainya aplikasi tersebut dibuat dengan *platform* yang berbeda-beda sehingga menimbulkan tuntutan kebutuhan metodologi pengembangan yang dapat mengakomodasi ke semua jenis aplikasi.

Rosa A. Sukamto juga memaparkan beberapa keuntungan dalam menggunakan metodologi berorientasi objek. Beberapa keuntungannya adalah sebagai berikut (Sukamto, 2013:100):

1. Meningkatkan produktivitas

   Dapat dikatakan untuk meningkatkan produktivitas karena pemakaian kelas dan objek yang ditemukan dalam suatu maslah dapat dipakai ulang untuk masalah lainnya yang melibatkan objek tersebut (*reusable*).

2. Kecepatan pengembangan

   Karena sistem yang dibangun dengan baik dan benar pada saat analisis dan perancangan, akan menyebabkan berkurangnya kesalahan pada saat pengkodean.

3. Kemudahan pemeliharaan

   Mudah dalam pemeliharaannya karena dengan model objek dan pola-pola yang cenderung tetap dan stabil dapat dipisahkan dan pola-pola yang mungkin sering diubah-ubah.

4. Adanya konsistensi

   Karena sifat pewarisan dan penggunaan notasi yang sama pada saat analisis, perancangan maupun pengkodean.

5. Meningkatkan kualitas perangkat lunak

   Karena pendekaan pengembangan lebih dekat dengan dunia nyata dan adanya knsistensi pada saat pengembangannya, perangkat lunak yang dihasilkan mampu memenuhi kebutuhan pemakai serta mempunyai sedikit kesalahan.

Analisis berorientasi objek alau sering disebut *Object Oriented Analysis* (OOA) adalah tahapan dalam meganalisis spesifikasi atau kebutuhan untuk sistem yang akan dibangun dengan konsep berorientasi objek, agar kebutuhan yang ada dapat diimplementasikan menjadi sistem berorientasi objek. Analisis ini sebaiknya dilakukan oleh orang yang benar-benar memahami implementasi sistem, karena tanpa pemahaman itu, maka sistem yang dihasilkan bisa jadi tidak realistis jika diimplementasikan dengan berbasis objek.

Desain berorientasi objek atau sering disebut *Object Oriented Design* (OOD) adalah tahapan perantara untuk memetakan spesifikasi atau kebutuhan sistem yang akan dibangun dengan konsep berorientasi objek ke desain pemodelan agar lebih mudah diimplementasikan dengan premrograman berorientasi objek. Desai berorientasi objek biasanya dituangkan dalam dokumentasi *software* dengan menggunakan perangkat pemodelan berorientasi objek, diantaranya adalah *Unified Modeling Language* (UML).

Rosa A. Sukamto menjelaskan bahwa OOA dan OOD yang ada dalam proses yang berulang-ulang sering kali memiliki batasan yang samar, sehingga kedua tahapan ini sering juga disebut *Object Oriented Analysis and Design* (OOAD). OOAD dalam bahasa Indonesia berarti Analisis dan Desain Berorientasi Objek.

Hanid Al Fatta menjelaskan bahwa ada beberapa ciri khas dalam pendekatan OOAD, yaitu *object, inheritance,* dan *object class.* Penjelasannya adalah sebagai berikut:

6. *Object*

   Object adalah struktur yang mengenkapsulasi atribut dan metode yang beroperasi berdasarkan atribut-atribut. *Object* disebut juga abstraksi dari benda nyata di mana

data dan proses diletakkan bersama untuk memodelkan struktur dan perilaku dari objek dunia nyata

7. *Object class*

   Object class adalah sekumpulan objek yang berbagi struktur dan perilaku yang sama.

8. *Inheritance*

   Inheritance adalah properti yang muncul ketika tipe entitas atau *object class* disusun secara hierarki dan setiap tipe entitas menerima atau mewarisi atribut dan metode pendahulunya.

2.3 UML

Grady Booch, Jim Rumbaugh, dan Ivar Jacobson adalah pengembang UML pada pertengahan 1990-an dengan banyak *feedback* dari komunitas pengembangan perangkat lunak. Pada tahun 1997, UML 1.0 diserahkan kepada *Object Management Group*, sebuah konsorsium nirlaba yang terlibat dalam menjaga spesifikasi untuk digunakan oleh industri komputer. UML 1.0 direvisi menjadi UML 1.1 dan diadopsi akhir tahun itu. Standar saat ini adalah UML 2.0 dan sekarang menjadi standar ISO.

UML menurut Martin Fawler adalah

"The Unified Modeling Language (UML) is a family of graphical notations, backed by single meta-model, that help in describing and designing software systems, particularly software systems built using the object-oriented (OO) style ." (Fawler, 2004:1)

Sedangkan menurut Adi Nugroho adalah

"UML (Unified Modeling Language) adalah 'bahasa' pemodelan untuk sistem atau perangkat lunak yang erparadigma 'berorientasi objek'. " (Nugroho, 2010:6)

Jadi UML adalah bahasa pemodelan dalam sebuah sistem, yang membantu dalam menjelaskan dan merancang sistem perangkat lunak, khususnya sistem perangkat lunak berbasis objek. Pemodelan tersebut digunakan untuk menyederhanakan permasalahan-permasalahan yang kompleks agar lebih mudah dipelajari dan dipahami.

Tujuan dari pemodelan perangkat lunak ini adalah sebagai sarana analisis, pemahaman, visualisasi, dan komunikasi antar anggota tim pengembang perangkat lunak, serta sebgai sarana dokumentasi yang bermanfaat untuk mempelajari perilaku dari perangkat lunak serta bermanfaat untuk melakukan pengujian terhadap perangkat lunak yang telah disediakan.

Brend Bruegge menjelaskan bahwa UML memiliki beberapa notasi. Notasi tersebut antara lain: (Bruegge, 2004)

1. *Use Case Diagram*

   Use case diagram adalah penggambaran fungsionalitas yang diharapkan dari sebuah sistem. Sebuah *use case* mempresentasikan sebuah interaksi antara aktor dengan sistem. Seorang/sebuah aktor adalah penggambaran sebuah entitas yang berinteraksi dengan sistem (misalnya pengguna, sistem lain, dan lingkungan fisik dari sistem) untuk melakukan pekerjaan-pekerjaan tertentu.

   Sebuah *use case* dapat meng-*include* fungsi *use case* lain sebagai bagian dari proses. Diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* juga dapat di-*include* lebih dari satu use case.

   Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behavior*-nya sendiri. Sementara hubungan generelasisasi antar *use case* menunjukkan bahwa *use case* merupakan spesialisasi dari yang lain.

2. *Class Diagram*

   Class Diagram digunakan untuk menggambarkan struktur dan deskripsi *class*, *package* dan objek yang ada pada sistem, serta hubungannya satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain. *Class* sendiri adalah abstraksi yang menentukan struktur umum dan perilaku suatu objek. Objek adalah contoh dari kelas yang dibuat dan dimodifikasi. Sebuah objek memiliki suatu kondisi yang menyertakan nilai-nilai dari atribut dan hubungannya dengan objek lain. *Class* menggambarkan keadaan (atribut) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda).

3. *Activity Diagram*

   Activity diagram atau diagram aktivitas menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Dalam diagram ini perlu diperhatikan bahwa *activity diagram* menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan oleh sistem.

2.4 Pemrograman Berorientasi Objek

Pemrograman berorientasi objek mempunyai tiga karakteristik utama. Karakteristik tersebut antara lain:

1. *Encapsulation* (pengkapsulan)

   Encapsulation merupakan dasar untuk pembatasan ruang lingkup program dengan data yang diproses. Data dan prosedur serta fungsi dikumpulkan menjadi satu objek, sehingga prosedur atau fungsi dari luar dapat mengaksesnya. Namun data tetap dapat terlindungi dari prosedur atau objek lain dari luar, selain prosedur yang sudah berada di dalam objek tersebut.

2. *Inheritance* (Pewarisan)

   Roger S. Pressman menjelaskan bahwa

   "Inheritance is one of the key differentiators between conventional and object-oriented systems." (Pressman, 2010:866)

   Inheritance adalah teknik yang menyatakan bahwa anak dari objek akan mewarisi data-data maupun atribut dan metode langsung dari induknya. Dengan kata lain bahwa atribut dan metode yang ada pada induk objek diturunkan langsung kepada anak objek.

   Suatu kelas dapat ditentukan secara umum, kemudian ditentukan secara spesifik menjadi subkelas. Setiap subkelas mempunyai hubungan atau mewarisi semua sifat yang dimiliki oleh kelas induknya, kemudian ditambah dengan sifat unik yang dimilikinya. Kelas objek sendiri dapat didefinisikan sebagai atribut dan service dari kelas objek lainnya.

Inheritance dapat diartikan sebagai penggambaran generalisasi dari sebuah kelas, contoh dari *inheritance* yaitu mobil dan sepeda motor adalah subkelas dari kendaraan bermotor. Kedua subkelas tersebut mewarisi sifat yang dimiliki oleh kendaraan bermotor, yaitu memiliki mesin dan dapat berjalan. Kedua subkelas tersebut pun mempunyai sifat masing-masing yang berbeda, misalnya dari jumlah roda dan kemampuan untuk berjalan mundur yang tidak dimiliki oleh sepeda motor.

3. *Polymorphism* (Polimorfisme)

   Menurut Roger S. Pressman

   "Polymorphism is a characteristic that greatly reduces the effort required to extend the design of an existing object-oriented system." (Pressman, 2010:868)

   Polymorphism adalah konsep yang menyatakan bahwa sesuatu yang sama dapat mempunyai bentuk dan perilaku yang berbeda. Dapat pula berarti bahwa operasi yang sama mungkin mempunyai perbedaan dalam kelas yang berbeda. Kemampuan dari objek-objek yang berbeda untuk melakukan sebuah metode dalam merespon pesan yang sama. Seleksi dari metode yang sesuai bergantung pada kelas yang seharusnya menciptakan objek tersebut.

Dalam buku "Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek" yang dibuat oleh Rosa A. Sukamto, dijelaskan bahwa ada beberapa konsep dasar atau istilah-istilah yang perlu diperhatikan dalam pemrograman berorientasi objek (Sukamto, 2013:104) :

1. Kelas (*class*)

   Kelas adalah kumpulan objek-objek dengan karakteristik yang sama. Kelas merupakan definisi statik dan himpunan objek yang sama yang mungjin lahir atau diciptakan dari kelas tersebut. Sebuah kelas akan mempunyai sifat (attribut), kelakuan (operasi/metode), hubungan (*relationship*) dan arti. Secara teknis, kelas adalah sebuah struktur tertentu dalam pembuatan perangkat lunak. Kelas secara fisik adalah berkasi atau *file* yang berisi kode program, di mana kode program merupakan seumua hal yangterkait dengan nama kelas.

2. Objek (*object*)

   Objek adalah abstraksi dan sesuatu yang mewakili dunia nyata seperti benda, manusia, satuan organisasi, tempat, kejadian, struktur, status, atau hal-hal yang bersifat abstrak. Objek merupakan suatu entitas yang mampu menyimpan informasi (status) dan mempunyai operasi yang dapat diterapkan atau dapat berpengaruh pada status objeknya. Objek mempunyai siklus hidup yaitu diciptakan, dimanipulasi, dan dihancurkan. Objek jika dilihat dalam segi teknis adalah elemen pada saat *runtime* yang akan diciptakan, dimanipulasi, dan dihancurkan saat eksekusi sehingga sebuah objek hanya ada saat sebuah program dieksekusi, jika masih dalam bentuk kode, disebut sebagai kelas jadi pada saat *runtime* (saat sebuah program dieksekusi) yang terbentuk adalah objek, di dalam teks progam yang terbentuk hanyalah kelas.

3. Metode (*method*)

   Operasi atau metode pada sebuah kelas hampir sama dengan fungsi atau prosedur pada metodologi struktural. Sebuah kelas boleh memiliki lebih dari satu metode untuk memanipulasi objek.

   Metode atau operasi dapat berasal dari *event*, aktivitas atau keadaan, fungsi, atau kelakuan dunia nyata. Contoh metode misalnya *read, wite, move copy,* dan sebagainya. Kelas sebaiknya memiliki metoe *get* dan *set* pada setiap attribute agar konsep enkapsulasi tetap terjaga. Metode *get* digunkan untuk memberikan akses kelas lain dalam mengakses atribut, sedangkan *set* digunakan mengisi atribut, agar kelas lain dapat mengakses atribut secara langsung.

4. Atribut (*attribute*)

   Atribut dari sebuah kelas adalah variabel global yang dmiliki sebuah kelas. Atribut dapat berupa nilai atau elemen-elemen data yang dimiliki oleh objek dalam kelas objek, misalnya berat, jenis, nama, dan sebagainya. Atribut sebaiknya bersifat privat untuk menjaga konsep enkapsulasi.

5. Abstraksi (*abstraction*)

   Prinsip untuk mempresentasikan dunia nyata yang kompleks menjadi satu bentuk model yang sederhana dengan mengabaikan aspek-aspek lain yang tidak sesuai dengan permasalahan.

6. Enkapsulasi (*encapsulation*)

   Pembungkusan atribut data dan layanan (operasi-operasi) yang dipunyai objek untuk menyembunyikan implementasi dan objek sehingga objek lain tidak mengetahui cara kerjanya.

7. Pewarisan (*inheritance*)

   Mekanisme yang memungkinkn satu objek mewarisi sebagia atau seluruh definisi dan objek lain sebagai bagian dari dirinya.

8. Antarmuka (*interface*)

   Antarmuka atau yang sering disebut *interface* sangat mirip dengan kelas, tapi tanpa atribut kelas dan metode yang dideklarasikan tanpa isi. Deklarasi metode pada sebuah *interface* dapat diimplementasikan oleh kelas lain. Sebuah kelas dapat mengimpllementasikan lebih dari satu *interface* di mana kelas tersebut akan mendeklarasikan metode pada *interface* yang dibutuhkan oleh kelas itu sekaligus mendefinisikan isinya pada kode program di dalam kelas. Metode yang ada pada *interface* dan diimplementasikan pada suatu kelas harus sama persis sama dengan yang ada pada *interface*. *Interface* biasanya digunakan agar kelas lain yang tidak mengakses *interface*nya langsung ke suatu kelas.

9. *Reusability*

   Pemanfaatan kembali objek yang sudah didefinisiskan untuk suatu permasalahan pada masalah lainnya yang melibatkan objek tersebut. Contohnya dalam sebuah aplikasi peminjaman buku diperlukan kelas anggota, maka ketika membuat aplkasi penyewaan VCD, kelas anggota ini dapat dipakai kembal dengan sedikit perubahan untuk aplikasi penyewaan VCD tanpa harus membuat dari awal lagi.

10. Generelasisasi dan spesialisasi

    Menunjukkan hubungan antar kelas dan objek yang umum dengan kelas dan objek yang khusus. Misalnya kelas yang lebih ummum (generalisasi) adalah kendaraan darat dan kelas khususnya (spesialisasi) adalah mobil, motor, dan kereta.

11. Komunikasi antar objek

    Komunikasi antar objek dilakukan lewat pesan (*message*) yang dikirm dari satu objek ke objek lainnya.

12. Polimorfisme (*polymorphism*)

    Kemampuan suatu objek untuk digunakan di beberapa tujuan yang berbeda dengan nama yang sama sehingga menghemat baris pada program.

13. *Package*

    Package adalah sebuah container atau kemasan yang dapat digunakan untuk mengelompokkan kelas-kelas sehingga memungknkan beberapa kelas yang bernama sama disimpan dalam *package* yang berbeda.

2.5 *Game*

2.5.1 Definisi *Game*

Kata game berasal dari bahasa Inggris yang artinya permainan. Definisi *game* atau permainan menurut Ernest Adams adalah

"A game is a type of play activity, conducted in the context of a pretended reality in which the participant(s) try to achieve at least one arbitrary, nontrivial goal by acting in accordance with rules." (Adams, 2010:3)

Game adalah salah satu media yang berkembang sangat pesat. Ivan C. Sibero memaparkan bahwa *game* atau permainan dapat dikategorikan menjadi dua bagian yaitu *game* fisik dan *game* elektronik. *Game* fisik merupakan permainan yang berhubungan langsung dengan gerak fisik. Permainan ini sering dimainkan ketika masih anak-anak, contohnya seperti permainan lompat tali, petak umpet, dan lain sebagainya. Seiring dengan berkembangnya permainan, permainan fisik sudah bannyak ditinggalkan, dan *game* elektronik merupakan permainan yang banyak dimainkan dewasa ini. *Electronic*

game saat ini berkembang dengan pesat yang membuat berbagai konsol baru yang lebih canggih dan mulai bermunculan. (Sibero, 2009:9)

2.5.2 Sejarah *Game*

Electrinic game atau yang sering disebut sebagai *video game* ini pertama kali ditemukan oleh Thomas T. Goldsmith Jr. dan Estle Ray Mann pada tahun 1947. Mereka menemukan *Cathode-Ray Tube* yang merupakan sebuah tabung *vacuum* yang digunakan sebagai media untuk membuat simulasi kecepatan tembakan dan arah tembakan sebuah roket yang mendasari perkembangannya saat ini. Kemudian pada bulan Februari tahun 1951 Christopher Strachey mulai menggunakan memori dimana aplikasinya diterapkan untuk kebutuhan para pilot. Penemuan-penemuan terus berkembang hingga tahun 1959. Namun era perkembangan konsol *game* ini baru dimulai setelah masa ini.

Berikut sejarah *game* menurut (Sibero, 2009:10):

1. Generasi Pertama (1972-1977)

   Generasi pertama ini diawali dari ide seorang teknisi televisi bernama Ralph Baer, yang mulai membuat *game* yang dapat dimainkan oleh dua pemain. Permainan ini bernama *chese* di mana dua buah titik saling berkejaran. Generasi pertama ini merupakan awal dari interaktif *game*.

2. Generas Kedua (1976-1983)

   Generasi kedua ini sering disebut sebagai generasi konsol 8 bit, di mana pemrograman video *game* yang dibuat lebih *advance.* Sekitar tahun 1976 Fairchild Channel F merupakan produk yang mulai dirilis.

3. Generasi Ketiga (1983-1992)

   Pada generasi sebelumnya, perkembangan *game* banyak dilakukan di Eropa dan Amerika, namun pada generasi ini Jepang juga mulai merilis konsol barunya yang dinamakan Famicom dan berubah nama enjadi Nitendo. *Pasar di Amerika Utara turut dipenuhi oleh Nintendo Entertainment System* (NES). Sedangkan di Eropa dan Brasil, selain NES, *Sega Master System* pun turut berkembang. Sedangkan ATARI membuat produk barunya yaitu ATARI 7800.

4. Generasi Keempat (1987-1996)

   Generasi keempat ini dikenal dengan generasi 16 bit, yang pengembangannya dibulai sekitar Oktober 1987. Beberapa *game* yang cukup dikenal pada generasi ini yaitu *Metroid, Zelda, Star Fox, Kirby, Dragon Quest, Final Fantasy, Seiken Densetsu (Secret of Mana), Donkey Kong, Street Fighter, Mortal Kombat,* dan *Mega Man X.*

5. Generasi Kelima (1993-2002)

   Pada generasi ini mulai muncul Konsol 64 bit dan grafis pada *game* yang mulai berkembang menjadi 3D. Pengembang konsol Nintendo mengeluarkan *Super Nintendo*, dan Sega mengeluarkan *Sega Saturn*.

6. Generasi Keenam (1998-2006)

   Generasi ini merupakan generasi 128 bit. Pada generasi ini, beberapa produk mulai bermunculan, seperti Sony yang mengeluarkan *Playstation 2*, *Microsoft* mengeluarkan *Sega's Dreamcast, Xbox*, dan Nontendo dengan *GameCube*.

7. Generasi Ketujuh(2004)

   Pada generasi ini beberapa pengembang konsol mulai berfokus pada pengembangan saja, dan hanya beberapa perusahaan konsol yang masih bertahan dengan pengembangannya. Perusahaan tersebut adalah Nintendo, Sony, dan Microsoft. Perkembangan konsol ini juga mempengaruhi tumbuhnya para pengembang *software game*.

2.5.3 *Genre Game*

*S*ebuah klasifikasi *game* yang didasari oleh interaksi pemain dan visualisasnya merupakan pengertian dari *genre game*. Namun untuk beberapa kasus pengembang *game* membuat kompilasi antar berbagai *genre* yang membuat variasi format *game* menjadi banyak. Berikut adalah beberapa contoh *genre game* menurut (Sibero, 2009:18):

1. *Action*

   Genre ini adalah jenis *game* yang membutuhkan kecermatan reaksi waktu dan gerak. *Genre* ini juga memiliki banyak rintangan di dalamnya. *Game action* memiliki beberapa jenis game, antara lain:

a. *Ball and Paddle*

Jenis *game* ini merupakan jenis yang pertama dibuat. Bola dan penangkisnya menjadi alat dalam permainan, contohnya *Ping Pong*.

b. *Beat 'em up, hack and slash*

Game ini menekan pada reaksi dari pemainnya. Aksi dalam *genre* ini yaitu berpetualang sambil bertempur. Pemain dihibur dengan beberapa alat seperti penggunaan pedang, dengan aksi pukulan dan tendangan, serta lawan dari pemain yang bervariasi.

c. *Fighting*

Jenis *game* ini sekilas hampir mirip dengan jenis sebelumnya, namun berbeda pada pertempurannya. Dalam permainan ini pemain berhadapan satu lawan satu dengan musuh yang memiliki berbagai keahlian dan digerakkan oleh komputer.

