**BAB IV**

**ANALISIS DAN PERANCANGAN SISTEM**

**4.1 Analisis Sistem**

Analisis sistem merupakan suatu tahapan yang bertujuan untuk mengetahui dan mengamati apa saja yang terlibat dalam suatu sistem. Pembahasan yang ada pada analisis sistem ini yaitu analisis kebutuhan fungsional meliputi perancangan sistem menggunakan bahasa pemodelan UML (*Unified Modeling Language*), perancangan tampilan *interface* serta analisis kebutuhan fungsoinal dan nonfungsional yang meliputi kebutuhan perangkat keras dan perangkat lunak yang akan digunakan.

**4.1.1 Analisis Kebutuhan Fungsional**

Analisis kebutuhan fungsional adalah segala bentuk data yang dibutuhkan oleh sistem agar sistem dapat berjalan sesuai dengan prosedur yang dibangun melalui perancangan sistem. Adapun kebutuhan fungsional dari sistem yang akan dibangun, yaitu;

1. Perancangan diagram sistem menggunakan bahasa pemodelan UML (*Unified Modeling Language*) yang meliputi pembuatan *flowchart* sistem, *flowchart* metode, *use case diagram*, *activity diagram, class diagram*  serta *sequence diagram.*
2. Menganalisis kebutuhan data yangterdiri dari data yang akan diambil dari *e-marketplace*, melakukan proses *web scraping* dari beberapa *e-marketplace* untuk menemukan beberapa data barang dan melakukan proses pencarian menggunakan algoritma *levenshtein distance* serta *output* dari aplikasi berupa hasil pencarian barang dari beberapa *e-marketplace*.

**4.1.2 Analisis Kebutuhan Non-fungsional**

Analisis kebutuhan non-fungsional adalah sebuah langkah dimana pembangun aplikasi menganalisis sumber daya yang dibutuhkan untuk membangun aplikasi yang akan dibangun. Analisis kebutuhan nonfungsional yang dilakukan dibagi dalam dua tahap, yaitu analisis kebutuhan perangkat keras dan analisis kebutuhan perangkat lunak. Kebutuhan perangkat keras yaitu kebutuhan perangkat atau komponen yang dibutuhkan pada sistem dan perangkat lunak yaitu kebutuhan perangkat lunak untuk membantu agar komponen perangkat keras dapat berfungsi dan dapat dijalankan pada sistem.

**4.1.2.1 Kebutuhan Perangkat Keras**

Untuk menerapkan rancangan yang telah dijelaskan sebelumnya, dibutuhkan beberapa perangkat keras sebagai sarana untuk mengimplementasikan aplikasi yang dibangun. Berikut ini spesifikasi perangkat keras yang dibutuhkan.

**Tabel 4.1 Spesifikasi perangkat keras**

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Perangkat** | **Spesifikasi** |
| 1. | *Notebook* | *Asus A455LD* |
| 2. | *Processor* | *Intel Core i5 2.10 GHz* |
| 3. | *Monitor* | *Monitor 14 inch* |
| 4. | *Memori* | *RAM 6 GB DDR 3L* |
| 5. | *Harddisk* | *500 GB HDD* |

**4.1.2.2 Kebutuhan Perangkat Lunak**

Perangkat lunak yang digunakan pada pembangunan aplikasi pencarian barang. Adapun rincian kebutuhan perangkat lunak dapat dilihat pada Tabel 4.2 berikut ini.

**Tabel 4.2 Spesifikasi perangkat lunak**

|  |  |  |
| --- | --- | --- |
| **No** | **Nama Perangkat** | **Spesifikasi** |
| 1. | *Operating System* | *Windows 10 Profesional 64 bit* |
| 2. | *Python* | *Python 3.6* |
| 3. | *Mysql* | *Mysqli* |
| 4. | *Apache* | *Php 5* |
| 5. | *Browser* | *Google Crome* |
| 6. | *Text Editor* | *Notepad++* |

**4.2 Analisis Perancangan Sistem**

Perancangan sistem yang akan dibangun terdiri atas perancangan *flowchar*t dan perancangan UML serta perancangan *user interface.*

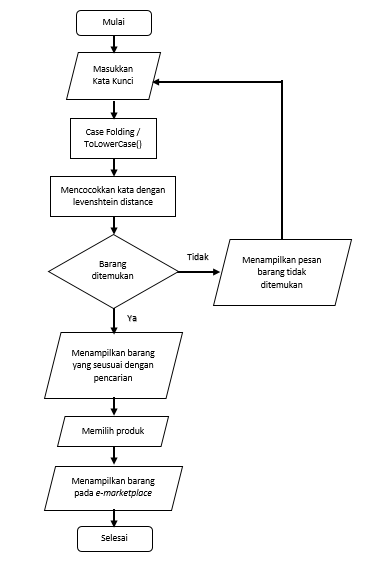
* + 1. **Perancangan *Flowchart***

*Flowchart* adalah simbol-simbol pekerjaan yang menunjukkan bagan aliran proses yang saling terhubung. Perancangan *flowchart* yang akan dibangun terdiri atas *flowchart* sistem, *flowchart* algoritma *Levenshtein Distance*.

