PEMANFAATAN API MIDTRANS SEBAGAI PAYMENT GATEWAY PADA PEMBUATAN WEBSITE *E-COMMERCE* MENGGUNAKAN FRAMEWORK LARAVEL (STUDI KASUS : CHESSPEDIA)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Reza Yusuf Maulana
NIM: 165150700111025

PROGRAM STUDI TEKNOLOGI INFORMASI
JURUSAN SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
TAHUN 2021

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

PENGESAHAN

PEMANFAATAN *API* MIDTRANS SEBAGAI *PAYMENT GATEWAY* PADA PEMBUATAN WEBSITE *E-COMMERCE* MENGGUNAKAN *FRAMEWORK* LARAVEL
(STUDI KASUS : CHESSPEDIA)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Reza Yusuf Maulana
NIM: 165150700111025

Skripsi ini telah diuji dan dinyatakan lulus pada
10 Juni 2021
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing 2
Dian Eka Ratnawati, S.Si., M.Kom	Issa Arwani, S.Kom., M.Sc.
NIP. 197306192002122001	NIP. 198309222012121003

Mengetahui
Ketua Jurusan Sistem Informasi

KEMENTERIAN PENDIDIKAN DAN KEBUDAYAAN UNIVERSITAS BRAWIJAYA FAKULTAS ILMU KOMPUTER JURUSAN SISTEM INFORMASI

Issa Arwani, S.Kom., M.Sc.
NIP. 198309222012121003

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA
Universitas Brawijaya

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur- unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 10 Juli 2021

Reza Yusuf Maulana

NIM: 165150700111025

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

PRAKATA

Puji Syukur penulis panjatkan kepada Tuhan Yang Maha Esa, karena dengan rahmat dan karunia-Nya, penulis dapat menyelesaikan penelitian yang berjudul "Pemanfaatan *Api* Midtrans Sebagai *Payment Gateway* Pada Pembuatan Website *E-Commerce* Menggunakan *Framework Laravel* (Studi Kasus : Chesspedia)". Penelitian ini dapat diselesaikan berkat dukungan, doa, dan bantuan dari banyak pihak. Penulis ingin menyampaikan rasa terima kasih kepada:

1. Ibunda tercinta yang selalu memberikan semangat, doa, dan kasih sayang yang tulus.
2. Bu Dian Eka Ratnawati, S.Si., M.Kom selaku dosen pembimbing satu yang telah bersedia membimbing dan mengarahkan penulis selama pelaksanaan penelitian ini.
3. Bapak Issa Arwani, S.Kom., M.Sc. selaku Ketua Jurusan Sistem Informasi sekaligus dosen pembimbing dua yang telah bersedia membimbing dan mengarahkan penulis selama pelaksanaan penelitian ini.
4. Segenap pengelola Chesspedia yang telah bekerjasama untuk menyelesaikan penelitian ini.
5. Ayahanda, Ibunda, , Shofya, Tere, Rocha, dan teman-teman lainnya atas perhatian dan kesabarannya dalam mendidik penulis, serta yang senantiasa tiada henti - hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini

Penulis menyadari bahwa penelitian ini masih belum sempurna karena keterbatasan ilmu dan pengalaman yang dimiliki. Oleh karena itu, semua kritik dan saran yang membangun akan penulis terima dengan senang hati. Semoga, penelitian ini dapat bermanfaat bagi siapapun yang memerlukannya.

Malang, 10 Juli 2021

Penulis

Reza.yusuf98@student.ub.ac.id

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

ABSTRAK

Reza Yusuf Maulana, Pemanfaatan Api Midtrans Sebagai *Payment Gateway* Pada Pembuatan Website *E-Commerce* Menggunakan Framework Laravel (Studi Kasus : Chesspedia)

Pembimbing: Dian Eka Ratnawati, S.Kom., M.Kom. dan Issa Arwani, S.Kom., M.Sc.

Chesspedia, merupakan merek catur yang berlokasi di Jawa Timur. Saat ini, proses pemesanan Chesspedia dilakukan melalui *whatsapp* dan web Chesspedia yang dapat diakses pada link *https://chesspedia.my.id*. Untuk pemesanan melalui web Chesspedia, pemilik memiliki kendala yang cukup signifikan, karena web *e-commerce* Chesspedia saat ini hanya mendukung pembayaran via transfer saja. Hal ini mengakibatkan proses verifikasi pembayaran pesanan memerlukan waktu yang cukup lama. Kendala proses pemesanan dan pembayaran yang dialami oleh pembeli menyebabkan Chesspedia kehilangan banyak calon pembeli. Untuk mengatasi kendala yang terjadi, dibangun sebuah *website e-commerce* dengan menggunakan metode *waterfall* lengkap dengan sistem manajemen pesanan, pengiriman dan pembayaran. Selain itu juga dilakukan integrasi dengan Midtrans, sebuah penyedia layanan *payment gateway*, untuk sistem pembayaran otomatis. Pengujian fungsionalitas sistem menggunakan pengujian validasi kepada pengguna *website e-commerce* chesspedia untuk keseluruhan fungsi sistem diperoleh hasil 100% valid. Pada pengujian non-fungsionalitas sistem memakai metode *System Usability Scale* (SUS) dengan responden pengguna *website* chesspedia. Hasil pengujian *website e-commerce chesspedia* mendapatkan *score* 74 dengan kategori *acceptable.* Dan pengujian menggunakan *Compatibility Testing,* website dapat dijalankan di berbagai macam jenis browser.

Kata kunci: *E-commerce*, *Payment Gateway*, *Waterfall*, *Black Box Testing*, *Usability Testing, Compatibility Testing*

ABSTRACT

Reza Yusuf Maulana, Utilizing Midtrans Api as a *Payment Gateway* for *E-Commerce* Website Development Using the Laravel Framework (Case Study: Chesspedia)

Supervisor : Dian Eka Rahmawati, S.Kom., M.Kom. and Issa Arwani, S.Kom., M.Sc.

Chesspedia, is a chess brand located in East Java. Currently, Chesspedia ordering process is carried out via WhatsApp and the Chesspedia web which can be accessed on https://chesspedia.my.id . For ordering via the Chesspedia web, the owner has significant obstacles, because the Chesspedia e-commerce web currently only supports payments via transfer only. This resulted in the order payment verification process taking quite a long time. Obstacles in the ordering and payment process experienced by buyers have caused Chesspedia to lose many prospective buyers. To overcome the obstacles that occur, an e-commerce website was built using the waterfall method complete with an order management system, shipping and payment. In addition, integration with Midtrans, a payment gateway service provider, for automatic payment systems is also carried out. Testing system functionality using validation testing to users of the Chesspedia e-commerce website for all system functions obtained 100% valid results. In testing the non-functionality of the system using the System Usability Scale (SUS) method with chesspedia website users as respondents. Chesspedia e-commerce website test results get a score of 74 in the acceptable categories. testing using Compatibility Testing, the website can be run on various types of browsers.

Keyword: *E-commerce*, *Website*, *Payment Gateway*, *Waterfall Method*, *black box testing*, *Usability Testing*, *Compatibility Testing*

DAFTAR ISI

ABSTRAK ..... **Error! Bookmark not defined.**
ABSTRACT vi
DAFTAR ISI vii
DAFTAR GAMBAR xi
DAFTAR TABEL xiii
BAB 1 PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Tujuan 2
1.4 Manfaat 3
1.5 Batasan Masalah 3
1.6 Sistematika Pembahasan 3
BAB 2 LANDASAN KEPUSTAKAAN 5
2.1 Kajian Pustaka 5
2.2 Chesspedia 6
2.3 E-Commerce 6
2.4 Rekayasa Perangkat Lunak 7
2.5 Software Development Life Cycle 7
2.5.1 Waterfall Model 8
2.5.2 *Unified Modeling Language* 10
2.5.3 Teknologi Pengembangan Sistem 13
2.6 Pengujian Perangkat Lunak 18
2.6.1 Blackbox Testing 19
2.6.2 Usability Testing 19
2.6.3 Compatibility Testing 21
BAB 3 METODOLOGI 22
3.1 Studi Literatur 22
3.2 Pengumpulan Data 23
3.3 Analisis Kebutuhan 23
3.4 Perancangan Sistem 23
3.5 Implementasi 24

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

3.6 Pengujian 24
3.7 Kesimpulan dan Saran 25
BAB 4 ANALISIS KEBUTUHAN 26
4.1 Deskripsi Umum Sistem 26
4.2 Identifikasi Aktor 26
4.3 Identifikasi Kebutuhan 27
4.3.1 Kebutuhan Fungsional 27
4.3.2 Kebutuhan Non-Fungsional 29
4.4 Pemodelan *Use Case* 29
4.4.1 Use Case Diagram 29
4.4.2 Use Case Scenario 30
BAB 5 PERANCANGAN SISTEM 44
5.1 Arsitektur Sistem 44
5.2 Sequence Diagram 44
5.2.1 *Sequence Diagram Register* 44
5.2.2 *Sequence Diagram Login* 45
5.2.3 *Sequence Diagram Logout* 46
5.2.4 *Sequence Diagram* Melihat Daftar Seluruh Produk 46
5.2.5 *Sequence Diagram* Melihat Detail Produk 47
5.2.6 *Sequence Diagram* Melihat Keranjang Belanja 47
5.2.7 *Sequence Diagram* Menampilkan Halaman *Checkout* 48
5.2.8 *Sequence Diagram* Melakukan *Checkout* 48
5.2.9 *Sequence Diagram* Melihat User Area 49
5.2.10 *Sequence Diagram* Melakukan Pembayaran 50
5.2.11 *Sequence Diagram* Melihat *Dashboard* Admin 51
5.2.12 *Sequence Diagram* Mengelola Produk 52
5.2.13 *Sequence Diagram* Mengelola Pesanan 53
5.2.14 *Sequence Diagram* Mengelola Pengiriman 54
5.3 *Class* Diagram 55
5.4 Perancangan Basis Data 58
5.4.1 Entity Relationship Diagram 58
5.4.2 Physical Data Model 58

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

5.5 Perancangan Antarmuka Sistem 59
5.5.1 Tampilan Halaman *Dashboard* Admin 59
5.5.2 Tampilan Halaman Pengaturan Produk 60
5.5.3 Tampilan Halaman Pesanan 61
5.5.4 Tampilan Halaman Pengiriman 61
5.5.5 Tampilan Halaman Awal 62
5.5.6 Tampilan Halaman Seluruh Daftar Produk 62
5.5.7 Tampilan Halaman Detail Produk 63
5.5.8 Tampilan Halaman Keranjang Belanja 63
BAB 6 Implementasi 65
6.1 Spesifikasi Lingkungan Implementasi 65
6.1.1 Spesifikasi Perangkat keras 65
6.1.2 Spesifikasi Perangkat Lunak 65
6.2 Implementasi Basis Data 65
6.3 Implementasi Kode Program 67
6.3.1 Integrasi *API* Midtrans 67
6.3.2 Integrasi *API* Raja Ongkir 71
6.3.3 Implementasi Fungsi *Register*, *Login* dan *Logout* 74
6.3.4 Implementasi Tampilkan Seluruh Produk 76
6.3.5 Implementasi Fungsi Keranjang Belanja 77
6.3.6 Implementasi Fungsi *Checkout* 79
6.3.7 Implementasi Fungsi Order 80
6.3.8 Implementasi Fungsi Pembayaran 82
6.3.9 Implementasi Fungsi Dashboard Admin **Error! Bookmark not defined.**
6.4 Implementasi Antarmuka 84
6.4.1 *Dashboard* Admin 84
6.4.2 Menu Produk 85
6.4.3 Menu Kategori 85
6.4.4 Menu Attribut 86
6.4.5 Menu Pesanan 86
6.4.6 Menu *Trash* 87
6.4.7 Menu Pengiriman Pesanan (*Shipment)* 87

6.4.8 Menu Laporan Pendapatan (*Revenue Reports)* 88
6.4.9 Menu Laporan Produk (Products Reports) 88
6.4.10 Menu Laporan Inventori *(Inventory Report)* 89
6.4.11 Menu Laporan Pembayaran *(Payments Reports)* 90
6.4.12 Menu Pengaturan *Slide* 90
6.4.13 Menu Pengaturan User 91
6.4.14 Menu Pengaturan *Role* 91
6.4.15 Halaman Utama Chesspedia 92
6.4.16 Halaman Seluruh Produk 93
6.4.17 Halaman Detail Produk 94
BAB 7 Pengujian Sistem 95
7.1 Pengujian Fungsional 95
7.2 Pengujian Non-Fungsional 105
7.2.1 Usability Testing 106
7.2.2 Compatibility Testing 107
BAB 8 KESIMPULAN DAN SARAN 110
8.1 Kesimpulan 110
8.2 Saran 110
DAFTAR REFERENSI 112
LAMPIRAN A WAWANCARA TERHADAP PEMILIK CHESSPEDIA 114
LAMPIRAN B RESPONDEN SUS 1 116
LAMPIRAN C RESPONDEN SUS 2 117
LAMPIRAN D RESPONDEN SUS 3 118
LAMPIRAN E RESPONDEN SUS 4 119
LAMPIRAN F RESPONDEN SUS 5 120

DAFTAR GAMBAR

Gambar 2.1 Diagram *Waterfall Model* 8
Gambar 2.2 *Use Case Relationship Web Service* 14
Gambar 2.3 Model dasar RESTful *Web service* 15
Gambar 2.4 Contoh Struktur Data JSON 17
Gambar 2.5 *Sistem Usability Scale* 20
Gambar 3.1 Diagram Alir Metode Penelitian 22
Gambar 4.1 Kode Kebutuhan 27
Gambar 4.2 *Use Case Diagram* pada Website Chesspedia 30
Gambar 5.1 Arsitektur Sistem 44
Gambar 5.2 *Sequence Diagram Register* 45
Gambar 5.3 *Sequence Diagram Login* 46
Gambar 5.4 *Sequence Diagram Logout* 46
Gambar 5.5 *Sequence Diagram* Melihat Daftar Seluruh Produk 47
Gambar 5.6 *Sequence Diagram* Melihat Detail Produk 47
Gambar 5.7 *Sequence Diagram* Melihat Keranjang Belanja 48
Gambar 5.8 *Sequence Diagram* Tampilkan Halaman *Checkout* 48
Gambar 5.9 *Sequence Diagram* Tampilkan *Checkout* 49
Gambar 5.10 Melihat Tampilan Menu User 50
Gambar 5.11 *Sequence Diagram* Pembayaran 51
Gambar 5.12 *Sequence Diagram* Melihat *Dashboard* Admin 52
Gambar 5.13 *Sequence Diagram* Mengelola Produk 53
Gambar 5.14 *Sequence Diagram* Mengelola Pesanan 54
Gambar 5.15 *Sequence Diagram* Mengelola Pengiriman 55
Gambar 5.16 *Class* Diagram Website *E-Commerce* Chesspedia 57
Gambar 5.17 *Entity Relationship Diagram Website E-Commerce* Chesspedia 58
Gambar 5.18 Physical Data Model Website *E-commerce* Chesspedia 59
Gambar 5.19 *Dashboard* Halaman Admin 60
Gambar 5.20 Halaman Pengaturan Produk 60
Gambar 5.21 Tampilan Halaman Pesanan 61
Gambar 5.22 Tampilan Halaman Pengiriman 62

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

Gambar 5.23 Tampilan Halaman Awal 62
Gambar 5.24 Tampilan Seluruh Produk 63
Gambar 5.25 Tampilan Halaman Detail Produk 63
Gambar 5.26 Tampilan Halaman Keranjang Belanja 64
Gambar 6.1 Daftar Tabel Pada DBMS 66
Gambar 6.2 Tampilan *Dashboard* Panel Admin 85
Gambar 6.3 Menu Produk 85
Gambar 6.4 Menu Kategori 86
Gambar 6.5 Menu Attribut 86
Gambar 6.6 Menu Pesanan 87
Gambar 6.7 Menu *Trash* 87
Gambar 6.8 Menu Pengiriman Pesanan (*Shipment*) 88
Gambar 6.9 Menu Laporan Pendapatan (*Revenue Reports*) 88
Gambar 6.10 Menu Laporan Produk (Products Reports) 89
Gambar 6.11 Menu Laporan Inventori *(Inventory Report)* 90
Gambar 6.12 Menu Laporan Pembayaran *(Payments Reports)* 90
Gambar 6.13 Menu Pengaturan Slide 91
Gambar 6.14 Menu Pengaturan *Role* 92
Gambar 6.15 Tampilan Halaman Utama Chesspedia 93
Gambar 6.16 Halaman Seluruh Produk 94
Gambar 6.17 Halaman Detail Produk 94
Gambar 7.1 Aturan Penomoran Pengujian Fungsional 95
Gambar 7.2 Hasil Pengujian *Compatibility Testing* 108
Gambar 7.3 *Critical Issues* Pada Hasil Pengujian Compatibility Testing 108
Gambar 7.4 *Major & Minor Issues* Pada Hasil Pengujian Compatibility Testing 109

DAFTAR TABEL

Tabel 2.1 *use case diagram* 10
Tabel 2.2 Sequence Diagram 11
Tabel 2.3 *Class Diagram* 13
Tabel 2.4 Metode HTTP dan Penggunaannya dalam REST 16
Tabel 4.1 Identifikasi Aktor 26
Tabel 4.2 Kebutuhan Fungsional Pengunjung & Pelanggan 27
Tabel 4.3 Kebutuhan Fungsional Admin 28
Tabel 4.4 Kebutuhan Non Fungsional 29
Tabel 4.5 *Use Case Scenario Register* 30
Tabel 4.6 *Use Case Scenario* Melihat Daftar Produk 31
Tabel 4.7 *Use Case Scenario* Melihat Detail Produk 32
Tabel 4.8 *Use Case Scenario* Mencari Produk 33
Tabel 4.9 *Use Case Scenario Login* 33
Tabel 4.10 *Use Case Scenario* Melakukan *Logout* 34
Tabel 4.11 Melihat Daftar Produk 35
Tabel 4.12 *Use Case Scenario* Melihat Detail Produk 35
Tabel 4.13 *Use Case Scenario* Menambah Produk ke Dalam Keranjang Belanja . 36
Tabel 4.14 *Use Case Scenario* Hitung Ongkos Kirim 36
Tabel 4.15 *Use Case Scenario* Melakukan *Checkout* 37
Tabel 4.16 Use Case Scenario Melakukan Pembayaran 38
Tabel 4.17 *Use Case Scenario* Mengubah Data Pribadi 39
Tabel 4.18 *Use Case Scenario* Melihat Riwayat Transaksi 39
Tabel 4.19 *Use Case Scenario* Melihat Status Transaksi 40
Tabel 4.20 *Use Case Scenario* Melakukan *Login* 41
Tabel 4.21 *Use Case Scenario* Mengelola Produk 41
Tabel 4.22 Mengelola Pesanan 42
Tabel 4.23 *Use Case Scenario* Mengelola Pengiriman 42
Tabel 4.24 *Use Case Scenario* Melakukan *Logout* 43
Tabel 6.1 Spesifikasi Perangkat Keras 65
Tabel 6.2 Kode Program *PaymentController.php* 67
Tabel 6.3 Kode Program *Library* Midtrans.php 69

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

Tabel 6.4 Kode Program *Shipping Controller* 71
Tabel 6.5 Potongan Kode Fungsi rajaOngkirRequest 71
Tabel 6.6 Potongan Fungsi *getProvinces* dan *getCity* 72
Tabel 6.7 Potongan Fungsi *Register* pada *register*Controller.php 74
Tabel 6.8 Potongan Kode Fungsi *Login* dan *Logout* pada *login*Controller.php 75
Tabel 6.9 Potongan Kode Fungsi Tampilkan Produk 76
Tabel 6.10 Implementasi Kode Program Fungsi Keranjang Belanja 77
Tabel 6.11 Implementasi Kode Fungsi *Checkout* 79
Tabel 6.12 Implementasi Potongan Kode Fungsi Order 80
Tabel 6.13 Implementasi Kode Program Fungsi Pembayaran 82
Tabel 7.1 Validasi Melakukan Proses *Register* 96
Tabel 7.2 Uji Validasi Melihat Daftar Seluruh Produk 96
Tabel 7.3 Validasi Melihat Halaman Detail Produk 97
Tabel 7.4 Uji Validasi Proses *Login* 98
Tabel 7.5 Uji Validasi Melakukan Proses *Logout* 98
Tabel 7.6 Uji Validasi Menambahkan Produk ke Keranjang 99
Tabel 7.7 Uji Validasi Melakukan *Checkout* 100
Tabel 7.8 Uji Validasi Melihat Riwayat Transaksi 101
Tabel 7.9 Uji Validasi Melakukan *Login* Admin 102
Tabel 7.10 Uji Validasi Mengelola Pesanan 103
Tabel 7.11 Uji Validasi Mengelola Produk 103
Tabel 7.12 Uji Validasi Mengelola Pengiriman 104
Tabel 7.13 Uji Validasi Mencetak Laporan Penjualan 104
Tabel 7.14 Uji Validasi Mencetak Laporan Stok Produk 105
Tabel 7.15 Daftar Pertanyaan Pengujian *Usability* menggunakan metode SUS. 106
Tabel 7.16 Hasil Kuisioner Responden terhadap pertanyaan Metode SUS. 107
Tabel 7.17 Hasil Perhitungan Skor Akhir Penilaian Kuisioner. 107

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Electronic Commerce (E-Commerce) adalah transaksi bisnis yang terjadi secara elektronik diantara organisasi maupun individual (Laudon dan Laudon, 2014). Sehingga, seluruh transaksi bisnis yang memakai teknologi internet bisa disebut sebagai *E-Commerce.* Dengan beragamnya platform *E-Commerce* yang menghadirkan pelbagai macam produk serta kemudahan dalam bertransaksi, menjadikan *E-Commerce* sangat diminati oleh pengguna transaksi bisnis. Dalam hal ini pengguna cukup memilih barang, membayar, dan menunggu barang yang dipesan datang. Sedangkan, penjual hanya perlu memasarkan produknya melalui media *E-Commerce* lalu menunggu pesanan masuk dan mengirimkannya sesuai dengan kurir pilihan pelanggan. Menurut survei GlobalWebIndex pada tahun 2020, Indonesia adalah negara dengan pengguna *E-Commerce* tertinggi di dunia. Sebanyak 93 persen pengguna internet di Indonesia pada kisaran umur 16 - 64 tahun, setidaknya pernah mencari atau membeli produk melalui *E- Commerce* (DataReportal, 2021).