2. *Maze*

Menurut beberapa pengembang, g*ame* ini menjadi dasar untuk pengembangan beberapa *game* lainnya. Dasar dari *game* ini adalah reaksi gerakan pemain yang cepat agar terhindar dari musuh. Contohnya adalah *Pacman* dan *Digger.*

3. *Pinball*

Game ini merupakan aplikasi dari permainan *pinball table.* Kontrol dalam permainan ini berupa dua buah lengan pemukul bola. Nilai yang berbeda di setiap zona didapatkan dari bola yang memantul, namun sedikit sulit untuk mencapai nilai yang tinggi karena bola yang dipukul tidak dapat dikontrol gerakannya.

4. *Shooter*

Jenis *game* ini membuat pemainnya tertantang dengan persenjataan yang dapat dipilih dan umumnya musuh yang disusun acak. Beberapa jenis *game genre* ini antara lain:

a. *First-Person Shooter* (FPS)

Game ini termasuk jenis *game modern* karena membutuhkan teknologi *hardware* yang dapat mendefinisikan visual 3 dimensi (3D) secara *real*. Contoh *game* pada jenis ini adalah *Counter Strike, Condition Zero* dan *Half-life 2*.

b. *Third Person Shooter (TPS)*

Jenis *game* ini mirip dengan FPS, namun perbedaannya berada pada tampilannya. TPS juga menggunakan orang ketiga sebagai sudut pandangnya sehingga gerakan karakter dapat terlihat. Beberapa contoh jenis *game* ini adalah *Tom Rider, Gears of War* dan *Max Payne*.

5. *Role Play Game* (RPG)

RPG memiliki beberapa jenis, seperti *Action Role Play Game* dan *Tactical Role Playing*. *Game* ini biasanya mengandung bayak unsur petualangan. Untuk *game* yang mengandung unsur peperangan, pemain memiliki perlengkapan tambahan seperti mengembangkan kekuatan (*strength*), kecerdasan (*intelligence*), atau kecepatan (*agility*). Beberapa contoh *game* ini seperti *Shining Force* dan *Front Mission Series.*

2.6 Visual Novel

2.6.1 Definisi Novel

Dari sekian banyak bentuk sastra seperti puisi, novel, cerita pendek dan drama, Novel dan cerita pendek merupakan bentuk yang paling banyak dibaca. Karya-karya modern klasik dalam kesusasteraan, kebanyakan juga berisi karya-karya novel. Menurut Kamus Besar Bahasa Indonesia

"Novel merupakan karangan prosa yang panjang mengandung rangkaian cerita kehidupan seseorang dengan orang di sekelilingnya dengan menonjolkan watak dan sifat setiap pelaku." (KBBI, 2008:1008)

Menurut Burhan Nurgiyantoro novel dibagi menjadi dua jenis, yaitu: novel populer dan novel serius.

"Novel popular novel yang populer pada masanya dan banyak penggemarnya" (Nurgiyantoro, 2010:18).

Sedangkan novel serius adalah

"Novel ini disoroti dan diungkapkan sampai ke inti hakikat kehidupan yang bersifat universal". (Nurgiyantoro, 2010: 18)

Sebaliknya novel hiburan hanya dibaca untuk kepentingan santai belaka, yang penting memberikan keasyikan pada pembaca untuk menyelesaikannya. Dengan demikian dapat dikatakan bahwa novel serius punya fungsi sosial, sedang novel hiburan hanya berfungsi personal. Novel berfungsi sosial lantaran novel yang baik ikut membina orang tua masyarakat menjadi manusia. Sedang novel hiburan tidak memperdulikan apakah cerita yang disajikan membina manusia atau tidak, yang penting adalah bahwa novel menarik dan orang mau cepat-cepat membacanya.

2.6.2 Definisi Novel Visual

Visual novel atau dalam bahasa Indonesia dikenal sebagai Novel Visual merupakan sebuah fiksi interaktif yang biasanya menggunakan *anime-style* dalam penerapannya. Novel visual ini merupakan *genre game* yang cukup terkenal di Jepang. Namun ada beberapa novel visual juga menggunakan gambar/foto asli, mauppun video. Novel visual termasuk salah satu jenis permainan petualangan, yang difokuskan di bagian penceritaan, sehingga pemain seringkali hanya membaca novel di komputer yang menampilkan gambar beserta teks dan suara. Seringkali pemain diharuskan untuk memilih di antara pilihan-pilihan yang ada untuk melanjutkan ke jalan cerita selanjutnya. Setiap pilihan yang diambil dapat memengaruhi jalan cerita dari novel visual yang sedang dimainkan. Istilah lain untuk novel visual adalah novel game atau *sound novel* (novel bersuara).

Menurut Dani Cavallaro dalam bukunya yang berjudul "*Anime and the Visual Novel*", novel visual memiliki arti :

"Some of the most cherished, visually appealing and diegetically complex anime titles of recent years have been based upon a very specific type of computer game generally known as visual novel" (Cavallaro, 2010:8)

Menurut Josiah Lebowitz dan Chris Klug, novel visual adalah

"A visual novel is a type of game that is in many ways similar to reading a book." (Lebowitz, 2011:193)

Dalam website aplikasi Ren'Py dijelaskan bahwa novel visual adalah

"Visual novels are computer-based stories that are told through words, images, sounds, and music. Many visual novels also present the player with menu choices that allow the player to control how the story is told." (http://www.renpy.org/why.html, diakses tanggal 26 November 2015, jam 15.00 WIB)

Jadi, dapat disimpulkan bahwa *visual novel* adalah game fiksi interaktif yang menampilkan gambar statis menggunakan gambar anime, foto atau lukisan sehingga mirip dengan rekaman video dan menyerupai media campuran yang mirip dengan membaca buku dalam memainkannya. Dalam novel visual juga terdapat pilihan-pilihan menu yang memunginkan pemain untuk mengontrol bagaimana cerita tersebut dapat diceritakan.

Menurut Josiah Lebowitz dan Chris Klug juga bahwa

"The story is told through large blocks of text, generally written in a first-person perspective from the main character's point of view." (Lebowitz, 2011 :193)

Jadi cerita pada novel visual diceritakan melalui blok teks besar, yang pada umumnya ditulis dalam sudut pandang orang pertama dari titik pandang karakter utama. Novel visual tidak seperti *ebook*, namun terdapat gambar latar belakang yang dapat berubah-ubah berdasarkan lokasi dari cerita dan potret atau gambaran seluruh atau sebagian dari setiap karakter yang sedang berbicara. Terkadang teks dan gambaran visual pada *game* yang ada disertai dengan efek suara dan latar belakang musik, dan di beberapa *game* juga dialog yang diutarakan karakter pun disuarakan.

Salah satu *game visual novel* yang cukup populer di Jepang adalah *Fate/Stay Night*. Kepopuleran yang didapatkan cukup bertahan hingga beberapa tahun sejek rilis, keberhasilan tersebut karena gambaran dari *anime*, manga, film, serial dari novel, dan beberapa bagian lain yang terkait. Namun alasan yang paling penting di balik keberhasilan yang terus menerus didapatkan adalah *Fate/Stay Night* memiliki jalan cerita yang menarik, dan karakter yang mudah diingat.

2.7 Ren'Py

Ren'Py merupakan aplikasi *open source* dan juga gratis untuk penggunaan komersial. Tidak perlu membayar biaya untuk mendistribusikaan *game* yang sudah dibuat menggunakan Ren'Py.

"Ren'Py is a visual novel engine – used by hundreds of creators from around the world – that helps you use words, images, and sounds to tell interactive stories that run on computers and mobile devices." (http://www.renpy.org/, diakses tanggal 26 November 2015, jam 15.06 WIB)

Dalam kutipan tersebut dapat dikatakan bahwa Ren'Py merupakan *visual novel engine* yang dapat mempermudah dalam menggunakan kata-kata, gambar, juga suara untuk menceritakan suatu kisah interaktif yang dapat dijalankan pada komputer.

Ren'Py sudah mengalami perubahan sebanyak 104 kali sejak tahun 2004 hingga akhir tahun 2015. Seperti yang sudah dituliskan dalam *website*-nya bahwa versi pertama diluncurkan tanggal 24 Agustus 2004, dengan versi Ren'Py 4pr1 kemudian mengalami perubahan yang diluncurkan tanggal 6 September 2004 yaitu Ren'Py 4.0. Versi-versi tersebut terus mengalami perubahan hingga perubahan ke 104 yaitu Ren'Py 6.99.8 dengan nama *"Here's to the crazy ones."* pada tanggal 25 Desember 2015. (http://www.renpy.org/release_list.html, diakses tanggal 25 Januari 2016, jam 13.50 WIB)

Salah satu keunggulan dari Ren'Py yaitu dapat berjalan hampir di setiap komputer. Tiga *platform* utama yang mendukung yaitu:

1. Windows XP + (x86)
2. Mac OS X 10.6+ (x86_64)
3. Linux (x86,x86_64)

Ren'Py *launcher* memungkinkan untuk membangun versi Ren'Py *game* untuk ketiga *platform* tersebut. Bahasa pemrograman yang dipakai dalam aplikasi ini adalah bahasa pemrograman Python.

Kelebihan dari aplikasi Ren'Py sendiri adalah:

1. Gratis untuk digunakan dan gratis untuk disebar - luaskan.
2. Kemampuan *cross platform* dimana kita dapat membuat *visual novel* untuk dijalankan pada OS Mac, Linux dan Windows. Bahkan dapat melakukan *Convert* program ke Android.
3. Penggunaan bahasa script yang mudah dipahami.
4. Memiliki berbagai macam fitur dari *game visual novel* secara default.
    a. Dapat dikustomisasi oleh pengguna.
    b. Mendukung berbagai macam tipe *file*.
    c. Mendukung pembuatan cerita bercabang.
    d. Tidak memerlukan komputer dengan spesifikasi tinggi untuk menjalankannya.

Sedangkan kelemahan dari aplikasi Ren'Py adalah :

1. Pada keadaan defaultnya tidak memiliki kemampuan untuk membuat game yang sangat kompleks. Membutuhkan tambahan coding baru, dan Framework tambahan untuk membuat lebih kompleks.
2. Tidak mendukung grafik 3 dimensi.
3. Pembuatan game yang terlalu sederhana memberi efek cepat bosan.

2.8 Phyton

Menurut Erik Westra dalam bukunya yang berjudul *Python GeoSpatial Develompent,* menjelaskan bahwa :

"Python is a modern, high-level language suitable for a wide variety of programming tasks. Technically, it is often referred to as a "scripting language, though this distinction isn't very important nowadays." (Westra, 2010:7)

Dikatakan bahwa Python adalah bahasa tingkat tinggi yang cocok untuk berbagai pemrograman. Dapat dibilang, hal ini sering disebut sebagai bahasa skrip, walaupun perbedaan yang ada tidak terlalu penting.

Berdasarkan pengertian dari *website* python :

"Python is a clear and powerful object-oriented programming language, comparable to Perl, Ruby, Scheme, or Java. Is an easy-to-use language that makes it simple to get your program working. This makes Python ideal for prototype development and other ad-hoc programming tasks, without compromising maintainability." (http://python.org/, diakses tanggal 5 Desember 2015, jam 14.10 WIB)

Menurut *website*-nya tersebut Python adalah bahasa pemrograman berorientasi objek, sebanding dengan bahasa Perl Ruby, Scheme, ataupun Java. Bahasa ini pun mudah digunakan sehingga membuat *programer* mudah dalah bekerja. Dengan adanya hal tersebut, membuat Python ideal untuk pengembangan prototipe, tanpa harus ada pemeliharaan.

Dari kedua pengertian tersebut, dapat disimpulkan bahwa bahasa Python adalah bahasa pemrograman berorientasi objek yang cocok untuk berbagai pemrograman. Bahasa Python sering disebut sebagai bahasa skrip, namun dapat dikatakan bahasa ini tetap sebanding dengan bahasa pemrograman lainnya, seperti bahasa Perl, Ruby, Scheme, ataupun Java.

BAB III

ANALISIS DAN PERANCANGAN

3.1 Gambaran Umum

Perancangan perangkat lunak yang dibahas pada dokumen ini, adalah *game visual novel*. Dalam *game* ini, diharapkan pemain dapat membaca novel dengan tampilan visual yang lebih menarik. *Game* ini juga memiliki animasi tokoh pemain berupa *avatar*. Dengan adanya *avatar*, pemain dapat membuat tokoh yang dimainkan sesuai dengan yang diinginkan. Pemain juga perlu menjawab pertanyaan yang tersedia pada cerita. Pertanyaan-pertanyaan yang diajukan dimaksudkan untuk menentukan kelanjutan dari cerita novel yang dibaca.

Pada saat membaca novel yang bercerita tentang seseorang yang harus memilih antara melanjutkan kuliah atau bekerja, akan muncul beberapa pertanyaan yang harus dijawab. Pemain dapat menjawab pertanyaan yang muncul dengan memilih jawaban yang sudah disediakan dalam bentuk *optional* (sudah tersedia beberapa jawaban dan pemain memilih jawaban tersebut). Setiap jawaban akan menentukan alur cerita dari novel yang dibaca, sehingga kelanjutan dari ceritapun akan berbeda dan membuat akhir cerita yang berbeda pula. Jika pemain sudah menyelesaikan *game*, dan ingin mendapatkan akhir cerita yang berbeda, pemain dapat memulai *game* lagi dari awal dengan pilihan jawaban yang berbeda.

Game ini juga memiliki *mini game* yang akan dimainkan sesuai dengan lanjutan cerita yang dipilih. Jika pemain memainkan *mini game* tersebut dan dapat menyelesaikannya, maka pemain akan melanjutkan *game* sesuai dengan hasil yang didapatkan. Pemain juga dapat mengganti dan memilih *avatar* yang tersedia. Bagian *avatar* yang dapat diganti seperti kepala, mata, mimik wajah, rambut, dan pakaian.

3.2 Analisis Kebutuhan Perangkat Lunak

Game visual novel ini diharapkan memenuhi kebutuhan-kebutuhan sebagai berikut :

1. Dapat memunculkan dialog pada tempat yang tersedia sesuai dengan alur cerita.
2. Dapat menampilkan pertanyaan-pertanyaan yang disediakan dan melanjutkan cerita sesuai dengan pilihan jawaban.
3. Dapat menampilkan *avatar* dan mengganti bagian-bagianya sesuai keinginan.
4. Dapat menampilkan *avatar* sesuai dengan yang sudah ditentukan sebelumnya pada masing-masing dialognya.
5. Dapat memainkan *mini-game* pada percabangan yang sudah ditentukan.

3.3 Pemodelan Perangkat Lunak

3.3.1 Use Case Diagram

Pada *use case* diagram di bawah ini, terdapat 6 *use case* dalam pemodelan perangkat lunak yang dibangun. Diagram ini menggambarkan interaksi antara pengguna dengan sistem, dalam hal ini adalah pemain dengan *game.*

1. *New Game*

   Pada *use case* ini pemain akan memulai *game* yang baru, tetapi harus membuat avatar terlebih dahulu.

2. Buat *Avatar*

   Seperti yang tergambar pada *use case diagram* gambar 3.1, *use case* ini dilakukan saat pemain memulai *game* baru. Pada *use case* ini pemain dapat membuat dan mengubah-ubah *avatar* miliknya.

3. Ubah *Avatar*

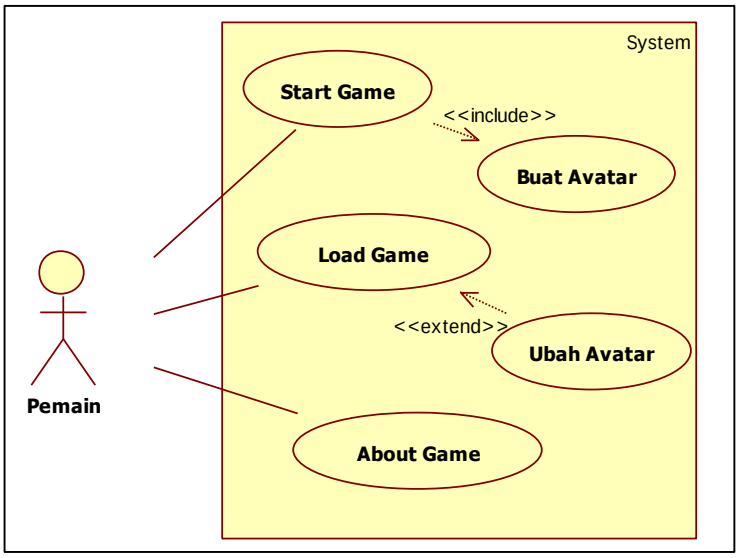
   Use case ubah *avatar* dapat dilakukan ketika pamain memilih *load game.* Pemain dapat juga mengubah properti dari *avatar*. Pada *use case* ini pemain dapat membuat dan mengubah-ubah *avatar* yang sebelumnya sudah dibuat.

4. *Load Game*

   Pada *use case* ini pemain daapat melanjutkan *game* yang sebelumnya sudah disimpan (di-*save*).

5. *About Game*

   Pada *use case* ini pemain dapat membaca deskripsi singkat mengenai *game* atau mempelajari cara memainkan *game*.



Gambar 3. 1
Use Case Diagram

3.3.2 Skenario

Penjelasan setiap *use case*, akan dijelaskan pada tabel-tabel yang ada berikut ini:

1. *Use case* : *Start Game*

   Aktor : Pemain

   Pre-condition : pemain berada pada tampilan *main menu*

   Definsi : Pemain memulai *game* dengan data yang baru

   Skenario normal

Tabel 3. 1
Skenario *use case Start Game*

Aksi Aktor	Reaksi Sistem
1. Klik tombol "*Start Game".*	
	2. Sistem akan menampilkan *form new game.*
3. Pemain mengisi nama.	
4. Klik "enter*".*	
	5. Sistem merekam data yang diisikan.

Aksi Aktor	Reaksi Sistem
	6. Sistem menampilkan *screen avatar*
7. Pemain mengganti bagian *avatar*	
	8. Sistem akan menyimpan hasil perubahan *avatar*
	9. Sistem menampilkan *screen* main *game*.

Post-condition : data pemain disimpan dan menampilkan tampilan utama *game*

Skenario alternatif 1 : pemain memilih kuliah lalu pilih STMIK IKMI, dan menjawab benar soal USM lebih dari 7, serta memilih menyerah saat lanjut kuliah.

Tabel 3. 2
Skenario Alternatif 1 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan *screen* dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “STMIK IKMI”	
	17. Sistem menampilkan *screen* dialog STMIK IKMI
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan *mini game* soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog diterima kuliah
	24. Sistem menampilkan *screen* dialog kuliah normal
25. Klik “enter”	
	26. Sistem menampilkan *screen mini game* ujian
	27. Sistem menampilkan soal Ujian
28. Pemain memilih jawaban soal ujian	
	29. Sistem menampilkan *mini game* soal ujian sampai 5 soal
30. Klik “enter”	
	31. Sistem menampilkan dialog pilihan lanjut kuliah.
32. Klik tombol “berhenti kuliah”	
	33. Sistem menampilkan dialog berhenti kuliah
	34. Sistem menampilkan *screen* tamat

Skenario alternatif 2 : pemain memilih kuliah lalu pilih STMIK IKMI, dan menjawab benar soal USM lebih dari 7, serta memilih cuti kuliah saat lanjut kuliah.

Tabel 3. 3
Skenario Alternatif 2 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “STMIK IKMI”	
	17. Sistem menampilkan screen dialog STMIK IKMI
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog diterima kuliah
	24. Sistem menampilkan screen dialog kuliah normal
25. Klik “enter”	
	26. Sistem menampilkan screen mini game ujian
	27. Sistem menampilkan soal Ujian
28. Pemain memilih jawaban soal ujian	
	29. Sistem menampilkan mini game soal ujian sampai 5 soal
30. Klik “enter”	
	31. Sistem menampilkan dialog pilihan lanjut kuliah.
32. Klik tombol “cuti kuliah”	
	33. Sistem menampilkan dialog cuti kuliah
	34. Sistem menampilkan screen tamat

Skenario alternatif 3 : pemain memilih kuliah lalu pilih STMIK IKMI, dan menjawab benar soal USM kurang dari 7, serta memilih menyerah dari *mini game*.

Tabel 3. 4
Skenario Alternatif 3 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “STMIK IKMI”	
	17. Sistem menampilkan screen dialog STMIK IKMI
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog tidak diterima kuliah
24. Klik “enter”	
	25. Sistem menampilkan pilihan untuk lanjut
	26. Sistem menampilkan soal Ujian
27. Klik “menyerah”	
	28. Sistem menampilkan screen tamat

Skenario alternatif 4 : pemain memilih kuliah lalu pilih STMIK IKMI, dan menjawab benar soal USM kurang dari 7, serta memilih cari kampus lain.

Tabel 3. 5
Skenario Alternatif 4 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	

Aksi Aktor	Reaksi Sistem
	15. Sistem menampilkan pilihan kampus
16. Klik “STMIK IKMI”	
	17. Sistem menampilkan screen dialog STMIK IKMI
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog tidak diterima kuliah
24. Klik “enter”	
	25. Sistem menampilkan pilihan untuk lanjut
	26. Sistem menampilkan soal Ujian
27. Klik “Cari kampus lain”	
	28. Sistem menampilkan pilihan kampus

Skenario alternatif 5 : pemain memilih kuliah lalu pilih ISB, dan menjawab benar soal USM lebih dari 7, serta memilih menyerah saat lanjut kuliah.