* + - 1. ***Flowchart* Sistem**

Setelah menganalisis sistem, maka didapatkan *flowchart diagram* sistem pada aplikasi pencarian barang di berbagai *e-marketplace* yang ditunjukkan oleh Gambar 4.1. Adapun alur kerja *flowchart diagram* sistem adalah sebagai berikut:

1. *User* memasukkan kata yang ingin dicari.
2. Kata yang dimasukkan akan melalui proses *case folding/toLowerCase* untuk mengubah kata menjadi huruf kecil.
3. Kata yang dicari akan dicocokkan menggunakan algoritma *Levenshtein Distance*.
4. Menampilkan daftar produk yang berhubungan dengan kata kunci.
5. Jika produk tidak ditemukan, maka menampilkan pesan “Produk tidak ditemukan” dan *user* dapat melakukan pencarian ulang. Tapi jika barang ditemukan, maka akan menampilkan produk.
6. *User* memilih produk yang diinginkan.
7. Menampilkan halaman penjualan produk yang telah dipilih.

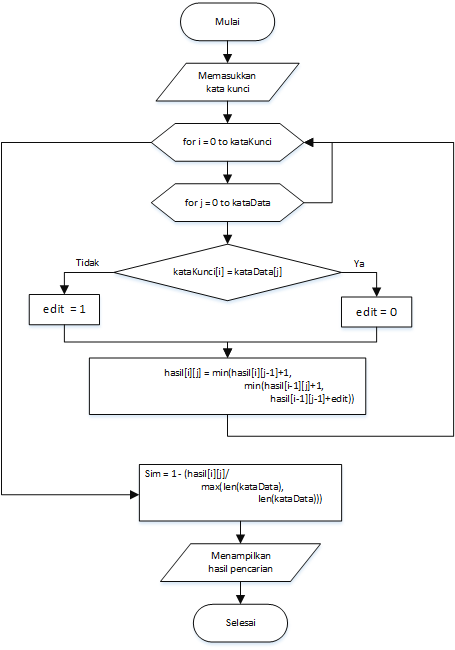


**Gambar 4.1 *Flowchart* Sistem**

* + - 1. ***Flowchart* algoritma *Levenshtein Distance***

Setelah menganalisis sistem, maka didapatkan *flowchart diagram* algoritma *Levenshtein Distance* pada aplikasi pencarian barang di berbagai *e-marketplace* yang ditunjukkan oleh Gambar 4.2. Adapun alur kerja *flowchart diagram* algoritma *Levenshtein Distance* adalah sebagai berikut:

1. Pertama *user* memasukkan kata kunci yang akan dicari.
2. Sistem melakukan proses pencarian dengan syarat *for i =* panjang kata kunci dimana *i=0*.
3. Sistem melakukan proses pencarian dengan syarat *for j =* panjang kata data dimana *j=0*.
4. Kemudian mengecek huruf pada kata kunci dan kata data. Bila huruf tersebut berbeda, maka memberikan tambahan nilai 1 kata sebelumnya.
5. Selanjutnya mencari nilai minimal dari nilai kata kunci, kata data dan kata sebelumnya.
6. Setelah itu, huruf selanjutnya akan kembali diproses dengan cara yang sama. Apabila huruf yang akan diproses sudah tidak ada lagi, maka system akan menampilkan hasil pencarian.

 **Gambar 4.2 *Flowchart* Algoritma *Levenshtein Distance***

Berdasarkan gambar 4.3. pencarian nilai Levenshtein Distance dapat dilakukan dengan memasukkan kedalam tabel kemudian melakukan perhitungan sebagai berikut:

**Tabel 4.3 Inisialisasi Awal *Levenshtein Distance***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | - | K | A | M | I |
| - | 0 | 1 | 2 | 3 | 4 |
| K | 1 |  |  |  |  |
| A | 2 |  |  |  |  |
| M | 3 |  |  |  |  |
| U | 4 |  |  |  |  |

B­erdasarkan tabel 4.3, inisialisasi awal dilakukan dengan menyusun kata pertama secara vertikal dan kata kedua secara horizontal, lalu diberikan nilai secara berurutan pada setiap katanya.