Dunia *E-Commerce* saat ini telah mencakup banyak sektor, mulai dari sektor produk fashion, produk makanan, produk rumah tangga dan sebagainya. Salah satu kategori produk yang laris di Indonesia adalah produk mainan dan hobi. Hal ini dapat dibuktikan oleh data Statista (2021) yaitu pada Hari Belanja Online Nasional (Harbolnas) 12.12 tahun 2020, masyarakat Indonesia menghabiskan sekitar 29 Trilyun Rupiah untuk membeli produk mainan dan hobi pada *E-commerce* Indonesia. Satu diantara banyak produk pada kategori mainan dan hobi di *E-commerce* adalah produk catur. Di Indonesia sendiri terdapat beberapa merk catur yang memiliki pangsa pasar cukup besar, seperti Purling, Proteam, Staunton hingga DGT. Chesspedia juga merupakan produsen catur lokal berlokasi di Provinsi Jawa Timur.

Saat ini, proses pemesanan Chesspedia dilakukan melalui WhatsApp dan web Chesspedia yang dapat diakses pada link https://chesspedia.my.id. Untuk pemesanan melalui web Chesspedia, pemilik memiliki kendala yang cukup signifikan, karena web *E-Commerce* Chesspedia saat ini hanya mendukung pembayaran via transfer manual ke satu rekening bank saja. Hal ini mengakibatkan proses verifikasi pembayaran pesanan memerlukan waktu yang cukup lama, karena admin harus melakukan proses cek secara manual yaitu mengecek validasi bukti transfer yang sudah diberikan oleh pembeli untuk dapat dilanjutkan ke proses verifikasi pesanan pada *website*. Kendala proses pemesanan dan pembayaran yang dialami oleh pembeli dalam hal ini menyebabkan produk Chesspedia kehilangan banyak calon pembeli. Hal ini dapat dibuktikan dari data pemesanan pada web chesspedia pada kuartal 3 2020, yang mana dalam kurun waktu tersebut terdapat 11 pesanan yang di cancel akibat terkendala masalah metode pembayaran.

Untuk mengatasi kendala yang terjadi, Chesspedia memerlukan sebuah solusi atas permasalahan yang dialami dalam proses pembayarannya. Diperlukan jenis pembayaran yang beragam untuk memudahkan pengguna melakukan pembayaran. *Payment gateway* merupakan bentuk solusi dari permasalahan ini yaitu suatu sistem penyedia layanan *e-commerce* yang berfungsi sebagai jembatan antara situs web dan lembaga keuangan yang memproses transaksi secara online. *Payment gateway* dapat memberikan layanan transaksi secara efisien, mudah dan aman. Salah satu layanan *payment gateway* yang sering digunakan ialah midtrans. Midtrans merupakan sistem pembayaran yang dapat memfasilitasi penjual dan pembeli untuk melakukan transaksi (Yuninda, 2019). Midtrans menyediakan *tools* terintegrasi kedalam sistem *e-commerce* sesuai kebutuhan pembayaran secara online melalui kartu kredit, kartu debit, dan pengiriman uang. Melalui *payment gateway* midtrans, pengguna dapat melakukan pembayaran, donasi, dan penarikan uang dengan mudah, cepat dan aman.

Berdasarkan uraian diatas, pemilik Chesspedia menginginkan pengembangan *website E-commerce* Chesspedia secara lengkap dengan proses pembayaran terintegrasi dengan *payment gateway* untuk memudahkan pelanggan serta admin dalam proses transaksi di Chesspedia. Pengembangan *E-Commerce* Chesspedia ini diolah menggunakan *framework* Laravel dan juga memanfaatkan layanan *Payment Gateway* API dari Midtrans.

Dengan adanya layanan *Payment Gateway Midtrans diharapkan dapat* memudahkan pembeli dan admin dalam melakukan transaksi di *website e-commerce* Chesspedia.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang sudah dijelaskan dapat dirumuskan permasalahan dalam penelitian ini adalah sebagai berikut :

1. Bagaimana analisis kebutuhan *e-commerce* Chesspedia dengan memanfaatkan layanan API *Payment Gateway* Midtrans?
2. Bagaimana rancangan dan implementasi *website e-commerce Chesspedia* ke dalam framework Laravel?
3. Bagaimana hasil pengujian fungsional dan non-fungsional dari pembuatan *website e-commerce* Chesspedia?

1.3 Tujuan

Dibawah ini merupakan tujuan penelitian :

1. Mendapatkan hasil analisis kebutuhan *website e-commerce* Chesspedia yang memanfaatkan layanan *API Payment Gateway Midtrans*.
2. Mendapatkan hasil perancangan dan Implementasi *website e-commerce* Chesspedia ke dalam framework Laravel dengan memanfaatkan *API Payment Gateway Midtrans*.
3. Mendapatkan hasil pengujian fungsional dan non-fungsional dari pembuatan *website e-commerce* Chesspedia.

1.4 Manfaat

Manfaat yang ingin didapat dari penelitian ini adalah sebagai berikut:

1. Meringankan beban admin Chesspedia pada saat verifikasi pembayaran serta pencatatan pesanan.
2. Mempermudah proses metode pembayaran baik bagi admin maupun pelanggan.

1.5 Batasan Masalah

Berdasarkan permasalahan diatas adapun batasan masalah, yaitu sebagai berikut :

1. Penelitian ini hanya sampai pengembangan sistem, tanpa melanjutkan pemeliharaan sistem.
2. Implementasi sistem ini ditujukan untuk mengelola pesanan yang masuk dan konfirmasi pembayaran yang masuk secara otomatis.
3. Penelitian ini menggunakan bahasa pemrograman PHP dan Java dengan *framework Laravel* dan *database* MySQL serta di hosting menggunakan layanan CPanel di JagoanHosting.
4. Cakupan penelitian ini adalah bagian manajemen Chesspedia .
5. Penelitian hanya menguji hingga fungsionalitas sistem yang berjalan tanpa melakukan pengujian terhadap *User Interface* dan *User Experience*

1.6 Sistematika Pembahasan

Sistematika pembahasan pada penelitian ini disusun dalam beberapa bab sebagai acuan, adapun rancangannya sebagai berikut :

BAB 1 PENDAHULUAN

Pembahasan pada bab ini berisi tentang latar belakang dari Chesspedia, diantaranya terdapat rumusan masalah, tujuan, manfaat, batasan masalah dan sistematika pembahasan atas penyusunan skripsi ini.

BAB 2 LANDASAN KEPUSTAKAAN

Pada bab ini berisi kajian pustaka dari penelitian-penelitian sebelumnya yang digunakan sebagai dasar penelitian, dan uraian tentang dasar teori untuk merancang *e-commerce* pada Chesspedia.

BAB 3 METODOLOGI

Bab ini berisi tentang metode penelitian yang digunakan untuk menyelesaikan penelitian pembangunan *website E-commerce* Chesspedia

BAB 4 ANALISIS KEBUTUHAN

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

Pembahasan bab ini menjabarkan berkaitan dengan penggambaran arsitektur, pemodelan bisnis, identifikasi aktor dan persyaratan fungsional sistem berdasarkan aktornya. Digunakan untuk melakukan analisis kebutuhan dari Chesspedia sebagai acuan dalam melakukan perancangan.

BAB 5 PERANCANGAN SISTEM

Pada bab ini berisi tentang perancangan berdasarkan hasil analisis yang telah dilakukan. Perancangan dilakukan untuk mendapatkan dokumen *sequence diagram*, PDM dan *class diagram*.

BAB 6 IMPLEMENTASI SISTEM

Bab ini membahas implementasi dari sistem yang dikembangkan. Implementasi berupa basis data , kode program dan antarmuka.

BAB 7 PENGUJIAN SISTEM

Pada bab ini membahas tentang pengujian yang dilakukan ketika menguji *website e-commerce* Chesspedia

BAB 8 PENUTUP

Bab ini membahas hasil dari penelitian yang telah dilakukan dengan menyajikan kesimpulan serta saran untuk penelitian yang akan datang.

BAB 2 LANDASAN KEPUSTAKAAN

Landasan kepustakaan berisi kajian pustaka , profil perusahaan serta landasan teori. Dalam kajian pustaka terdapat beberapa ringkasan tentang penelitian sebelumnya yang berhubungan dengan topik peneliti sebagai acuan penyelesaian permasalahan . Profil perusahaan menjelaskan tentang detail profil Chesspedia. Landasan teori berisi teori-teori yang berhubungan dengan topik peneliti.

2.1 Kajian Pustaka

Kajian pustaka yang menjadi landasan dan digunakan untuk penelitian ini berguna untuk mebahas tentang penelitian yang sudah ada pada waktu sebelumnya mengenai pembangunan aplikasi dengan menggunakan layanan *web service*.

Penelitian pertama adalah penelitian yang dilakukan oleh Tri May Mega Puspasari dan Dina Maulina pada tahun 2019 terkait pemanfaatan sistem *payment gateway* pada *website* travel Travnesia. Sistem ini dibangun dengan tujuan agar mempermudah pengguna Travnesia dalam melakukan proses verifikasi pembayaran. Dimana layanan pembayaran yang digunakan oleh Travnesia menggunakan Midtrans *payment gateway* yang membuat wisatawan dapat melakukan pembayaran online dalam pemesanan paket wisata. Midtrans juga memiliki integrasi yang mudah dan keamanan yang baik karena menggunakan sistem anti-fraud.

Penelitian kedua oleh Yuninda Eka Nisrina, Widhy Hayuhardhika dan Buce Trias Hanggara pada tahun 2019 terkait pemanfaatan sistem *payment gateway* yang mana melakukan pengembangan terhadap sistem yang ada di Wisata Kampung Sapi *Adventure* dengan menggunakan layanan *Payment Gateway*. Untuk proses jual beli tiket masuk tempat wisata tersebut saat ini masih menggunakan metode manual, yakni wisatawan mengirimkan foto bukti pembayaran transfer ke admin di sosial media Kampung Sapi *Adventure.* Hal ini mengakibatkan dapat terjadinya data tidak valid, dikarenakan proses verifikasi yang dilakukan masih manual. Untuk mengatasi hal ini, peneliti menggunakan layanan *Payment Gateway* Midtrans dengan konsep *Service Oriented* Architecture*.* Melalui aplikasi yang akan dibuat, para petugas akan dimudahkan dalam pendataan transaksi penjualan tiket.

Penelitian ketiga dilakukan oleh Erick Febriyanto, Untung Rahardja dan Nika Alnabawi pada tahun 2018 terkait pemanfaatan fitur pencatatan transaksi otomatis yang memanfaatkan layanan *Payment Gateway Midtrans* sebagai sarana penjualan hosting serta domain. Fitur pembayaran yang ada saat ini masih menggunakan uang tunai (konvensional) serta pembuatan laporan penjualan masih dilakukan secara manual (*Excel Spreadsheet)* sehingga hal ini mengakibatkan terbuangnya waktu bagi para pekerja iPanda. Selain itu sistem yang ada saat ini juga masih semi komputerisasi, sehingga cara ini sangat tidak efisien karena menyulitkan pembeli dalam melakukan pembayaran serta

pencatatan tidak sesuai. Untuk mengatasi hal tersebut, peneliti menggunakan layanan *Payment Gateway Midtrans* untuk menghandle proses pembayaran serta transaksi. Dengan begitu pembeli tidak akan kerepotan dalam meilih metode pembayaran yang mereka inginkan.

Penelitian keempat di lakukan oleh Habib Yafi Ardi, Bayu Primbadha dan Arief Andy Soebroto pada tahun 2019 terkait pemanfaatan fitur pada sistem *payment gateway* dengan merancang suatu marketplace *homestay* dan kost dengan fasilitas layanan *payment gateway*. Tujuan dari perangkat lunak ini adalah untuk menciptakan suatu pasar bagi para pemilik properti persewaan tempat tinggal dimana pemilik dan pelanggan penyewa tempat tinggal dapat dipertemukan. Homie memiliki 3 tipe pengguna yakni guest, owner, dan admin. Disini dilakukan juga integrasi dengan metode pembayaran secara dua arah yakni dari pembeli dan juga kepada owner. Pembeli dapat melakukan pembayaran melalui layanan *payment gateway midtrans* yang telah terintegrasi. Lalu para owner juga dapat mencairkan dananya di fitur yang ada di midtrans.

Penelitian kelima oleh Calvine Jardinia Tanuwidjaja dan Alexander Setiawan terkait pemanfaatan sistem *payment gateway* serta api raja ongkir yang mana pembuatan *website* Ecommerce ini dapat membantu pemilik toko untuk memasarkan produknya dengan jangka wilayah yang lebih luas. Pada penelitian ini peneliti mengusulkan sistem pembayaran melalui internet menggunakan *payment gateway* untuk menangani transaksi pembayaran kartu kredit antara pelanggan, dan pemilik toko. Untuk menguji dan mengevaluasi sistem pembayaran secara online, penulis membangun aplikasi berbasis *website e-commerce* NeoGenesis. Layanan yang disediakan pada aplikasi NeoGenesis yakni pembelian dan penjualan barang elektronik oleh toko aksesoris Komputer di Surabaya. Aplikasi NeoGenesis memanfaatkan sistem pembayaran yang diusulkan yaitu dengan *payment gateway* untuk menangani pembayaran secara transfer antara pelanggan dan pemilik toko. Peneliti menyatakan dari evaluasi yang telah dilakukan menunjukkan bahwa sistem pembayaran gateway dapat digunakan dengan mudah, aman, dan hemat biaya.

2.2 Chesspedia

Chesspedia merupakan toko online yang menjual peralatan set catur lengkap mulai dari tas catur, bidak catur hingga jam catur yang terletak di Jl.Aris Munandar no.99 Kota Malang, Jawa Timur. Toko online yang didirikan sejak tahun 2018 ini pada awalnya hanya melayani pembelian via *whatsapp* saja, namun seiring menjamurnya *e-commerce* pada saat ini, chesspedia juga membuka toko online melalui berbagai marketplace yang ada, seperti Tokopedia, Lazada dan shopee maupun sosial media lainnya yakni Instagram.

2.3 E-Commerce

Menurut Kotler & Armstrong (2018) *E-commerce* adalah saluran online yang dapat dijangkau seseorang melalui computer, yang digunakan oleh pebisnis dalam melakukan aktifitas bisnisnya dan digunakan konsumen untuk

mendapatkan informasi dengan menggunakan bantuan computer yang dalam prosesnya diawali dengan memberi jasa informasi pada konsumen dalam penentuan pilihan. Menurut Wong (2018) *E-Commerce* adalah proses jual beli dan memasarkan barang serta jasa melalui system elektronik, seperti radio, televisi dan jaringan computer atau internet

Selain itu, *e-commerce* juga memiliki model bisnis. Model bisnis ini memiliki tujuan untuk memaksimalkan potensi keuntungan dari target pasar yang diinginkan (Turban et al., 2015). Dalam penerapannya sendiri, pelaku bisnis bisa menggunakan beberapa model bisnis sekaligus jika dirasa perlu untuk demi meningkatkan keuntungan. Berikut beberapa model bisnis yang umum digunakan oleh *e-commerce:*

1. ***Online direct marketing*.**
   Model bisnis dimana barang atau jasa dijual secara langsung oleh pembuatnya tanpa perantara, sehingga proses distribusi menjadi lebih efisien terutama untuk produk yang sifatnya digital.
2. ***Electronic tendering systems*.**
   Model bisnis dimana organisasi atau perusahaan melakukan pembelian secara besar - besaran melalui lelang. Proses lelang ini dapat dilakukan secara daring untuk menghemat waktu dan biaya. Biasanya, model bisnis ini digunakan oleh perusahaan besar atau instansi pemerintah.
3. ***Electronic marketplaces and exchanges*.**
   Model bisnis dimana sebuah *marketplace* hanya menangani proses transaksi dari produk atau jasa yang sepenuhnya disediakan oleh pihak ketiga.
4. ***Viral marketing*.**
   Model bisnis yang populer digunakan dalam sosial media. Pada dasarnya, *viral marketing* ini adalah promosi dari mulut ke mulut menggunakan internet untuk mempopulerkan suatu produk atau jasa.

2.4 Rekayasa Perangkat Lunak

Ilmu yang menjelaskan seluruh aspek serta proses dalam pengembangan perangkat lunak, mulai dari pengembangan hingga produksi perangkat lunak disebut sebagai rekayasa perangkat lunak. Tujuan proses ini yaitu menghasilkan aplikasi dengan cara yang efektif dan dapat memenuhi semua aspek mulai dari waktu, biaya serta tenaga. Dalam proses rekayasa perangkat lunak terdapat aspek yang mendasar seperti analisis dan spesifikasi kebutuhan, pengembangan perangkat lunak meliputi perancangan dan implementasi, proses validasi yaitu pengujian, serta pemeliharaaan (Sommerville, 2011).

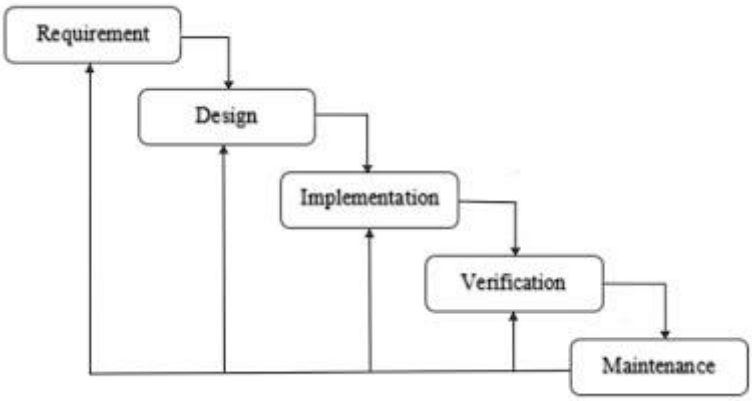
2.5 Software Development Life Cycle

SDLC mempunyai banyak sekali model yang dapat dilakukan oleh pengembang dan dapat disesuaikan dengan keadaan proyek yang akan dikerjakan. Model yang ada mempunyai keunikan dan ciri khas masing-masing dan digunakan dengan cara

menyesuaikan dengan kebutuhan agar dapat meningkatkan peluang keberhasilan pada proses pengembangan suatu aplikasi (Pressman, 2012).

2.5.1 Waterfall Model

Waterfall model merupakan salah satu model yang biasa disebut juga *classic life cycle*. Pendekatan dengan metode *waterfall* mempunyai suatu ciri khas yaitu hal yang sistematis, berurutan dalam pengembangan suatu perangkat lunak. Tahapan dimulai dari analisis kebutuhan, perancangan, mengimplementasikan perancangan menjadi baris kode yang dapat di proses oleh komputer, pengujian, hingga pendistribusian kepada pengguna. Fase-fase tersebut di representasikan pada Gambar 2.1 yang merupakan ilustrasi dari siklus *waterfall* (Pressman, 2012).