Tabel 3. 6
Skenario Alternatif 5 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “ISB”	
	17. Sistem menampilkan screen dialog ISB
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	23.
	24. Sistem menampilkan dialog diterima kuliah
	25. Sistem menampilkan screen dialog kuliah normal
26. Klik “enter”	

Aksi Aktor	Reaksi Sistem
	27. Sistem menampilkan screen mini game ujian
	28. Sistem menampilkan soal Ujian
29. Pemain memilih jawaban soal ujian	
	30. Sistem menampilkan mini game soal ujian sampai 5 soal
31. Klik “enter”	
	32. Sistem menampilkan dialog pilihan lanjut kuliah.
33. Klik tombol “berhenti kuliah”	
	34. Sistem menampilkan dialog berhenti kuliah
	35. Sistem menampilkan screen tamat

Skenario alternatif 6 : pemain memilih kuliah lalu pilih ISB, dan menjawab benar soal USM lebih dari 7, serta memilih cuti kuliah saat lanjut kuliah.

Tabel 3. 7
Skenario Alternatif 6 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “ISB”	
	17. Sistem menampilkan screen dialog ISB
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog diterima kuliah
	24. Sistem menampilkan screen dialog kuliah normal
25. Klik “enter”	
	26. Sistem menampilkan screen mini game ujian
	27. Sistem menampilkan soal Ujian
28. Pemain memilih jawaban soal ujian	
	29. Sistem menampilkan mini game soal ujian sampai 5 soal

Aksi Aktor	Reaksi Sistem
30. Klik “enter”	
	31. Sistem menampilkan dialog pilihan lanjut kuliah.
32. Klik tombol “cuti kuliah”	
	33. Sistem menampilkan dialog cuti kuliah
	34. Sistem menampilkan screen tamat

Skenario alternatif 7 : pemain memilih kuliah lalu pilih ISB, dan menjawab benar soal USM kurang dari 7, serta memilih menyerah dari *mini game*.

Tabel 3. 8
Skenario Alternatif 7 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “STMIK IKMI”	
	17. Sistem menampilkan screen dialog STMIK IKMI
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog tidak diterima kuliah
24. Klik “enter”	
	25. Sistem menampilkan pilihan untuk lanjut
	26. Sistem menampilkan soal Ujian
27. Klik “menyerah”	
	28. Sistem menampilkan screen tamat

Skenario alternatif 8 : pemain memilih kuliah lalu pilih ISB, dan menjawab benar soal USM kurang dari 7, serta memilih cari kampus lain.

Tabel 3. 9
Skenario Alternatif 8 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “ISB”	
	17. Sistem menampilkan screen dialog ISB
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog tidak diterima kuliah
24. Klik “enter”	
	25. Sistem menampilkan pilihan untuk lanjut
	26. Sistem menampilkan soal Ujian
27. Klik “Cari kampus lain”	
	28. Sistem menampilkan pilihan kampus

Skenario alternatif 9 : pemain memilih kuliah lalu pilih UNPAT, dan menjawab benar soal USM lebih dari 7, serta memilih menyerah saat lanjut kuliah.

Tabel 3. 10
Skenario Alternatif 9 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	

Aksi Aktor	Reaksi Sistem
	15. Sistem menampilkan pilihan kampus
16. Klik “UNPAT”	
	17. Sistem menampilkan screen dialog UNPAT
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog diterima kuliah
	24. Sistem menampilkan screen dialog kuliah normal
25. Klik “enter”	
	26. Sistem menampilkan screen mini game ujian
	27. Sistem menampilkan soal Ujian
28. Pemain memilih jawaban soal ujian	
	29. Sistem menampilkan mini game soal ujian sampai 5 soal
30. Klik “enter”	
	31. Sistem menampilkan dialog pilihan lanjut kuliah.
32. Klik tombol “berhenti kuliah”	
	33. Sistem menampilkan dialog berhenti kuliah
	34. Sistem menampilkan screen tamat

Skenario alternatif 10 : pemain memilih kuliah lalu pilih UNPAT, dan menjawab benar soal USM lebih dari 7, serta memilih cuti kuliah saat lanjut kuliah.

Tabel 3. 11
Skenario Alternatif 10 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “UNPAT”	
	17. Sistem menampilkan screen dialog UNPAT
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM

Aksi Aktor	Reaksi Sistem
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog diterima kuliah
	24. Sistem menampilkan screen dialog kuliah normal
25. Klik “enter”	
	26. Sistem menampilkan screen mini game ujian
	27. Sistem menampilkan soal Ujian
28. Pemain memilih jawaban soal ujian	
	29. Sistem menampilkan mini game soal ujian sampai 5 soal
30. Klik “enter”	
	31. Sistem menampilkan dialog pilihan lanjut kuliah.
32. Klik tombol “cuti kuliah”	
	33. Sistem menampilkan dialog cuti kuliah
	34. Sistem menampilkan screen tamat

Skenario alternatif 11 : pemain memilih kuliah lalu pilih UNPAT, dan menjawab benar soal USM kurang dari 7, serta memilih menyerah dari *mini game*.

Tabel 3. 12
Skenario Alternatif 11 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “UNPAT”	
	17. Sistem menampilkan screen dialog UNPAT
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog tidak diterima kuliah
24. Klik “enter”	
	25. Sistem menampilkan pilihan untuk lanjut

Aksi Aktor	Reaksi Sistem
	26. Sistem menampilkan soal Ujian
27. Klik “menyerah”	
	28. Sistem menampilkan screen tamat

Skenario alternatif 12 : pemain memilih kuliah lalu pilih UNPAT, dan menjawab benar soal USM kurang dari 7, serta memilih cari kampus lain.

Tabel 3. 13
Skenario Alternatif 12 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “UNPAT”	
	17. Sistem menampilkan screen dialog UNPAT
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog tidak diterima kuliah
24. Klik “enter”	
	25. Sistem menampilkan pilihan untuk lanjut
	26. Sistem menampilkan soal Ujian
27. Klik “Cari kampus lain”	
	28. Sistem menampilkan pilihan kampus

Skenario alternatif 13 : pemain memilih kuliah lalu pilih Maramatha, dan menjawab benar soal USM lebih dari 7, serta memilih menyerah saat lanjut kuliah.

Tabel 3. 14
Skenario Alternatif 13 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	

Aksi Aktor	Reaksi Sistem
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “Maramatha”	
	17. Sistem menampilkan screen dialog Maramatha
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog diterima kuliah
	24. Sistem menampilkan screen dialog kuliah normal
25. Klik “enter”	
	26. Sistem menampilkan screen mini game ujian
	27. Sistem menampilkan soal Ujian
28. Pemain memilih jawaban soal ujian	
	29. Sistem menampilkan mini game soal ujian sampai 5 soal
30. Klik “enter”	
	31. Sistem menampilkan dialog pilihan lanjut kuliah.
32. Klik tombol “berhenti kuliah”	
	33. Sistem menampilkan dialog berhenti kuliah
	34. Sistem menampilkan screen tamat

Skenario alternatif 14 : pemain memilih kuliah lalu pilih Maramatha, dan menjawab benar soal USM lebih dari 7, serta memilih cuti kuliah saat lanjut kuliah.

Tabel 3. 15
Skenario Alternatif 14 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “Maramatha”	

Aksi Aktor	Reaksi Sistem
	17. Sistem menampilkan screen dialog Maramatha
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog diterima kuliah
	24. Sistem menampilkan screen dialog kuliah normal
25. Klik “enter”	
	26. Sistem menampilkan screen mini game ujian
	27. Sistem menampilkan soal Ujian
28. Pemain memilih jawaban soal ujian	
	29. Sistem menampilkan mini game soal ujian sampai 5 soal
30. Klik “enter”	
	31. Sistem menampilkan dialog pilihan lanjut kuliah.
32. Klik tombol “cuti kuliah”	
	33. Sistem menampilkan dialog cuti kuliah
	34. Sistem menampilkan screen tamat

Skenario alternatif 15 : pemain memilih kuliah lalu pilih Maramatha, dan menjawab benar soal USM kurang dari 7, serta memilih menyerah dari *mini game*.

Tabel 3. 16
Skenario Alternatif 15 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “Maramatha”	
	17. Sistem menampilkan screen dialog Maramatha
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	

Aksi Aktor	Reaksi Sistem
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog tidak diterima kuliah
24. Klik “enter”	
	25. Sistem menampilkan pilihan untuk lanjut
	26. Sistem menampilkan soal Ujian
27. Klik “menyerah”	
	28. Sistem menampilkan screen tamat

Skenario alternatif 16 : pemain memilih kuliah lalu pilih Maramatha, dan menjawab benar soal USM kurang dari 7, serta memilih cari kampus lain.

Tabel 3. 17
Skenario Alternatif 16 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kuliah”	
	13. Sistem menampilkan screen dialog kuliah
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan kampus
16. Klik “Maramatha”	
	17. Sistem menampilkan screen dialog Maramatha
18. Pemain lanjut membaca cerita dengan klik “enter”	
	19. Sistem menampilkan mini game soal USM
20. Pemain klik jawaban USM	
	21. Sistem menampilkan sampai 10 soal
22. Klik “enter”	
	23. Sistem menampilkan dialog tidak diterima kuliah
24. Klik “enter”	
	25. Sistem menampilkan pilihan untuk lanjut
	26. Sistem menampilkan soal Ujian
27. Klik “Cari kampus lain”	
	28. Sistem menampilkan pilihan kampus

Skenario alternatif 17 : pemain memilih pekerjaan, benar dalam pengetikan surat lamaran, dan menjawab benar soal Psikotest lebih dari 7, serta menang dalam *mini game* pilihan pemain

Tabel 3. 18
Skenario Alternatif 17 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kerja”	
	13. Sistem menampilkan screen dialog kerja
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan pekerjaan
16. Klik pilihan pekerjaan	
	17. Sistem menampilkan mini game surat lamaran
18. Pemain input teks surat lamaran	
	19. Sistem menampilkan screen kirim surat lamaran
	20. Sistem menampilkan dialog kerja
21. Klik “enter”	
	22. Sistem menampilkan mini game Psikotest
	23. Sistem menampilkan soal psikotest
24. Pemain klik jawaban psikotest	
	25. Sistem menampilkan sampai 10 soal
26. Klik “enter”	
	27. Sistem menampilkan dialog diterima kerja
28. Klik “enter”	
	29. Sistem menampilkan screen mini game sesuai pilihan pemain
30. Pemain memilih gambar jawaban	
	31. Sistem menampilkan soal mini game sesuai pilihan pemain sampai 10
32. Klik “enter”	
	33. Sistem menampilkan dialog happy ending
	34. Sistem menampilkan screen tamat

Skenario alternatif 18 : pemain memilih pekerjaan, benar dalam pengetikan surat lamaran, dan menjawab benar soal Psikotest lebih dari 7, serta kalah dalam *mini game* sesuai pilihan pemain

Tabel 3. 19
Skenario Alternatif 18 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kerja”	
	13. Sistem menampilkan screen dialog kerja
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan pekerjaan
16. Klik pilihan pekerjaan	
	17. Sistem menampilkan mini game surat lamaran
18. Pemain input teks surat lamaran	
	19. Sistem menampilkan screen kirim surat lamaran
	20. Sistem menampilkan dialog kerja
21. Klik “enter”	
	22. Sistem menampilkan mini game Psikotest
	23. Sistem menampilkan soal psikotest
24. Pemain klik jawaban psikotest	
	25. Sistem menampilkan sampai 10 soal
26. Klik “enter”	
	27. Sistem menampilkan dialog diterima kerja
28. Klik “enter”	
	29. Sistem menampilkan screen mini game sesuai pilihan pemain
30. Pemain memilih gambar jawaban	
	31. Sistem menampilkan soal mini game sesuai pilihan pemain sampai 10
32. Klik “enter”	
	33. Sistem menampilkan dialog sad ending
	34. Sistem menampilkan screen tamat

Skenario alternatif 19 : pemain memilih pekerjaan, benar dalam pengetikan surat lamaran, dan menjawab benar soal Psikotest kurang dari 7, serta memilih menyerah dalam pilihan kerja

Tabel 3. 20
Skenario Alternatif 19 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kerja”	
	13. Sistem menampilkan screen dialog kerja
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan pekerjaan
16. Klik pilihan pekerjaan	
	17. Sistem menampilkan mini game surat lamaran
18. Pemain input teks surat lamaran	
	19. Sistem menampilkan screen kirim surat lamaran
	20. Sistem menampilkan dialog kerja
21. Klik “enter”	
	22. Sistem menampilkan mini game Psikotest
	23. Sistem menampilkan soal psikotest
24. Pemain klik jawaban psikotest	
	25. Sistem menampilkan sampai 10 soal
26. Klik “enter”	
	27. Sistem menampilkan dialog tidak diterima kerja
28. Klik “enter”	
	29. Sistem menampilkan screen pilihan lanjut kerja
30. Klik “menyerah”	
	31. Sistem menampilkan screen tamat

Skenario alternatif 20 : pemain memilih pekerjaan, benar dalam pengetikan surat lamaran, dan menjawab benar soal Psikotest kurang dari 7, serta memilih cari pekerjaan lain

Tabel 3. 21
Skenario Alternatif 20 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	

Aksi Aktor	Reaksi Sistem
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kerja”	
	13. Sistem menampilkan screen dialog kerja
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan pekerjaan
16. Klik pilihan pekerjaan	
	17. Sistem menampilkan mini game surat lamaran
18. Pemain input teks surat lamaran	
	19. Sistem menampilkan screen kirim surat lamaran
	20. Sistem menampilkan dialog kerja
21. Klik “enter”	
	22. Sistem menampilkan mini game Psikotest
	23. Sistem menampilkan soal psikotest
24. Pemain klik jawaban psikotest	
	25. Sistem menampilkan sampai 10 soal
26. Klik “enter”	
	27. Sistem menampilkan dialog tidak diterima kerja
28. Klik “enter”	
	29. Sistem menampilkan screen pilihan lanjut kerja
30. Klik “cari pekerjaan lain”	
	31. Sistem menampilkan pilhan pekerjaan

Skenario alternatif 21 : pemain memilih pekerjaan, salah dalam pengetikan surat lamaran kemudian memilih menyerah, dan menjawab benar soal Psikotest lebih dari 7, serta kalah dalam *mini game* sesuai pilihan pemain

Tabel 3. 22
Skenario Alternatif 21 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kerja”	
	13. Sistem menampilkan screen dialog kerja
14. Pemain lanjut membaca cerita dengan klik “enter”	

Aksi Aktor	Reaksi Sistem
	15. Sistem menampilkan pilihan pekerjaan
16. Klik pilihan pekerjaan	
	17. Sistem menampilkan mini game surat lamaran
18. Pemain input teks surat lamaran	
	19. Sistem menampilkan salah dalam pengetikan
20. Klik “menyerah”	
	21. Sistem menampilkan screen kirim surat lamaran
	22. Sistem menampilkan dialog kerja
23. Klik “enter”	
	24. Sistem menampilkan mini game Psikotest
	25. Sistem menampilkan soal psikotest
26. Pemain klik jawaban psikotest	
	27. Sistem menampilkan sampai 10 soal
28. Klik “enter”	
	29. Sistem menampilkan dialog diterima kerja
30. Klik “enter”	
	31. Sistem menampilkan screen mini game sesuai pilihan pemain
32. Pemain memilih gambar jawaban	
	33. Sistem menampilkan soal mini game sesuai pilihan pemain sampai 10
34. Klik “enter”	
	35. Sistem menampilkan dialog sad ending
	36. Sistem menampilkan screen tamat

Skenario alternatif 22 : pemain memilih kerja lalu pilih pilihan pemain, salah dalam pengetikan surat lamaran kemudian memilih menyerah, dan menjawab benar soal Psikotest lebih dari 7, serta kalah dalam *mini game* pilihan pemain

Tabel 3. 23
Skenario Alternatif 22 *use case Start Game*

Aksi Aktor	Reaksi Sistem
1 – 9 Sama seperti skenario normal	
10. Pemain membaca lanjutan cerita dengan klik “enter”.	
	11. Sistem menampilkan pilihan lanjut
12. Klik tombol “Kerja”	
	13. Sistem menampilkan screen dialog kerja
14. Pemain lanjut membaca cerita dengan klik “enter”	
	15. Sistem menampilkan pilihan pekerjaan

Aksi Aktor	Reaksi Sistem
16. Klik “Pilihan pemain”	
	17. Sistem menampilkan mini game surat lamaran
18. Pemain input teks surat lamaran	
	19. Sistem menampilkan salah dalam pengetikan
20. Klik “coba lagi”	
	21. Sistem menampilkan mini game surat lamaran

2. *Use case :* Buat *Avatar*

   Aktor : Pemain

   Pre-condition : pemain masuk ke form *start game*

   Skenario normal

Tabel 3. 24
Skenario *use case Avatar*

Aksi Aktor	Reaksi Sistem
1. Pemain mengisi *Form input* nama	
2. Klik “*Enter*”	
	3. Sistem merekam data pemain
	4. Sistem menampilkan *screen avatar*
5. Buat / pilih bagian untuk *avatar*	
6. Klik tombol “kembali”.	
	7. *Avatar* tersimpan
	8. Kembali ke *form* utama *game*

Post-condition : *avatar* disimpan dan menampilkan *form* utama *game*

3. *Use case :* Ubah *Avatar*

   Aktor : Pemain

   Pre-condition : pemain melanjutkan permainan pada “*load game”*

   Skenario normal

Tabel 3. 25
Skenario *use case Avatar*

Aksi Aktor	Reaksi Sistem
1. Klik “*Avatar*”	
	2. Sistem menampilkan *screen avatar*
3. Buat / pilih bagian untuk *avatar*	
4. Klik tombol “kembali”.	
	5. *Avatar* tersimpan
	6. Kembali ke *form* utama *game*

4. *Use case : Load Game*

Aktor : Pemain

Pre-condition : Pemain berada pada tampilan *main menu*

Definsi : Pemain melanjutkan *game* berdasarkan *game* yang sudah disimpan

Skenario normal

Tabel 3. 26
Skenario *use case Load Game*

Aksi Aktor	Reaksi Sistem
1. Klik tombol "*Load Game".*	
	2. Sistem akan melakukan validasi apakah terdapat *game* yang di-*save.*
	3. Sistem akan menampilkan *screen load game*
4. Pilih data *save game* yang ingin dimainkan	
	5. Sistem akan mengambil data pemain.
	6. Sistem menampilkan *form* utama.

Post-condition : Pemain berada pada tampilan utama dan dapat melanjutkan *game*

Skenario alternatif tidak ada *game* yang tersimpan

Tabel 3. 27
Skenario alternatif *use case Load Game*

Aksi Aktor	Reaksi Sistem
1. Klik tombol "*Load Game".*	
	2. Sistem akan melakukan validasi. apakah terdapat *game* yang di-*save*
	3. Sistem akan menampilkan *form load game* yang kosong.
4. Klik tombol "return"	
	5. Sistem akan menampilkan *form main menu*

Post-condition : Pemain kembali ke *form main menu*

5. *Use case : About Game*

Aktor : Pemain

Pre-condition : Pemain berada pada *form main menu*

Definsi : Pemain membaca cara memainkan *game*

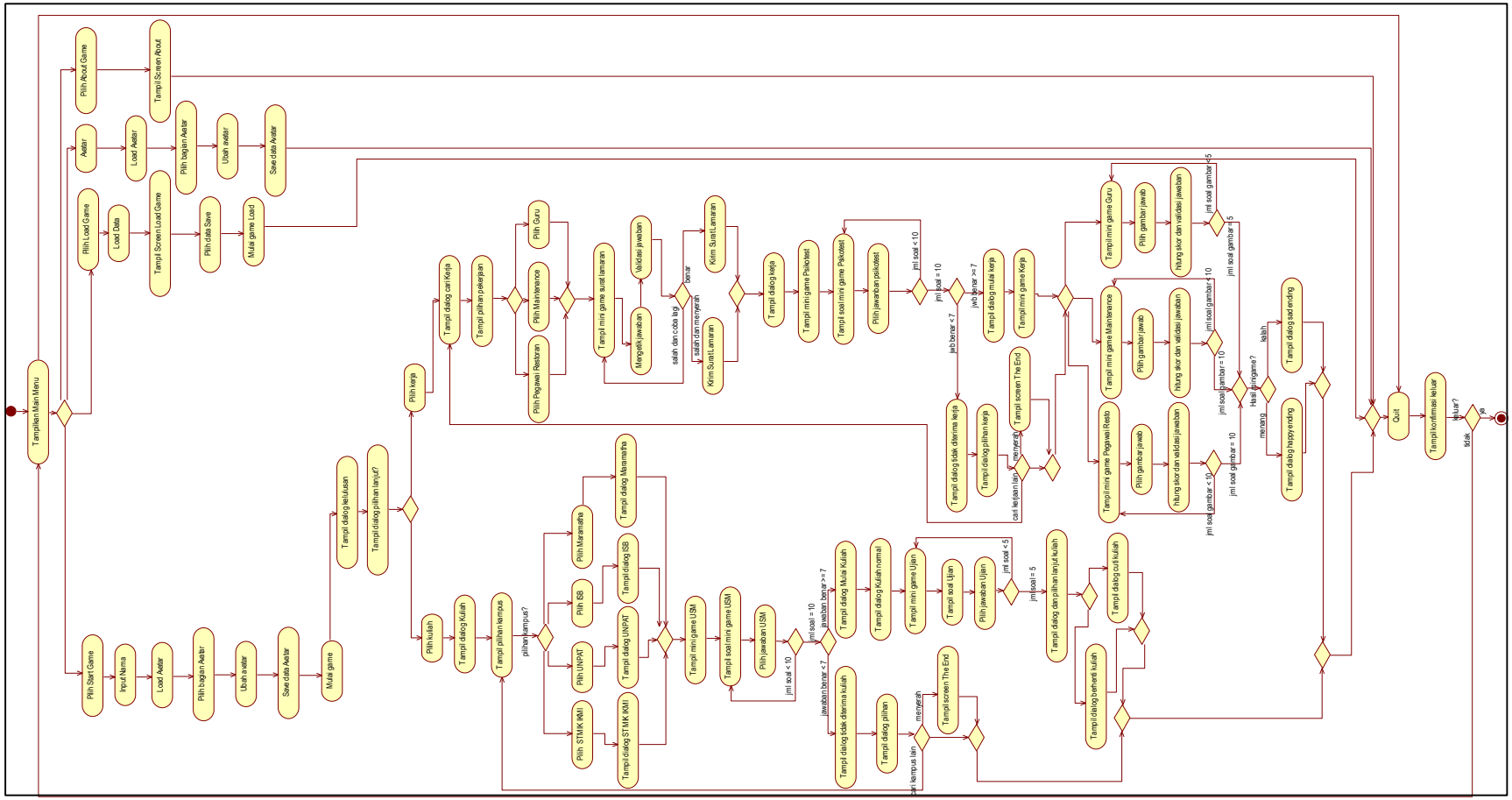
Skenario normal

Tabel 3. 28
Skenario use case About Game

Aksi Aktor	Reaksi Sistem
1. Klik tombol "*About Game".*	
	2. Sistem menampilkan *form about game*
3. Klik tombol "*OK*".	
	4. Sistem kembali *form main menu*