**Tabel 4.4 Pencarian Baris Pertama *Levenshtein Distance***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | - | K | A | M | I |
| - | 0 | 1 | 2 | 3 | 4 |
| K | 1 | 0 | 1 | 2 | 3 |
| A | 2 |  |  |  |  |
| M | 3 |  |  |  |  |
| U | 4 |  |  |  |  |

Pada tabel 4.4 terdapat nilai “0” yang telah didapat dari pencarian nilai minimum dari nilai kotak diatasnya yang telah ditambahkan 1 dan disampingnya yang telah ditambahkan 1 beserta nilai kotak diatasnya yang ditambahkan 1 bila huruf yang dicocokkan bernilai sama.

**Tabel 4.5 Pencarian Baris Kedua *Levenshtein Distance***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | - | K | A | M | I |
| - | 0 | 1 | 2 | 3 | 4 |
| K | 1 | 0 | 1 | 2 | 3 |
| A | 2 | 1 | 0 | 1 | 2 |
| M | 3 |  |  |  |  |
| U | 4 |  |  |  |  |

Pada tabel 4.5 terbapat lanjutan baris yang dilakukan sama dengan cara sebelumnya. Dengan mencari nilai minimum dari beberapa nilai kotak.

**Tabel 4.6 Mendapatkan Nilai *Levenshtein Distance***

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | - | K | A | M | I |
| - | 0 | 1 | 2 | 3 | 4 |
| K | 1 | 0 | 1 | 2 | 3 |
| A | 2 | 1 | 0 | 1 | 2 |
| M | 3 | 2 | 1 | 0 | 1 |
| U | 4 | 3 | 2 | 1 | 1 |

Pada tabel 4.6 terdapat hasil dari algoritma ini dengan mengisi semua kotak. Hasil dari algoritma ini dapat dilihat pada nilai akhir yang telah didapatkan yaitu “1”. Dengan ini menyatakan hasil dari algoritma *Levenshtein Distance* dari perbandingan kata “kamu” dan “kami” bernilai “1”.

Tahap selanjutnya memasukkan hasil *Levenshtein Distance* kedalam *similarity* sebagai berikut:

**(4.1)**

Keterangan:

* *Similarity* = nilai kemiripan
* Dis = nilai *Levenshtein Distance*
* MaxLength = jumlah string terpanjang

Sim = 1 – () = 0.75

Dengan menggunakan rumus *similarity*, nilai kemiripan dari kata “KAMU” dan “KAMI” menghasilkan nilai 0.75.

* + 1. ***Unified Modeling Language* (UML)**

Aplikasi dibangun dengan menggunakan *Unified Modeling Language* (UML). UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram yang terdiri dari *Use Case Diagram, Activity Diagram, Class Diagram* dan *Sequence Diagram.*

* + - 1. ***Use Case Diagram***

*Use Case Diagram* adalah sebuah diagram yang dapat merepresentasikan interaksi yang terjadi antara user dengan sistem. *Use Case Diagram* ini mendeskripsikan siapa saja yang menggunakan sistem dan bagaimana cara mereka berinteraksi dengan sistem. *Use Case Diagram* dari sistem yang akan dibangun dapat ditunjukkan pada Gambar 4.3.



**Gambar 4.3 *Use Case Diagram* Aplikasi**

**Tabel 4.7 Keterangan *Use Case Diagram***

|  |  |
| --- | --- |
| **Aktor** | **Sistem** |
| *User* memilih menu beranda | Sistem akan menampilkan menu beranda. |
| *User* memilih pencarian | Sistem akan menyiapkan *form* pencarian*.* |
| *User* memilih menu petunjuk | Sistem akan menampilkan menu petunjuk bantuan penggunaan aplikasi. |
| *User* memilih menu tentang | Sistem akan menampilkan menu tentang aplikasi. |
| *User* memilih menu login | Sistem akan menampilkan *form* *login*. |
| *User* memilih menu cari data | Sistem akan menampilkan form beserta pilihan *website* yang akan diambil datanya. |
| *User* memilih menu lihat data | Sistem akan menampilkan data-data yang tersedia pada *website*. |
| *User* memilih menu halaman admin | Sistem akan menampilkan halaman awal admin |

* + - 1. ***Activity Diagram***

*Activity diagram* menggambarkan berbagai alur aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity* *diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Berikut ini adalah *activity diagram* yang akan menggambarkan alir aktivitas sistem.

1. ***Activity Diagram* Beranda**

Gambar 4.4 merupakan *activity diagram* yang menunjukkan aktivitas *user* ketika memilih menu beranda, lalu sistem akan menampilkan menu beranda yang berisikan tampilan menu beranda.



**Gambar 4.4 *Activity Diagram* Beranda**

1. ***Activity Diagram* Pencarian**

Gambar 4.5 tersebut merupakan *activity diagram* yang menunjukkan aktivitas *user* ketika telah memilih menu cariproduk*,* lalu sistem akan menampilkan *form input* kata, *user* kemudian memasukkan kata yang dicari lalu menekan tombol cari dan sistem akan menampilkan daftar produk yang dicari.