Gambar 2.1 Diagram *Waterfall Model*

Sumber: Pressman (2012)

Dalam pengembangannya metode *waterfall*memiliki beberapa tahapan yang berurut yaitu: *requirement* (analisis kebutuhan), *system design* (desain sistem), *Coding* (pengkodean) & *Testing* (pengujian), Penerapan Program, pemeliharaan. Tahapan tahapan dari metode *waterfall* adalah sebagai berikut :

1. *Requirement Analisis*
   Tahap pengembang sistem diperlukan dengan tujuan untuk memahami perangkat lunak yang diharapkan oleh pengguna dan batasan perangkat lunak tersebut. Informasi ini biasanya dapat diper*oleh* melalui wawancara, diskusi atau survei langsung. Informasi dianalisis untuk mendapatkan data yang dibutuhkan oleh pengembang.
2. *System Design*
   Spesifikasi kebutuhan dari tahap sebelumnya akan dipelajari dalam fase ini serta desain sistem disiapkan. Desain Sistem membantu dalam menentukan perangkat keras*(hardware)* dan sistem persyaratan dan juga membantu dalam mendefinisikan arsitektur sistem menyeluruh.
3. *Implementation*
   Pada tahap ini, sistem pertama kali dikembangkan di program kecil yang disebut *unit.* Setiap *unit* dikembangkan dan diuji untuk fungsionalitas yang disebut sebagai *unit testing*.

4. *Integration & Testing*
   Seluruh *unit* yang dikembangkan dalam tahap implementasi diintegrasikan ke dalam sistem setelah pengujian yang dilakukan masing-masing *unit*. Setelah integrasi, sistem diuji untuk mengecek setiap kegagalan maupun kesalahan.
5. *Operation & Maintenance*
   Tahap akhir model *waterfall.* Perangkat lunak yang sudah jadi, dijalankan serta dilakukan pemeliharaan. Pemeliharaan termasuk dalam memperbaiki kesalahan yang tidak ditemukan pada langkah sebelumnya. Perbaikan implementasi *unit* sistem dan peningkatan jasa sistem sebagai kebutuhan baru.

Pada tulisan dari Powel (2016) memberikan penjelasan mengenai kelebihan dan kekurangan dari *waterfall model*. Berikut ini adalah kelebihannya :

1. *Adapts to Shifting Teams*
   Metode *Waterfall* mempunyai ruang lingkup yang sudah jelas, rinci, serta mempunyai struktur yang kuat karena pada setiap tahap dari awal perencanaan sudah mempunyai dokumentasi. Hal ini cocok pada suatu tim yang besar melihat anggota yang mungkin datang dan pergi sepanjang pengerjaan proyek, dikarenakan dokumentasi yang jelas dan beban pekerjaan lebih sedikit pada anggota tim atau individu.
2. *Structured Organization*
   Metode ini memaksa untuk adanya struktur organisasi pada proyek yang jelas dan mengharuskan disiplin dalam desain dan strukturnya. Dengan metode yang disiplin ini prosedur akan menjadi terperinci untuk mengelola setiap aspek pada proyek dari desain, pengembangan hingga pengujian.
3. *Allows for Early Design Changes*
   Perubahan hanya akan terjadi pada awal pengumpulan kebutuhan antara *client* dan *development* dan hal ini membuat tidak akan terjadi perubahan lagi saat pengembangan atau pengkodean telah dimulai.

Selain beberapa kelebihan yang ada *Waterfall model* juga memiliki kekurangan yang ada antara lain:

1. *Non-adaptive Design Constraints*
   Metode ini tidak bisa beradaptasi jika diketahui ada kesalahan mendasar pada sebuah sistem. Apabila jika ada kesalahan mendasar pada sebuah sistem tidak hanya membutuhkan lompatan untuk kembali ke tahap sebelumnya namun bisa membuat pengulangan dari tahap paling awal. Sehingga perancangan yang baik sejak awal sangat diperlukan agar tidak terjadi hal-hal seperti ini.
2. *Ignores Mid-Process User/Client Feedback*
   Karena proses yang sangat ketat pada setiap tahapan, maka akan menjadi masalah jika tiba-tiba ada *feedback* dari *client* yang terlambat masuk pada siklus pengembangan, karena pada siklus ini semua berjalan terus turun dan tidak bisa kembali.
3. *Delayed Testing Period*
   Berbeda dengan model SDLC yang lebih modern yang berusaha melakukan pengujian pada setiap tahapan, metode ini hanya dapat melakukan pengujian pada tahap akhir.

2.5.1.1 Unified Modeling Language

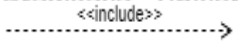
Unified Modeling Language (UML) merupakan metode pemodelan berupa visual yang dapat digunakan untuk memvisualisasikan atau merepresentasikan, sebuah artefak dalam suatu perangkat lunak. UML memberikan gambaran mengenai bagaimana sebuah sistem harus dibangun serta digunakan untuk pemahaman, desain, mencari, mengatur, mempertahankan serta mengkontrol infromasi dari perangkat lunak tersebut dengan pengembangan perangkat lunak berbasis *Object Oriented* (Rumbaugh,dkk, 2004).

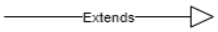
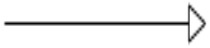
2.5.1.2 Use Case Diagram

Menurut (Whitten and Bentley, 2007), *Use case* adalah diagram yang menunjukkan interaksi antar sistem maupun pengguna dengan sistem. Sehingga dapat diartikan berupa bagaimana sistem akan bekerja dan apa saja yang diharpkan oleh pengguna ketika menggunakan atau berinteraksi dengan sistem.

Use case dapat disimbolkan melalui elips yang berisi nama dari *activity* yang dilakukan sistem terhadap aktor-aktor yang bersangkutan dan hubungan antar elips mewakili satu tujuan dari sistem serta menjelaskan urutan langkah yang diambil untuk mencapai suatu tujuan. Simbol dan juga deskripsi yang digunakan pada *use case diagram* dapat dilihat pada Tabel 2.1.

Tabel 2.1 *Use Case Diagram*

Simbol	Deskripsi
Use case	*Use case* merupakan fungsionalitas yang akan disediakan sistem sebagai aksi yang saling bertukar pesan antar aksi atau aktor, biasanya dinyatakan dengan kata kerja.
Aktor	Aktor dapat berupa orang, proses atau sistem lain yang berinteraksi dengan sistem yang akan dibuat diluar sistem dari sistem itu sendiri, aktor belum tentu merupakan orang, biasanya dinyatakan dengan kata benda.
Asosiasi/*Assosiation*	Asosiasi merupakan komunikasi antara aktor dengan *use case* yang berpartisipasi.
 *Include*	*Include* menggambarkan suatu *use case* yang termasuk dalam *use case* lainnya. *Include* bertujuan untuk mengurangi redundansi antara dua *use case* atau lebih.

 Ekstensi/*Extend*	Extensi merupakan relasi tambahan *use case* kesebuah *use case* yang dapat berdiri sendiri, mirip dengan prinsip *inheritance* pada pemrograman beriorientasi objek. Biasanya namaya *use case* tambahan ini memiliki nama depan yang sama dengan *use case* yang ditambahkan.
 Generalisasi/ *Generalization*	Generalisasi merupakan hubungan generalisasi dan spesialisasi antara 2 buah *use case* dimana salah satu fungsi nya lebih umum dari yang lainnya.

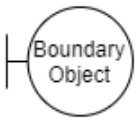
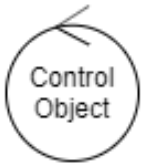
Sumber: Rosa, A.S and Shalahuddin,(2014)

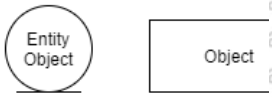
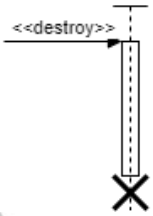
2.5.1.3 Sequence Diagram

Menurut (Whitten and Bentley, 2007), berupa *diagram* yang digunakan untuk mendeskripsikan interaksi antara sistem dengan pengguna yang di definisikan dari *use case scenario* yang telah dibuat.

Sequence diagram menggambarkan perilaku objek *use case* dengan mendeskripsikan waktu hidup objek dan *massage* yang dikirim dan diterima oleh objek. Untuk membuat *sequence diagram* dibutuhkan adanya melihat skenario yang ada pada *use case* (Rosa, A.S and Shalahuddin, 2014). Simbol dan juga deskripsi yang digunakan pada dapat dilihat pada Tabel 2.3.

Tabel 2.2 *Sequence Diagram*

Simbol	Deskripsi
Aktor	Aktor dapat berupa orang, proses atau sistem lain yang berinteraksi dengan sistem yang akan dibuat diluar sistem dari sistem itu sendiri, aktor belum tentu merupakan orang, biasanya dinyatakan dengan kata benda diawal frase nama aktor.
	*Boundary* digunakan untuk menghubungkan *user* dengan sistem.
	*Control* digunakan untuk mengontrol aktivitas yang dilakukan oleh kegiatan.

	Entity atau entitas yang mempunyai atribut yang datanya bisa direkam.
Lifeline / Garis Hdup	*Lifeline* merupakan garis vertikal yang memanjang kebawah yang mengindikasikan urutan kehidupan
Pesan tipe *call*	Tipe *call* menyatakan suatu objek memanggil operassi yang ada pada objek lain atau dirinya sendiri.
Pesan tipe *create*	Tipe *create* menyatakan suatu objek yang mengarah objek lain yang dibuat.
Pesan tipe *send*	Tipe *send* menyatakan suatu objek mengirimkan atau memasukkan data/informasi ke objek lainnya.
Pesan tipe *return*	Tipe *return* menyatakan objek yang telah menjalankan operasi dan menghasilkan suatu kembalian ke objek tertentu.
Pesan tipe *destroy* 	Tipe *destroy* menyatakan suatu objek mengakhiri objek lain. Jika ada *create* maka ada *destroy*.

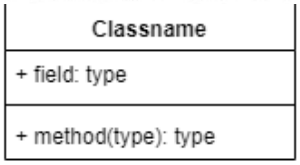
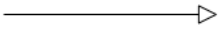
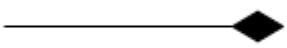
Sumber : Rosa, A.S and Shalahuddin (2014) & (Whitten and Bentley (2007)

2.5.1.4 Class Diagram

Class Diagram merupakan statik objek terstruktur dari sistem yang memperlihatkan kelas objek pada sistem yang saling berhubungan antara objek kelas (Whitten and Bentley, 2007).

Pada Tabel 2.4 menjelaskan simbol dan deskripsi yang digunakan pada *class diagram* dapat dilihat. *Class diagram* menggambarkan struktur sistem dari pendefinisian beberapa kelas yang akan dibuat untuk pengembangan suatu sistem. Kelas yang dibangun tersebut memiliki atribut dan metode atau operasi (Rosa, A.S and Shalahuddin, 2014).

Tabel 2.3 *Class Diagram*

Simbol	Deskripsi
Class/Kelas 	Kelas pada struktur sistem.
Interface/Antarmuka	Interface yang memiliki konsep sama pada pemrograman berorientasi objek
Association/Asosasiasi	Asosiasi merupakan relasi antar kelas dengan makna umum yang biasanya disertai *multiplicity*.
Directed Association/ Asosiasi berarah	Merupakan relasi antar kelas dengan makna kelas yang digunakan oleh kelas lain, biasanya disertai *multiplicity*.
Generalization/Generalisasi 	Merupakan relasi antar kelas dengan makna umum generalisai dan spesialisasi.
Dependency/ Kebergantungan	Merupakan relasi antar kelas dengan makna kebergantungan antar kelas
Aggregation/Agregasi	Merupakan relasi yang menunjukkan satu objek dari kelas terdiri dari objek kelas lain.
Compocition/komposisi 	Merupakan relasi yang menunjukan hubungan satu kelas bertanggung jawab dengan bagian dalam kelas lainnya. Jika satu kelas rusak, maka kelas lain juga rusak.

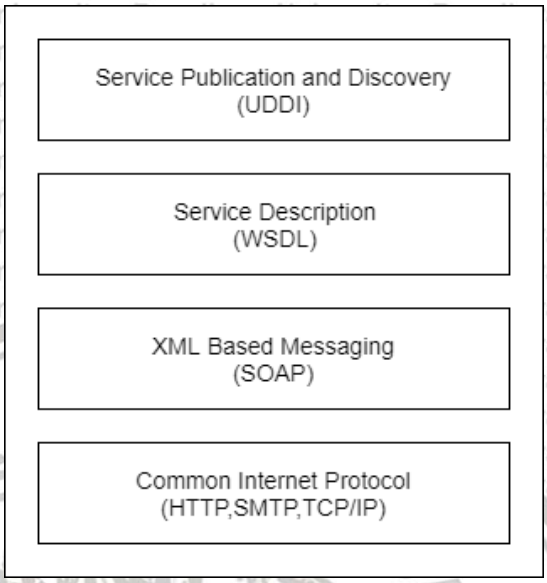
Sumber : Rosa, A.S and Shalahuddin (2014) & (Whitten and Bentley (2007)

2.5.2 Teknologi Pengembangan Sistem

2.5.2.1 Web service

Web service sebenarnya merupakan kumpulan dari fungsi dan method yang terdapat pada sebuah server yang dapat dipanggil oleh klien dari jarak jauh, kemudian untuk memanggil method-method tersebut kita bebas menggunakan aplikasi yang akan dibuat dengan bahasa pemrograman apa saja yang dijalankan pada *platform* apa saja (Marthasari, 2010).

Adanya teknologi *web service* dapat menjembatani perbedaan-perbedaan teknologi dari masing-masing sumber. Dapat ditarik kesimpulan bahwa *web service* merupakan kumpulan layanan yang disediakan melalui jaringan berbasis *web* dengan standar yang telah ditetapkan mampu menunjang interoperabilitas, dan dapat berjalan diberbagai *platform* dan *framework*.



Gambar 2.2 *Use Case Relationship Web Service*

(Deviana, 2011)

Gambar 2.2 merupakan blok bangunan *web service* yang mana menyediakan fasilitas komunikasi jarak jauh antara dua aplikasi yang merupakan layer arsitektur *web service*.

a. Layer 1 : protokol internet standar yang digunakan sebagai sarana transportasi adalah HTTP dan TCP/IP.
b. Layer 2 : *Simple Object Access Protocol* (SOAP) berbasiskan XML dan digunakan untuk pertukaran informasi antar sekelompok layanan.
c. Layer 3 : *Web service Definition Language* (WSDL) digunakan untuk mendiskripsikan attribute layanan.
d. Layer 4 : *Universal Description Discovery and Integration*, yang mana merupakan direktori pusat untuk deskripsi layanan.

Marthasari dkk (2010) dalam penelitiannya menjelaskan komponen *web service* yaitu:

a. *Extensible Markup Language* (XML)

XML adalah dasar yang penting atas terbentuknya *Web service*. *Web service* dapat berkomunikasi dengan aplikasi-aplikasi yang memanggilnya dengan menggunakan XML , karena XML berbentuk teks sehingga mudah untuk ditransportasikan menggunakan protokol HTTP.

b. *Simple Object Access Protocol* (SOAP)

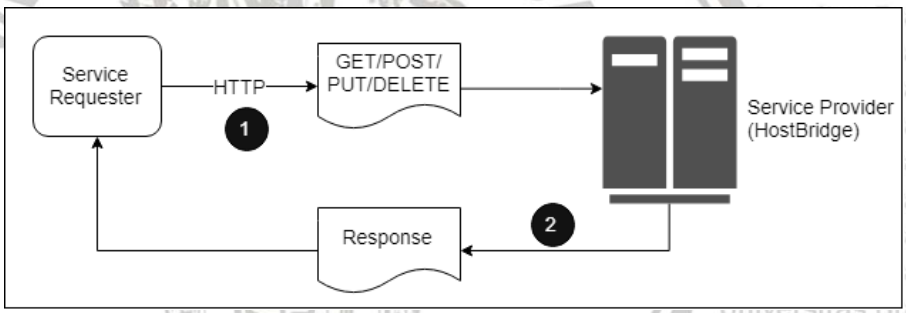
SOAP (Simple Object Access Protocol) adalah format standard dokumen berbentuk XML yang digunakan untuk melakukan proses request dan responses antara *web service* dengan aplikasi yang memanggilnya. Dokumen SOAP digunakan untuk melakukan request disebut dengan SOAP request sedangkan dokumen SOAP yang diperoleh dari *Web service* disebut dengan SOAP responses.

c. *Web service Definition Language* (WSDL)

WSDL (*Web service* Description Language) merupakan sebuah dokumen dalam format XML, isinya menjelaskan informasi detail sebuah *Web service*. Di dalam WSDL dijelaskan method-method apa saja yang tersedia dalam *Web service*, parameter apa saja yang diperlukan untuk memanggil sebuah method, dan apa hasil atau tipe data yang dikembalikan oleh method yang dipanggil tersebut.

2.5.2.2 Teknologi Representational State Transfer (REST)

REST merupakan singkatan dari *Representative State Transfer*. Pertama kali dikemukakan dalam disertasi seorang program doktor bernama Roy Thomas Fielding pada tahun 2000 (Rozali, 2011). REST adalah sebuah metode dalam menyampaikan *resource* melalui media *web*.



Gambar 2.3 Model dasar RESTful *Web service*

(HostBridge,2009)

Gambar 2.3 merupakan model dasar dari RESTful *Web service*. Berikut merupakan langkah-langkah yang dilakukan dalam model dasar RESTful *Web service* (HostBridge, 2009):

1. *Query Request Provider* melalui HTTP dengan menggunakan URI (*Uniform Resource Identifier*). *Request* menggunakan methods (metode) HTTP untuk menentukan apakah *request* tersebut dimaksudkan untuk *Create* (menciptakan), *Read* (membaca), *Update* (memperbarui), atau *Delete* (menghapus) data.
2. *HostBridge* mengembalikan sebuah dokumen dalam bentuk XML untuk *Requester* (pemohon) dengan CICS data *enclosed*. Diagram di atas menunjukkan dua *web service* : sebuah *Provider* (*HostBridge*) yang menyediakan layanan *web service* dan sebuah Request yang digunakan pada *web service*. Dalam model RESTful layanan dideskripsikan secara sendiri (*self-*

describing), sehingga tidak diperlukan WSDL dan Server UDDI yang bertindak sebagai Broker (HostBridge, 2009).

Pada development *web* modern aksi CRUD tersebut dapat dimappingkan dengan HTTP method sebagai berikut : *CREATE* sebagai *POST*, *RETREIVE* sebagai *GET*, *UPDATE* sebagai *PUT*, dan *DELETE* sebagai *DELETE*. Tabel 2.5 merupakan penggunaan methods(metode) HTTP dalam REST *Web service*.

Tabel 2.4 Metode HTTP dan Penggunaannya dalam REST

Metode	Keterangan
GET	Mendapatkan (*read*) sebuah sumber daya (*resource*) yang diidentifikasi dengan URI (*Uniform Resource Identifier*)
POST	Mengirimkan sumber daya (*resource*) ke server. Digunakan untuk membuat (*create*) sumber daya baru.
PUT	Mengirimkan sumber daya (*resource*) ke server. Digunakan untuk memasukkan (*insert*) atau memperbarui (*update*) sumber daya yang tersimpan.
DELETE	Menghapus (*delete*) sumber daya (*resource*) yang diidentifikasi dengan URI.
HEAD	Mendapatkan metadata (*response header*) dari sumber daya(*resource*) yang diidentifikasi dengan URI.

Sumber : (Riyadi, 2013).

2.5.2.3 JavaScript Object Notation (JSON)

Javascript Object Notation (JSON) merupakan sebuah *general-purpose* data *encoding* format yang populer. Implementasi JSON banyak digunakan pada *database* dan *web service*. Struktur dokumen pada JSON secara opsional dibatasi berdasarkan dengan skema yang terdiri atas dua hal yakni map (pemetaan struktur nilai berdasarkan klasifikasi jenisnya) dan list (pengelompokan nilai berdasarkan klasifikasijenisnya) (Kleppmann dan Alastair, 2017).

Sebuah objek JSON merupakan *unordered set* dari name atau value *pairs*. Sebuah objek dimulai dengan tanda "{" (*left brace*) dan diakhiri dengan tanda "}"(*right brace*). Setiap name diikuti oleh tanda :"(*colon*) dan name atau value pairs dipisahkan oleh tanda "," (koma). Gambar 2.4 merupakan contoh struktur data dengan format JSON.

```
{
  "first_name" : "Sammy",
  "last_name" : "Shark",
  "location" : "Ocean",
  "online" : true,
  "followers" : 987
}
```

Gambar 2.4 Contoh Struktur Data JSON

Kelebihan menggunakan JSON menurut nurseitov dkk (2009) adalah kecepatan proses jika dibandingkan dengan XML yang merupakan format data dalam *web service* yang telah ada selama ini. Kecepatan ini disebabkan karena untuk memparsing sebuah XML *file* maka dibutuhkan sebuah *external* Sementara JSON sudah dapat diproses langsung oleh javascript tanpa memerlukan lainnya. Format data JSON dapat terbaca dengan mudah oleh manusia dan dapat diparsing dengan cepat oleh komputer.