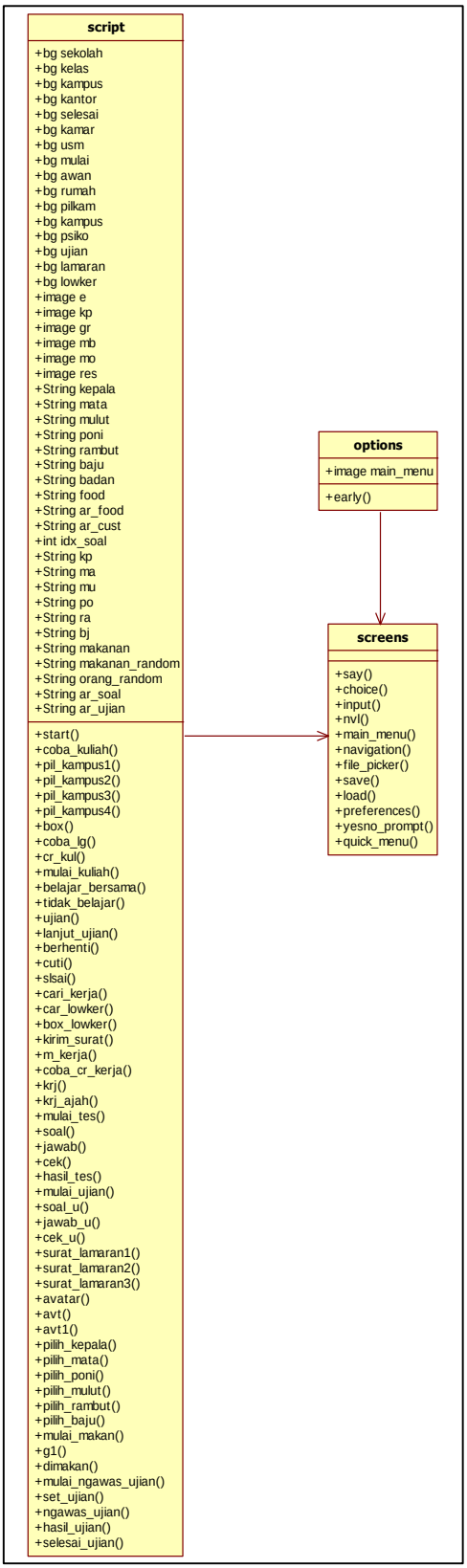
Post-condition : Pemain berada pada *form main menu*

3.3.3 Activity Diagram



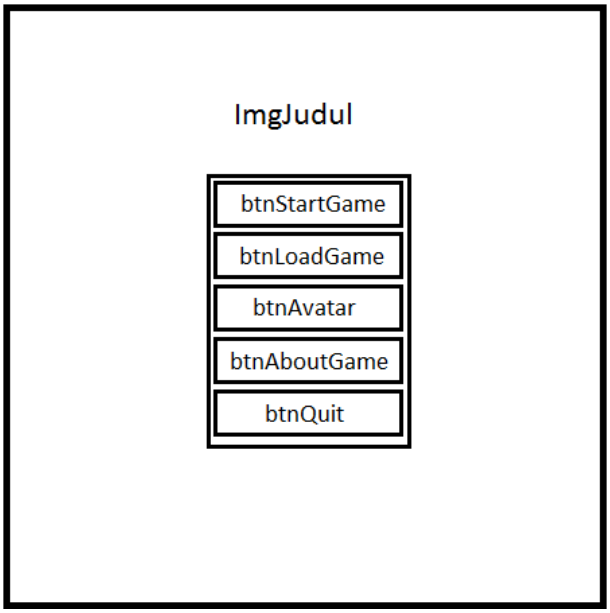
Gambar 3.2
Activity Diagram Visual Novel After Graduate

3.3.4 Class Diagram



Gambar 3.3
Class Diagram visual novel After Graduate

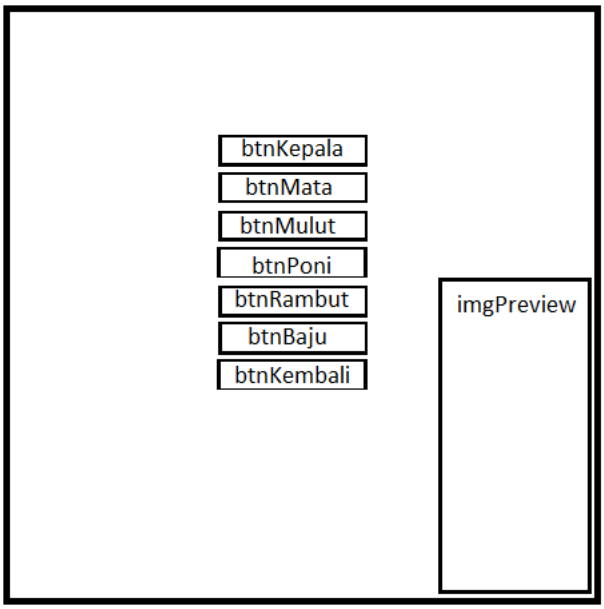
3.3.5 Rancangan Antar Muka



Gambar 3. 4
Rancangan *Main Menu*

Penjelasan mengenai komponen-komponen pada Gambar 3.4 sebagai berikut :

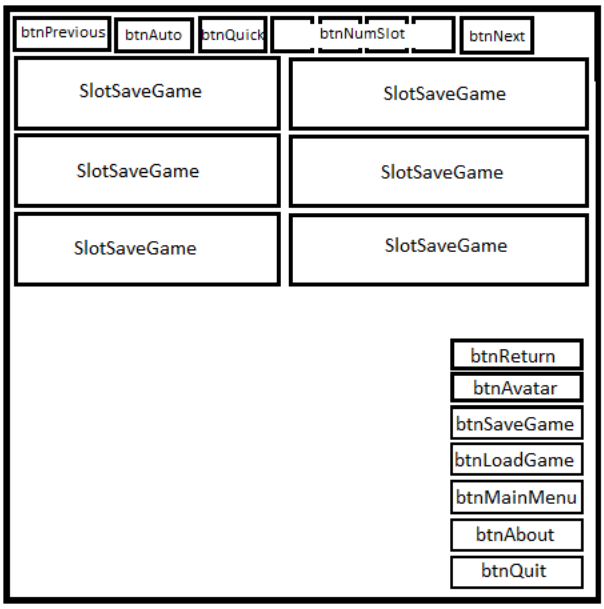
1. imgJudul : berfungsi untuk menampilkan gambar judul dari *game After Graduate.*
2. btnStartGame : berfungsi untuk memulai *game.*
3. btnLoadGame : berfungsi untuk melanjutkan *game* yang sudah tersimpan sebelumnya.
4. btnAvatar : berfungsi untuk menampilkan *screen avatar.*
5. btnAboutGame : berfungsi untuk menampilakan *screen about.*
6. btnQuit : berfungsi untuk keluar dari *game.*



Gambar 3. 5
Rancangan *Avatar*

Penjelasan mengenai komponen-komponen pada Gambar 3.5 sebagai berikut :

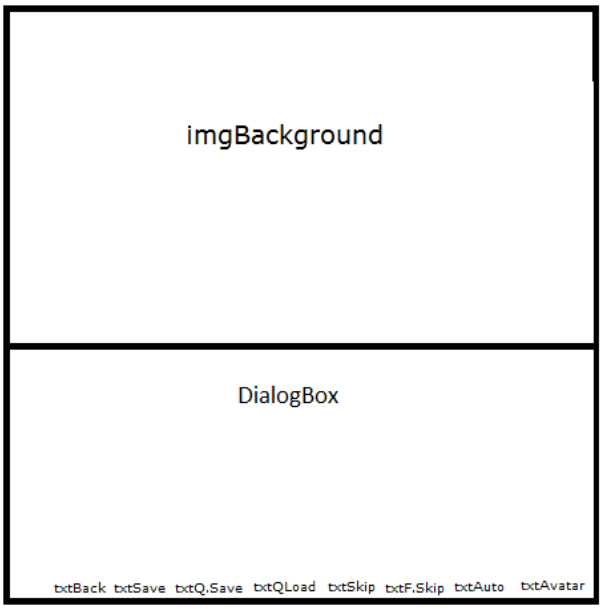
1. btnKepala : berfungsi untuk mengganti kepala dari *avatar.*
2. btnMata : berfungsi untuk mengganti mata dari *avatar.*
3. btnMullut : berfungsi untuk mengganti mulut dari *avatar*.
4. btnPoni : berfungsi untuk mengganti poni dari *avatar.*
5. btnRambut : berfungsi untuk mengganti rambut dari *avatar.*
6. btnBaju : berfungsi untuk mengganti baju dari *avatar.*
7. imgPreview : berfungsi untuk menampilkan *preview* dari *avatar.*



Gambar 3. 6
Rancangan *Load Game*

Penjelasan mengenai komponen-komponen pada Gambar 3.6 sebagai berikut :

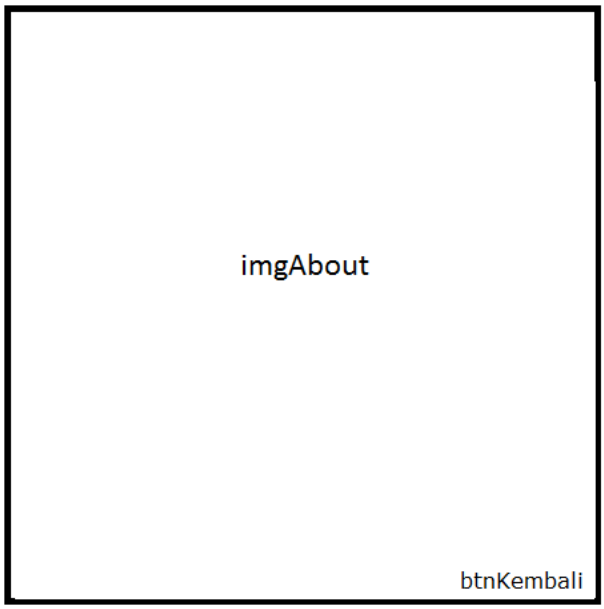
1. btnPrevious : berfungsi untuk menampilkan *slot* sebelumnya.
2. btnAuto : berfungsi untuk menampilkan hasil penyimpanan *game* secara otomatis.
3. btnQuick : berfungsi untuk menampilkan hasil penyimpanan *game* pada Quick Save.
4. btnNumSlot : berfungsi untuk menampilkan nomor *slot* penyimpanan *game.*
5. btnNext : berfungsi untuk menampilkan slot selanjutnya.
6. slotSaveGame : berfungsi sebagai tempat meyimpan *game.*
7. btnReturn : berfungsi untuk kembali.
8. btnAvatar : berfungsi untuk masuk ke *screen avatar.*
9. btnSaveGame : berfungsi untuk menyimpan *game.*
10. btnLoadGame : berfungsi untuk melanjutkan *game* yang sudah tersimpan sebelumnya.
11. btnMainMenu : berfungsi untuk kembali ke *screen main menu.*
12. btnAbout : berfungsi untuk menampilakan *screen about.*
13. btnQuit : berfungsi untuk keluar dari *game.*



Gambar 3. 7
Rancangan Main *Game*

Penjelasan mengenai komponen-komponen pada Gambar 3.7 sebagai berikut :

1. imgBackground : berfungsi untuk menampilkan gambar *background.*
2. dialogBox : berfungsi untuk menampilkan teks cerita.
3. txtBack : berfungsi untuk kembali ke cerita sebelumnya.
4. txtSave: berfungsi untuk menampilkan *screen save game.*
5. txtQ.Save : berfungsi untuk menyimpan *game* secara otomatis dan tersimpan di dalam slot *quick.*
6. txtQLoad : berfungsi untuk menampilkan hasil penyimpnan secara otomatis.
7. txtSkip : berfungsi untuk melewati ke teks cerita selanjutnya.
8. txtF.Skip : berfungsi untuk melewati cerita langsung kepada pilhan selanjutnya..
9. txtAuto : berfungsi untuk melanjutkan ke teks cerita selanjutnya.
10. txtAvatar : berfungsi untuk menampilkan *screen avatar*.



Gambar 3. 8
Rancangan *About*

Penjelasan mengenai komponen-komponen pada Gambar 3.8 sebagai berikut :

1. imgAbout : berfungsi untuk menampilkan gambar *about game.*
2. btnKembali : berfungsi untuk kembali.

BAB IV
IMPLEMETASI DAN PENGUJIAN PERANGKAT LUNAK

4.1 Konfigurasi Sistem Komputer

Konfigurasi sistem komputer dimaksudkan untuk memberi gambaran perangkat keras yang digunakan untuk membuat *visual novel After Graduate* yang dikembangkan pada Tugas Akhir ini. Adapun spesifikasi perangkat kerasnya adalah sebagai berikut :

Tabel 4. 1
Spesifiksi Perangkat Keras Komputer

Konfigurasi Perangkat	Spesifikasi
Processor	AMD A8-6410 APU 2.0GHz
Memory	4 GB
Video Graphics Array (VGA)	AMD Radeon R4 Graphics
Harddisk	500 GB
Monitor	Resolusi 1366x768

4.2 Pemilihan Perangkat Lunak

Selain perangkat keras yang sudah disebutkan, digunakan pula perangkat lunak yang mendukung pengembangan *game* ini. Perangkat lunak tersebut yaitu :

1. Windows 8.1

   Sistem Operasi yang digunakan dalam pengerjaan *game* ini adalah Windows 8.1. Sistem operasi ini digunakan karena dapat mendukung hampir semua *software* yang ada.

2. Ren'Py 6.99.6.739

   Ren'Py merupakan *visual novel engine* yang menggunakan bahasa pemrograman python. Ren'Py menggunakan fungsi-fungsi yang memudahkan dalam pembuatan *game* ini.

4.3 Tampilan Antar Muka

Berikut adalah hasil *screenshot* dari seluruh kemungkinan *interface* yang ada di dalam *game* ini :

1. Menu Utama

Gambar 4. 1
Tampilan halaman menu utama

Menu utama merupakan tampilan awal saat *game* dijalankan. Terdapat lima tombol pada menu utama, yaitu tombol *start game,* tombol *load game,* tombol *avatar,* tombol *about game,* dan tombol *quit.* Penjelasan mengenai tombol-tombol tersebut adalah sebagai berikut :

a. Tombol *start game*

Tombol ini akan membawa pemain masuk ke dalam *game*, dan menampilkan *screen* yang akan diisi oleh pemain, seperti pada gambar 4.2.

Gambar 4. 2
Tampilan *screen input* nama pemain

b. Tombol *load* game

Tombol ini akan membawa pemain masuk ke *screen load* dan dapat memilih hasil simpan *game* sebelumnya. *Screen load* ditampilkan pada gambar 4.3.

Disediakan beberapa *slot* penyimpanan game, dengan klik pada *empty slot* yang tersedia, maka *game* otomatis akan tersimpan.

Gambar 4. 3
Tampilan *screen load*

c. Tombol *avatar*

Tombol ini akan membawa pemain pada *screen avatar* sehingga pemain dapat mengubah *avatar* sesuai denga keinginannya. *Screen avatar* ditampilkan pada gambar 4.4.

Gambar 4. 4
Tampilan *screen avatar*

Pada *screen* tersebut pemain dapat mengubah bagian-bagian *avatar* sesuai keinginan. Bagian-bagian tersebut seperti kepala, mata, mimik wajah, poni, rambut, juga baju yang dipakai. Pemilihan/penggantian bagian dapat dilakukan dengan menekan tombol-tombol yang terdapat pada *screen* tersebut. Sedangkan tombol kembali, digunakan untuk keluar dari *screen avatar.*

d. Tombol *about game*

Tombol ini merupakan penjelasan mengenai versi dari *game visual novel After Graduate* dan cara memainkan *game* ini, dapat dilihat dalam gambar 4.5.

Gambar 4. 5
Tampilan *screen about game*

2. *Screen* dialog kelulusan

Screen ini menunjukkan dialog saat pemain lulus dari SMA, pada gambar 4.6. Pada *screen* ini, terdapat tombol navigasi di kanan bawah yang memiliki fungsi, diantaranya :

a. *back* : tombol ini berfungsi untuk kembali ke cerita sebelumnya.

b. *save*: tombol ini berfungsi untuk menampilkan *screen save game.*

c. *Q.Save* : tombol ini berfungsi untuk menyimpan *game* secara otomatis dan tersimpan di dalam slot *quick.*

d. *QLoad* : tombol ini berfungsi untuk menampilkan hasil penyimpnan secara otomatis.

e. *Skip* : tombol berfungsi untuk melewati ke teks cerita selanjutnya.

f. *F.Skip* : tombol ini berfungsi untuk melewati cerita langsung kepada pilhan selanjutnya.

g. *Auto* : tombol ini berfungsi untuk melanjutkan ke teks cerita selanjutnya.

h. *Avatar* : tombol ini berfungsi untuk menampilkan *screen avatar*.

Gambar 4. 6
Tampilan *screen* dialog kelulusan

3. Pada gambar 4.7 terdapat *screen pilihan* lanjut

Gambar 4. 7
Tampilan pilihan lanjut

Pada *screen* ini terdapat dua tombol yang harus dipilih oleh pemain. Kedua tombol tersebut yaitu :

a. Tombol kuliah

Ketika tombol ini diklik, maka *game* akan masuk pada dialog kuliah untuk lanjutan dari cerita. Dalam cerita, pemain akan menemui pilihan dari kampus, *screen* pilihan seperti pada gambar 4.8.

Gambar 4. 8
Tampilan pilihan kampus

Jika pemain sudah memilih salah satu dari kampus yang tersedia, maka pemain akan dihadapkan dengan *mini game* USM dimana pemain harus menjawab beberapa pertanyaan. Pertanyaan terlihat pada gambar 4.9 dan pilihan jawaban pada gambar 4.10

Gambar 4. 9
Tampilan *mini game* soal USM

Gambar 4. 10
Tampilan jawaban soal USM

Dengan menjawab beberapa pertanyaan, dan jika hasil benar kurang dari tujuh nomor, maka pemain dinyatakan tidak lulus dalam USM. Kemudian pemain akan dihadapkan dengan pilihan untuk melanjutkan cerita. Dialog pilihan lanjut seperti pada gambar 4.11, untuk pilihan untuk melanjutkan pada gambar 4.12.

Gambar 4. 11
Tampilan dialog tidak lulus USM

Gambar 4. 12
Tampilan *screen* pilihan lanjut cerita

Untuk hasil jawaban benar yang lebih dari tujuh, maka pemain dinyatakan lulus dari USM. Kemudian pemain akan dihadapkan dengan *mini game* ujian. Pada *mini game* ujian, pemain harus menjawab beberapa pertanyaan.

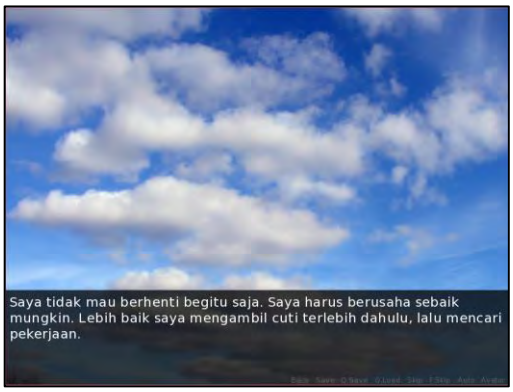
Gambar 4. 13
Tampilan *screen* diterima kuliah

Gambar 4. 14
Tampiln *mini game* soal Ujian

Gambar 4. 15
Tampilan jawaban soal ujian

Setelah menghadapi ujian , dan melanjutkan cerita, maka akan tampil pilihan untuk melanjutkan cerita seperti pada gambar 4.16. Jika sudah memilih jawaban, maka akan ditamilkan dialog lanjut seperti pada gambar 4.17.