**Gambar 4.5 *Activity Diagram* Pencarian**

1. ***Activity Diagram* Petunjuk**

Gambar 4.6 merupakan gambar *activity diagram* yang menunjukkan aktivitas *user* memilih menu petunjuk, maka sistem akan menampilkan menu petunjuk. Setelah itu, menu petunjuk tersebut akan menampilkan cara penggunaan sistem.

 **Gambar 4.6 *Activity Diagram* Petunjuk**

1. ***Activity Diagram* Tentang**

Gambar 4.7 merupakan gambar *activity diagram* yang menunjukkan aktivitas *user* memilih menu tentang, maka sistem akan menampilkan menu tentang. Setelah itu, menu tersebut akan menampilkan penjelasan sistem.

 **Gambar 4.7 *Activity Diagram* Tentang**

1. ***Activity Diagram* Login**

Gambar 4.8 merupakan gambar *activity diagram* yang menunjukkan aktivitas adminmemilih menu *login*, maka sistem akan menampilkan *form login*. Setelah itu, admin mengisi *form login* untuk masuk ke halaman admin.



**Gambar 4.8 *Activity Diagram Login***

1. ***Activity Diagram* Cari Data**

Gambar 4.9 merupakan gambar *activity diagram* yang menunjukkan aktivitas admin yang telah masuk ke halaman admin untuk memilih menu cari data, maka sistem akan menampilkan *form* cari data. Setelah itu, admin mengisi *form* sesuai kebutuhan.



**Gambar 4.9 *Activity Diagram* Cari Data**

1. ***Activity Diagram* Lihat Data**

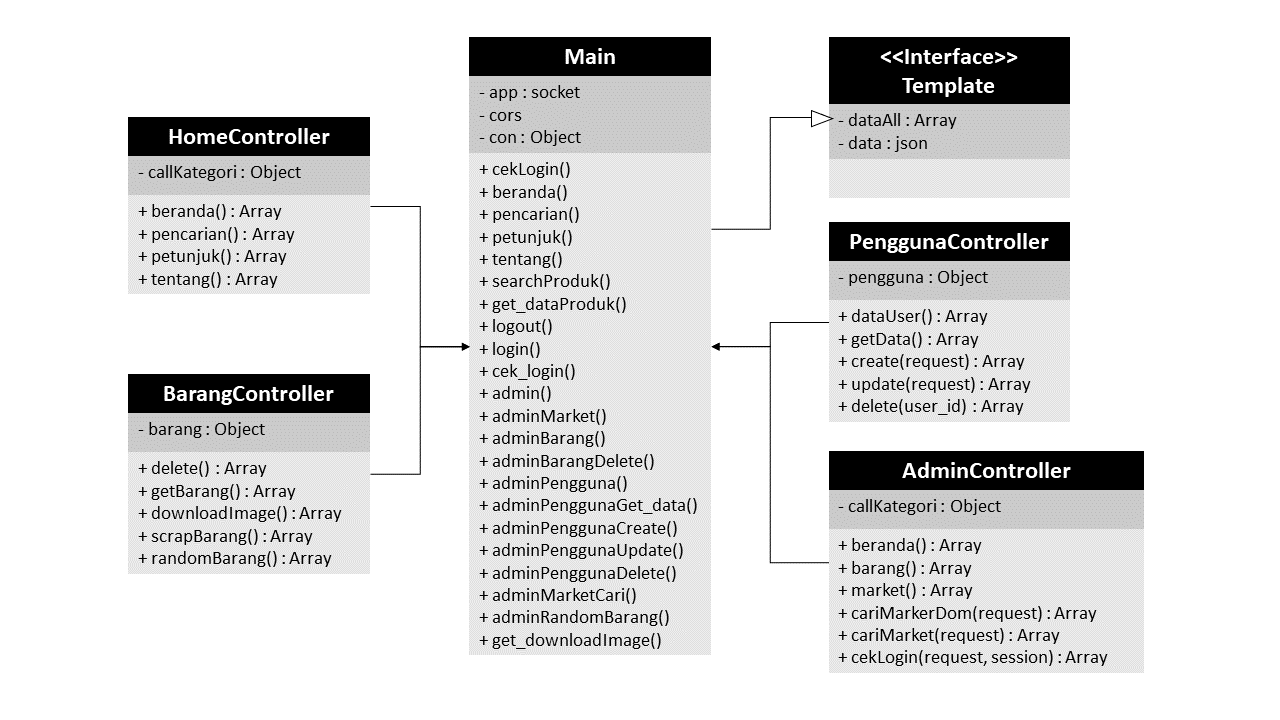
Gambar 4.10 merupakan gambar *activity diagram* yang menunjukkan aktivitas admin yang telah masuk ke halaman admin untuk memilih menu lihat data, maka sistem akan menampilkan daftar data yang ada.