2.5.2.4 PHP

PHP atau Hypertext Preprocessor merupakan salah satud bahasa pemrograman. Berbeda dengan javascript yang bekerja di dalam tiap browser, PHP memerlukan suatu program buat dapat menerjemahkan ataupun membaca kode tersebut. PHP memiliki seluruh fitur yang biasa terdapat dalam bermacam berbagai bahasa pemrograman sehingga ini dapat menjadikan PHP membuat suatu aplikasi yang berdiri sendiri. Dengan memakai PHP kita bisa membangun sesuatu *website* dengan konten yang dinamis. Semacam contoh seseorang pengguna tiba pada suatu *website*, dengan menekan tombol tertentu hingga melaksanakan suatu proses yang diberikan dari klien serta dipulangkan oleh server dengan hasil tertentu, proses tersebut ditangani oleh PHP yang diproses di dalam server sehingga PHP juga bisa dituturkan sebagai server- side programming languange(Wellens, 2015).

2.5.2.5 Laravel

Laravel merupakan framework PHP yang menekankan pada kesederhanaan dan fleksibilitas pada designnya. Sama seperti framework PHP lainnya seperti codeigniter, laravel dibangun dengan basis MVC (Model-*View*-Controller). Laravel adalah suatu pola bentuk pemrograman yang memiliki tujuan untuk dapat memisahkan antara logika bisnis, logika data serta logika untuk tampilan. Pemisahan ini dilakukan untuk kemudahan dalam mengatur suatu arsitektur terhadap sebuah aplikasi (Luthfi, 2017).

2.5.2.6 Payment Gateway Midtrans

Merujuk pada Peraturan Bank Indonesia Nomor 18/40/PBI/2016 tentang Penyelenggaraan Pemrosesan Transaksi Pembayaran, *Payment gateway* marupakan sebuah layanan elektronik yang digunakan dalam melakukan proses transaksi pembayaran dengan menggunakan alat pembayaran dengan media kartu, uang elektronik, dan/atau Proprietary Channel. *Payment gateway* merupakan sistem yang membantu efisiensi proses pembayaran pembeli kepada penjual. Sistem pembayaran seperti ini sering dipakai dalam *e-commerce* atau toko online yang terpercaya. Untuk melakukan aktivasi akun agar dapat menggunakan layanan dari *payment gateway* akan membutuhkan syarat-syarat yang itu tidak dapat dipenuhi oleh sembarang toko online.

Maka, toko online yang sudah menggunakan sistem *payment gateway* dapat dipastikan merupakan toko online yang terpercaya dan konsumen tidak perlu khawatir untuk bertransaksi dengan toko online tersebut (Cermati, 2017). Alur kerja dari *payment gateway* sebagai berikut, sesudah pembeli memutuskan untuk memilih produk maupun jasa yang terdapat pada *e-commerce* dan telah melewati proses pembayaran, selanjutnya informasi mengenai detail transaksi akan diteruskan ke penyedia sistem payment geteway. Kemudian penyedia sistem *payment gateway* akan meneruskan detail informasi tranaksi ke prosesor pembayaran dari bank penjual. Lalu prosesor pembayaran akan meneruskan detail informasi transaksi ke asosiasi untuk mendapatkan informasi kartu yang digunakan adalah Mastercard atau Visa. Kemudian bank terkait menerima sebuah request, kemudian bank akan mengirim response ke prosesor dengan sebuah kode yang berbeda dari response sebelumnya, response ini yang akan menunjukkan apakah transasksi berhasil atau gagal. Kemudian, prosesor pembayaran mengirim sebuah pesan kepada penyedia payment gateway. Pesan tersebut yang selanjutnya diteruskan ke web penjual serta pemilik kartu pembayaran.

Midtrans merupakan salah satu platform untuk melakukan payment gateway. Midtrans merupakan sistem pembayaran yang dapat memfasilitasi penjual dan pembeli untuk melakukan transaksi ((Midtrans (PT Midtrans), 2019). Midtrans menyediakan tools terintegrasi ke *e-commerce* sesuai kebutuhan pembayaran secara online dengan kartu debit, kartu kredit, bahkan penarikan uang, dan pengiriman uang. Dengan midtrans dapat melakukan pembayaran belanja online, donasi, produk berlangganan, dan penarikan uang dengan mudah, cepat dan aman.

2.5.2.7 Raja Ongkir

RajaOngkir merupakan sebuah situs dan web service (API) yang menyediakan informasi ongkos kirim dari berbagai kurir di Indonesia seperti POS Indonesia, JNE, TIKI, PCP, ESL, dan RPX. Secara umum, RajaOngkir ditujukan kepada pengguna yang ingin mengetahui dan membandingkan ongkos kirim dari berbagai kurir dan secara khusus bagi pemilik toko online, maupun bagi orang yang sering berbelanja online. Keunikan dari sistem RajaOngkir adalah data yang terpadu. pengguna cukup melakukan request dengan parameter nama kota asal, kota tujuan, jenis kurir, dan berat, sistem RajaOngkir otomatis melakukan pengecekan ke semua kurir yang didukung. RajaOngkir menyediakan API ongkos kirim yang bisa dimanfaatkan oleh para pengembang untuk mengembangkan aplikasi untuk berbagai platform, seperti Android, BlackBerry, iOS, Desktop, dan lain-lain.

2.6 Pengujian Perangkat Lunak

Pengujian perangkat lunak adalah suatu proses dalam rekayasa perangkat lunak yang berisi kegiatan yang telah disusun dan dilakukan dengan membuat penempatan kasus uji dan metode pengujian yang sudah disusun sebelumnya. Pengujian bertujuan untuk memastikan bahwa kode-kode dari sumber terkecil

telah diimplementasikan dengan benar serta tes validasi untuk melakukan validasi fungsi yang ada pada sistem telah memenuhi kebutuhan yang telah didapatkan sebelumnya (Pressman, 2009).

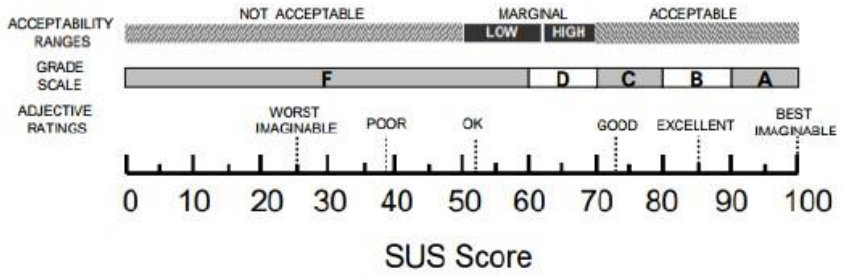
2.6.1 Blackbox Testing

Suatu perangkat lunak mempunyai banyak kebutuhan yang harus dipenuhi, untuk menguji kebutuhan apakah sudah dipenuhi sepenuhnya oleh sistem dilakukan pengujian *black-box testing*. Pengujian ini berguna untuk melakukan validasi apakah sistem sudah memenuhi kebutuhan fungsional atau tidak. Pengujian ini dilakukan dengan cara memberikan kondisi yang diberikan pada perangkat lunak untuk menguji semua kebutuhan fungsional pada sistem (Pressman, 2009).

2.6.2 Usability Testing

Usability testing pada penelitian ini dilakukan untuk mengetahui seberapa efektif dan efisien sistem ketika digunakan. *Usability testing* adalah metode dimana pengguna dari suatu perangkat lunak diminta untuk melakukan tugas-tugas tertentu dalam upaya untuk mengukur kemudahan penggunaan perangkat lunak, waktu, dan persepsi pengguna tentang pengalaman dalam menggunakan perangkat lunak. *Usability* sendiri memiliki 5 komponen kualitas yaitu efisiensi, efektifitas, mudah dipelajari, mudah diingat dan memberi kepuasan kepada pengguna. (Nielsen, 2020). Usability testing dapat dilakukan secara resmi ataupun secara informal seperti hanya dengan menggunakan kertas *mockup* aplikasi atau situs *web* (Niranjaanamurthy, 2014).

Kinerja Usability diukur dengan menggunakan pengujian *post Study* dengan metode *Sistem Usability Scale* (Brooke, 1996). SUS terdiri dari 10 pertanyaan dengan menggunakan skala *likert* 1-5 dimana 1 bernilai Sangat tidak setuju, 2 tidak setuju, 3 netral, 4 setuju dan 5 sangat setuju. Pertanyaan nomor (1,3,6,7,10) merupakan pertanyaan bernada positif. Sedangkan pertanyaan nomor (2,4,5,8,9) merupakan pertanyaan bernada negatif. Setiap pertanyaan diberi bobot minimal 0 dan maximal 4. Penilaian dilakukan dengan mengurangi 1 nilai respon pengguna untuk pertanyaan bernada positif. Dan 5 dikurangi nilai respon pengguna untuk pertanyaan bernada negatif. Setelah itu nilai dari pertanyaan 1-10 dijumlah dan dikali dengan 2.5. skor akhir SUS akan berada pada rentang nilai 0-100. Sesuai dengan gambar 2.5. dapat diketahui berapa tingkat *usability* dan *acceptable* desain sistem aplikasi yang dikembangkan. Penilaian berdasarkan 3 kategori yaitu *not acceptable* dengan rentang skor SUS 0-50,9; *marginal* dengan rentang poin 51-70,9; dan *acceptable* dengan rentang poin 71-100. Untuk Adjective Rating di bagi menjadi 6 kriteria yakni *worst imaginable* dengan rentang nilai 0-37.9, *poor* dengan rentang nilai 38-51.9, *ok* dengan rentang nilai 52-71.9, *good* dengan rentang nilai 72-84.9, *excellent* dengan rentang nilai 85-91.9, dan *best imaginable* dengan rentang nilai diatas 92.



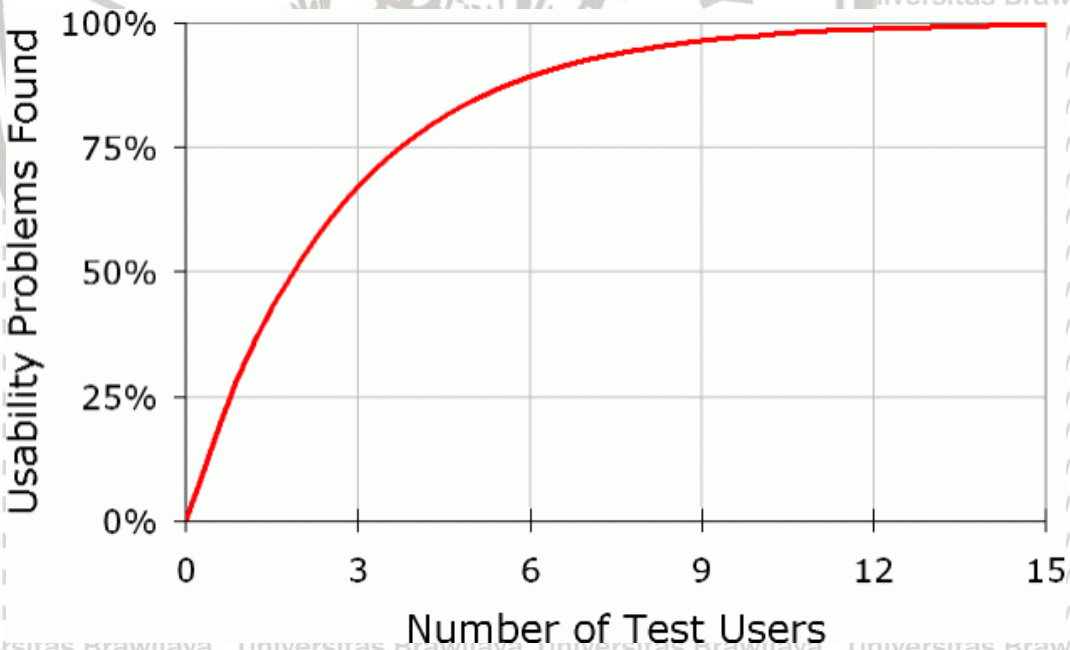
Gambar 2.5 *Sistem Usability Scale*

Sumber : (John Brooke, 1996)

Pada penelitian yang dilakukan oleh Tom Landauer dan Jakob Nielsen menyatakan bahwa hasil dari *usability testing* yang ditemukan dalam usability dengan nilai n responden ialah :

$$N\,(1-(1-L)^{n})$$

di mana N adalah jumlah total masalah kegunaan dalam desain dan L adalah proporsi masalah usability yang ditemukan saat menguji satu responden. Nilai khas L adalah 31% yang memberikan hasil sebagai berikut.



Gambar 2.6 Kurva Banyak Responden Terhadap Hasil Uji *Usability Testing*

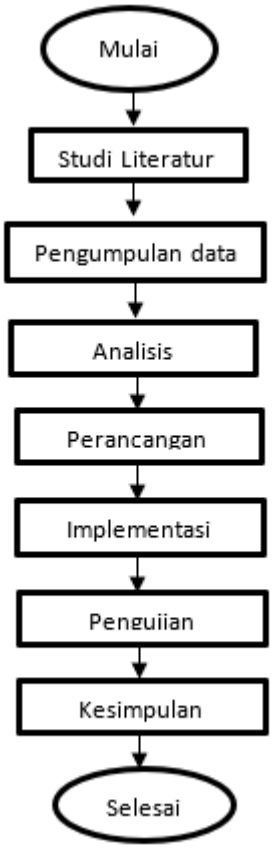
Dari kurva diatas dapat dilihat bahwa titik kurva optimal terdapat pada jumlah responden 5 orang yang mana memberikan hasil yang sangat signitifkan dibandingkan dengan jumlah 6 atau lebih responden.

2.6.3 Compatibility Testing

Compatibility Testing, atau pengujian kompatibilitas adalah suatu metode pengujian perangkat lunak yang digunakan untuk memastikan bahwa sebuah perangkat lunak dapat berjalan dengan baik pada berbagai macam perangkat. Untuk pengujian pada *website*, maka pengujian kompatibilitas ini adalah untuk memeriksa kompatibiltas *website* dengan berbagai macam browser, apakah *website* yang dibangun dapat berjalan dengan lancar atau tidak. Untuk melakukan uji kompatibilitas, dapat dilakukan dengan memeriksa secara manual menggunakan banyak jenis browser sekaligus. Namun, cara ini tidak efektif. Untuk itu, diperlukan sebuah *tools* yang mampu melakukan pengujian sekaligus di banyak browser secara otomatis. Salah satu contoh dari *tools* ini adalah *Sortsite*. *Sortsite* adalah salah satu *tools* yang dikembangkan oleh *PowerMapper*, sebuah perusahaan asal Edinburgh, UK. *Tools* ini digunakan untuk menganalisa seluruh halaman yang terdapat pada sebuah *website*. Hasil pengujian kompatibilitas menggunakan *tools* ini, yaitu berupa daftar *browser* dengan keterangan mengenai lancar tidaknya suatu *website* jika dijalankan pada daftar *browser* tersebut.

BAB 3 METODOLOGI

Dalam bab ini akan menjelaskan mengenai metodologi yang digunakan dalam penelitian ini meliputi identifikasi masalah, analisis kebutuhan, perancangan sistem, dan pengambilan kesimpulan. Diagram alir penelitian dapat dilihat pada Gambar 3.1.



Gambar 3.1 Diagram Alir Metode Penelitian

3.1 Studi Literatur

Proses untuk melakukan tinjauan teori atau studi literatur digunakan untuk menunjang penelitian. Teori-teori tersebut adalah sebagai berikut:

a. Penelitian yang telah dilakukan terdahulu dan berkaitan dengan *Payment Gateway Midtrans* serta penerapannya.
b. Chesspedia
c. Proses Rekayasa Perangkat Lunak
d. *Unified Modeling Language* yang berupa *use case diagram*, dan *class diagram.*
e. Teknologi pengembangan menggunakan REST pada *Payment Gateway* Midtrans sebagai *web service*, php dan *Laravel* sebagai pengembang sistem *web*

3.2 Pengumpulan Data

Pengumpulan data merupakan tahap awal yang bisa dilakukan yaitu melakukan pengumpulan data sebagai dasar dalam proses analisis kebutuhan. Dalam proses pengumpulan data, berdasarkan sumbernya terdapat 2 jenis data yang digunakan yakni data primer dan data sekunder.

1. Data primer yakni data yang dipe*roleh* langsung dari sumbernya. Pada penelitian ini, data primer bersumber dari wawancara kepada pemilik Chesspedia yang dilakukan secara langsung berupa hasil wawancara terkait permasalahan yang saat ini sedang di hadapi oleh chesspedia.
2. Data sekunder, yakni data yang dikumpulkan diolah dan diproses oleh pihak luar, yang berupa publikasi atau jurnal. Dalam penelitian ini, data sekunder dipe*roleh* dengan menggunakan metode jurnal. Yang berisi penelitian terkait mengenai sistem pembayarannya.

3.3 Analisis Kebutuhan

Tujuan dari analisis kebutuhan adalah untuk mendapatkan semua kebutuhan yang dibutuhkan oleh sistem yang akan dibuat. Analisis kebutuhan dimulai dengan mengidentifikasi semua kebutuhan sistem dan siapa saja yang terlibat didalamnya. Analisis kebutuhan juga berfungsi untuk mengetahui kondisi yang ada di lapangan sehingga dapat mengetahui implementasi perangkat lunak yang akan digunakan. Pada tahap ini akan dilakukan analisis terhadap hasil wawancara yang telah dilakukan sebelumnya kepada pemilik Chesspedia untuk disesuaikan dengan kebutuhan pemangku kepentingan. Hasil analisis ini akan didokumentasikan ke dalam tabel pernyataan masalah yang berisi solusi untuk menyelesaikan masalah yang telah diidentifikasikan.

Berikutnya adalah proses pemodelan *Use case* yang digunakan untuk mengetahui tujuan aktor menggunakan sistem yang dikembangkan. Kemudian dilakukan pembuatan spesifikasi *Use case* yang digunakan untuk menjelaskan secara singkat mengenai setiap *Use case* yang telah diidentifikasikan, memberikan informasi mengenai aktor yang menjalankan *use case*, keadaan sebelum dan sesudah *Use case* dijalankan serta tahap-tahap yang dilakukan ketika *Use case* berjalan untuk mecapai tujuan yang aktor lakukan.

3.4 Perancangan Sistem

Perancangan berfungsi sebagai acuan dalam implementasi dan pengujian dalam pembuatan sistem ini. Tahapan-tahapan perancangan dalam membangun sistem ini adalah sebagai berikut :

1. **Perancangan Arsitektur.**
   Pemodelan sistem akan dibuat dalam bentuk gambar arsitektur serta diagram UML seperti dan *Class Diagram.*

2. **Perancangan Basis Data**
   Rancangan Tabel database dijelaskan menggunakan ERD *(Entity Relationship Diagram)*. Rancangan tersebut akan menjadi dasar dalam implementasi database pada sistem.
3. **Perancangan Antarmuka**
   Antarmuka dari sistem yang akan dibangun akan dibuat tata letak komponen yang harus disedikan oleh sistem berdasarkan kebutuhan sistem dan diberi beberapa sampel antarmuka utama berdasarkan level pengguna. Rancangan tersebut akan menjadi dasar Implementasi Antarmuka sistem.

3.5 Implementasi

Tahap implementasi merupakan tahapan untuk melakukan implementasi dari hasil perancangan yang telah dilakukan. Hasil dari perancangan diterjemahkan menjadi bahasa pemrograman yang disusun untuk membangun perangkat lunak. Pada penelitian ini menggunakan bahasa pemrograman PHP dan menggunakan kerangka kerja laravel. Serta untuk menjunjang tampilan antarmuka digunakan kerangka kerja *bootstrap*. Selain itu pada tahap ini juga dilakukan implementasi basis data.

3.6 Pengujian

Perangkat lunak yang telah diimplementasikan akan melalui tahap pengujian. Tahap ini merupakan tahapan dimana dalam perangkat lunak akan diuji untuk memastikan bahwa aplikasi yang telah dibangun memenuhi seluruh fungsionalitas sistem yang dibutuhkan. Pengujian dilakukan menggunakan *black-box testing*, *usability testing,* dan *compatibility testing.*

Pengujian Validasi dilaksanakan untuk mengetahui apakah program dapat berjalan sesuai dengan fungsi yang ditetapkan, dan tidak memiliki kesalahan ketika melakukan suatu fungsi tertentu.

Usability Testing dilakukan untuk mengukur seberapa mudah sistem untuk digunakan. Pengujian *usability* dilakukan dengan menggunakan metode SUS (*System Usability Scale*). Pengujian ini dilakukan dengan cara memberikan gambaran sistem berupa *mockup* atau *screenshot* beberapa halaman sistem kepada pengguna dengan tujuan pengguna dapat memperkirakan kemudahan sistem saat digunakan, kemudian pengguna diminta untuk mengisi sebuah kuesioner dan dari jawaban kuesioner tersebut yang akan dikalkulasi sebagai nilai *usability* atau kemudahan sistem.