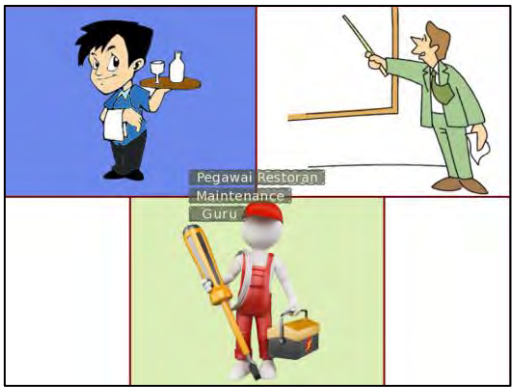
Gambar 4. 16
Tampilan pilihan lanjut cerita



Gambar 4. 17
Tampilan dialog setelah pilih cuti kuliah & mencari kerja

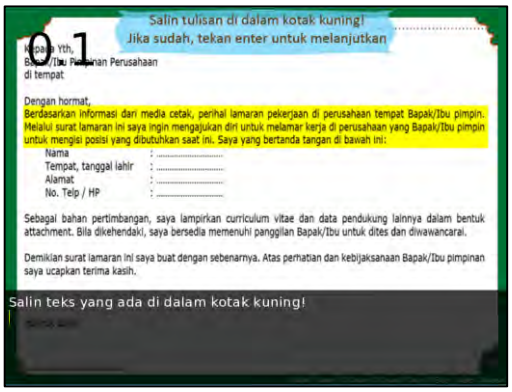
b. Tombol kerja

Pada s*creen* ini pemain akan memainkan simulasi dari pekerjaan yang dipilih. Beberapa contoh pekerjaan yang ada pada cerita ini adalah pegawai restoran, maintenance, dan guru.



Gambar 4. 18
Tampilan tombol pilihan pekerjaan

Sebelum memainkan simulasi, pemain diharuskan memainkan *mini game* surat lamaran, seperti pada gambar 4.19.



Gambar 4. 19
Tampilan *mini game* surat lamaran

Setelah memainkan *mini game* surat lamaran tersebut, pemain harus memainkan *mini game* psikotes gambar 4.20, dan menjawab soal psikotes tersebut pada gambar 4.21 dengan klik tombol jawaban yang tersedia sesuai dengan soal yang tertera.

Gambar 4. 20
Tampilan soal *mini game* psikotes

Gambar 4. 21
Tampilan tombol jawaban psikotes

Gambar 4.22 menunjukkan salah satu contoh simulasi pekerjaan sebagai pegawai restoran cepat saji. Pada *screen* ini pemain harus menyajikan makanan dengan cara memilih gambar sesuai dengan permintaan pelanggan pada gambar.

Gambar 4. 22
Tampilan *mini game* pekerjaan pegawai restoran

Jika pemain telah menyelesaikan *game* maka akan muncul *screen* tamat seperti pada gambar 4.23.

Gambar 4. 23
Tampian *screen* tamat

4.4 Pengujian

4.4.1 Kondisi Pengujian

Pengujian *game visual novel* ini menggunakan metode *black box* yang berfokus melakukan pengujian melalui tampilan perangkat lunak. Pengujian akan menguji fungsi – fungsi dari tombol dan navigasi yang ada pada *game*, apakah sudah berjalan sebagaimana mestinya atau belum dengan melihat hasil dari masing – masing percobaan pengujian. Fungsi – fungsi yang akan diuji dalam pengujian adalah sebagai berikut :

1. Kontrol dan navigasi halaman Menu Utama
2. Navigasi *avatar*
3. Navigasi *start game*

4. Navigasi pilihan setelah kelulusan

5. Pengujian *mini game* Kuliah

6. Pengujian *mini game* Kerja

7. Navigasi *about game*

8. Navigasi *quit*

4.4.2 Pelaksanaan Pengujian

1. Kontrol dan Navigasi halaman Menu Utama

Tabel 4. 2
Tabel pengujian kontrol dan navigasi halamam menu utama

No	Jenis Pengujian	Ekspektasi	Output	Ket
1	Menu utama pada saat *first run*	Menu utama ditampilkan	Menu utama ditampilkan	OK
2	Tombol *start game* diklik	*Screen* pengisian nama pemain ditampilkan	*Screen* pengisian nama pemain ditampilkan	OK
3	Tombol *load game* diklik	*Screen load game* ditampilkan	*Screen load game* ditampilkan	OK
4	Tombol *avatar* diklik	*Screen avatar* ditampilkan	*Screen avatar* ditampilkan	OK
5	Tombol *about game* diklik	*Screen about game* ditampilkan	*Screen about game* ditampilkan	OK
6	Tombol *quit* diklik	Pilihan konfirmasi keluar ditampilkan	Pilihan konfirmasi keluar ditampilkan	OK

2. Navigasi *avatar*

Tabel 4. 3
Tabel pengujian navigasi avatar

No	Jenis Pengujian	Ekspektasi	Output	Ket
1	Tombol *avatar* diklik	*Screen Avatar* diampilkan	*Screen Avatar* diampilkan	OK
2	Tombol kepala diklik	Menu pilihan bagian kepala ditampilkan	Menu pilihan bagian kepala ditampilkan	OK
3	Tombol bagian kepala diklik	*Preview avatar* berubah sesuai dengan bagian kepala yang dipilih	*Preview avatar* berubah sesuai dengan bagian kepala yang dipilih	OK
4	Tombol mata diklik	Menu pilihan bagian mata ditampilkan	Menu pilihan bagian mata ditampilkan	OK
5	Tombol bagian mata diklik	*Preview avatar* berubah sesuai dengan bagian mata yang dipilih	*Preview avatar* berubah sesuai dengan bagian mata yang dipilih	OK
6	Tombol mulut diklik	Menu pilihan bagian mulut ditampilkan	Menu pilihan bagian mulut ditampilkan	OK
7	Tombol bagian mulut diklik	*Preview avatar* berubah sesuai dengan bagian mulut yang dipilih	*Preview avatar* berubah sesuai dengan bagian mulut yang dipilih	OK
8	Tombol poni diklik	Menu pilihan bagian poni ditampilkan	Menu pilihan bagian poni ditampilkan	OK

No	Jenis Pengujian	Ekspektasi	Output	Ket
9	Tombol bagian poni diklik	*Preview avatar* berubah sesuai dengan bagian poni yang dipilih	*Preview avatar* berubah sesuai dengan bagian poni yang dipilih	OK
10	Tombol rambut diklik	Menu pilihan bagian rambut ditampilkan	Menu pilihan bagian rambut ditampilkan	OK
11	Tombol bagian rambut diklik	*Preview avatar* berubah sesuai dengan bagian rambut yang dipilih	*Preview avatar* berubah sesuai dengan bagian rambut yang dipilih	OK
12	Tombol baju diklik	Menu pilihan bagian baju ditampilkan	Menu pilihan bagian baju ditampilkan	OK
13	Tombol bagian baju diklik	*Preview avatar* berubah sesuai dengan bagian baju yang dipilih	*Preview avatar* berubah sesuai dengan bagian baju yang dipilih	OK
14	Tombol kembali diklik	Keluar dari *screen avatar*	Keluar dari *screen avatar*	OK

3. Navigasi *start game*

Tabel 4. 4
Tabel pengujian navigasi *start game*

No	Jenis Pengujian	Ekspektasi	Output	Ket
1	Tombol *start game* diklik	*Screen* pengisian nama pemain ditampilkan	*Screen* pengisian nama pemain ditampilkan	OK
2	*Input* nama pemain	Nama pemain berubah sesuai dengan yang sudah diinput	Nama pemain berubah sesuai dengan yang sudah diinput	OK

4. Navigasi pilihan setelah kelulusan

Tabel 4. 5
Tabel pengujian Navigasi pilihan setelah kelulusan

No	Jenis Pengujian	Ekspektasi	Output	Ket
1	Tombol kuliah diklik	Dialog kuliah ditampilkan	Dialog kuliah ditampilkan	OK
2	Tombol kampus STMIK IKMI pada menu pilihan kampus	Tampil dialog STMIK IKMI	Tampil dialog STMIK IKMI	OK
3	Tombol kampus UNPAT pada menu pilihan kampus	Tampil dialog UNPAT	Tampil dialog UNPAT	OK
4	Tombol kampus ISB pada menu pilihan kampus	Tampil dialog ISB	Tampil dialog ISB	OK
5	Tombol kampus Maramatha pada menu pilihan kampus	Tampil dialog Maramatha	Tampil dialog Maramatha	OK
6	Tombol kerja diklik	Dialog kerja ditampilkan	Dialog kerja ditampilkan	OK
7	Tombol Pegawai Restoran diklik	Tampil *mini game* surat lamaran	Tampil *mini game* surat lamaran	OK
8	Tombol Maintenance diklik	Tampil *mini game* surat lamaran	Tampil *mini game* surat lamaran	OK
9	Tombol Guru diklik	Tampil *mini game* surat lamaran	Tampil *mini game* surat lamaran	OK

5. Pengujian *mini game* kuliah

Tabel 4. 6
Tabel pengujian *mini game* kuliah

No	Jenis Pengujian	Ekspektasi	Output	Ket
1	Menjawab soal *mini game* USM	Tampil soal berikutnya	Tampil soal berikutnya	OK
2	Mejawab 10 soal *mini game* USM	Tampil dialog selanjutnya	Tampil dialog selanjutnya	OK
3	*Mini game* USM dimainkan dengan menjawab soal dengan benar lebih dari tujuh	Tampil dialog diterima dan mulai kuliah	Tampil dialog diterima dan mulai kuliah	OK
4	*Mini game* USM dimainkan dengan menjawab soal dengan benar kurang dari tujuh	Tampil dialog tidak diterima kuliah	Tampil dialog tidak diterima kuliah	OK
5	Menjawab soal *mini game* Ujian	Tampil soal berikutnya	Tampil soal berikutnya	OK
6	Mejawab 5 soal *mini game* Ujian	Tampil dialog selanjutnya	Tampil dialog selanjutnya	OK

6. Pengujian *mini game* kerja

Tabel 4. 7
Tabel pengujian *mini game* kerja

No	Jenis Pengujian	Ekspektasi	Output	Ket
1	Menjawab soal *mini game* Psikotes	Tampil soal berikutnya	Tampil soal berikutnya	OK
2	Mejawab 10 soal *mini game* Psikotest	Tampil dialog selanjutnya	Tampil dialog selanjutnya	OK
3	*Mini game* Psikotest dimainkan dengan menjawab soal dengan benar lebih dari tujuh	Tampil dialog diterima dan mulai kerja	Tampil dialog diterima dan mulai kerja	OK
4	*Mini game* Psikotes dimainkan dengan menjawab soal dengan benar kurang dari tujuh	Tampil dialog tidak diterima kerja	Tampil dialog tidak diterima kerja	OK
5	Klik gambar jawaban benar pada *mini game* Pegawai Restoran	Tampil soal gambar berikutnya	Tampil soal gambar berikutnya	OK
6	Menjawab 10 soal *mini game* Pegawai Restoran	Tampil dialog selanjutnya	Tampil dialog selanjutnya	OK
7	*Mini game* Pegawai restoran dimainkan dengan menjwab benar lebih dari tujuh	Tampil dialog *happy ending*	Tampil dialog *happy ending*	OK
8	*Mini game* Pegawai restoran dimainkan dengan menjawab salah lebih dari tujuh	Tampil dialog *sad ending*	Tampil dialog *sad ending*	OK

No	Jenis Pengujian	Ekspektasi	Output	Ket
9	Klik gambar jawaban benar pada *mini game* Maintenance	Tampil soal gambar berikutnya	Tampil soal gambar berikutnya	OK
10	Menjawab 10 soal *mini game* Maintenance	Tampil dialog selanjutnya	Tampil dialog selanjutnya	OK
11	*Mini game* Maintenance dimainkan dengan menjawab benar lebih dari tujuh	Tampil dialog *happy ending*	Tampil dialog *happy ending*	OK
12	*Mini game* Maintenance dimainkan dengan menjawab salah lebih dari tujuh	Tampil dialog *sad ending*	Tampil dialog *sad ending*	OK
13	Klik gambar jawaban benar pada *mini game* Pengawas Ujian	Tampil soal gambar berikutnya	Tampil soal gambar berikutnya	OK
14	Menjawab 5 soal *mini game* Pengawas Ujian	Tampil dialog selanjutnya	Tampil dialog selanjutnya	OK
15	*Mini game* Pengawas Ujian dimainkan dengan menjawab benar lebih dari tiga	Tampil dialog *happy ending*	Tampil dialog *happy ending*	OK
16	*Mini game* Pengawas Ujian dimainkan dengan menjwab salah lebih dari tiga	Tampil dialog *sad ending*	Tampil dialog *sad ending*	OK

7. Navigasi *about game*

Tabel 4. 8
Tabel pengujian navigasi *about game*

No	Jenis Pengujian	Ekspektasi	Output	Ket
1	Tombol *about game* pada menu utama diklik	*Screen about game* ditampilkan	*Screen about game* ditampilkan	OK

8. Navigasi *quit*

Tabel 4. 9
Tabel pengujian navigasi *quit*

No	Jenis Pengujian	Ekspektasi	Output	Ket
1	Tombol *quit* pada menu utama diklik	Keluar dari *game*	Keluar dari *game*	OK
2	Tombol *exit* (x) saat sedang bermain *game* diklik	Pilihan konfirmasi ditampilkan	Pilihan konfirmasi ditampilkan	OK
3	Tombol *yes* diklik	Keluar dari *game*	Keluar dari *game*	OK
4	Tombol *no* diklik	Tidak jadi keluar dari *game,* dan kembali ke permainan	Tidak jadi keluar dari *game,* dan kembali ke permainan	OK

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Setelah selesai melalui semua tahap pembuatan dan pengembangan dari *game visual novel After Graduate* dari perencanaan, analisa, perancangan, hingga implementasi maka penulis menarik beberapa kesimpulan, yaitu :

1. *Game visual novel* pada *engine* Ren'Py dibuat dengan menggunakan bahasa python dan memanfaatkan beberapa fungsi yang sudah ada sehingga memudahkan pembuatan *game visual novel*. Fungsi-fungsi tersebut berupa perpindahan layar, menampilkan pertanyaan yang disediakan, tombol yang dapat berasal dari gambar. Percabangan alur cerita pada *visual novel* dibuat dengan memanfaatkan fitur menu pada Ren'Py dan menambahkan *mini game* untuk menentukan hasil lanjutan dari cerita.
2. Penyimpanan soal dan jawaban pada *game* ini menggunakan *list* yang terdapat pada bahasa pemrograman Python. Penyimpanan soal dan jawaban dilakukan dengan menyimpan *list* yang memiliki urutan soal, pilihan jawaban, jawaban sebenarnya.
3. Kustomisasi *avatar* pada *game* dilakukan dengan cara membagi *avatar* dalam beberapa bagian yang dapat diganti oleh pemain. Bagian-bagian yang disediakan antara lain bagian kepala, bagian mata, bagian mimik wajah, bagian poni pada rambut, bagian rambut belakang, serta pakaian yang dipakai oleh avatar. Bagian-bagian avatar yang sudah dipilih akan disimpan ke dalam sebuah variabel dan akan ditampilkan sesuai dengan data yang sudah disimpan sebelumnya.

5.2 Saran

Produk yang dihasilkan dalam Tugas Akhir ini dapat dikembangkan lebih lagi karena masih banyak hal dari segi teknis pemrograman hingga sistem permainan yang belum

lengkap. Adapun saran yang diberikan untuk pengembangan proyek Tugas Akhir ini diantaranya:

1. Membuat *game visual novel* dengan mnggunakan *Platform* lain yang lebih menarik.
2. Membuat *game visual novel* yang dapat dimainkan secara *online.*
3. Menambahkan variasi suara sesuai dengan suasana atau *backgound* dalam cerita dan menambahkan *sound efect* yang sesuai.
4. Meningkatkan segi grafis dan animasi agar *game* terlihat lebih menarik.
5. Menambahkan *mini game* yang lebih variatif.

DAFTAR PUSTAKA

____, "*Kamus Besar Bahasa Indonesia",* 2008.

A. Sukamto, Rosa dan M. Shalahudin, 2013, "*REKAYASA PERANGKAT LUNAK TERSTRUKTUR dan BERORIENTASI OBJEK",* Bandung: Informatika Bandung.

Adams, Ernest, 2010, "*Fundamentals of Game Design",* Pearson Education, Inc.

Bruegge, Brend dan H. Dutoit, Allen, 2004, "*Object-Oriented Software Engineering Using UML", Patterns, and Java.* Pearson Education, Inc.

Buckland, Mat, 2005, "*Programming Game AI by Example",* USA: Wordware Publishing, Inc.

Cavallaro, Dani, 2010, "*Anime and the visual novel".*

Fatta, Hanif Al, 2007, "*Analisis dan Perancangan Sistem Informasi untuk Keunggullan Bersaing Perusahaan dan Organisasi Modern",* Andi.

Fawler, Martin dan Scott, Kendall, 2004, "*UML Distilled Third Edition",* Addison Wesley Longman, Inc.

Griffith, Christoper, 2010, "*Real World Flash Game Development",* Elsevier.

Kumar, A. Anand, 2009, "*FUNDAMENTALS OF DIGITAL CIRCUITS Second edition",* PHI Learning.

Lebowitz, Josiah dan Klug, Chris, 2011, "*Interactive Storytelling for Video Games",* Elsevier.

Marsic, Ivan, 2012, "*Software Engineering",* New Jersey: Rutgers University.

Mulyanto, Aunur R., 2008, "*REKAYASA PERANGKAT LUNAK JILID 1",* Jakarta: Direktorat Pembinaan Sekolah Menengah Kejuruan.

Nugroho, Adi, 2010, "*Rekayasa Perangkat Lunak Berorientasi Onjek dengan Metode USDP",* Yogyakarta: ANDI.

Nurgiantoro, Burhan, 2010, "*Sastra Anak : Pengantar Pemahaman Dunia Anak",* Yogyakarta: Gadjah Mada University Press

Pressman, Roger S., 2010, "*Software Engineering",* McGraw-Hill.

Sibero, Ivan C., 2009, "*Langkah Mudah Membuat Game 3D",* Yogyakarta: MediaKom.

Simarmata, Janner, 2010, "*Rekayasa Perangkat Lunak",* Yogyakarta: ANDI.

Siswoutomo, Wiwit, 2005, "*Membangun Aplikasi Database Berbasis Flash",* Jakarta: PT. Elex Media Komputindo.

Sunyoto, Andi, 2010, "*Adobe Flash + XML = Rich Multimedia Application",* Yogyakarta: ANDI.

Teoh, Soon Tee, 2011, "*Game AI : Finite State Machines and Variants",* San Jose University.

Westra, Erik, 2010, "*Python Geospatial Development*", Birmingham: Pact Publishing Ltd.

http://python.org/moin/BeginnersGuide/Overview, diakses tanggal 5 Desember 2015, jam 14.10 WIB

http://www.renpy.org/, diakses tanggal 26 November 2015, jam 15.06 WIB

http://www.renpy.org/release_list.html, diakses tanggal 25 Januari 2016, jam 13.50 WIB