**Gambar 4.10 *Activity Diagram* Lihat Data**

* + - 1. ***Class Diagram***

*Class diagram* merupakan diagram yang selalu ada dipemodelan sistem berorientasi objek. *Class diagram* menunjukkan hubungan antar *class* dalam sistem yang sedang dibangun dan bagaimana mereka saling berkolaborasi untuk mencapai suatu tujuan. Berikut ini adalah *class diagram* sistem:

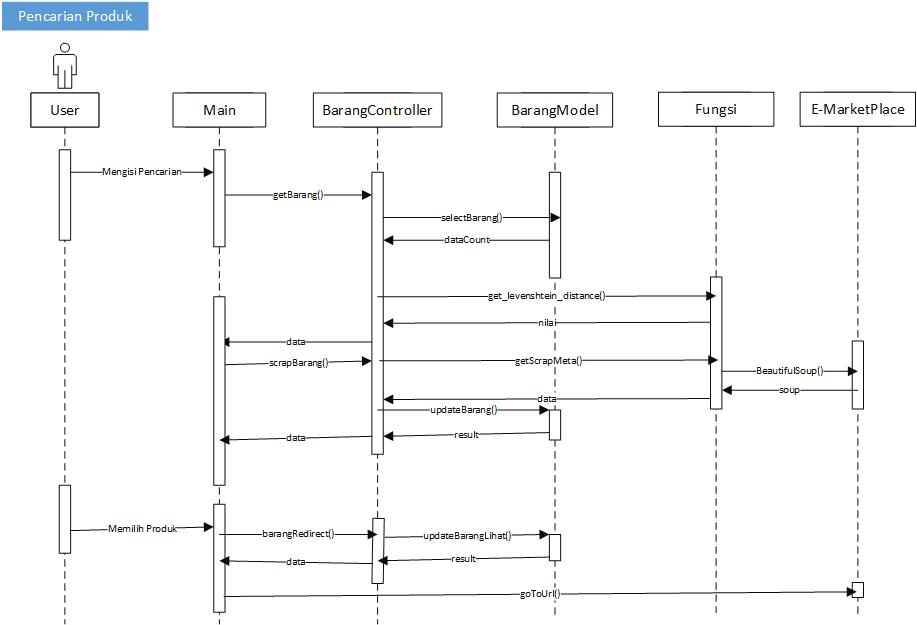
****

**Gambar 4.11 *Class Diagram* Aplikasi Pencarian**

* + - 1. ***Sequence Diagram***

*Sequence Diagram* menggambarkan interaksi antar objek di dalam dan di sekitar sistem yang digambarkan terhadap waktu. Berikut ini adalah *Sequence Diagram* yang akan menggambarkan interkasi antar objek dan sistem.

1. ***Sequence Diagram* Pencarian Produk**

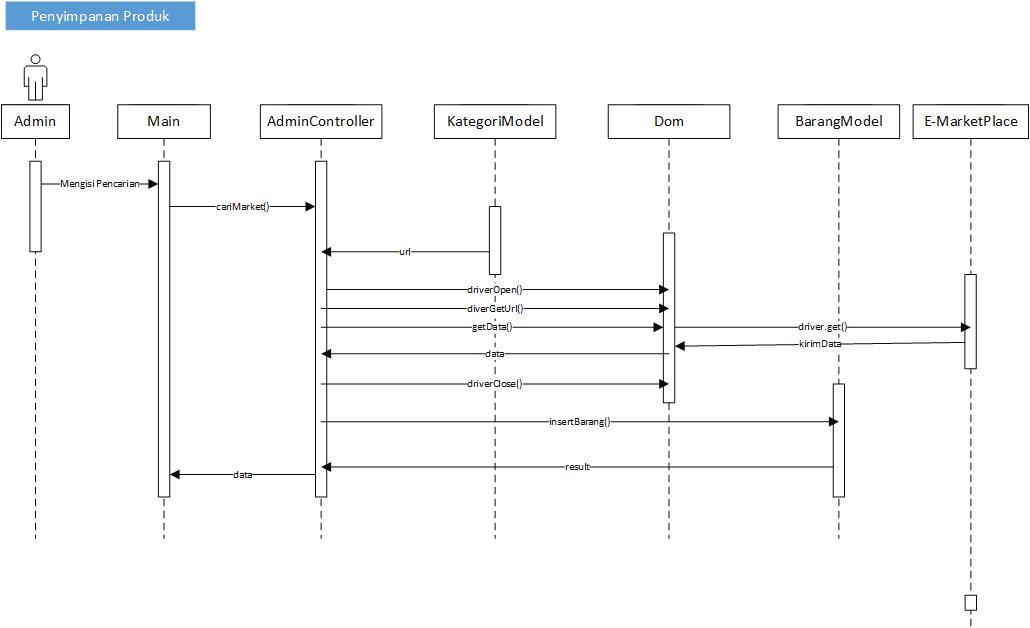
******

**Gambar 4.12 *Sequence Diagram* Pencarian Produk**

Gambar 4.12 adalah *sequence* *diagram* yang menunjukkan hubungan dari sistem terhadap *user*, dimana sistem akan menampilkan *form* pencarian, lalu *user* memasukkan kata kunci untuk mencari barang yang diinginkan.