Compability Testing dilakukan untuk memastikan bahwa sebuah perangkat lunak dapat berjalan dengan baik pada berbagai macam perangkat. Untuk pengujian pada *website*, maka pengujian kompatibilitas ini adalah untuk memeriksa kompatibiltas *website* dengan berbagai macam browser, apakah *website* yang dibangun dapat berjalan dengan lancar atau tidak. Untuk melakukan uji kompatibilitas, dapat dilakukan dengan menggunakan *tools* yang bernama *Sortsite.*

3.7 Kesimpulan dan Saran

Setelah semua tahapan telah selesai dilakukan maka pada tahap terakhir diberikan kesimpulan dari penelitian yang telah dilakukan. Selain kesimpulan juga diberikan saran yang menjadi catatan yang dapat digunakan untuk pengembangan perangkat lunak dalam penelitian berikutnya.

BAB 4 ANALISIS KEBUTUHAN

Pada penelitian ini, analisis kebutuhan bertujuan untuk memberikan gambaran mengenai kebutuhan yang harus terpenuhi oleh pengguna sistem . Hasil dari analisis kebutuhan ini akan dijadikan sebagai acuan dalam perancangan *website e-commerce* Chesspedia. Pembahasan dibagi menjadi deskripsi umum sistem, pemodelan proses bisnis, identifikasi aktor, identifikasi kebutuhan dan pemodelan kebutuhan.

4.1 Deskripsi Umum Sistem

Sistem yang dikembangkan merupakan *website* e-commerce yang dapat digunakan oleh pelanggan untuk melakukan berbelanja, dan juga dapat digunakan oleh admin untuk mengelola produk dan pesanan yang masuk. Pelanggan dapat mengakses halaman katalog produk dimana pelanggan dapat melihat semua produk yang ditawarkan. Pelanggan dapat melihat detail dari produk yang dipilih seperti harga dan gambar produk sebelum melakukan transaksi pembelian. Saat melakukan *checkout*, pelanggan dapat memilih kurir dan durasi pengiriman yang diinginkan, dengan ongkos kirim yang berbeda-beda. Pembayaran dapat dilakukan dengan berbagai metode pembayaran yang disediakan, seperti transfer bank, kartu kredit, kartu debit, maupun via *e-money* seperti Gopay, OVO dan shopeePay.

Untuk mengelola produk, Admin memiliki akses menuju halaman panel admin. Pada halaman panel admin, Admin dapat melihat seluruh pesanan yang masuk, dan dapat melakukan filter terhadap pesanan masuk berdasarkan status pesanan tersebut. Admin dapat menyetujui pesanan jika bukti pembayaran sudah diverifikasi oleh sistem *Payment Gateway* Midtrans. Jika pesanan sudah disetujui, maka admin dapat melanjutkan ke proses pengiriman, dan menambahkan nomor resi pengiriman jika pesanan sudah mendapatkan nomor resi dari ekspedisi yang digunakan. Selain itu, Admin juga dapat menambahkan produk baru, serta mengubah produk yang sudah ada.

4.2 Identifikasi Aktor

Individu/aktor yang akan melakukan interaksi dengan sistem. Pada sistem ini aktor yang terlibat adalah pengguna. Pengguna merupakan sebuah entitas berupa orang yang melakukan interaksi dengan sistem. Penjelasan mengenai aktor yang akan berinteraksi dengan sistem terdapat pada Tabel 4.1.

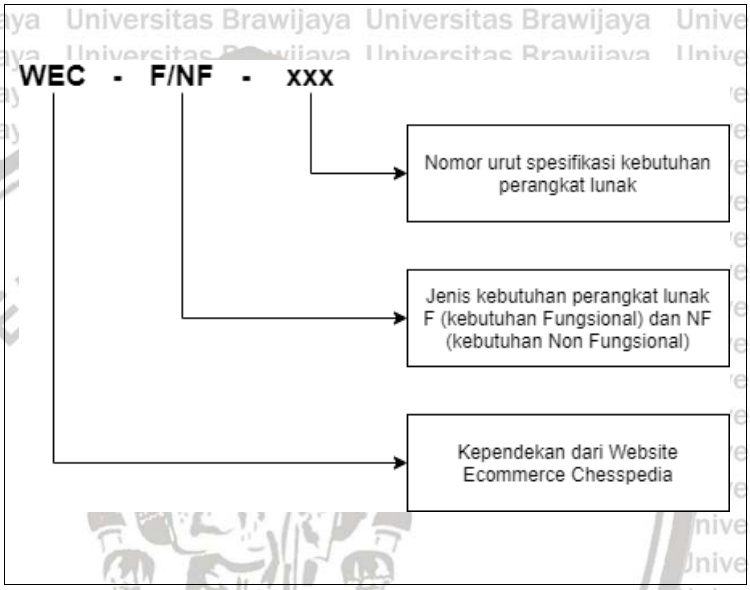
Tabel 4.1 Identifikasi Aktor

No	Aktor	Deskripsi
1	Pengunjung	Pengunjung merupakan seseorang yang mengunjungi web *e-commerce Chesspedia*
2	Pelanggan	Pelanggan merupakan pengunjung yang telah melakukan register dan masuk kedalam sistem web *e-commerce Chesspedia*

3	Admin	Mempunyai fungsi untuk mengelola dan mengatur segala sesuatu yang berkaitan dengan web *e-commerce Chesspedia*

4.3 Identifikasi Kebutuhan

Tahap indentifikasi kebutuhan merupakan proses untuk dapat memperoleh daftar kebutuhan yang diperlukan oleh sistem. Terdapat dua jenis kebutuhan pada sistem ini yaitu kebutuhan fungsional dan kebutuhan non fungsional. Penamaan kode ditunjukkan dalam Gambar 4.1.



Gambar 4.1 Kode Kebutuhan

4.3.1 Kebutuhan Fungsional

Kebutuhan fungsional dibagi berdasarkan jenis aktor yang terlibat. Kebutuhan fungsional pada Website E-Commerce Chesspedia dibagi menjadi dua sesuai dengan jumlah aktor, yaitu kebutuhan fungsional untuk pengunjung & pengguna, dan kebutuhan fungsional untuk admin sebagaimana masing - masing ditunjukkan dalam Tabel 4.2 dan Tabel 4.3,

Tabel 4.2 Kebutuhan Fungsional Pengunjung & Pelanggan

No	Pelanggan		
	Kode Fungsi	**Nama Fungsi**	**Deskripsi**
1	WEC-F-PNG1	*Register*	Sistem dapat melakukan pendaftaran user baru
2	WEC-F-PNG2	Melihat Daftar Produk	Sistem dapat digunakan melihat daftar produk

3	WEC-F-PNG3	Melihat Detail Produk	Sistem dapat melihat halaman detail produk
4	WEC-F-PNG4	Search	Sistem dapat melakukan pencarian produk
5	WEC-F-PLG1	*Login*	Sistem dapat melakukan *login*
6	WEC-F-PLG2	*Logout*	Sistem dapat melakukan *logout*
7	WEC-F-PLG3	Melihat Daftar Produk	Sistem dapat melihat daftar produk
8	WEC-F-PLG4	Melihat Detail Produk	Sistem dapat melihat detail produk
9	WEC-F-PLG5	Menambah produk ke dalam keranjang belanja	Sistem dapat menambahkan produk kedalam keranjang belanja
10	WEC-F-PLG6	*Search*	Sistem dapat melakukan pencarian produk
11	WEC-F-PLG7	Melakukan *Checkout*	Sistem dapat melakukan *checkout* produk yang sudah ditambahkan ke keranjang belanja
12	WEC-F-PLG8	Melakukan pembayaran	Sistem dapat melakukan pembayaran dari produk yang di beli pelanggan
13	WEC-F-PLG9	Mengubah data pribadi	Sistem dapat mengubah data pelanggan
14	WEC-F-PLG10	Melihat Riwayat transaksi	Sistem dapat melihat Riwayat transaksi pelanggan
15	WEC-F-PLG11	Melihat status Transaksi	Sistem dapat melihat status transaksi pelanggan yang sedang aktif / berjalan

Dan Tabel 4.3 menjelaskan kebutuhan fungsional pada aktor admin, terdapat lima kebutuhan fungsional.

Tabel 4.3 Kebutuhan Fungsional Admin

No	Kode Fungsi	Nama Fungsi	Deskripsi

1	WEC-F-ADM1	Melakukan *Login*	Sistem dapat melakukan *Login* yang terhubung ke server agar dapat masuk ke dalam sistem.
2	WEC-F-ADM2	Melakukan *Logout*	Sistem dapat melakukan *Logout*
3	WEC-F-ADM3	Mengelola pesanan	Sistem dapat mengelola pesanan
4	WEC-F-ADM4	Mengelola Produk	Sistem dapat mengelola produk
5	WEC-F-ADM5	Mengelola Pengiriman	Sistem dapat mengelola pengiriman

4.3.2 Kebutuhan Non-Fungsional

Selain kebutuhan fungsional, terdapat sebuah kebutuhan yang dapat menunjang pengalaman pengguna dalam penggunaan sistem ini yaitu kebutuhan non-fungsional yang terdapat dalam Tabel 4.4.

Tabel 4.4 Kebutuhan Non Fungsional

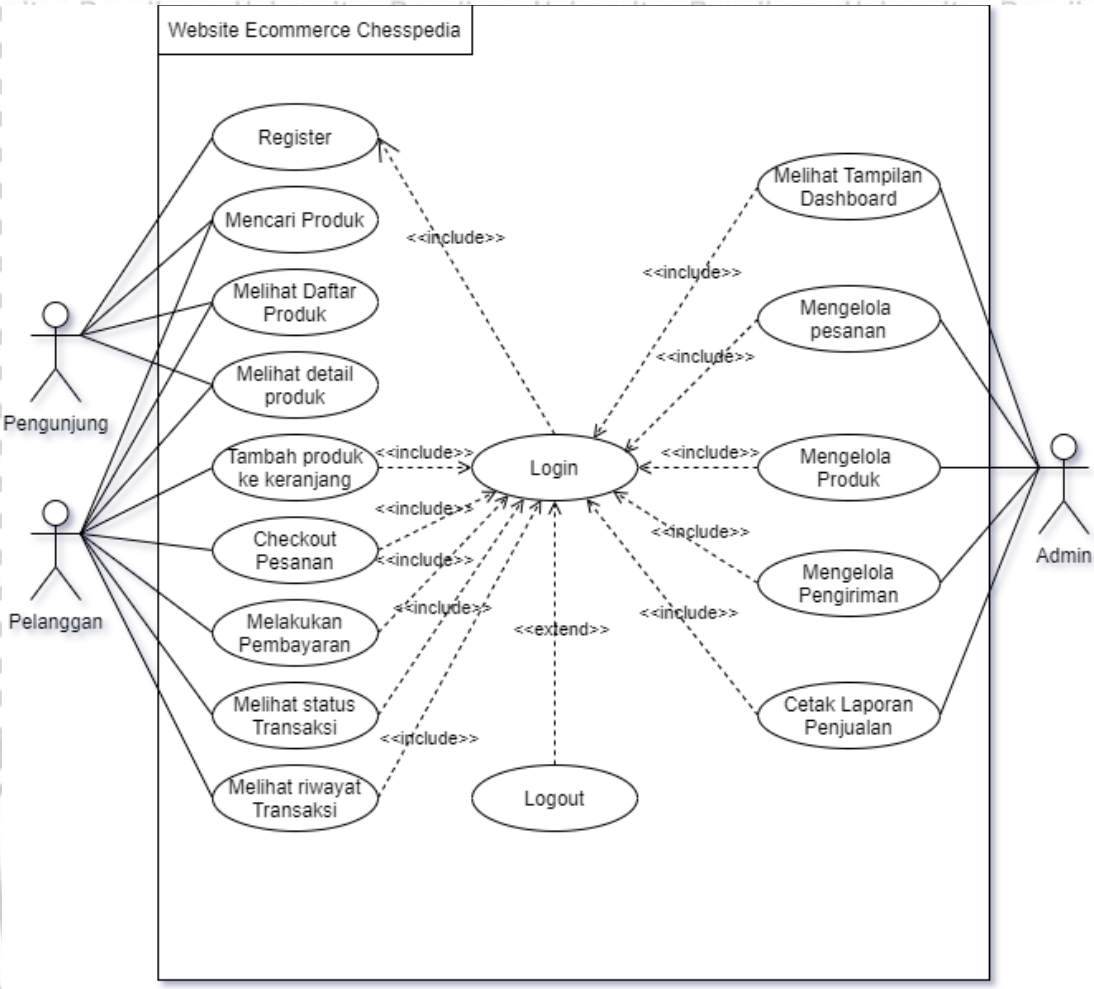
No	Kode Fungsi	Nama Fungsi	Deskripsi
1	WEC-NF-ALL1	*Usability*	Kemudahan penggunaan sistem atau perangkat lunak oleh user.

4.4 Pemodelan *Use Case*

Pemodelan *use case* merupakan cara dalam mendeskripsikan kebutuhan fungsional sistem ke dalam bentuk diagram yang bertujuan untuk memudahkan antara pengembang dengan pengguna sistem dalam memahami kebutuhan sistem dan sebagai acuhan untuk melakukan validasi kebutuhan.

4.4.1 Use Case Diagram

Use case Diagram merupakan jenis pemodelan *Unified Markup Language* (UML) yang berfokus untuk menggambarkan kebutuhan-kebutuhan dan fungsionalitas dari sistem serta digunakan untuk menunjukkan aksi-aksi yang dilakukan oleh aktor terhadap sistem dan mampu memberikan deskripsi tentang tipe interaksi yang terjadi. Pada Gambar 4.2 menggambarkan Use Case Diagram pada web *E-Commerce* Chesspedia yang mana memiliki 3 aktor.



Gambar 4.2 *Use Case Diagram* pada Website Chesspedia

4.4.2 Use Case Scenario

4.4.2.1 Use Case Scenario Melakukan *Register*

Use Case Scenario melakukan *register* yang terdapat pada Tabel 4.5 dilakukan oleh aktor *Pengunjung*. Aktor memasukan data pribadi seperti email, nama, dan *password* untuk terdaftar dalam sistem. Ketika aktor telah berhasil melakukan registrasi, user menginput *username* dan *password* kembali, maka sistem mengarah ke halaman *home.* Jika aktor telah memasukkan *username* dan *password* maka sistem akan menampilkan pemberitahuan dan tampilan tetap halaman *login*.

Tabel 4.5 *Use Case Scenario Register*

Melakukan *Register*	
Code	WEC-F-PNG1
Actors	*Pengunjung*

Description	Skenario untuk melakukan *Register*
Use Case Goal	Aktor dapat melakukan *Register*
Preconditions	Aktor telah mengakses *website* Chesspedia
Main Flow	1. Aktor memilih menu *register* 2. Modal untuk *register* akan muncul 3. Aktor mengisi data diri yang dibutuhkan, beserta email dan *password* 4. Sistem memverifikasi akun 5. Aktor dialihkan ke halaman *home*, dengan modal *login* yang akan ditampilkan oleh sistem
Alternative Flow	Jika main *flow* pada Langkah ke 3 tidak berhasil : 4. ditampilkan informasi mengapa proses registrasi tidak berhasil serta menampilkan sebab registrasi tidak berhasil.
Postconditions	Aktor berhasil melakukan registrasi

4.4.2.2 Use Case Scenario Melihat Daftar Produk

Pada Tabel 4.6 merupakan *Use Case Scenario* untuk melihat daftar produk. Pada *homepage* chesspedia terdapat *carousell* untuk melihat berbagai produk yang dijual. Sistem akan menampilkan daftar produk apa saja yang mendapatkan highlight seperti diskon, keluaran terbaru.

Tabel 4.6 *Use Case Scenario* Melihat Daftar Produk

Melihat Daftar Produk	
Code	WEC-F-PNG2
Actors	*Pengunjung*
Description	Skenario untuk melihat daftar produk
Use Case Goal	Aktor dapat melihat daftar produk
Preconditions	Aktor berada pada halaman selain halaman daftar produk
Main Flow	1. Aktor memilih menu daftar produk yang berada pada menu navigasi 2. Sistem menampilkan halaman katalog produk
Alternative Flow	-
Postconditions	Aktor berhasil mengakses halaman daftar produk

4.4.2.3 Use Case Scenario Melihat Detail Produk

Pada Tabel 4.7 merupakan *Use Case Scenario* untuk melihat detail produk. Pada *homepage* terdapat berbagai menu melihat produk, salah satunya melihat detail produk. Setelah memilih produk, maka sistem akan mengarahkan ke halaman detail produk. Kemudian aktor dapat melihat detail dari produk seperti foto produk lebih banyak, deskripsi, harga, serta informasi stok yang tersedia.

Tabel 4.7 *Use Case Scenario* Melihat Detail Produk

Melihat Detail Produk	
Code	WEC-F-PNG3
Actors	*Pengunjung, Pelanggan*
Description	Skenario untuk melihat detail produk
Use Case Goal	Aktor dapat melihat halaman detail produk
Preconditions	Aktor berada pada halaman daftar produk
Main Flow	1. Aktor memilih produk yang diinginkan pada daftar produk 2. Sistem menampilkan halaman detail produk dari produk yang di pilih oleh aktor
Alternative Flow	Jika produk yang dipilih aktor memiliki status kosong atau *sold out* : 2. informasi mengenai habisnya stok produk ditampilkan pada halaman detail produk 3. Tombol beli dan tambah keranjang di *nonaktifkan* fungsinya pada halaman detail produk
Postconditions	Aktor dapat mengakses halaman produk

4.4.2.4 Use Case Scenario Mencari Produk

Pada Tabel 4.8 merupakan *Use Case Scenario* untuk mencari produk. Pada *homepage* terdapat search bar pada bagian *footer*. Aktor dapat mencari produk yang diinginkan dengan menginputkan kata kunci maupun nama produk yang diinginkan kedalam search bar. Setelah aktor melakukan klik pada search bar, sistem akan menampilkan daftar produk yang terkait dengan kata kunci yang diinputkan oleh aktor. Jika produk yang diinginkan tidak ada, maka sistem akan menampilkan halaman yang berisikan keterangan bahwa produk dengan kata kunci tersebut tidak ditemukan atau tidak ada.

Tabel 4.8 *Use Case Scenario* Mencari Produk

Mencari Produk	
Code	WEC-F-PNG4
Actors	*Pengunjung*
Description	Skenario untuk mencari produk
Use Case Goal	Aktor dapat mencari produk yang diinginkan
Preconditions	Aktor berada pada halaman *homepage, Daftar Produk, maupun halaman detail produk*
Main Flow	1. Aktor mengisikan kata kunci pencarian pada kolom pencarian yang berada di *footer* halaman 2. Aktor klik tombol cari atau menekan tombol enter 3. Sistem menampilkan hasil pencarian produk yang sesuai dengan kata kunci pencarian
Alternative Flow	3. Jika data produk tidak tersedia dengan kata kunci yang diberikan 4. Sistem tidak menampilkan data
Postconditions	Aktor berhasil menampilkan daftar produk

4.4.2.5 Use Case Scenario Melakukan *Login*

Pada Tabel 4.9 merupakan *Use Case Scenario* untuk melakukan *login*. Aktor memasukan data seperti email, *password* yang telah terdaftar dalam sistem. Selanjutnya, sistem akan melakukan verifikasi terhadap data yang telah diinputkan. Ketika aktor telah berhasil melakukan *login* maka aktor akan di teruskan ke *homepage* chesspedia. Namun, jika aktor gagal melakukan *login*, maka sistem akan menampilkan pesan error kepada aktor terkait data apay an tidak sesuai, seperti kesalahan *username* maupun *password*.

Tabel 4.9 *Use Case Scenario Login*

Login	
Code	WEC-F-PLG1
Actors	*Pelanggan*
Description	Skenario untuk melakukan *login*
Use Case Goal	Aktor berhasil melakukan *login*
Preconditions	Aktor berada pada halaman *home*, daftar produk, maupun detail produk
Main Flow	1. Aktor memilih menu *login* pada *footer*

	2. Sistem akan membuka halaman *login* untuk ditampilkan 3. Aktor mengisi *username* dan *password* 4. Sistem memverifikasikan input dari aktor 5. Aktor berhasil masuk ke dalam sistem
Alternative Flow	Jika sistem gagal melakukan validasi pada Langkah nomor 4 pada main *flow* maka : 1. sistem menampilkan informasi penyebab gagalnya *login*. Aktor akan Kembali diminta untuk melakukan input Kembali *username* dan *password* dengan benar.
Postconditions	Aktor berhasil melakukan *login*

4.4.2.6 Use Case Scenario Melakukan *Logout*

Pada Tabel 4.10 merupakan *Use Case Scenario* untuk melakukan *Logout* dari sistem. Aktor memilih icon *logout* kemudian menghapus *session* pengguna. Sistem menampilkan halaman *Login.*

Tabel 4.10 *Use Case Scenario* Melakukan *Logout*

	Melakukan *Logout*
Code	WEC-F-PLG2
Actors	*Pelanggan*
Description	Skenario untuk melakukan *Logout*
Use Case Goal	Aktor dapat *Logout* dari sistem
Preconditions	Aktor telah *Login* ke sistem
Main Flow	1. Aktor memilih menu *logout* pada *footer* 2. Sistem menghapus *session* milik aktor 3. Sistem Kembali menampilkan *homepage chesspedia*
Alternative Flow	-
Postconditions	Aktor berhasil melakukan *Logout*

4.4.2.7 Use Case Scenario Melihat Daftar Produk

Pada Tabel 4.11 merupakan *Use Case Scenario* untuk melihat daftar produk. Pada *home*page chesspedia terdapat carousell untuk melihat berbagai produk yang dijual. Sistem akan menampilkan daftar produk apa saja yang mendapatkan highlight seperti diskon, keluaran terbaru.