LAMPIRAN

LISTING PROGRAM

1. Script.rpy

```
image bg sekolah = "bg1.jpg"
image bg kelas = "kelas.jpg"
image bg kampus = "bg_kampus.jpg"
image bg kantor = "bg3.jpg"
image bg selesai = "the_end.jpg"
image bg kamar = "kamar.jpg"
image bg usm = "bg_usm.jpg"
image bg mulai = "bg_mulai.jpg"
image bg awan = "awan.jpg"
image bg rumah = "rumah.jpg"
image bg pilkam = "Pil_kamp.jpg"
image bg kampus1 = "bg_kam1.jpg"
image bg kampus2 = "bg_kam2.jpg"
image bg kampus3 = "bg_kam3.jpg"
image bg kampus4 = "bg_kam4.jpg"
image bg psiko = "psikotest.jpg"
image bg ujian = "bg_ujian.jpg"
image bg lamaran = "mini_lamaran.jpg"
image bg lowker = "lowker.jpg"
image e normal = "a1.png"
image e smile = "a2.png"
image kp normal = "Kepsek.png"
image gr normal = "guru.png"
image gr senyum = "Guru_Senyum.png"
image mo malu = "momo_malu.png"
image mo normal = "momo1.png"
image re normal = "res1.png"
image re senyum = "res2.png"
define siswa_baik = "ujian/siswa1_%s.jpg"
define siswa_nyontek = "ujian/nyontek1_%s.jpg"
define e = Character('Eileen', color="#c8ffc8")
define n = Character('Narator', color="#c8ffc8")
define s = Character('Siswa', color="#00ffff")
define k = Character('Kepala Sekolah', color="#ffff00")
define g = Character('Guru', color="#00ff00")
define mama = Character('Mama', color="#00ffff")
define mb = Character('.......?', color="#ffffff")
define mo = Character('Momo', color="#c8ffc8")
define res = Character('Resepsionis', color="#c8ffc8")
define kepala = ConditionSwitch(
  "kp == 'k1'","Kepala/ke1.png",
  "kp == 'k2'","Kepala/ke2.png",
  "kp == 'k3'","Kepala/ke3.png",
  "kp == 'k4'","Kepala/ke4.png",
  "kp == 'k5'","Kepala/ke5.png",
  "kp == 'k6'","Kepala/ke6.png",
  "kp == 'k7'","Kepala/ke7.png",
  "kp == 'k8'","Kepala/ke8.png",
  "kp == 'k9'","Kepala/ke9.png",
  "kp == 'k10'","Kepala/ke10.png",
  "kp == 'none'","none.png",
  )
define mata = ConditionSwitch(
  "ma == 'ma1'","mata/ma1.png",
  "ma == 'ma2'","mata/ma2.png",
  "ma == 'ma3'","mata/ma3.png",
  "ma == 'ma4'","mata/ma4.png",
  "ma == 'ma5'","mata/ma5.png",
  "ma == 'ma6'","mata/ma6.png",
  "ma == 'ma7'","mata/ma7.png",
  "ma == 'ma8'","mata/ma8.png",
  "ma == 'ma9'","mata/ma9.png",
  "ma == 'ma10'","mata/ma10.png",
  "ma == 'ma11'","mata/ma11.png",
  "ma == 'ma12'","mata/ma12.png",
  "ma == 'ma13'","mata/ma13.png",
  "ma == 'ma14'","mata/ma14.png",
```

```
  "ma == 'ma15'","mata/ma15.png",
  )
define mulut = ConditionSwitch(
  "mu == 'mu1'","mulut/mu1.png",
  "mu == 'mu2'","mulut/mu2.png",
  "mu == 'mu3'","mulut/mu3.png",
  "mu == 'mu4'","mulut/mu4.png",
  "mu == 'mu5'","mulut/mu5.png",
  "mu == 'mu6'","mulut/mu6.png",
  "mu == 'mu7'","mulut/mu7.png",
  "mu == 'mu8'","mulut/mu8.png",
  "mu == 'mu9'","mulut/mu9.png",
  "mu == 'mu10'","mulut/mu10.png",
  "mu == 'mu11'","mulut/mu11.png",
  "mu == 'mu12'","mulut/mu12.png",
  "mu == 'mu13'","mulut/mu13.png",
  "mu == 'mu14'","mulut/mu14.png",
  "mu == 'mu15'","mulut/mu15.png",
  "mu == 'mu16'","mulut/mu16.png",
  "mu == 'mu17'","mulut/mu17.png",
  "mu == 'mu18'","mulut/mu18.png",
  )
define poni = ConditionSwitch(
  "po == 'po1'","poni/po1.png",
  "po == 'po2'","poni/po2.png",
  "po == 'po3'","poni/po3.png",
  "po == 'po4'","poni/po4.png",
  "po == 'po5'","poni/po5.png",
  "po == 'po6'","poni/po6.png",
  "po == 'po7'","poni/po7.png",
  "po == 'po8'","poni/po8.png",
  "po == 'po9'","poni/po9.png",
  "po == 'po10'","poni/po10.png",
  "po == 'po11'","poni/po11.png",
  "po == 'po12'","poni/po12.png",
  "po == 'po13'","poni/po13.png",
  "po == 'po14'","poni/po14.png",
  "po == 'po15'","poni/po15.png",
  "po == 'po16'","poni/po16.png",
  "po == 'po17'","poni/po17.png",
  "po == 'po18'","poni/po18.png",
  )
define rambut = ConditionSwitch(
  "ra == 'ra1'","rambut/ra1.png",
  "ra == 'ra2'","rambut/ra2.png",
  "ra == 'ra3'","rambut/ra3.png",
  "ra == 'ra4'","rambut/ra4.png",
  "ra == 'ra5'","rambut/ra5.png",
  "ra == 'ra6'","rambut/ra6.png",
  "ra == 'ra7'","rambut/ra7.png",
  "ra == 'ra8'","rambut/ra8.png",
  "ra == 'ra9'","rambut/ra9.png",
  "ra == 'ra10'","rambut/ra10.png",
  "ra == 'ra11'","rambut/ra11.png",
  "ra == 'ra12'","rambut/ra12.png",
  "ra == 'ra13'","rambut/ra13.png",
  "ra == 'ra14'","rambut/ra14.png",
  "ra == 'ra15'","rambut/ra15.png",
  "ra == 'ra16'","rambut/ra16.png",
  "ra == 'ra17'","rambut/ra17.png",
  "ra == 'ra18'","rambut/ra18.png",
  )
define baju = ConditionSwitch(
  "bj == 'bj1'","baju/bj1.png",
  "bj == 'bj2'","baju/bj2.png",
  "bj == 'bj3'","baju/bj3.png",
  "bj == 'bj4'","baju/bj4.png",
  "bj == 'bj5'","baju/bj5.png",
```

```
  "bj == 'bj6'","baju/bj6.png",
  "bj == 'bj7'","baju/bj7.png",
  "bj == 'bj8'","baju/bj8.png",
  "bj == 'bj9'","baju/bj9.png",
  "bj == 'bj10'","baju/bj10.png",
  "bj == 'bj11'","baju/bj11.png",
  "bj == 'bj12'","baju/bj12.png",
  "bj == 'bj13'","baju/bj13.png",
  "bj == 'bj14'","baju/bj14.png",
  "bj == 'bj15'","baju/bj15.png",
  "bj == 'bj16'","baju/bj16.png",
  "bj == 'bj17'","baju/bj17.png",
  "bj == 'bj18'","baju/bj18.png",
  )
define badan = "badan/ba1.png"
define food1 = "food/burger1_%s.png"
define food2 = "food/burger2_%s.png"
define food3 = "food/hotdog1_%s.png"
define food4 = "food/hotdog2_%s.png"
define food5 = "food/kentang1_%s.png"
define food6 = "food/minum1_%s.png"
define food7 = "food/minum2_%s.png"
define food8 = "food/minum3_%s.png"
define
ar_food=["food/burger1.png","food/burger2.png","food/hotdog1.png","food/hotdog2.png","food/kentang1.png","food/
minum1.png","food/minum2.png","food/minum3.png"]
define ar_cust=["food/or1.png","food/or2.png","food/or3.png","food/or4.png","food/or5.png"]
$ idx_soal=0
define kp="k1"
define ma="ma1"
define mu="mu5"
define po="po17"
define ra="ra1"
define bj="bj15"
define makanan="gk bisa dimakan"
define makanan_random=0
define orang_random=0
define ar_soal=[
 [
  ["2-5-7-3-6-8-4-....-....",
  ["5-6","6-7","7-8","7-9"],
  3
  ],
  ["9-5-1-2-10-6-2-3-11-7-....",
  ["3","5","12","9"],
  0
  ],
  ["1-9-2-2-9-3-3-9-4-4-....",
  ["7","8","9","10"],
  2
  ],
  ["a-b-c-i-j-d-e-f-i-j-....",
  ["k","g","l","m"],
  1
  ],
  ["a-b-c-c-d-e-f-f-f-g-h-i-i-....",
  ["i","j","k","l"],
  0
  ],
  ["Seminar : sarjana ",
  ["akademi : taruna","perpustakaan : peneliti","rumah sakit : pasien","konservator : seniman"],
  1
  ],
  ["bata : tanah liat",
  ["beton : semen","batu : pasir","bunga : buah","kertas : buku"],
  0
  ],
  ["panas : api",
```

```
 ["hujan : awan","abu : arang","terang : matahari","hangat : listrik"],
 2
 ],
 ["kulit : sisik",
 ["tegel : lantai","rumah : kamar","dinding : cat","atap : genteng"],
 3
 ],
 ["beras : nasi goreng ;; kayu :....",
 ["pasak","kursi tamu","papan","tripleks"],
 1
 ]
],
[
 ["Mobil - Bensin = Pelari - ....",
 ["Makan","Sepatu","Kaos","Lintasan"],
 0
 ],
 ["Pisang - Buah = Sapi - ....",
 ["Banteng","Gemuk","Binatang","Kulit"],
 2
 ],
 ["Datang - Pergi = Akhir - ....",
 ["Awal","Kemudian","Setelah","Zaman"],
 0
 ],
 ["Bersih - Kotor = Tinggi - ....",
 ["Jangkung","Jauh","Dekat","Rendah"],
 3
 ],
 ["Kertas - Pena = Dinding - ....",
 ["dempul","kuas","amplas","cat"],
 1
 ],
 ["Lapar : Nasi ",
 ["Mual : Obat","haus : air","bosan : tidur","baca : gambar"],
 1
 ],
 ["Kepala : Helm",
 ["sabut : kelapa","jari : cincin","rumah : atap","album : foto"],
 2
 ],
 ["Rambut : Gundul",
 ["pakaian : bugil","lantai : licin","botak : kepala","mobil : mogok"],
 0
 ],
 ["diameter : lingkaran",
 ["luas : panjang","tinggi : segitiga","sudut : tegak lurus","diagonal : segi empat"],
 3
 ],
 ["Senapan : Berburu",
 ["kapal : berlabuh","perangkap : menangkap","parang : mengasah","pancing : ikan"],
 1
 ]
],
[
 ["2-5-7-3-6-8-4-....-....",
 ["5-6","6-7","7-8","7-9"],
 3
 ],
 ["9-5-1-2-10-6-2-3-11-7-....",
 ["3","5","12","9"],
 0
 ],
 ["Pisang - Buah = Sapi - ....",
 ["Banteng","Gemuk","Binatang","Kulit"],
 2
 ],
 ["Lapar : Nasi ",
```

```
  ["Mual : Obat","haus : air","bosan : tidur","baca : gambar"],
  1
  ],
  ["a-b-c-c-d-e-f-f-f-g-h-i-i-....",
  ["i","j","k","l"],
  0
  ],
  ["Seminar : sarjana ",
  ["akademi : taruna","perpustakaan : peneliti","rumah sakit : pasien","konservator : seniman"],
  1
  ],
  ["Mobil - Bensin = Pelari - ....",
  ["Makan","Sepatu","Kaos","Lintasan"],
  0
  ],
  ["Kertas - Pena = Dinding - ....",
  ["dempul","kuas","amplas","cat"],
  1
  ],
  ["kulit : sisik",
  ["tegel : lantai","rumah : kamar","dinding : cat","atap : genteng"],
  3
  ],
  ["beras : nasi goreng ;; kayu :....",
  ["pasak","kursi tamu","papan","tripleks"],
  1
  ]
 ]
]
define ar_ujian=[
 [
  ["Jika tabung P tingginya dua kali tabung Q dan jari-jarinya setengah dari tabung Q, perbandingan isi tabunga P terhadap isi tabung Q adalah ....",
  ["1:4","1:2","1:1","2:1"],
  1
  ],
  ["Seorang pedagang menjual sebuah barang dengan harga Rp 80.000,- dan memperoleh laba 23% dari harga beli. Berapakan harga beli?",
  ["Rp 100.000,-","Rp 96.000,-","Rp 64.000,-","Rp 80.000,-"],
  2
  ],
  ["Seorang pekerja dibayar Rp 800,- per jam. Dia bekerja dari pukul 8.00 s/d 16.00. Dia akan dibayar tambahan 50% per jam untuk selewatnya pukul 16.00, jika dia memperoleh bayaran Rp 8.000,- pada hari itu, pukul berapa dia pulang?",
  ["16.20","17.00","17.10","17.20"],
  3
  ],
  ["Seorang mahasiswa melakukan percobaan di laboratorium dan mendapatkan kesimpulan bahwa perbandingan populasi kuman yang bersifat (x) dengan populasi uman yang tidak bersifat (x) adalah 5:3, dan bahwa 3/8 dari kuman yang tidak bersifat (x) adalah jantan. Berapa populasi kuman (x) jantan terhadap populasi kuman seluruhnya?",
  ["1:1","5:8","9:40","15:64"],
  3
  ],
  ["Sebuah perusahaan mengurangi jam kerja pegawainya dari 40 jam per minggu menjadi 36 jam per minggu tanpa mengurangi gaji. Jika seorang pegawai tadinya diberi gaji Rp x/jam. Berapa rupiahkah/jam gajinya sekarang?",
  ["10x","x/10","9x/10","10x/9"],
  3
  ]
 ],
 [
  ["Panitia mengedarkan undangann pertemuan untuk 50 wanita dan 70 pria. Jika ternyata 40% dari undangan wanita dann 50% undangan pria hadir, kira-kira berapa persen yang hadir?",
  ["90","86","48","46"],
  3
  ],
  ["Seorang dari titik x berjalan ke timur 1 km, kemudian 2 km ke utara, lalu 1 km ke timur, terus 1 km ke utara, lalu 1 km ke timur, terakhir 1 km ke utara sampai titik y. Berapa kilometer jarak titik x dari titik y?",
  ["12","9","7","5"],
  3
```

```
  ],
  ["Seorang pekerja mengecet tembok yang tingginya 3 meter dan telah sepertiga selesai, Jika dia selanjutnya mengecat tembok 10 meter persegi lagi, dia sudah akan tiga per empat selesai. Berapa meterkah panjang tembok itu?",
  ["10","8","6","4"],
  1
  ],
  ["Sebuah bejana berbentuk silinder berisi air 1/3nya. Jika ditambah air 3 lagi, bejana ini menjadi berisi 1/2nya. Berapa liter kapasitas bejana itu?",
  ["18","24","27","30"],
  0
  ],
  ["Jika tabung P tingginya dua kali tabung Q dan jari-jarinya setengah dari tabung Q, perbandingan isi tabunga P terhadap isi tabung Q adalah ....",
  ["1:4","1:2","1:1","2:1"],
  1
  ]
 ],
 [
  ["Bambang meninggalkan kota A pukul 6.15 dan sampai di kota B pukul 9.45. Jika dia mengendarai mobilnya dengan kecepatan rata-rata 60km/jam, dan dia beristirahat satu jam untuk makan, berapa km jarak A ke B?",
  ["210 km","175 km","150 km","135 km"],
  2
  ],
  ["Jika sebuah bujur sangkar P luasnya 64 dan sisinya = x dan Q adalah sebuah empat persegi panjang, dimana salah satu sisinya 4, dan sisi lainnya y, bila P=Q maka ....",
  ["X > y","X < y","X = y","X = 2y"],
  1
  ],
  ["Jika seorang berjalan menempuh jarak 2/5 km dalam 5 menit, berapa kecepatan rata-rata perjalanan orang tersebut dalam 1 jam?",
  ["4 km","4,2 km","0.4 km","4,8 km"],
  3
  ],
  ["Seorang siswa memperoleh nilai 91, 88, 86 dan 78 untuk empat mata pelajaran. Berapa nilai yang harus diperoleh untuk mata pelajaran ke lima agar dia memperoleh rata-rata 85?",
  ["82","84","85","86"],
  0
  ],
  ["Sebuah bejana berbentuk silinder berisi air 1/3 nya. Jika ditambah air 3 liter lagi, bejana ini menjadi berisi 1/2 nya. Berapa liter kapasitas bejana itu?",
  ["15","18","24","27"],
  1
  ]
 ],
 [
  ["Sebuah bejana berbentuk silinder berisi air 1/3nya. Jika ditambah air 3 lagi, bejana ini menjadi berisi 1/2nya. Berapa liter kapasitas bejana itu?",
  ["18","24","27","30"],
  0
  ],
  ["Panitia mengedarkan undangann pertemuan untuk 50 wanita dan 70 pria. Jika ternyata 40% dari undangan wanita dann 50% undangan pria hadir, kira-kira berapa persen yang hadir?",
  ["90","86","48","46"],
  3
  ],
  ["Seorang pedagang menjual sebuah barang dengan harga Rp 80.000,- dan memperoleh laba 23% dari harga beli. Berapakan harga beli?",
  ["Rp 100.000,-","Rp 96.000,-","Rp 64.000,-","Rp 80.000,-"],
  2
  ],
  ["Seorang siswa memperoleh nilai 91, 88, 86 dan 78 untuk empat mata pelajaran. Berapa nilai yang harus diperoleh untuk mata pelajaran ke lima agar dia memperoleh rata-rata 85?",
  ["82","84","85","86"],
  0
  ],
  ["Seorang pekerja mengecet tembok yang tingginya 3 meter dan telah sepertiga selesai, Jika dia selanjutnya mengecat tembok 10 meter persegi lagi, dia sudah akan tiga per empat selesai. Berapa meterkah panjang tembok itu?",
  ["10","8","6","4"],
```

```
  1
  ]
 ]
]

label start:
 $ idx_mkn=0
 $ skr = 0
 jump mulai_makan
 scene bg mulai
 with fade
 $ renpy.music.play("lagu.mp3","music",True)
 $ povname = renpy.input(_("Siapa nama anda?")) or _("Guy Shy")
 $ x = Character(povname, color="#c8ffc8")

 "Hai %(povname)s..... \nSelamat datang di Game After Graduate....\nSemoga kamu terhibur dengan membaca Visual Novel ini..."
 "Selamat membaca.... ^.^ \nJangan lupa buat avatarmu dulu ya.... >.<"

 call screen avatar

 scene bg awan
 with fade

 "Hari ini adalah hari yang sangat menegangkan, karena merupakan hari yang sangat menentukan."

 scene bg kamar
 with fade
 "Sesaat setelah terbangun, %(povname)s pun langsung mempersiapkan diri dan berangkat ke sekolah"

 scene bg awan
 with fade
 "Sesampainya di sekolah, ia bertemu dengan teman-teman dan sahabatnya"

 show screen avt1
 x "Hai! Bagaimana kabarmu?"

 hide screen avt1

 show e normal
 e "Hai juga...\nHuft.... Saya benar-benar gugup untuk menghadapi hari ini."

 scene bg awan
 show screen avt1
 x "Sudah pasti. Saya juga merasakan hal yang sama..."
 x "Sudahlah, mari kita masuk ke dalam kelas"

 hide screen avt1
 scene bg kelas
 with fade
 "Kami pun masuk ke dalam kelas sambil menunggu dengan cemas. Tidak lama kemudian bel sekolah pun berbunyi"
 "Teng....Teng....Teng...."
 "Guru dan Kepala Sekolah pun memasuki ruang kelas."
 "Suasana di ruangan pun mulai berubah dan semakin menegangkan. kami semua menjadi semakin gugup"

 show kp normal
 "Akhirnya Kepala Sekolah pun mengumumkan bahwa seluruh siswa dinyataan 'LULUS UN' dan telah selesai menempuh SMA."
 k "Selamat untuk kalian semua. Semoga semua pelajaran-pelajaran yang sudah kalian terima, dapat bermanfaat."

 scene bg kelas
 "Mendengar hal tersebut, kami semua langsung berteriak kegirangan."

 s "Akhirnya....!!! Penantian kami telah selesai.....!"

 "Kami pun keluar kelas."
 scene bg awan
 with fade
```

```
"Lalu Salah satu guru menghampiri kami dan mulai berbicara pada kami"

show gr senyum
g "Selamat ya... Akhirnya kalian lulus juga"

hide gr senyum
s "Terima Kasih Pak..."

scene bg awan
show gr normal
g "Wah... Setelah ini, apa rencana kalian? Apakah langsung melanjutan kuliah atau langsung bekerja? "

hide gr normal
"Beberapa teman pun menjawab ingin kuliah, dan ada pula yang menjawab akan langsung bekerja."
"Saya pun mulai berpikir untuk melanjutkan ke mana?"


menu:
  "Kuliah":
   jump coba_kuliah
  "Kerja":
   jump cari_kerja

label coba_kuliah:
 scene bg awan
 "Saya menjawab pertanyaan guru tersebut, bahwa saya akan melanjutkan kuliah, sama seperti teman saya."

 scene bg rumah
 with fade
 x "Akhirnya...\nSampai juga di rumah...\nBenar-benar hari yang menegangkan..."
 x "Ma.... Saya Lulus....\nTerima kasih untuk doa nya Ma... ^.^"
 mama "Wah... Selamat sayang....\nSemoga kamu bisa meneruskan kuliah dengan baik ya..."
 x "Iya ma...\nTerima kasih. "
 "Kemudian saya pun mulai mencari Kampus dan juga persyaratan-persyaratan yang dibutuhkan."

 scene bg pilkam
 with fade
 call screen box

label pil_kampus1:

 scene bg rumah
 with fade
 "Setelah melihat berbagai macam kampus, saya pun menetapkan untuk mendaftar di salah satu kampus tersebut."

 scene bg kampus1
 with fade
  "Keesokan harinya saya langsung membawa persyaratan yang diperlukan ke tempat tersebut. Tidak lama kemudian saya
diberi tahu untuk mengikuti USM pada hari Senin."


  "Hari Senin pun tiba... \nSaya harus mengikuti USM. Semoga saya dapat mengerjakan soal USM dengan baik."

  "Saya pun memasuki ruangan, dan mulai mengerjakan sal USM."

 scene bg usm
 with fade
 $ i = 1
 jump mulai_tes

label pil_kampus2:

 scene bg rumah
 with fade
 "Setelah melihat berbagai macam kampus, saya pun menetapkan untuk mendaftar di salah satu kampus tersebut."

 scene bg kampus2
 with fade
```

```
 "Keesokan harinya saya langsung membawa persyaratan yang diperlukan ke tempat tersebut. Tidak lama kemudian saya diberi tahu untuk mengikuti USM pada hari Senin."


 "Hari Senin pun tiba... \nSaya harus mengikuti USM. Semoga saya dapat mengerjakan soal USM dengan baik."

 "Saya pun memasuki ruangan, dan mulai mengerjakan sal USM."

 scene bg usm
 with fade
 $ i = 1
 jump mulai_tes

label pil_kampus3:

 scene bg rumah
 with fade
 "Setelah melihat berbagai macam kampus, saya pun menetapkan untuk mendaftar di salah satu kampus tersebut."

 scene bg kampus3
 with fade
 "Keesokan harinya saya langsung membawa persyaratan yang diperlukan ke tempat tersebut. Tidak lama kemudian saya diberi tahu untuk mengikuti USM pada hari Senin."


 "Hari Senin pun tiba... \nSaya harus mengikuti USM. Semoga saya dapat mengerjakan soal USM dengan baik."

 "Saya pun memasuki ruangan, dan mulai mengerjakan sal USM."

 scene bg usm
 with fade
 $ i = 1
 jump mulai_tes

label pil_kampus4:

 scene bg rumah
 with fade
 "Setelah melihat berbagai macam kampus, saya pun menetapkan untuk mendaftar di salah satu kampus tersebut."

 scene bg kampus4
 with fade
 "Keesokan harinya saya langsung membawa persyaratan yang diperlukan ke tempat tersebut. Tidak lama kemudian saya diberi tahu untuk mengikuti USM pada hari Senin."


 "Hari Senin pun tiba... \nSaya harus mengikuti USM. Semoga saya dapat mengerjakan soal USM dengan baik."

 "Saya pun memasuki ruangan, dan mulai mengerjakan sal USM."

 scene bg usm
 with fade
 $ i = 1
 jump mulai_tes


screen box():
   vbox:
      style_group "box1"
      xalign .5
      yalign .5

      textbutton _(" STMIK IKMI ") action Jump("pil_kampus1")
      textbutton _(" U N P A T ") action Jump("pil_kampus2")
      textbutton _(" I S B ") action Jump("pil_kampus3")
      textbutton _(" MARAMATHA ") action Jump("pil_kampus4")
```

```
label coba_lg:
 scene bg kampus
 with fade
 "USM pun selesai...."

 x "Huft... Akhirnya selesai juga, saya tinggal menunggu pengumuman.\nSekarang lebih baik saya pulang ke rumah untuk beristirhat."

 "HARI PENGUMUMAN"

 "Saya pun pergi ke kampus."
 "Lalu saya diberitahu bahwa saya tidak lulus dalam USM, kemudian saya pun pulang."
 "Bagaimana ini?"

 menu:
  "Cari Kampus Lain":
   scene bg pilkam
   with fade
   call screen box
  "Menyerah":
   jump slsai

label cr_kul:
 "Saya tidak bisa menyerah...."
 "Saya akan mencoba mencari di kampus lain."
 scene bg pilkam
 with fade
 call screen box

label mulai_kuliah:
 scene bg kampus
 with fade
 "USM pun selesai...."

 x "Huft... Akhirnya selesai juga, saya tinggal menunggu pengumuman."

 "HARI PENGUMUMAN"

 "Saya pun pergi ke kampus. Setelah sampai di sana saya diberitahu untuk mengikuti CSPT pada hari Sabtu, dan saya diberi KTM."
 "Hari dimana saya harus mengikuti CSPT pun tiba, dan saya harus bergegas berangkat ke kampus. Sesampainya di sana saya mulai bergabung dengan teman-teman."
 "Kami terbagi menjadi beberapa kelompok dan mulai mengikuti CSPT dengan baik"
 "Selesai CSPT, kami diberitahu bahwa perkuliahan dimulai ada hari Senin. Kami pun mulai mencetak jadwal pelajaran yang harus kami tempuh."

 show mo malu
 mb "Hallo... bagaimana tadi saat CSPT?"

 scene bg kampus
 x "Cukup menyenangkan, dan melelahkan."

 show mo malu
 mb "Sama, saya pun begitu..."

 scene bg kampus
 x "Ngomong-omong dari tadi kita belum berkenalan... Nama saya %(povname)s. Siapa namamu?"

 show mo malu
 mb "Nama saya Momo...."

 scene bg kampus
 x "Salam kenal Momo..."

 show mo normal
 mo "%(povname)s Boleh lihat jadwalmu untuk hari Senin?"

 scene bg kampus
```

```
 x "oh.. Boleh...Boleh"

 show mo normal
 mo "Ternyata hari Senin jadwal kita sama."

 scene bg kampus
 x "Benarkan? Kalau begitu sampai bertemu hari Senin ya..."

 show mo normal
 mo "Ia... Sampai bertemu juga..."

 scene bg kampus
 "Hari Senin pun tiba..."

 "Saya datang ke kampus dan melihat jadwal kuliah hari tersebut. Kemudian saya menanyakan pada bagian sekretariat mengenai ruang kelas saya."

 "Hari-hari perkuliahan pun berlalu, saya juga sudah mengikuti perkuliaha dengan baik."

 x "Tidak terasa ya satu minggu lagi kita akan UAS"
 mo "Bagaimana kalau kita belajar bersama?"

 menu:
  "Boleh Juga":
   jump belajar_bersama
  "Tidak, terima kasih. Lain kali saja":
   jump tidak_belajar


label belajar_bersama:
 x "Boleh juga idemu Mo...  Baiklah kalau begitu, Kapan kita akan belajar?"
 mo "Bagaimana kalau hari Sabtu ini, selesai perkuliahan terakhir?"
 x "Oke... Siap..."

 "Hari Sabtu pun tiba, kami pun belajar bersama. Kami membahas beberapa soal dan mata kuliah lainnya yang masih belum dipahami."

 jump ujian
 return

label tidak_belajar:
 x "Tidak terima kasih. Lain kali saja"
 x "Saya ingin belajar sendiri dan berusaha sesuai dengan kemampuan saya. Terima kasih banyak untuk tawarannya."
 "Lalu teman saya pun pergi meninggalkan saya, tanpa berbicara."

 jump ujian
 return

label ujian:
 "Hari ujian pun tiba, saya mulai menjawab pertanyaan-pertanyaan ujian dengan baik sesuai dengan kemampuan saya."

 scene bg ujian
 with dissolve

 jump mulai_ujian
 return

label lanjut_ujian:
 "Ketika saya mengerjakan ujian, ada seorang dosen yang masuk ke dalam ruangan dn membagikan surat pemberitahuan mengenai pembayaran untuk semester selanjutnya."

 "Selesai mengerjakan ujian saya keluar lalu membaca surat tersebut."
 scene bg awan
 with fade
 x "Bagaimana cara saya membayar kuliah? Orang tua saya sudah semakin tua dan tidak memiliki cukup uang untuk membayar."
 x "Apa yang sebaiknya saya lakukan untuk semester selanjutnya?"
```

```
  menu:
   "Berhenti Kuliah":
    jump berhenti
   "Cuti kuliah & Mencari Kerja":
    jump cuti