1. ***Sequence Diagram* Penyimpanan Produk**

Gambar 4.13 adalah diagram *sequence* yang menunjukkan hubungan dari sistem terhadap *admin*, dimana *admin* memilih menu cari data. Kemudian *admin* memilih *website* yang akan secara otomatis mencari data pada *website* tersebut.

******

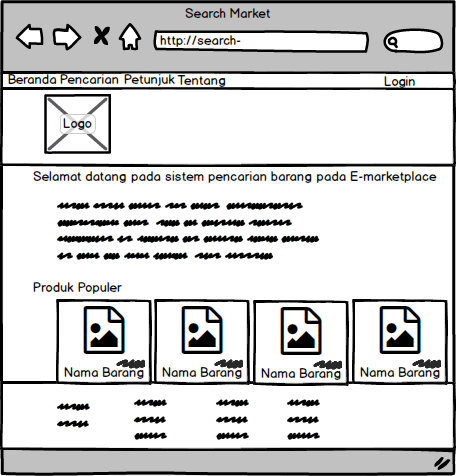
**Gambar 4.13 *Sequence Diagram* Penyimpanan Produk**

* 1. **Perancangan Antarmuka (Interface)**

Perancangan *user interface* adalah tahapan pembuatan antarmuka yang akan digunakan pada pembangunan aplikasi pencarian barang di berbagai *e-marketplace* yang dibagi menjadi sembilan bagian yaitu perancangan menu beranda, *form* pencarian, menu petunjuk, menu tentang, *form* login, menu halaman admin, *form* cari data dan *form* lihat data.

* + 1. **Rancangan Menu Beranda**

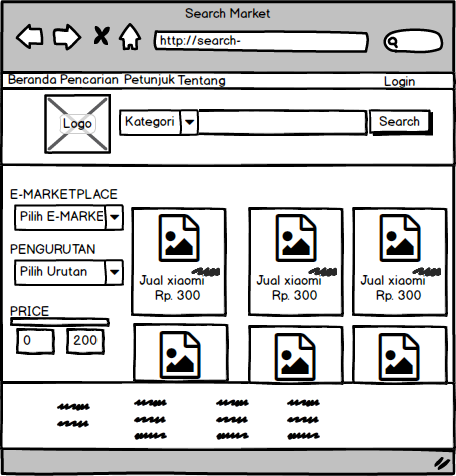
Menu beranda adalah tampilan pertama yang akan digunakan pada *website*. Pada menu beranda berisi logo aplikasi.



**Gambar 4.14 Rancangan Menu Beranda**

* + 1. **Rancangan *Form* Pencarian**

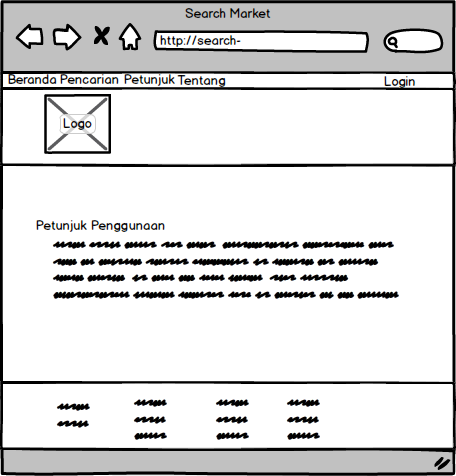
Pada *form* pencarian *user* memasukkan kata kunci untuk mencari produk yang sesuai dengan kata kunci. Kemudian sistem akan menampilkan daftar produk yag terkait.

****

**Gambar 4.15 Rancangan *Form* Pencarian**

* + 1. **Rancangan Menu Petunjuk**

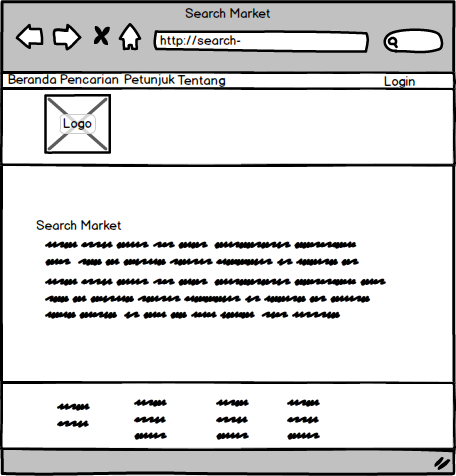
Pada saat *user* memilih menu petunjuk, sistem akan menampilkan petunjuk penggunaan sistem. Disini akan dijelaskan cara-cara penggunaan aplikasi untuk mencari produk.