Tabel 4.11 Melihat Daftar Produk

Melihat Daftar Produk	
Code	WEC-F-PLG3
Actors	Pelanggan
Description	Skenario untuk melihat daftar produk
Use Case Goal	Aktor dapat mengakses halaman daftar produk
Preconditions	Aktor berada pada halaman diluar daftar produk
Main Flow	1. Aktor memilih menu daftar produk yang berada pada menu navigasi 2. Sistem menampilkan halaman katalog produk
Alternative Flow	-
Postconditions	Aktor dapat mengakses halaman daftar produk

4.4.2.8 Use Case Scenario Melihat Detail Produk

Pada Tabel 4.12 merupakan *Use Case Scenario* untuk melihat detail produk. Pada *home*page chesspedia user memilih produk yang diinginkan pada daftar produk. Sistem akan menampilkan halaman detail produk dari produk yagng dipilih oleh aktor. Sistem akan menampilkan daftar produk apa saja yang mendapatkan highlight seperti diskon, keluaran terbaru.

Tabel 4.12 *Use Case Scenario* Melihat Detail Produk

Melihat Detail Produk	
Code	WEC-F-PLG4
Actors	Pelanggan
Description	Skenario untuk melihat halaman detail produk
Use Case Goal	Aktor dapat melihat halaman detail produk
Preconditions	Aktor berada pada halaman daftar produk
Main Flow	1. Aktor memilih produk yang diinginkan pada daftar produk 2. Sistem menampilkan halaman detail produk dari produk yang telah dipilih oleh aktor
Alternative Flow	Jika produk yang dipilih oleh aktor memiliki status kosong atau *sold out* : 1. Informasi mengenai stok produk habis akan ditampilkan pada halaman detail produk

	2. Tombol beli dan tambah keranjang di nonaktifkan fungsinya pada halaman detail produk
Postconditions	Aktor dapat mengakses halaman produk

4.4.2.9 Use Case Scenario Menambahkan Produk ke Keranjang

Pada Tabel 4.13 merupakan *Use Case Scenario* untuk menambahkan produk ke dalam keranjang. Ialah Use Case Scenario untuk memasukan produk yang dipilih oleh aktor ke dalam keranjang pembelian. Pada halaman detail produk, terdapat ikon di samping foto produk yang mana berbentuk sebagai keranjang. Setelah aktor melakukan klik pada icon keranjang, sistem akan menambahkan produk yang dipilih aktor kedalam keranjang pembelian. Jika stok produk tidak tersedia, maka fitur menambahkan produk ke keranjang akan dikunci.

Tabel 4.13 *Use Case Scenario* Menambah Produk ke Dalam Keranjang Belanja

Menambah Produk ke Dalam Keranjang Belanja	
Code	WEC-F-PLG5
Actors	Pelanggan
Description	Skenario untuk menambahkan produk ke dalam keranjang belanja
Use Case Goal	Aktor dapat melakukan penambahan produk ke dalam keranjang belanja
Preconditions	Aktor berada pada halaman detail produk dan memiliki status telah *login* ke dalam sistem
Main Flow	1. Aktor menekan tombol tambah ke keranjang 2. Sistem menambahkan produk yang dimaksud ke dalam keranjang belanja
Alternative Flow	Jika produk memiliki status habis atau *soldout* maka : 1. Tombol tambah ke keranjang di matikan
Postconditions	Aktor berhasil menambahkan produk ke keranjang belanja

4.4.2.10 Use Case Scenario Mencari Produk

Pada Tabel 4.14 merupakan *Use Case Scenario* untuk mencari produk. Pada *homepage* terdapat search bar pada bagian *footer*. Aktor dapat mencari produk yang diinginkan dengan menginputkan kata kunci maupun nama produk yang diinginkan kedalam search bar. Setelah aktor melakukan klik pada search bar, sistem akan menampilkan daftar produk yang terkait dengan kata kunci yang diinputkan oleh aktor. Jika produk yang diinginkan tidak ada, maka sistem akan

menampilkan halaman yang berisikan keterangan bahwa produk dengan kata kunci tersebut tidak ditemukan atau tidak ada.

Tabel 4.14 *Use Case Scenario* Mencari Produk

Mencari Produk	
Code	WEC-F-PLG6
Actors	*Pengunjung*
Description	Skenario untuk mencari produk
Use Case Goal	Aktor dapat mencari produk yang diinginkan
Preconditions	Aktor berada pada halaman *homepage, Daftar Produk, maupun halaman detail produk*
Main Flow	1. Aktor mengisikan kata kunci pencarian pada kolom pencarian yang berada di *footer* halaman 2. Aktor klik tombol cari atau menekan tombol enter 3. Sistem menampilkan hasil pencarian produk yang sesuai dengan kata kunci pencarian
Alternative Flow	3. Jika data produk tidak tersedia dengan kata kunci yang diberikan 4. Sistem tidak menampilkan data
Postconditions	Aktor berhasil menampilkan daftar produk

4.4.2.11 Use Case Scenario Melakukan *Checkout*

Pada Tabel 4.15 merupakan *Use Case Scenario* untuk melakukan *checkout* pada pesanan. Pada halaman keranjang, terdapat menu *checkout* yang nantinya bisa di setelah pembeli memilih produk yang ingin di *checkout*. Lalu sistem akan memunculkan halaman *checkout* yang mana aktor diharuskan melakukan pengisian alamat pengiriman, jenis pengiriman serta metode pembayaran yang akan di gunakan. Setelah data data tersebut lengkap, klik tombol bayar untuk melakukan pembayaran.

Tabel 4.15 *Use Case Scenario* Melakukan *Checkout*

Melakukan *Checkout*	
Code	WEC-F-PLG7
Actors	Pelanggan
Description	Skenario untuk melakukan proses *checkout*
Use Case Goal	Aktor berhasil melakukan *checkout*
Preconditions	Aktor berada pada halaman keranjang belanja

Main Flow	1. Aktor melakukan klik pada tombol *checkout* 2. Sistem menampilkan halaman *checkout* produk 3. Aktor diminta untuk mengisi data yang diperlukan seperti alamat, jenis pengiriman dan metode pembayaran yang diinginkan 4. Aktor melakukan klik pada tombol bayar
Alternative Flow	Jika data tidak lengkap maka : 1. Tombol bayar pesanan akan di nonaktifkan hingga aktor melengkapi data – data yang dibutuhkan
Postconditions	Aktor berhasil melakukan *checkout* produk

4.4.2.12 *Use Case Scenario* Melakukan Pembayaran

Use Case Scenario melakukan pembayaran terdapat pada Tabel 4.16 dilakukan oleh aktor pelanggan. Aktor melakukan klik bayar pada halaman *checkout* yang akan di lanjutkan di halaman pembayaran oleh *Payment Gateway Midtrans.* Selanjutnya aktor memilih jenis pembayaran apa yang diinginkan, seperti transfer bank, kartu kredit, maupun *e-money*. Jika aktor telah melakukan pembayaran maka user akan dialihkan kembali kehalaman *website e-commerce* chesspedia

Tabel 4.16 Use Case Scenario Melakukan Pembayaran

Melakukan Pembayaran	
Code	WEC-F-PLG8
Actors	Pelanggan
Description	Skenario untuk melakukan proses pembayaran
Use Case Goal	Aktor berhasil melakukan pembayaran
Preconditions	Aktor berada pada halaman *checkout*
Main Flow	1. Aktor melakukan klik pada tombol bayar 2. Sistem menampilkan halaman pembayaran pesanan 3. Aktor memilih metode pembayaran apa yang ingin digunakan 4. Aktor melakukan pembayaran sesuai metode pembayaran yang dipilih
Alternative Flow	Jika aktor gagal melakukan pembayaran maka : 5. status pesanan *"unpaid"* akan muncul di Riwayat transaksi pelanggan
Postconditions	Aktor berhasil melakukan pembayaran pesanan

4.4.2.13 *Use Case Scenario* Mengubah Data Pribadi

Use Case Scenario melakukan pembayaran terdapat pada Tabel 4.17 dilakukan oleh aktor pelanggan. Aktor melakukan klik bayar pada halaman *checkout* yang akan di lanjutkan di halaman pembayaran oleh *Payment Gateway Midtrans.* Selanjutnya aktor memilih jenis pembayaran apa yang diinginkan, seperti transfer bank, kartu kredit, maupun *e-money*. Jika aktor telah melakukan pembayaran maka user akan dialihkan kembali kehalaman *website e-commerce* chesspedia

Tabel 4.17 *Use Case Scenario* Mengubah Data Pribadi

Mengubah Data Pribadi	
Code	WEC-F-PLG9
Actors	Pelanggan
Description	Skenario untuk melakukan proses pembayaran
Use Case Goal	Aktor berhasil melakukan pembayaran
Preconditions	Aktor berada pada halaman *checkout*
Main Flow	1. Aktor melakukan klik pada tombol bayar 2. Sistem menampilkan halaman pembayaran pesanan 3. Aktor memilih metode pembayaran apa yang ingin digunakan 4. Aktor melakukan pembayaran sesuai metode pembayaran yang dipilih
Alternative Flow	Jika aktor gagal melakukan pembayaran maka : 5. status pesanan *"unpaid"* akan muncul di Riwayat transaksi pelanggan
Postconditions	Aktor berhasil melakukan pembayaran pesanan

4.4.2.14 Use Case Scenario Melihat Riwayat Transaksi

use case scenario melihat riwayat transaksi terdapat pada tabel 4.18 dilakukan oleh aktor pelanggan. aktor melakukan klik menu order pada halaman profil. sistem akan menampilkan riwayat orderan yang pernah dilakukan oleh aktor.

Tabel 4.18 *Use Case Scenario* Melihat Riwayat Transaksi

Melakukan Pembayaran	
Code	WEC-F-PLG9
Actors	Pelanggan

Description	Skenario untuk melihat riwayat transaksi
Use Case Goal	Aktor berhasil melihat riwayat transaksi
Preconditions	Aktor berada pada halaman profile
Main Flow	1. Aktor memilih menu profil pada halaman navigasi 2. Sistem menampilkan halaman *profile* 3. Aktor dapat melihat riwayat transaksi pada bagian daftar transaksi di halaman *profile*
Alternative Flow	-
Postconditions	Aktor berhasil melihat riwayat pesanan

4.4.2.15 *Use Case Scenario* Melihat Status Transaksi

Use Case Scenario melihat status transaksi terdapat pada Tabel 4.19 dilakukan oleh aktor pelanggan. Aktor melakukan klik bayar pada halaman *checkout* yang akan di lanjutkan di halaman pembayaran oleh *Payment Gateway Midtrans.* Selanjutnya aktor memilih jenis pembayaran apa yang diinginkan, seperti transfer bank, kartu kredit, maupun *e-money*. Jika aktor telah melakukan pembayaran maka user akan dialihkan kembali kehalaman *website e-commerce* chesspedia

Tabel 4.19 *Use Case Scenario* Melihat Status Transaksi

Melihat Status Transaksi	
Code	WEC-F-PLG9
Actors	Pelanggan
Description	Skenario untuk melihat status transaksi
Use Case Goal	Aktor berhasil melihat status transaksi
Preconditions	Aktor berada pada halaman *profile*
Main Flow	1. Aktor memilih menu profil pada halaman navigasi 2. Sistem menampilkan halaman *profile* 3. Aktor memilih transaksi yang ingin dilihat statusnya 4. Sistem menampilkan informasi lengkap terkait transaksi yang dipilih
Alternative Flow	-
Postconditions	Aktor berhasil melihat riwayat pesanan

4.4.2.16 *Use Case Scenario* Melakukan *Login Admin*

Use Case Scenario melakukan *Login* di *web* yang terdapat pada Tabel 4.20 dilakukan oleh aktor Admin. Aktor memasukkan *username* dan *password* yang sudah terdaftar dalam sistem. Ketika aktor telah memasukkan *username* dan *password* dengan benar, maka sistem mengarah ke halaman *dashboard*. Jika aktor telah memasukkan *username* dan *password* maka sistem akan menampilkan pemberitahuan dan tampilan tetap halaman *login*.

Tabel 4.20 *Use Case Scenario* Melakukan *Login*

Melakukan *Login*	
Code	WEC-F-ADM1
Actors	Admin
Description	Skenario untuk melakukan *Login*
Use Case Goal	Aktor dapat melakukan *Login*
Preconditions	Aktor mengakses sistem di *web*
Main Flow	1. Aktor mengakses sistem di *web* 2. Sistem menampilkan halaman *Login* 3. Aktor memasukkan *username* dan *password* dan menekan tombol *Login* 4. Sistem mengidentifikasi akun 5. Sistem menampilkan halaman *dashboard*
Alternative Flow	4.a Jika input *username* dan *password* tidak *valid* 4.b Sistem menampilkan pemberitahuan 4.c. Kembali ke *flow* 3
Postconditions	Aktor berhasil masuk ke dalam sistem

4.4.2.17 *Use Case Scenario* Mengelola Produk

Pada Tabel 4.21 merupakan *Use Case Scenario* admin mengelola produk. Aktor dapat melakukan tambah produk, edit produk, hapus produk serta melakukan tambah gambar produk.

Tabel 4.21 *Use Case Scenario* Mengelola Produk

Mengelola Produk	
Code	WEC-F-ADM2
Actors	Admin
Description	Skenario untuk mengelola produk
Use Case Goal	Aktor dapat mengelola pesanan yang masuk

Preconditions	Aktor *login* sebagai admin
Main Flow	1. Aktor memilih menu pesanan 2. Sistem menampilkan halaman pesanan 3. Aktor dapat memfilter pesanan berdasarkan status, menyetujui atau menolak pesanan
Alternative Flow	-
Postconditions	Aktor berhasil mengelola pesanan yang masuk

4.4.2.18 *Use Case Scenario* Mengelola Pesanan

Pada Tabel 4.22 merupakan *Use Case Scenario* admin mengelola pesanan. Aktor dapat melakukan terima pesanan, tolak pesanan, serta melakukan pengecekan pesanan.

Tabel 4.22 Mengelola Pesanan

Mengelola Pesanan	
Code	WEC-F-ADM3
Actors	Admin
Description	Skenario untuk mengelola produk
Use Case Goal	Aktor dapat mengelola produk
Preconditions	Aktor *login* sebagai admin
Main Flow	1. Aktor memilih menu produk 2. Sistem menampilkan halaman produk 3. Aktor dapat menambah, menghapus, meng-*update*, produk serta melakukan *update* stok.
Alternative Flow	-
Postconditions	Aktor berhasil mengelola produk

4.4.2.19 *Use Case Scenario* Mengelola Pengiriman

Pada Tabel 4.23 merupakan *Use Case Scenario* admin mengelola pengiriman. Aktor dapat mengisi *airwaybill (*resi) serta *update airwaybill.*

Tabel 4.23 *Use Case Scenario* Mengelola Pengiriman

Mengelola Pengiriman	
Code	WEC-F-ADM4

Actors	Admin
Description	Skenario untuk mengelola pengiriman
Use Case Goal	Aktor dapat mengelola pengiriman
Preconditions	Aktor *login* sebagai admin
Main Flow	1. Aktor memilih menu pengiriman 2. Sistem menampilkan halaman pengiriman 3. Aktor dapat melihat informasi terkait pengiriman dan melakukan penambahan resi
Alternative Flow	-
Postconditions	Aktor berhasil mengelola pengiriman

4.4.2.20 Use Case Scenario Melakukan *Logout*

Pada Tabel 5.15 merupakan *Use Case Scenario* untuk melakukan *Logout* sistem. Aktor memilih icon *logout* kemudian menghapus *session* pengguna . Sistem menampilkan halaman *Login.*

Tabel 4.24 *Use Case Scenario* Melakukan *Logout*

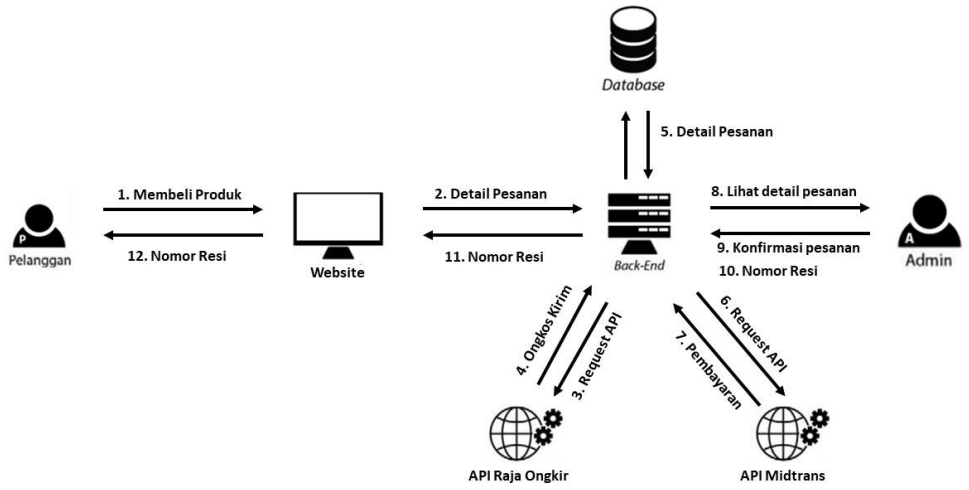
Melakukan *Logout*	
Code	SAL-F-017
Actors	Admin
Description	Skenario untuk melakukan *Logout*
Use Case Goal	Aktor dapat *Logout* dari sistem
Precondition	Aktor telah *Login* ke sistem
Main Flow	1. Aktor memilih *icon Logout* 2. Sistem menghapus *session* pengguna 3. Sistem menampilkan halaman *Login*
Alternative Flow	-
Postcondition	Aktor berhasil melakukan *Logout*

BAB 5 PERANCANGAN SISTEM

Pada bab 5 berisi tentang pembuatan perancangan diagram untuk menunjang perancangan tampilan sistem informasi. Diagram yang dibuat berupa *sequence diagram*. Selain diagram tersebut, juga dibuat arsitektur sistem, struktur data penyimpanan serta rancangan antarmuka sistem.

5.1 Arsitektur Sistem

Pada Gambar 5.1, merupakan rancangan arsitektur secara umum dari Website E-Commerce Chesspedia yang dikembangkan. Pelanggan melakukan transaksi pembelian produk pada Website E-Commerce Chesspedia. Detail pesanan pelanggan ditambahkan dengan ongkos kirim yang didapatkan dari API RajaOngkir untuk kemudian dimasukkan kedalam database. Admin dapat melihat detail dari pesanan yang masuk, dan menunggu pembayaran dari pelanggan yang didapatkan dari API *Payment Gateway* Midtrans. Jika pembayaran sudah diterima, maka admin akan melakukan konfirmasi dan segera melakukan proses pengiriman produk. Nomor resi yang didapatkan akan ditampilkan pada dasbor pelanggan, dan status pembayaran mengalami perubahan.