label berhenti:
  "Jika seperti ini, lebih baik saya berhenti kuliah saja. Saya sudah tidak memiliki cukup biaya untuk membayar perkuliahan semester depan."
  "Untuk apa saya menyusahkan orang tua dan juga mengeluarkan biaya yang banyak..."

  "Lalu %(povname)s pun berhenti kuliah dan menjadi pengangguran. Ia pun semakin menyusahkan orang tuanya."
  jump slsai
  return

label cuti:
  "Saya tidak mau berhenti begitu saja. Saya harus berusaha sebaik mungkin. Lebih baik saya mengambil cuti terlebih dahulu, lalu mencari pekerjaan."
  "Setelah saya sudah memiliki biaya yang cukup, saya akan melanjukan kuliah saya."

  x "Saya harus mulai menacari pekerjaan!!!"

  "Setelah itu saya bertanya pada bagian sekretariat bagaimana cara mengambil cuti."

  "Saya pun diberitahu cara-cara untuk mendapatkan cuti, dan mulai bekerja..."
  "Setelah cuti selama satu semester, akhirnya %(povname)s pun melanjutkan kuliahnya sambik bekerja hingga lulus. ^.^"

  jump slsai
  return


label slsai:
  scene bg selesai
  with dissolve
  "...TAMAT..."
  return

#$ uang += 10
#%(uang)d %(povname)s"


label cari_kerja:
  scene bg awan
  with fade
  "Saya menjawab pertanyaan guru tersebut, bahwa saya akan bekerja untuk membantu orang tua saya."

  scene bg rumah
  "Sesampainya di rumah, saya pun mulai mencari lowongan pekerjaan dan juga membuat surat lamaran pekerjaan."

  jump cari_lowker

label cari_lowker:
  scene bg lowker
  with fade


  "Pilih Pekerjaan"

  call screen box_lowker


screen box_lowker():
    vbox:
        style_group "box1"
        xalign .5
        yalign .5
```

```
    textbutton _("Pegawai Restoran") action Jump("surat_lamaran1")
    textbutton _("Maintenance") action Jump("surat_lamaran2")
    textbutton _(" Guru ") action Jump("surat_lamaran3")

label kirim_surat:
 scene bg rumah
 "Saya pun mengirimkan surat lamaran yang sudah saya buat melalui email."

 "Beberapa hari setelah saya mengirimkan surat lamaran pekerjaan tersebut, saya pun ditelepon untuk datang ke kantor tempat saya melamar."
 "Sesampainya di sana, saya harus mengikuti tes tertulis berupa psikotest.\nSaya pun memasuki ruangan untuk mengikuti tes tersebut."

 x "Semoga saja saya dapat menjawab soal-soal tes dengan baik..."

 scene bg psiko
 with fade
 $ i = 2
 jump mulai_tes

 return

label m_kerja:
 scene bg lowker
 with fade
 "Tes pun selesai, saya pun keluar dari dalam ruangan."
 x "Semoga jawabannya benar..."

 "Keesokan harinya saya pun ditelpon kembali dan diminta untuk datang kembali ke kantor tersebut."
 "Sesampainya di sana, saya pun diberitahu bahwa saya diterima untuk bekerja di sana.\nSaya pun merasa sangat gembira mendengar hal tersebut."

 x "Kapan saya dapat mulai bekerja?"

 show re normal
 res "Besok pagi, anda sudah dapat bekerja."

 hide re normal
 x "Baik..\nMulai pukul berapa saya harus datang?"

 show re normal
 res "Anda harus datang sebelum pukul setengah delapan pagi.\nJuga gunakan pakaian yang sopan dan rapi."

 hide re normal
 x "Baiklah, terima kasih untuk informasinya..."

 show re senyum
 res "Sama-sama"

 if (idx_krj==1):
  jump mulai_makan
 elif (idx_krj==3):
  jump mulai_ngawas_ujian

 return

label coba_cr_kerja:
 "Tes pun selesai, saya pun keluar dari dalam ruangan."
 x "Semoga jawabannya benar..."

 "Keesokan harinya saya pun ditelpon kembali dan diberitahu bahwa saya tidak diterima di tempat tersebut."

 x "Baiklah.... \nKalau begitu lebih baik saya"

 menu:
  "Cari Lowongann pekerjaan yang lain":
   jump cari_lowker
  "Menyerah":
```

```
    jump slsai

  return


label krj:
 x"Huft hari yang cukup melelahkan..."

 "Lalu %(povname)s pun melalui hari-hari kerja yang melelahkan. Karena ia dapat bekerja dengan baik, maka ia pun diangkat menjadi kepala Bagian."
 "Tidak lama kemudian ia menikah dan hidup bahagia. ^.^"

 jump slsai

label krj_ajah:
 x"Huft hari yang cukup melelahkan..."

 "Lalu %(povname)s pun melalui hari-hari kerja yang melelahkan. Karena ia tidak dapat bekerja dengan baik, maka ia pun mendapatkan potongan gaji.."
 "Ia pun menjadi pemalas dan keluar dari tempat kerja tersebut, kemudian menjadi pengangguran. -_-"

 jump slsai


label mulai_tes:
  $ idx_soal=0
  $ skor = 0
  $ idx_random = renpy.random.randint(0,(len(ar_soal)-1))
  "%(idx_random)d"
  $ rand_soal = ar_soal[idx_random]
  $ jml_soal = 10
  jump soal

label soal:
  $ idx_tampil =idx_soal+1
  $ tmp_soal = rand_soal[idx_soal]
  $ soal = tmp_soal[0]
  "Soal ke %(idx_tampil)d\n %(soal)s"
  jump jawab

label jawab:
  $ jawaban = -1
  $ idx_tampil =idx_soal+1
  $ tmp_soal = rand_soal[idx_soal]
  $ jawab0= tmp_soal[1][0]
  $ jawab1= tmp_soal[1][1]
  $ jawab2= tmp_soal[1][2]
  $ jawab3= tmp_soal[1][3]
  menu :
    "%(jawab0)s":
      $ jawaban =0
      jump cek
    "%(jawab1)s":
      $ jawaban=1
      jump cek
    "%(jawab2)s":
      $ jawaban=2
      jump cek
    "%(jawab3)s":
      $ jawaban=3
      jump cek
label cek :
  $ idx_tampil =idx_soal+1
  $ tmp_soal = rand_soal[idx_soal]
  if(tmp_soal[2]==jawaban):
   $ skor= skor+1
  $ idx_soal = idx_soal +1
  if (idx_soal<jml_soal) :
```

```
    jump soal
  jump hasil_tes
label hasil_tes:
  if(i == 1):
    if (skor<=7):
      jump coba_lg
    else :
      jump mulai_kuliah
  elif (i==2):
    if (skor<=7):
      jump coba_cr_kerja
    else :
      jump m_kerja
  return

label mulai_ujian:
  $ idx_soal=0
  $ skor = 0
  $ idx_random = renpy.random.randint(0,(len(ar_ujian)-1))
  $ rand_soal = ar_ujian[idx_random]
  $ jml_soal = 5
  jump soal_u

label soal_u:
  $ idx_tampil =idx_soal+1
  $ tmp_soal = rand_soal[idx_soal]
  $ soal = tmp_soal[0]
  "Soal ke %(idx_tampil)d\n %(soal)s"
  jump jawab_u

label jawab_u:
  $ jawaban = -1
  $ idx_tampil =idx_soal+1
  $ tmp_soal = rand_soal[idx_soal]
  $ jawab0= tmp_soal[1][0]
  $ jawab1= tmp_soal[1][1]
  $ jawab2= tmp_soal[1][2]
  $ jawab3= tmp_soal[1][3]
  menu :
    "%(jawab0)s":
      $ jawaban =0
      jump cek_u
    "%(jawab1)s":
      $ jawaban=1
      jump cek_u
    "%(jawab2)s":
      $ jawaban=2
      jump cek_u
    "%(jawab3)s":
      $ jawaban=3
      jump cek_u
label cek_u :
  #$ hasil = "Salah"
  $ idx_tampil =idx_soal+1
  $ tmp_soal = rand_soal[idx_soal]
  if(tmp_soal[2]==jawaban):
   $ skor= skor+1
  $ idx_soal = idx_soal +1
  if (idx_soal<jml_soal) :
    jump soal_u
  jump lanjut_ujian


init:

    python:
        remaining=0
        def countdown(st, at, length=0.0):
```

```
        global remaining
        remaining = length + st
        return Text("%.1f" % remaining, color="#000000", size=72), .1

  image countdown = DynamicDisplayable(countdown, length=0.01)


label surat_lamaran1:
 $ idx_krj = 1
 scene bg lamaran
 with fade

 show countdown at Position(xalign=.05, yalign=.05)

 $ jwb = renpy.input(_("Salin teks yang ada di dalam kotak kuning!")) or _("Salah")
 $ teks = "Berdasarkan informasi dari media cetak, perihal lamaran pekerjaan di perusahaan tempat Bapak/Ibu pimpin. Melalui surat lamaran ini saya ingin mengajukan diri untuk melamar kerja di perusahaan yang Bapak/Ibu pimpin untuk mengisi posisi yang dibutuhkan saat ini. Saya yang bertanda tangan di bawah ini:"
 if ((teks) == (jwb)):
  $ skor=0
  if remaining >100.0:
   $skor = 50
  elif remaining > 90.0:
   $skor = 60
  elif remaining > 80.0:
   $skor = 70
  elif remaining > 7.00:
   $skor = 80
  elif remaining > 6.00:
   $skor = 90
  else:
   $skor =100
  "%(remaining)d %(skor)d"
  jump kirim_surat
 "Maaf surat yang anda ketik tidak sama persis dengan soal."
 menu:
  "Coba lagi":
   jump surat_lamaran
  "Menyerah":
   jump kirim_surat

label surat_lamaran2:
 $ idx_krj = 2
 scene bg lamaran
 with fade

 show countdown at Position(xalign=.05, yalign=.05)

 $ jwb = renpy.input(_("Salin teks yang ada di dalam kotak kuning!")) or _("Salah")
 $ teks = "Berdasarkan informasi dari media cetak, perihal lamaran pekerjaan di perusahaan tempat Bapak/Ibu pimpin. Melalui surat lamaran ini saya ingin mengajukan diri untuk melamar kerja di perusahaan yang Bapak/Ibu pimpin untuk mengisi posisi yang dibutuhkan saat ini. Saya yang bertanda tangan di bawah ini:"
 if ((teks) == (jwb)):
  $ skor=0
  if remaining >100.0:
   $skor = 50
  elif remaining > 90.0:
   $skor = 60
  elif remaining > 80.0:
   $skor = 70
  elif remaining > 7.00:
   $skor = 80
  elif remaining > 6.00:
   $skor = 90
  else:
   $skor =100
  "%(remaining)d %(skor)d"
  jump kirim_surat
```

```
  "Maaf surat yang anda ketik tidak sama persis dengan soal."
  menu:
   "Coba lagi":
    jump surat_lamaran
   "Menyerah":
    jump kirim_surat

label surat_lamaran3:
  $ idx_krj = 3
  scene bg lamaran
  with fade

  show countdown at Position(xalign=.05, yalign=.05)

  $ jwb = renpy.input(_("Salin teks yang ada di dalam kotak kuning!")) or _("Salah")
  $ teks = "Berdasarkan informasi dari media cetak, perihal lamaran pekerjaan di perusahaan tempat Bapak/Ibu pimpin. Melalui surat lamaran ini saya ingin mengajukan diri untuk melamar kerja di perusahaan yang Bapak/Ibu pimpin untuk mengisi posisi yang dibutuhkan saat ini. Saya yang bertanda tangan di bawah ini:"
   if ((teks) == (jwb)):
    $ skor=0
    if remaining >100.0:
      $skor = 50
    elif remaining > 90.0:
      $skor = 60
    elif remaining > 80.0:
      $skor = 70
    elif remaining > 7.00:
      $skor = 80
    elif remaining > 6.00:
      $skor = 90
    else:
      $skor =100
    "%(remaining)d %(skor)d"
    jump kirim_surat
  "Maaf surat yang anda ketik tidak sama persis dengan soal."
  menu:
   "Coba lagi":
    jump surat_lamaran
   "Menyerah":
    jump kirim_surat


screen avatar:
   tag menu
   window:
      style "gm_root"
   use avt
   frame:
      style_group "gm_nav"
      xalign .50
      yalign .50

      has vbox

      textbutton _("Kepala") action [Hide("avatar"),Show("pilih_kepala")]
      textbutton _("Mata") action [Hide("avatar"),Show("pilih_mata")]
      textbutton _("Mulut") action [Hide("avatar"),Show("pilih_mulut")]
      textbutton _("Poni") action [Hide("avatar"),Show("pilih_poni")]
      textbutton _("Rambut") action [Hide("avatar"),Show("pilih_rambut")]
      textbutton _("Baju") action [Hide("avatar"),Show("pilih_baju")]
      textbutton _("Kembali") action Return()

screen avt():
 # tag menu
 add rambut xalign 1.1 yalign 1.0
 add badan xalign 1.1 yalign 1.0
 add kepala xalign 1.1 yalign 1.0
 add poni xalign 1.1 yalign 1.0
```

```
 add mata xalign 1.1 yalign 1.0
 add mulut xalign 1.1 yalign 1.0
 add baju xalign 1.1 yalign 1.0

screen avt1():
 # tag menu
 add rambut xalign 0.0 yalign 1.0 size (450,600)
 add badan xalign 0.0 yalign 1.0 size (450,600)
 add kepala  xalign 0.0 yalign 1.0 size (450,600)
 add poni xalign 0.0 yalign 1.0 size (450,600)
 add mata xalign 0.0 yalign 1.0 size (450,600)
 add mulut xalign 0.0 yalign 1.0 size (450,600)
 add baju xalign 0.0 yalign 1.0 size (450,600)

screen pilih_kepala:
  tag menu
  window:
     style "gm_root"
  use avt
  frame:
     style_group "gm_nav"
     xalign .50
     yalign .50

     has vbox

     textbutton _("kepala 1") action [SetVariable("kp","k1")]
     textbutton _("kepala 2") action [SetVariable("kp","k2")]
     textbutton _("kepala 3") action [SetVariable("kp","k3")]
     textbutton _("kepala 4") action [SetVariable("kp","k4")]
     textbutton _("kepala 5") action [SetVariable("kp","k5")]
     textbutton _("kepala 6") action [SetVariable("kp","k6")]
     textbutton _("Kembali") action [Hide("avatar"),Show("avatar")]

screen pilih_mata:
  tag menu
  window:
     style "gm_root"
  use avt
  frame:
     style_group "gm_nav"
     xalign .50
     yalign .50

     has vbox

     textbutton _("Mata 1") action [SetVariable("ma","ma1")]
     textbutton _("Mata 2") action [SetVariable("ma","ma2")]
     textbutton _("Mata 3") action [SetVariable("ma","ma3")]
     textbutton _("Mata 4") action [SetVariable("ma","ma4")]
     textbutton _("Mata 5") action [SetVariable("ma","ma5")]
     textbutton _("Mata 6") action [SetVariable("ma","ma6")]
     textbutton _("Mata 7") action [SetVariable("ma","ma7")]
     textbutton _("Mata 8") action [SetVariable("ma","ma8")]
     textbutton _("Mata 9") action [SetVariable("ma","ma9")]
     textbutton _("Mata 10") action [SetVariable("ma","ma10")]
     textbutton _("Mata 11") action [SetVariable("ma","ma11")]
     textbutton _("Mata 12") action [SetVariable("ma","ma12")]
     textbutton _("Mata 13") action [SetVariable("ma","ma13")]
     textbutton _("Mata 14") action [SetVariable("ma","ma14")]
     textbutton _("Mata 15") action [SetVariable("ma","ma15")]
     textbutton _("Kembali") action [Hide("avatar"),Show("avatar")]

screen pilih_poni:
  tag menu
  window:
     style "gm_root"
  use avt
```

```
  frame:
      style_group "gm_nav"
      xalign .50
      yalign .50

      has vbox

      textbutton _("Poni 1") action [SetVariable("po","po1")]
      textbutton _("Poni 2") action [SetVariable("po","po2")]
      textbutton _("Poni 3") action [SetVariable("po","po3")]
      textbutton _("Poni 4") action [SetVariable("po","po4")]
      textbutton _("Poni 5") action [SetVariable("po","po5")]
      textbutton _("Poni 6") action [SetVariable("po","po6")]
      textbutton _("Poni 7") action [SetVariable("po","po7")]
      textbutton _("Poni 8") action [SetVariable("po","po8")]
      textbutton _("Poni 9") action [SetVariable("po","po9")]
      textbutton _("Poni 10") action [SetVariable("po","po10")]
      textbutton _("Poni 11") action [SetVariable("po","po11")]
      textbutton _("Poni 12") action [SetVariable("po","po12")]
      textbutton _("Poni 13") action [SetVariable("po","po13")]
      textbutton _("Poni 14") action [SetVariable("po","po14")]
      textbutton _("Poni 15") action [SetVariable("po","po15")]
      textbutton _("Poni 16") action [SetVariable("po","po16")]
      textbutton _("Poni 17") action [SetVariable("po","po17")]
      textbutton _("Poni 18") action [SetVariable("po","po18")]
      textbutton _("Kembali") action [Hide("avatar"),Show("avatar")]

screen pilih_mulut:
  tag menu
  window:
      style "gm_root"
  use avt
  frame:
      style_group "gm_nav"
      xalign .50
      yalign .50

      has vbox

      textbutton _("Marah 1") action [SetVariable("mu","mu1")]
      textbutton _("Marah 2") action [SetVariable("mu","mu2")]
      textbutton _("Senyum 1") action [SetVariable("mu","mu3")]
      textbutton _("Senyum 2") action [SetVariable("mu","mu4")]
      textbutton _("Senyum 3") action [SetVariable("mu","mu5")]
      textbutton _("Senyum 4") action [SetVariable("mu","mu6")]
      textbutton _("Senyum 5") action [SetVariable("mu","mu7")]
      textbutton _("Senyum 6") action [SetVariable("mu","mu8")]
      textbutton _("Senyum 7") action [SetVariable("mu","mu9")]
      textbutton _("Datar 1") action [SetVariable("mu","mu10")]
      textbutton _("Datar 2") action [SetVariable("mu","mu11")]
      textbutton _("Datar 3") action [SetVariable("mu","mu12")]
      textbutton _("Takut 1") action [SetVariable("mu","mu13")]
      textbutton _("Takut 2") action [SetVariable("mu","mu14")]
      textbutton _("Bengong 1") action [SetVariable("mu","mu15")]
      textbutton _("Bengong 2") action [SetVariable("mu","mu16")]
      textbutton _("Takut") action [SetVariable("mu","mu17")]
      textbutton _("Marah") action [SetVariable("mu","mu18")]
      textbutton _("Kembali") action [Hide("avatar"),Show("avatar")]

screen pilih_rambut:
  tag menu
  window:
      style "gm_root"
  use avt
  frame:
      style_group "gm_nav"
      xalign .50
      yalign .50
```

```
        has vbox

        textbutton _("Sangat Pendek") action [SetVariable("ra","ra1")]
        textbutton _("Pendek ikat 1") action [SetVariable("ra","ra2")]
        textbutton _("Pendek Bob") action [SetVariable("ra","ra3")]
        textbutton _("Kepang 1") action [SetVariable("ra","ra4")]
        textbutton _("Pendek Bob acak") action [SetVariable("ra","ra5")]
        textbutton _("Panjang ikat 1") action [SetVariable("ra","ra6")]
        textbutton _("Panjang") action [SetVariable("ra","ra7")]
        textbutton _("Panjang ikat 2a") action [SetVariable("ra","ra8")]
        textbutton _("Panjang ikat 1a") action [SetVariable("ra","ra9")]
        textbutton _("Pendek ikat 2a") action [SetVariable("ra","ra10")]
        textbutton _("Pendek") action [SetVariable("ra","ra11")]
        textbutton _("Ikal ikat 1") action [SetVariable("ra","ra12")]
        textbutton _("Panjang ikat 1b") action [SetVariable("ra","ra13")]
        textbutton _("Pendek ikat 1") action [SetVariable("ra","ra14")]
        textbutton _("Ikat 2 acak") action [SetVariable("ra","ra15")]
        textbutton _("Ikat 2 ikal") action [SetVariable("ra","ra16")]
        textbutton _("Pendek ikat 2b") action [SetVariable("ra","ra17")]
        textbutton _("Panjang ikat 2b") action [SetVariable("ra","ra18")]
        textbutton _("Kembali") action [Hide("avatar"),Show("avatar")]

screen pilih_baju:
  tag menu
  window:
     style "gm_root"
  use avt
  frame:
     style_group "gm_nav"
     xalign .50
     yalign .50

     has vbox


     textbutton _("Suster") action [SetVariable("bj","bj1")]
     textbutton _("Seragam Perempuan 1") action [SetVariable("bj","bj2")]
     textbutton _("Seragam Perempuan 2") action [SetVariable("bj","bj3")]
     textbutton _("Seragam Perempuan 3") action [SetVariable("bj","bj4")]
     textbutton _("Seragam Perempuan 4") action [SetVariable("bj","bj5")]
     textbutton _("Sweater Perempuan") action [SetVariable("bj","bj6")]
     textbutton _("Pesta Perempuan") action [SetVariable("bj","bj7")]
     textbutton _("Kimono 1") action [SetVariable("bj","bj8")]
     textbutton _("Kimono 2") action [SetVariable("bj","bj9")]
     textbutton _("Dokter") action [SetVariable("bj","bj10")]
     textbutton _("Sweater Laki-laki") action [SetVariable("bj","bj11")]
     textbutton _("Kimono 3") action [SetVariable("bj","bj12")]
     textbutton _("Fantasi") action [SetVariable("bj","bj13")]
     textbutton _("Koki") action [SetVariable("bj","bj14")]
     textbutton _("Piyama") action [SetVariable("bj","bj15")]
     textbutton _("Seragam Laki-laki 1") action [SetVariable("bj","bj16")]
     textbutton _("Seragam Laki-laki 2") action [SetVariable("bj","bj17")]
     textbutton _("Seragam Laki-laki 3") action [SetVariable("bj","bj18")]
     textbutton _("Kembali") action [Hide("avatar"),Show("avatar")]


label mulai_makan :
 $ idx_mkn = idx_mkn+1
 $ makanan_random = renpy.random.randint(0,(len(ar_food)-1))
 $ orang_random = renpy.random.randint(0,(len(ar_cust)-1))
 call screen g1
screen g1:
 window:
   style "gm_root"
 frame:
   xalign .00
   has hbox
```

```
  imagebutton auto food1 action [SetVariable("makanan",0),Hide("g1"),Jump("dimakan")]
 frame:
  xalign .30
  has hbox
  imagebutton auto food2 action [SetVariable("makanan",1),Hide("g1"),Jump("dimakan")]
 frame:
  xalign .60
  has hbox
  imagebutton auto food3 action [SetVariable("makanan",2),Hide("g1"),Jump("dimakan")]
 frame:
  xalign .90
  has hbox
  imagebutton auto food4 action [SetVariable("makanan",3),Hide("g1"),Jump("dimakan")]
 frame:
  xalign .00
  yalign .90
  has hbox
  imagebutton auto food5 action [SetVariable("makanan",4),Hide("g1"),Jump("dimakan")]
 frame:
  xalign .30
  yalign .90
  has hbox
  imagebutton auto food6 action [SetVariable("makanan",5),Hide("g1"),Jump("dimakan")]
 frame:
  xalign .40
  yalign .45
  has hbox
  imagebutton auto food7 action [SetVariable("makanan",6),Hide("g1"),Jump("dimakan")]
 frame:
  xalign .10
  yalign .45
  has hbox
  imagebutton auto food8 action [SetVariable("makanan",7),Hide("g1"),Jump("dimakan")]
 frame:
  xalign 1.0
  yalign 1.1
  add ar_cust[orang_random] xalign 0 yalign 0 size (300, 420)
 frame:
  xalign .77
  yalign .45
  add ar_food[makanan_random] xalign 0 yalign 0 size (88,88)

label dimakan:
 scene bg sekolah
 if (idx_mkn==10):
  if (skr>=7):
   jump krj
  jump krj_ajah
 if(makanan==makanan_random):
  $ skr = skr+1
 $ skr = skr
 #"Skor : %(skr)s, soal : %(idx_mkn)s"
 jump mulai_makan

label mulai_ngawas_ujian:
 $ ar_siswa=[]
 $ jml_siswa=6
 $ jml_nyontek=1
 $ skor_ujian=0
 $ jumlah_mengawas=0
 jump set_ujian
label set_ujian:
 if(jumlah_mengawas>=5):
  jump selesai_ujian
 $ar_siswa=[0,0,0,0,0,0]
 $ i =0
 $tmp=""
 while i<jml_nyontek:
```

```
 $acak=renpy.random.randint(0,jml_siswa-1)
 while int(ar_siswa[acak])!=0:
  $acak=renpy.random.randint(0,jml_siswa-1)
  $tmp=ar_siswa[acak]
 $ar_siswa[acak]=1
 $i=i+1
 $pilih_siswa=0
call screen ngawas_ujian
screen ngawas_ujian:
 window:
  style "gm_root"
 frame:
  xalign .20
  has hbox
  if(ar_siswa[0]!=1):
   imagebutton auto siswa_baik action [SetVariable("pilih_siswa",'0a'),Hide("ngawas_ujian"),Jump("hasil_ujian")]
  else:
   imagebutton auto siswa_nyontek action [SetVariable("pilih_siswa",'0b'),Hide("ngawas_ujian"),Jump("hasil_ujian")]

  if(ar_siswa[1]!=1):
   imagebutton auto siswa_baik action [SetVariable("pilih_siswa",'1a'),Hide("ngawas_ujian"),Jump("hasil_ujian")]
  else:
   imagebutton auto siswa_nyontek action [SetVariable("pilih_siswa",'1b'),Hide("ngawas_ujian"),Jump("hasil_ujian")]

  if(ar_siswa[2]!=1):
   imagebutton auto siswa_baik action [SetVariable("pilih_siswa",'2b'),Hide("ngawas_ujian"),Jump("hasil_ujian")]
  else:
   imagebutton auto siswa_nyontek action [SetVariable("pilih_siswa",'2a'),Hide("ngawas_ujian"),Jump("hasil_ujian")]
 frame:
  xalign .20
  yalign .75
  has hbox
  if(ar_siswa[3]!=1):
   imagebutton auto siswa_baik action [SetVariable("pilih_siswa",'3a'),Hide("ngawas_ujian"),Jump("hasil_ujian")]
  else:
   imagebutton auto siswa_nyontek action [SetVariable("pilih_siswa",'3b'),Hide("ngawas_ujian"),Jump("hasil_ujian")]
  if(ar_siswa[4]!=1):
   imagebutton auto siswa_baik action [SetVariable("pilih_siswa",'4a'),Hide("ngawas_ujian"),Jump("hasil_ujian")]
  else:
   imagebutton auto siswa_nyontek action [SetVariable("pilih_siswa",'4b'),Hide("ngawas_ujian"),Jump("hasil_ujian")]
  if(ar_siswa[5]!=1):
   imagebutton auto siswa_baik action [SetVariable("pilih_siswa",'5a'),Hide("ngawas_ujian"),Jump("hasil_ujian")]
  else:
   imagebutton auto siswa_nyontek action [SetVariable("pilih_siswa",'5b'),Hide("ngawas_ujian"),Jump("hasil_ujian")]

label hasil_ujian:
 $ siswa = pilih_siswa[:1]
 $ pilihan_index = pilih_siswa[1:]
 $hasil_mengawas=0
 if(ar_siswa[int(siswa)]==1):
  if(pilihan_index=='b'):
   $ skor_ujian= skor_ujian+1
   $ hasil_mengawas=1
 if(hasil_mengawas==1):
  "Anda menangkap anak yang menyontek"
 else:
  "Anda salah menangkap"
 $ jumlah_mengawas=jumlah_mengawas+1
 jump set_ujian

label selesai_ujian:
 "%(skor_ujian)d"
```