****

**Gambar 4.16 Rancangan Menu Petunjuk**

* + 1. **Rancangan Menu Tentang**

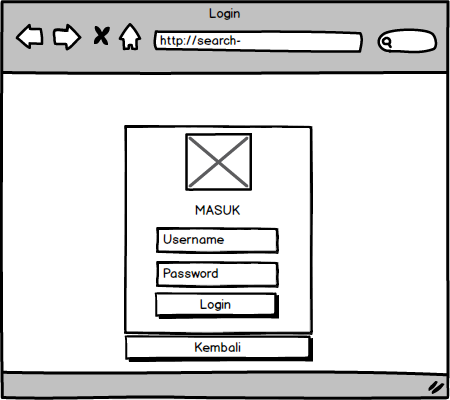
Pada saat *user* memilih menu tentang, sistem akan menampilkan penjelasan mengenai deskripsi sistem. Sehingga *user* dapat mengetahui manfaat dari sistem ini.



**Gambar 4.17 Rancangan MenuTentang**

* + 1. **Rancangan *Form* *Login***

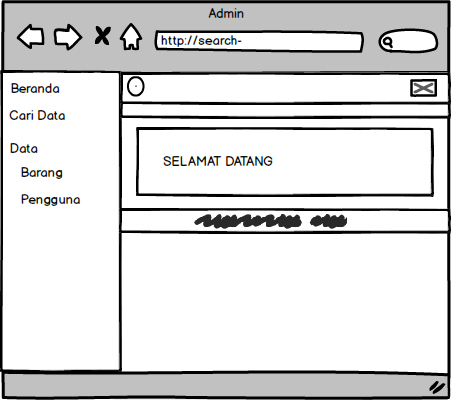
Pada saat *user* memilih menu login, sistem akan menampilkan *form* *login* sebagai *admin*. Dengan cara memasukkan *username* dan *password* dengan benar.

****

**Gambar 4.18 Rancangan *Form Login***

* + 1. **Rancangan Menu Halaman Admin**

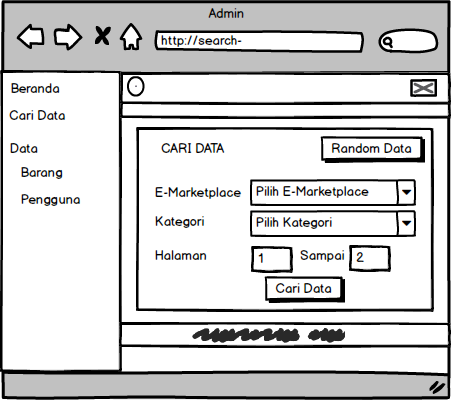
Pada saat *user* memilih menu Halaman Admin, sistem akan menampilkan tampilan awal setelah melakukan *login*. Menu ini berisi ucapan selamat dating.

****

**Gambar 4.19 Rancangan MenuHalaman Admin**

* + 1. **Rancangan Menu Cari Data**

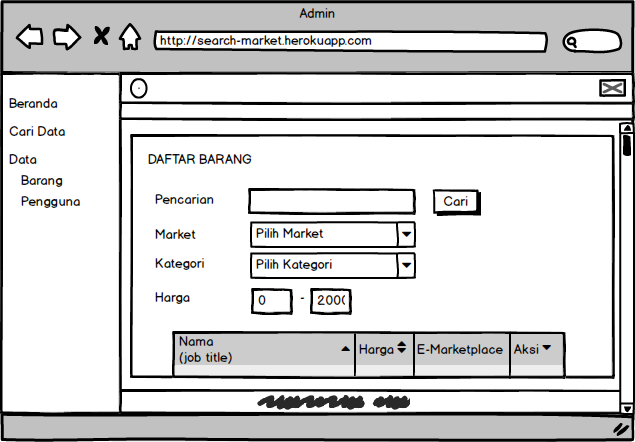
Pada saat *user* memilih menu Cari Data, sistem akan menampilkan pilihan website yang akan diambil datanya dengan menggunakan teknik *web scraping*.