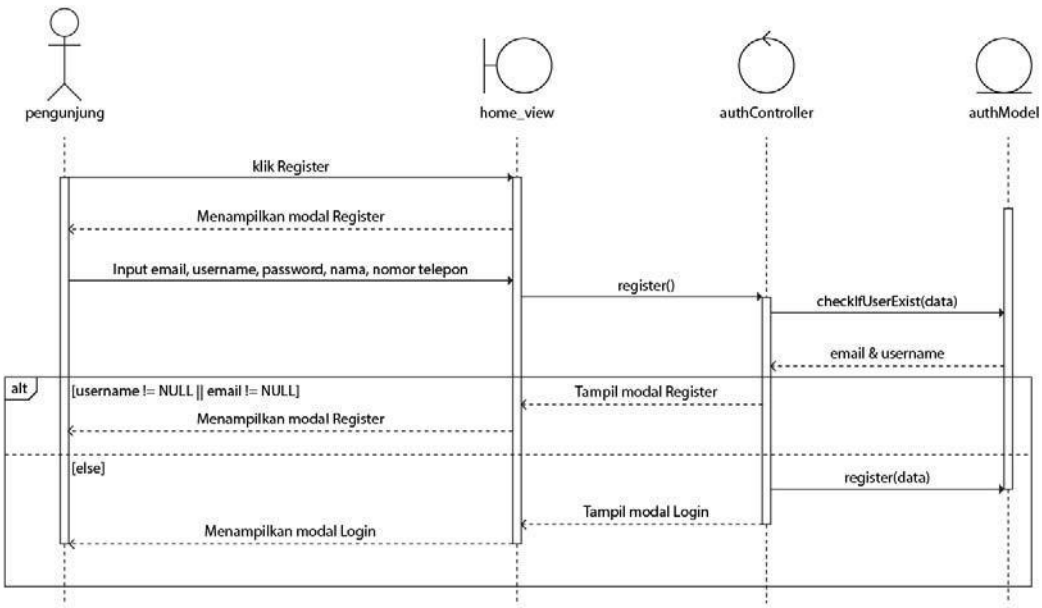
Gambar 5.1 Arsitektur Sistem

5.2 Sequence Diagram

5.2.1 *Sequence Diagram Register*

Pada gambar 5.2 merupakan *sequence diagram* untuk proses melakukan *register*. *Sequence diagram register* dimulai Ketika aktor yakni pengunjung melakukan klik pada menu *register* yang terletak di pojok kanan atas halaman utama. Saat menu tersebut diklik, halaman untuk *register* akan muncul dan meminta pengunjung untuk mengisikan data – data yang di butuhkan seperti

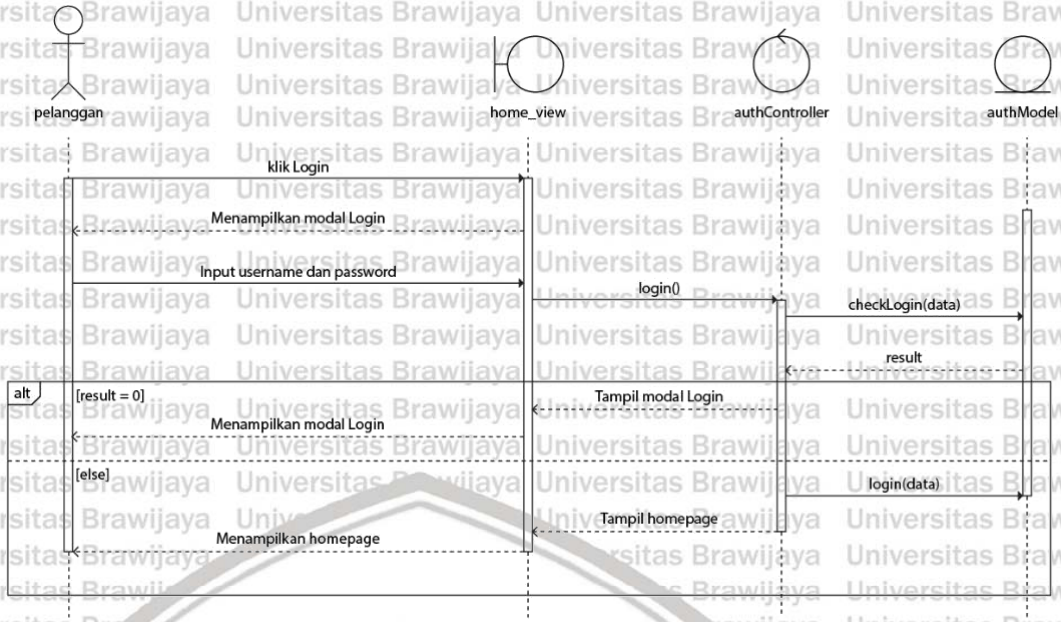
email, nama, *password* serta konfirmasi *password*. *authController* akan mengecek apakah data yang dimasukan oleh user pada *authModel* apakah data tersebut belum terdaftar atau sudah terdaftar. Jika user telah terdaftar maka akan dialihkan ke alternatif *flow* dimana saat aktor tersebut telah terdaftar akan dialihkan kembali ke halaman *login*. Namun, jika belum terdaftar maka pengunjung akan langsung dialihkan ke halaman utama



Gambar 5.2 *Sequence Diagram Register*

5.2.2 *Sequence Diagram Login*

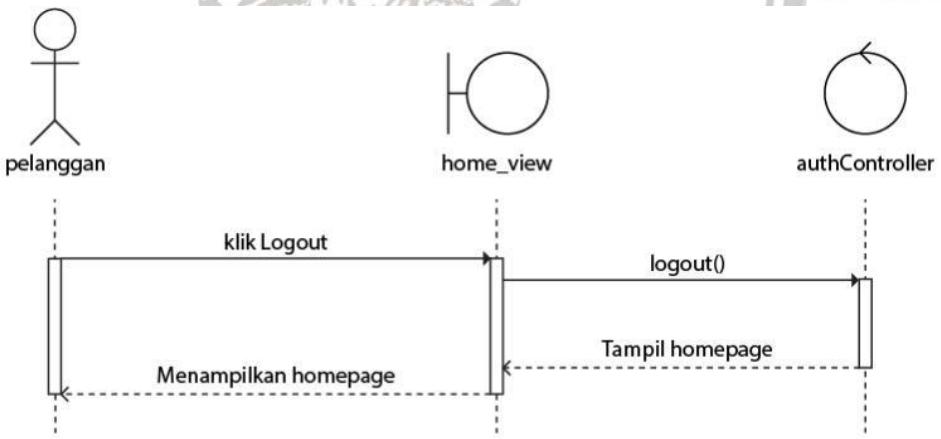
Gambar 5.3 merupakan *sequence diagram* pada proses *login*. *Sequence diagram* dimulai saat aktor yakni pelanggan melakukan klik pada menu *login* yang terletak pada pojok kanan atas pada halaman utama. Ketika menu ini diklik, modal untuk *login* tampil dan meminta pelanggan untuk mengisikan *username* dan juga *password*. *AuthController* akan melakukan cek data, apakah daya yang dimasukan telah sesuai atau tidak. Jika data tidak sesuai maka sistem akan menampilkan pesan error bahwa data tidak sesuai atau data tidak terdaftar. Jika data benar, maka aktor akan di alihkan ke halaman utama dengan kondisi aktor telah berhasil melakukan *login*.



Gambar 5.3 *Sequence Diagram Login*

5.2.3 *Sequence Diagram Logout*

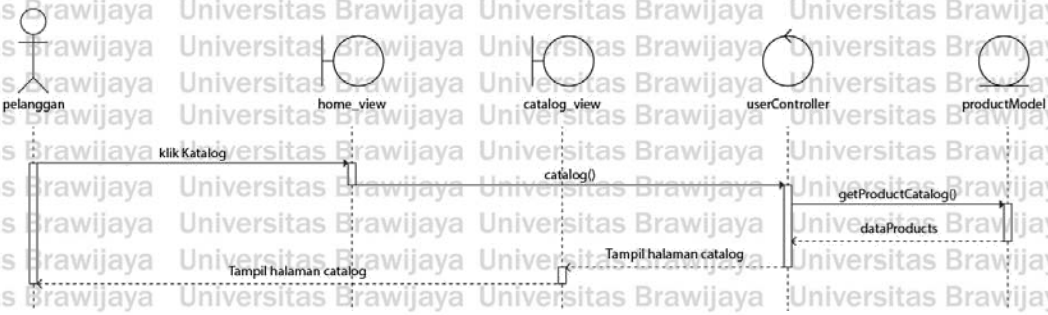
Gambar 5.4 berisi tentang *sequence diagram* untuk proses *logout*. Dimulai dari aktor melakukan klik pada menu *logout* yang terletak pada pojok kanan atas halaman utama. Controller akan melakukan *destroy session* pada *login* aktor, dan diarahkan kembali ke halaman utama.



Gambar 5.4 *Sequence Diagram Logout*

5.2.4 *Sequence Diagram* Melihat Daftar Seluruh Produk

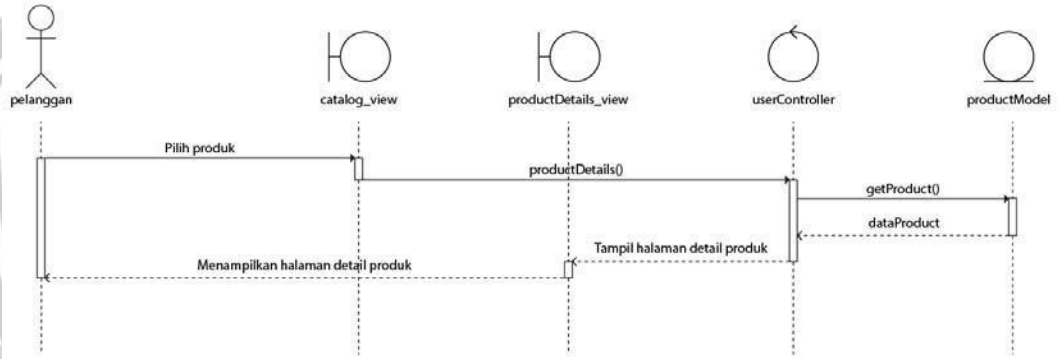
Gambar 5.5 merupakan *sequence diagram* untuk melihat daftar seluruh produk. *Sequence diagram* dimulai saat aktor yakni pelanggan melakukan klik show all products yang terdapat pada halaman utama. *userController* memanggil fungsi *getProduct*() pada *controllerProduct* untuk mengambil data produk yang akan ditampilkan pada halaman seluruh produk. Setelah itu, *userController* akan mengarahkan pelanggan ke halaman daftar seluruh produk.



Gambar 5.5 *Sequence Diagram* Melihat Daftar Seluruh Produk

5.2.5 *Sequence Diagram* Melihat Detail Produk

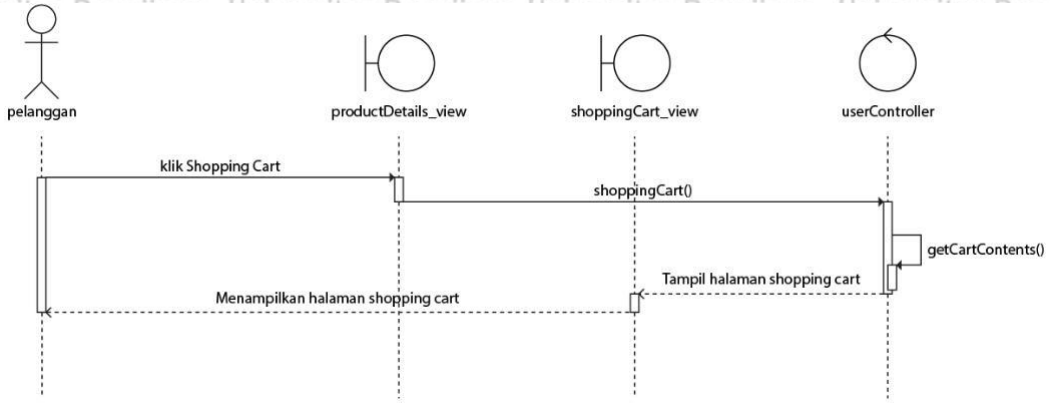
Pada Gambar 5.6 adalah *sequence diagram* untuk melihat katalog produk. *Sequence diagram* dimulai saat aktor yakni pelanggan memilih produk yang terdapat pada halaman *catalog*. *UserController* memanggil fungsi *getProduct*() pada *productModel* untuk mengambil data dari produk yang di pilih oleh aktor. Setelah itu, *userController* akan diarahkan oleh sistem aktor ke halaman detail produk.



Gambar 5.6 *Sequence Diagram* Melihat Detail Produk

5.2.6 *Sequence Diagram* Melihat Keranjang Belanja

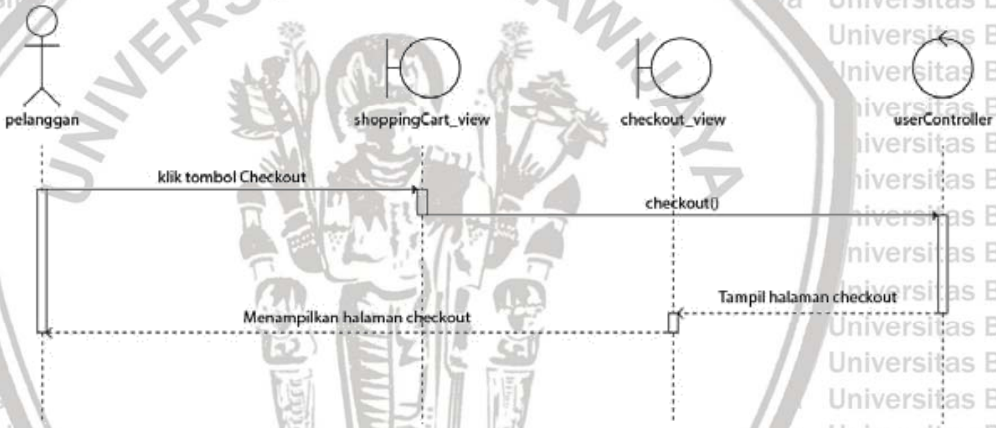
Pada Gambar 5.8 adalah *sequence diagram* untuk melihat *shopping cart*. *Sequence diagram* dimulai saat aktor (pelanggan) melakukan klik pada menu *shopping cart*. *UserController* memanggil fungsi getCartContents() untuk mengambil isi dari *shopping cart* yang terdapat pada *session* untuk ditampilkan pada halaman *shopping cart*. Setelah itu, *userController* akan mengarahkan aktor menuju halaman *shopping cart*.



Gambar 5.7 *Sequence Diagram* Melihat Keranjang Belanja

5.2.7 *Sequence Diagram* Menampilkan Halaman *Checkout*

Pada Gambar 5.8 adalah untuk menampilkan halaman *checkout*. dimulai saat aktor (pelanggan) melakukan klik pada menu *shopping cart*. *UserController* selanjutnya akan mengarahkan aktor menuju halaman *checkout*.



Gambar 5.8 *Sequence Diagram* Tampilkan Halaman *Checkout*

5.2.8 *Sequence Diagram* Melakukan *Checkout*

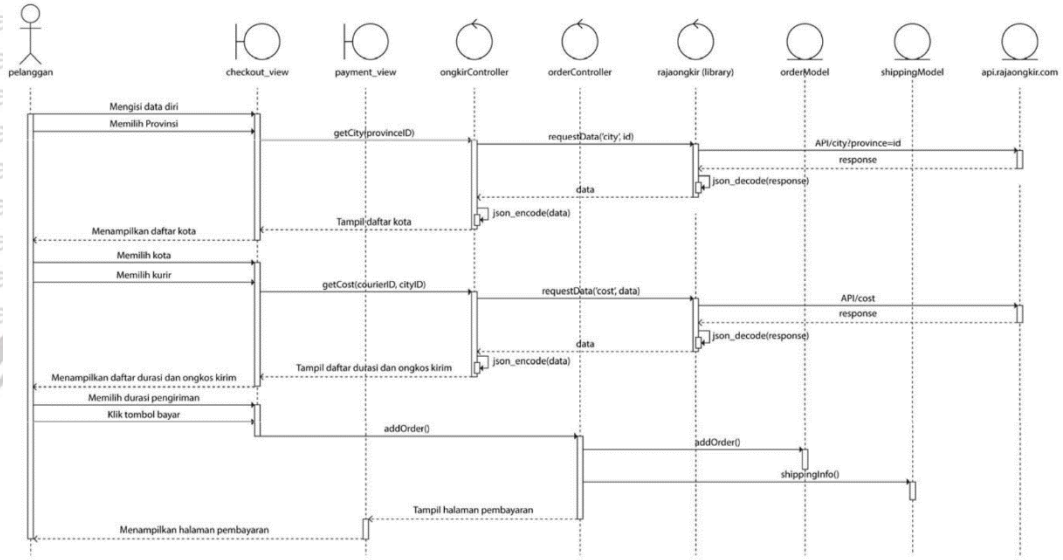
Pada Gambar 5.9 merupakan *sequence diagram* untuk melakukan *checkout*. *Sequence diagram* dimulai saat aktor pelanggan mengisi data diri dan memilih provinsi dari file json yang disediakan oleh sistem. Setelah provinsi dipilih maka akan memanggil fungsi *getCityID* dipanggil pada *shippingController* yang kemudian melakukan *request data* ke *api.rajaongkir.com* melalui RajaOngkir.

Hasil dari *decode* tersebut kemudian diolah untuk kemudian dilakukan *encode* kembali ke dalam bentuk JSON yang kemudian ditampilkan pada halaman *checkout.*

Aktor melanjutkan dengan memilih kota dan kurir. *getCost(courierID, cityID)* pada *ongkirController* dipanggil untuk melakukan *request data* ke

api.rajaongkir.com melalui RajaOngkir sama seperti fungsi *getCityID*. *Response* berbentuk JSON yang didapatkan dilakukan *decode* untuk diolah dan kembali dilakukan *encode* kedalam bentuk JSON untuk ditampilkan. Aktor melanjutkan memilih durasi pengiriman, dan klik tombol bayar.

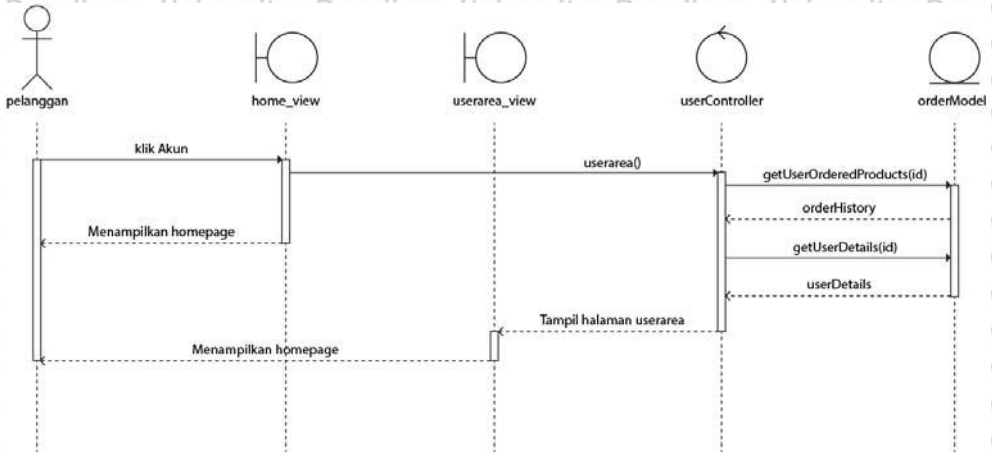
Ketika tombol bayar diklik, *orderController* memanggil *addOrder()* pada *orderModel* dan *shippingInfo()* pada *shippingModel* untuk menyimpan informasi pesanan dan pengiriman. Kemudian *orderController* akan mengarahkan aktor menuju halaman *payment*.



Gambar 5.9 *Sequence Diagram* Tampilkan *Checkout*

5.2.9 *Sequence Diagram* Melihat User Area

Pada Gambar 5.12 adalah untuk melihat user area/ *Sequence diagram* dimulai saat aktor (pelanggan) memilih menu akun. UserController memanggil fungsi getUserOrderedProducts(id) pada orderModel untuk mendapatkan orderHistory, dan memanggil fungsi getUserDetails(id) untuk mendapatkan userDetails. Setelah itu, userController mengarahkan aktor menuju halaman user area.



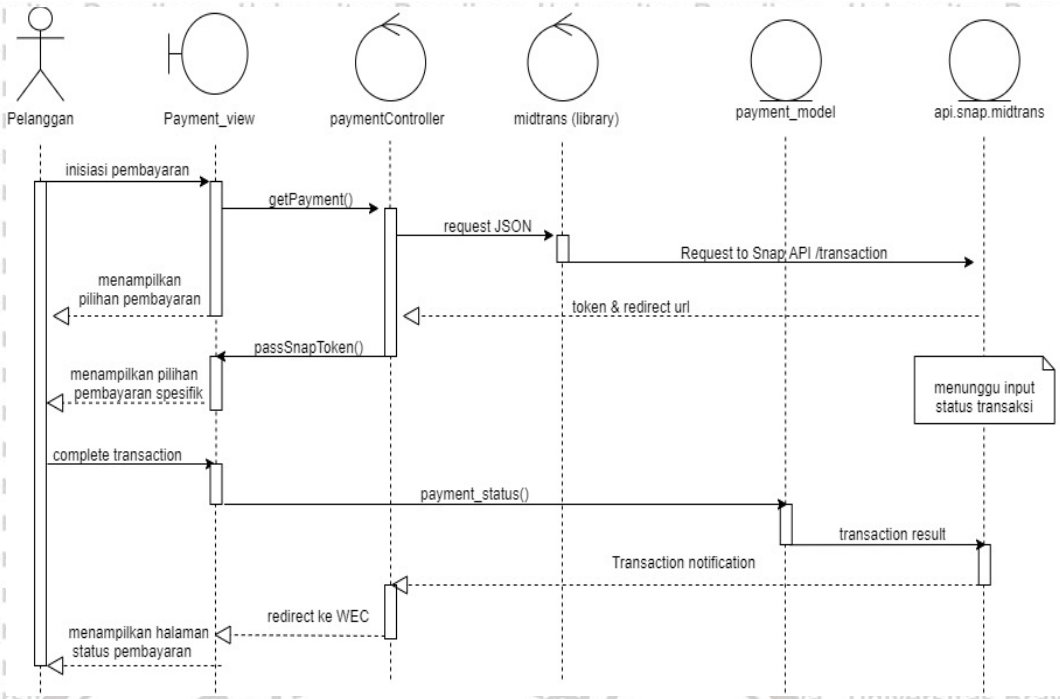
Gambar 5.10 Melihat Tampilan Menu User

5.2.10 *Sequence Diagram* Melakukan Pembayaran

Pada Gambar 5.9 merupakan *sequence diagram* untuk melakukan Pembayaran. *Sequence diagram* dimulai saat aktor pelanggan melakukan klik proses pembayaran. Lalu akan masuk ke tampilan dari *Payment Gateway* Midtrans. Sistem akan melakukan get Payment_Channels pada PaymentController, yang akan menampilkan metode pembayaran yang diinginkan seperti Atm/Bank Transfer, *E-wallet* maupun metode pembayaran lainnya. Setelah memilih metode pembayaran maka akan melakukan *request data* ke API Midtrans melalui Midtrans.