2. screens.rpy

```
screen say(who, what, side_image=None, two_window=False):

    if not two_window:
```

```
            window:
        id "window"

        has vbox:
            style "say_vbox"

        if who:
            text who id "who"
        text what id "what"
    else:
        vbox:
            style "say_two_window_vbox"
            if who:
                window:
                    style "say_who_window"
                    text who:
                        id "who"
            window:
                id "window"
                has vbox:
                    style "say_vbox"
                text what id "what"

    if side_image:
        add side_image
    else:
        add SideImage() xalign 0.0 yalign 1.0

    use quick_menu

screen choice(items):

    window:
        style "menu_window"
        xalign 0.5
        yalign 0.5

        vbox:
            style "menu"
            spacing 2

            for caption, action, chosen in items:

                if action:

                    button:
                        action action
                        style "menu_choice_button"

                        text caption style "menu_choice"

                else:
                    text caption style "menu_caption"

init -2:
    $ config.narrator_menu = True

    style menu_window is default

    style menu_choice is button_text:
        clear

    style menu_choice_button is button:
        xminimum int(config.screen_width * 0.45)
        xmaximum int(config.screen_width * 0.45)

screen input(prompt):
```

```
    window style "input_window":
        has vbox

        text prompt style "input_prompt"
        input id "input" style "input_text"

    use quick_menu

screen nvl(dialogue, items=None):

    window:
        style "nvl_window"

        has vbox:
            style "nvl_vbox"

        for who, what, who_id, what_id, window_id in dialogue:
            window:
                id window_id

                has hbox:
                    spacing 10

                if who is not None:
                    text who id who_id

                text what id what_id

            if items:

                vbox:
                    id "menu"

                    for caption, action, chosen in items:

                        if action:

                            button:
                                style "nvl_menu_choice_button"
                                action action

                                text caption style "nvl_menu_choice"

                        else:

                            text caption style "nvl_dialogue"

    add SideImage() xalign 0.0 yalign 1.0

    use quick_menu

screen main_menu():

    tag menu

    window:
        style "mm_root"

    frame:
        style_group "mm"
        xalign .50
        yalign .50

        has vbox

        textbutton _("Start Game") action Start()
        textbutton _("Load Game") action ShowMenu("load")
        textbutton _("Avatar") action ShowMenu("avatar")
```

```
        textbutton _("About Game") action Help()
        textbutton _("Quit") action Quit(confirm=False)

init -2:

    # Make all the main menu buttons be the same size.
    style mm_button:
        size_group "mm"

screen navigation():

    window:
        style "gm_root"
    frame:
        style_group "gm_nav"
        xalign .98
        yalign .98

        has vbox

        textbutton _("Return") action Return()
        #textbutton _("Preferences") action ShowMenu("preferences")
        textbutton _("Avatar") action ShowMenu("avatar")
        textbutton _("Save Game") action ShowMenu("save")
        textbutton _("Load Game") action ShowMenu("load")
        textbutton _("Main Menu") action MainMenu()
        textbutton _("About") action Help()
        textbutton _("Quit") action Quit()

init -2:

    # Make all game menu navigation buttons the same size.
    style gm_nav_button:
        size_group "gm_nav"

screen file_picker():

    frame:
        style "file_picker_frame"

        has vbox

        # The buttons at the top allow the user to pick a
        # page of files.
        hbox:
            style_group "file_picker_nav"

            textbutton _("Previous"):
                action FilePagePrevious()

            textbutton _("Auto"):
                action FilePage("auto")

            textbutton _("Quick"):
                action FilePage("quick")

            for i in range(1, 9):
                textbutton str(i):
                    action FilePage(i)

            textbutton _("Next"):
                action FilePageNext()

        $ columns = 2
        $ rows = 5

        grid columns rows:
            transpose True
```

```
        xfill True
        style_group "file_picker"

        for i in range(1, columns * rows + 1):
            button:
                action FileAction(i)
                xfill True

                has hbox
                add FileScreenshot(i)

                $ file_name = FileSlotName(i, columns * rows)
                $ file_time = FileTime(i, empty=_("Empty Slot."))
                $ save_name = FileSaveName(i)

                text "[file_name]. [file_time!t]\n[save_name!t]"

                key "save_delete" action FileDelete(i)

screen save():
    tag menu

    use navigation
    use file_picker

screen load():
    tag menu

    use navigation
    use file_picker

init -2:
    style file_picker_frame is menu_frame
    style file_picker_nav_button is small_button
    style file_picker_nav_button_text is small_button_text
    style file_picker_button is large_button
    style file_picker_text is large_button_text

screen preferences():

    tag menu
    use navigation
    grid 3 1:
        style_group "prefs"
        xfill True
        vbox:
            frame:
                style_group "pref"
                has vbox

                label _("Display")
                textbutton _("Window") action Preference("display", "window")
                textbutton _("Fullscreen") action Preference("display", "fullscreen")

            frame:
                style_group "pref"
                has vbox

                label _("Transitions")
                textbutton _("All") action Preference("transitions", "all")
                textbutton _("None") action Preference("transitions", "none")

            frame:
                style_group "pref"
                has vbox

                label _("Text Speed")
                bar value Preference("text speed")
```

```
vbox:
    frame:
        style_group "pref"
        has vbox

        label _("Skip")
        textbutton _("Seen Messages") action Preference("skip", "seen")
        textbutton _("All Messages") action Preference("skip", "all")

    frame:
        style_group "pref"
        has vbox

        textbutton _("Begin Skipping") action Skip()

    frame:
        style_group "pref"
        has vbox

        label _("After Choices")
        textbutton _("Stop Skipping") action Preference("after choices", "stop")
        textbutton _("Keep Skipping") action Preference("after choices", "skip")

    frame:
        style_group "pref"
        has vbox

        label _("Auto-Forward Time")
        bar value Preference("auto-forward time")

        if config.has_voice:
            textbutton _("Wait for Voice") action Preference("wait for voice", "toggle")

vbox:
    frame:
        style_group "pref"
        has vbox

        label _("Music Volume")
        bar value Preference("music volume")

    frame:
        style_group "pref"
        has vbox

        label _("Sound Volume")
        bar value Preference("sound volume")

        if config.sample_sound:
            textbutton _("Test"):
                action Play("sound", config.sample_sound)
                style "soundtest_button"

    if config.has_voice:
        frame:
            style_group "pref"
            has vbox

            label _("Voice Volume")
            bar value Preference("voice volume")

            textbutton _("Voice Sustain") action Preference("voice sustain", "toggle")
            if config.sample_voice:
                textbutton _("Test"):
                    action Play("voice", config.sample_voice)
                    style "soundtest_button"
```

```
init -2:
    style pref_frame:
        xfill True
        xmargin 5
        top_margin 5

    style pref_vbox:
        xfill True

    style pref_button:
        size_group "pref"
        xalign 1.0

    style pref_slider:
        xmaximum 192
        xalign 1.0

    style soundtest_button:
        xalign 1.0

screen yesno_prompt(message, yes_action, no_action):
    modal True

    window:
        style "gm_root"

    frame:
        style_group "yesno"

        xfill True
        xmargin .05
        ypos .1
        yanchor 0
        ypadding .05

        has vbox:
            xalign .5
            yalign .5
            spacing 30

        label _(message):
            xalign 0.5

        hbox:
            xalign 0.5
            spacing 100

            textbutton _("Yes") action yes_action
            textbutton _("No") action no_action
    key "game_menu" action no_action

init -2:
    style yesno_button:
        size_group "yesno"

    style yesno_label_text:
        text_align 0.5
        layout "subtitle"

    hbox:
        style_group "quick"

        xalign 1.0
        yalign 1.0

        textbutton _("Back") action Rollback()
        textbutton _("Save") action ShowMenu('save')
        textbutton _("Q.Save") action QuickSave()
```

```
        textbutton _("Q.Load") action QuickLoad()
        textbutton _("Skip") action Skip()
        textbutton _("F.Skip") action Skip(fast=True, confirm=True)
        textbutton _("Auto") action Preference("auto-forward", "toggle")
        textbutton _("Avatar") action ShowMenu("avatar")

init -2:
    style quick_button:
        is default
        background None
        xpadding 5

    style quick_button_text:
        is default
        size 12
        idle_color "#8888"
        hover_color "#ccc"
        selected_idle_color "#cc08"
        selected_hover_color "#cc0"
        insensitive_color "#4448"
```

3. Options.rpy

```
init -1 python hide:
    config.developer = True
    config.screen_width = 800
    config.screen_height = 600
    config.window_title = u"VN_Coba_TA"
    config.name = "VN_Coba_TA"
    config.version = "0.0"
    theme.crayon(
        widget = "#525748",
        widget_hover = "#f45c73",
        widget_text = "#e7f3cb",
        widget_selected = "#ffce95",
        disabled = "#b3c292",
        disabled_text = "#525748",
        label = "#525748",
        frame = "#d8ebae",
        mm_root = "Bg_TA.jpg",
        gm_root = "#e7f3cb",
        rounded_window = True,
        )

    config.has_sound = True
    config.has_music = True
    config.has_voice = False
    style.button.activate_sound = "click.wav"
    style.imagemap.activate_sound = "click.wav"
    config.enter_sound = "click.wav"
    config.exit_sound = "click.wav"
    config.sample_sound = "click.wav"
    config.help = "README.html"
    config.enter_transition = None
    config.exit_transition = None
    config.intra_transition = None
    config.main_game_transition = None
    config.game_main_transition = None
    config.end_splash_transition = None
    config.end_game_transition = None
    config.after_load_transition = None
    config.window_show_transition = None
    config.window_hide_transition = None
    config.adv_nvl_transition = dissolve
    config.nvl_adv_transition = dissolve
    config.enter_yesno_transition = None
    config.exit_yesno_transition = None
    config.enter_replay_transition = None
```

```
    config.exit_replay_transition = None
    config.say_attribute_transition = None

image main_menu:
    contains:
        "#000"

    contains:
        "Bg_TA"
        size (800, 509)

    contains:
        Text("Ren'Py " + config.version, size=18)
        yalign .98
        xalign .02

python early:
    config.save_directory = "VN_Coba_TA-1443234281"

init -1 python hide:

    config.default_fullscreen = False

    config.default_text_cps = 0

    config.default_afm_time = 10
```