****

**Gambar 4.20 Rancangan MenuCari Data**

* + 1. **Rancangan Menu Barang**

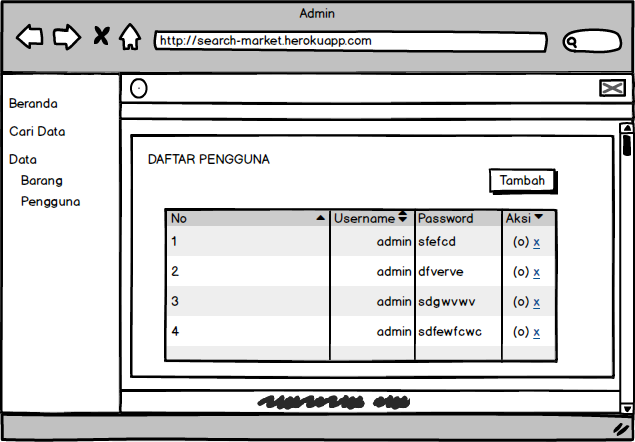
Pada saat *user* memilih menu Barang, sistem akan menampilkan daftar produk yang ada pada *database*.

****

**Gambar 4.21 Rancangan MenuBarang**

* + 1. **Rancangan Menu Pengguna**

Padas sat *user* memilih menu Pengguna, sistem akan menampilkan daftar pengguna yang berguna sebagai login *user*.



**Gambar 4.22 Rancangan MenuPengguna**

* 1. **Metode Pengujian**

Pengujian merupakan metode yang dilakukan untuk menjelaskan mengenai pengoperasian perangkat lunak yang terdiri dari perangkat pengujian, metode pengujian dan pelaksanaan pengujian.

Pengujian program ini menggunakan metode *black box*. Pengujian *black box* merupakan pengujian program berdasarkan fungsi dari program. Tujuan dari metode *black box* ini adalah untuk menemukan kesalahan fungsi pada program. Pengujian dengan metode ini dilakukan dengan cara memberikan sejumlah *input* pada program aplikasi yang kemudian diproses sesuai dengan kebutuhan fungsionalnya untuk melihat apakah program aplikasi menghasilkan keluaran yang diinginkan dan sesuai dengan fungsi dari program tersebut. Apabila dari masukan yang diberikan proses menghasilkan keluaran yang sesuai dengan kebutuhan fungsionalnya, maka program aplikasi yang bersangkutan telah benar, tetapi jika keluaran yang dihasilkan tidak sesuai dengan kebutuhan fungsionalnya, maka masih terdapat kesalahan pada program aplikasi.

Pengujian dilakukan dengan mencoba semua kemungkinan yang terjadi dan pengujian dilakukan berulang-ulang. Jika dalam pengujian ditemukan kesalahan, maka akan dilakukan penelusuran dan perbaikan untuk memperbaiki kesalahan yang terjadi. Jika telah selesai melakukan perbaikan, maka akan dilakukan pengujian kembali. Pengujian dan perbaikan dilakukan secara terus menerus hingga diperoleh hasil yang terbaik.

* 1. **Rencana Pengujian**

Pengujian perangkat lunak berikut menggunakan data uji berdasarkan dari masing-masing data. Rencana selengkapnya dapat dilihat pada tabel berikut:

**Tabel 4.8 Rencana Pengujian Black Box**

|  |  |  |
| --- | --- | --- |
| **Kelas Uji** | **Detail Uji** | **Jenis Pengujian** |
| Pengujian Menu | Beranda | *Black Box* |
| Pencarian | *Black Box* |
| Petunjuk | *Black Box* |
| Tentang | *Black Box* |
| Login | *Black Box* |
| Pengujian Menu Admin | Halaman Admin | *Black Box* |
| Lihat Data | *Black Box* |
| Cari Data | *Black Box* |
| Pengujian Algoritma *Levenshtein Distance* | Pencarian data yang ada di *database* | *Black Box* |
| Pengujian Teknik *Web Scraping* | Pencarian data pada website luar. | *Black Box* |

Setelah melakukan pengujian *black box*, sistem akan dilakukan pengujian kembali kepada para pengguna agar mengetahui pendapat pengguna mengenai sistem yang dikembangkan. Pengguna melakukan pengisian kuesioner untuk memberikan pendapatnya terhadap sistem. Kuesioner terdiri dari 11 pertanyaan dengan 5 opsi yang dapat dipilih sesuai tanggapan oleh pengguna. Setelah mendapatkan respon dari pengguna, data tersebut akan dihitung tinggat kesuksesan sistem sesuai pada tabel 4.9.

**Tabel 4.9 Rencana Kuesioner**

|  |  |  |  |
| --- | --- | --- | --- |
| **Optional** | **Bobot** | **Interval Interprestasi skor** | **Jenis Pengujian** |
| Sangat Setuju | 5 | 80% – 100% | Kuesioner |
| Setuju | 4 | 60% – 79,99% | Kuesioner |
| Netral | 3 | 40% – 59,99% | Kuesioner |
| Tidak Setuju | 2 | 20% – 39,99% | Kuesioner |
| Sangat Tidak Setuju | 1 | 0% – 19,99% | Kuesioner |
| **Rumus memperoleh skor** |  | | Kuesioner |