Hasil dari decode tersebut kemudian diolah untuk kemudian dilakukan *encode* Kembali kedalam bentuk JSON yang kemudian ditampilkan pada halaman Snap Pembayaran Midtrans. Aktor dapat melakukan pembayaran sesuai dengan metode pembayaran yang ditentukan pembeli.

Jika pelanggan telah menyelesaikan pembayaran transaksi, maka pelanggan akan di *redirect* menuju halaman status pemesanan. Di sisi *backend*, payment controller akan mengirimkan status transaksi terbaru yang mana akan men-*trigger* api midtrans dalam melakukan *update* status. Api midtrans akan melakukan *update notification* saat pembayaran telah di terima oleh pihak midtrans.

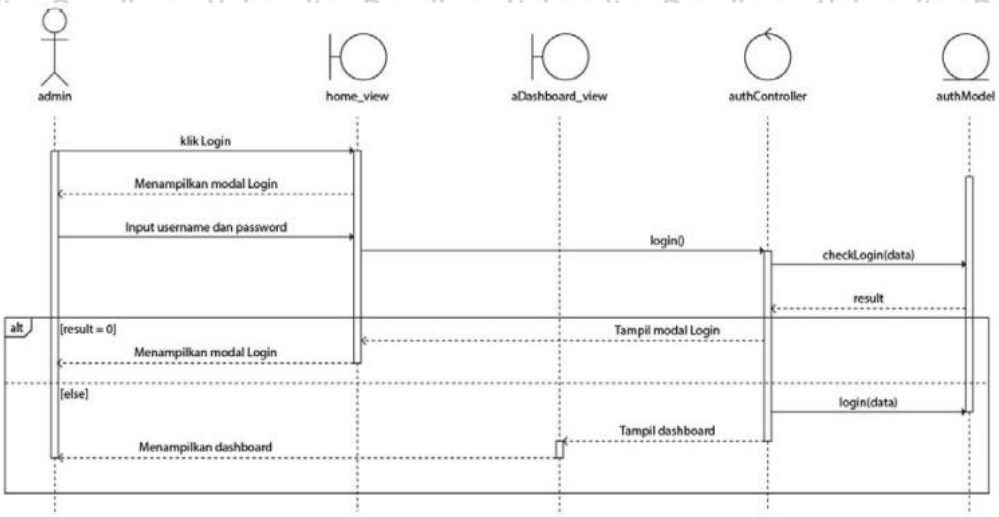


Gambar 5.11 *Sequence Diagram* Pembayaran

5.2.11 *Sequence Diagram* Melihat *Dashboard* Admin

Pada gambar 5.12 merupakan *sequence diagram* untuk melihat *dashboard* admin. Diagram ini dimulai saat admin melakukan *login* pada halaman utama chesspedia. Ketika admin telah melakukan *login*, *authController* akan melakukan pengecekan terkait data yang telah dimasukan oleh aktor, yang mana akan menentukan hak akses apa saja yang akan diberikan oleh authmodel. Disini authModel akan melakukan verifikasi apakah data yang di masukan sudah benar dan apakah aktor memiliki *role* sebagai admin.

Pada sequence ini terdapat *alternative flow* dimana jika *username* dan *password* salah, maka aktor akan dialihkan kehalaman utama chesspedia. Jika *username* dan *password* sudah benar maka akan di alihkan ke halaman *dashboard* admin.



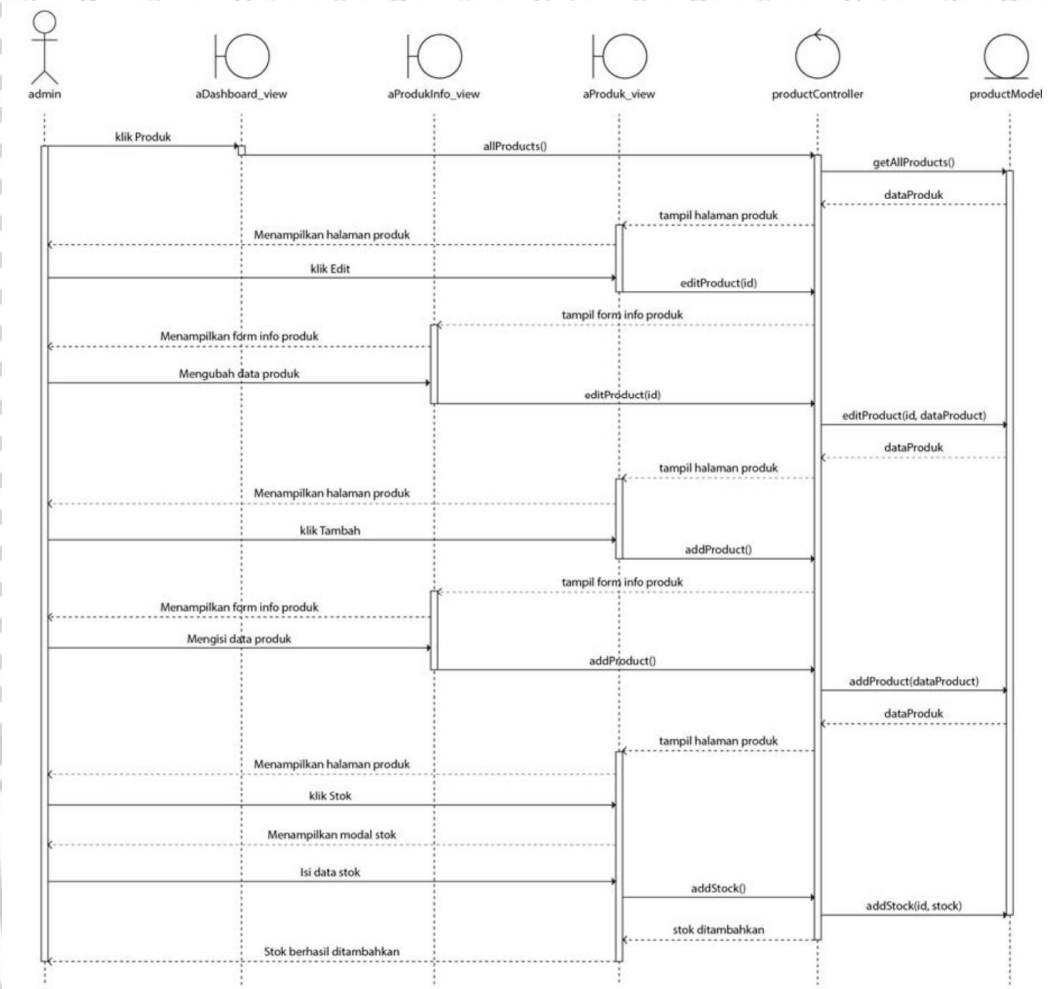
Gambar 5.12 *Sequence Diagram* Melihat *Dashboard* Admin

5.2.12 *Sequence Diagram* Mengelola Produk

Pada Gambar 5.13 merupakan *sequence diagram* untuk mengelola produk. *Sequence diagram* ini dimulai Ketika admin melakukan klik pada sub menu produk yang terdapat pada menu di bagian samping *dashboard*. ProductController memanggil fungsi get untuk menampilkan informasi terkait data apa saja yang dimiliki oleh suatu produk dan menampilkannya pada *view* produk. Di menu produk ini, admin dapat melakukan penambahan data produk, mengubah informasi terkait produk seperti harga, deskripsi, gambar dan juga menghapus produk. Untuk mengubah informasi produk, aktor melakukan klik pada tombol edit. Saat tombol edit di klik, akan mengaktifkan fungsi dari editProduct yang mana akan menampilkan info produk yang ingin dirubah.

Untuk menambah produk, admin dapat men-klik pada menu tambah produk. Hal ini akan menjalankan fungsi *addProduct* yang akan menampilkan halaman penambahan produk yang berisi informasi produk. Admin dapat mengisi data – data yang sesuai dan melakukan penyimpan data produk tersebut.

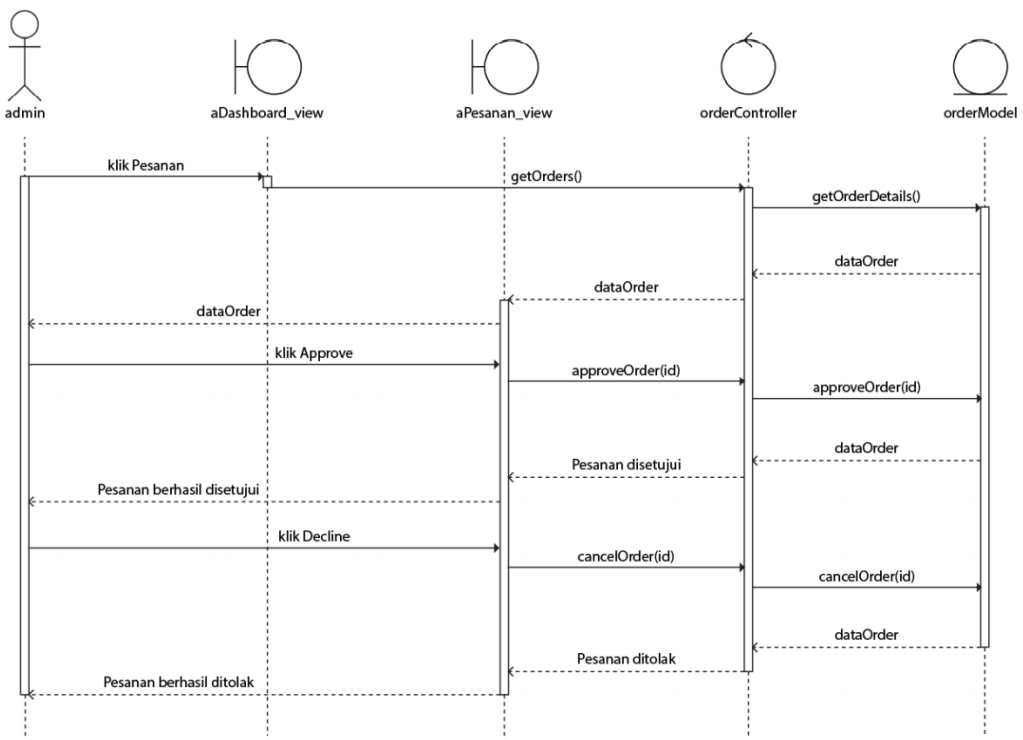
Admin juga dapat melakukan pengelolaan stok pada suatu produk. Dengan melakukan klik pada tombol stok di informasi produk. Yang mana akan membuat stok produk sesuai dengan yang diinginkan oleh admin.



Gambar 5.13 *Sequence Diagram* Mengelola Produk

5.2.13 *Sequence Diagram* Mengelola Pesanan

Gambar 5.14 merupakan gambar dari untuk mengelola pesanan. Sequence ini dimulai Ketika admin melakukan klik pada menu pesanan yang terdapat di halaman *dashboard*. *orderController* memanggil fungsi get untuk melakukan pengambilan seluruh data terkait pemesanan dan menampilkannya pada *view* pesanan. Pada halaman ini, admin dapat mengelola pesanan seperti menerima pesanan dan menolak pesanan.

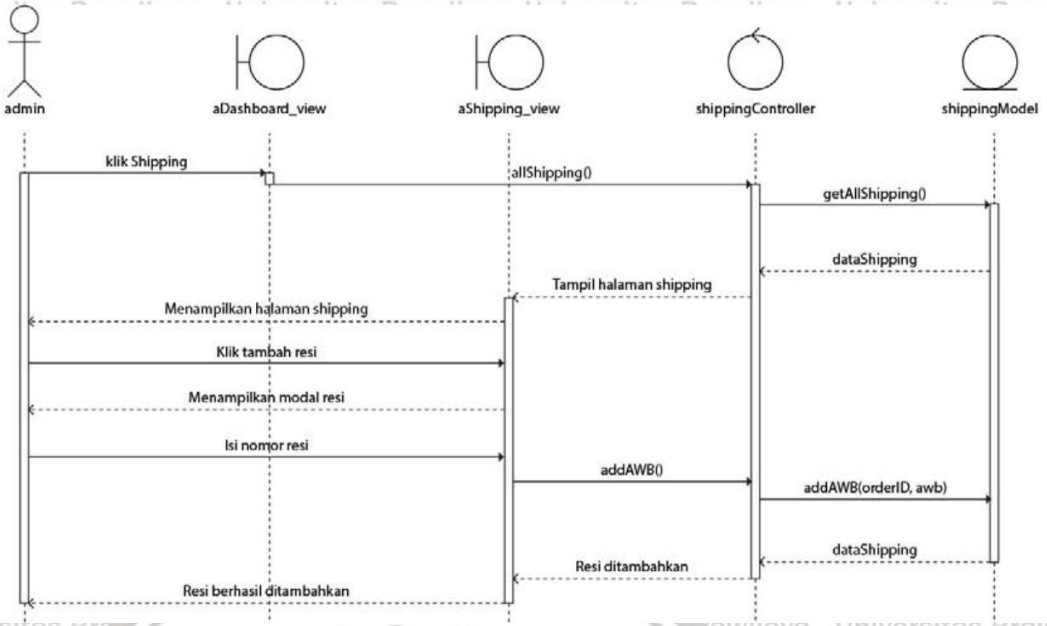


Gambar 5.14 *Sequence Diagram* Mengelola Pesanan

5.2.14 *Sequence Diagram* Mengelola Pengiriman

Gambar 5.15 merupakan *sequence diagram* untuk mengelola pengiriman pesanan. *Sequence Diagram* ini dimulai Ketika admin memilih sub-menu pengiriman yang terdapat dalam menu pesanan. *Shippingcontroller* akan memanggil fungsi *getAllShipping* pada *shippingmodel* untuk menampilkan seluruh data yang terkait dengan pengiriman.

Admin juga melakukan penambahan resi atau awb (*Air Way Bill)* pada transaksi yang berjalan dengan melakukan klik pada pesanan yang di inginkan, lalu menambahkan input nomor resi yang sesuai dengan pilihan kurir pembeli.



Gambar 5.15 *Sequence Diagram* Mengelola Pengiriman

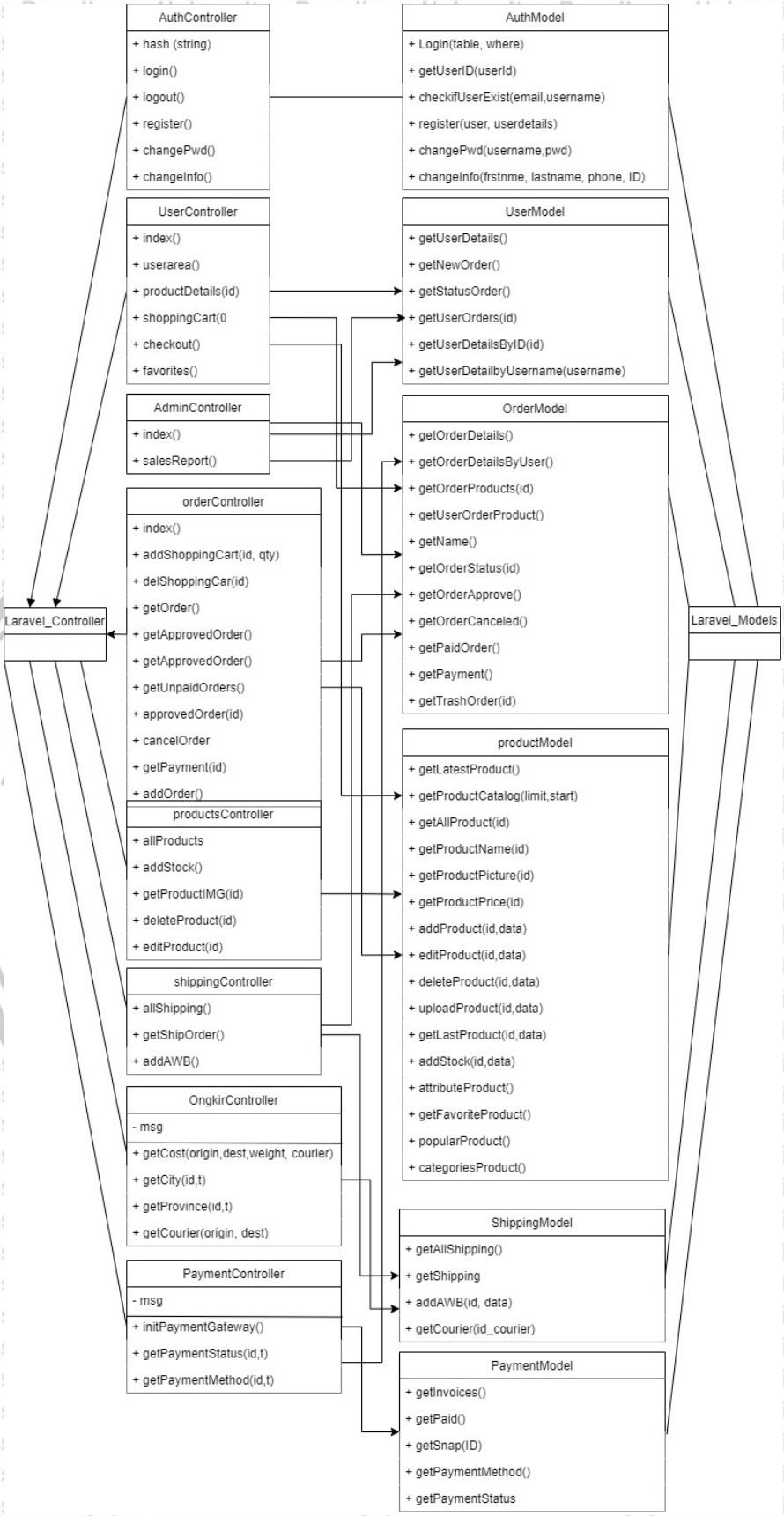
5.3 *Class* Diagram

Class Diagram adalah gambaran dari struktur kelas yang terdapat dalam sistem, bagaimana susunannya, dan hubungan apa saja yang terbentuk di antara kelas tersebut. Pada Gambar 5.10 adalah *class diagram* dari Website *E-commerce Chesspedia, userController, productsController, adminController orderController, ongkirController, shippingController, dan PaymentController. 8 class* ini memiliki relasi inheritance dengan *class Laravel_Controller.*

AuthController adalah *class* yang berfungsi untuk menangani proses *login*, *logout* dan *register*. *Class* ini mengakses *class* authModel untuk mengambil dan menyimpan data. *AdminController* adalah *class* yang berfungsi untuk menampilkan panel admin lengkap dengan data terkini mengenai penjualan, produk, dan pesanan yang diambil dari *class userModel, ordersModel, productsModel, shippingModel, PaymentModel* untuk mengambil dan menyimpan data. *ProductsController* merupakan *class* yang digunakan untuk mengelola data data produk seperti menambah produk, menampilkan produk, maupun menghapus data dari suatu produk. *Controller* ini juga berfungsi untuk menambahkan *attribute* dan *categories* pada suatu produk. *Class* ini mengakses *class producsModel* untuk menyimpan dan mengambil suatu data. *OrderController* adalah *class* yang berfungsi untuk menangani pesanan yang didapatkan dari pelanggan. *Class* ini mengakses *class ordersModel* dan juga productsModel untuk mengambil dan menyimpan data. *PaymentController* merupakan *class* yang digunakan untuk memproses mekanisme pembayaran pada *E-commerce Chesspedia. Class* ini terhubung dengan *paymentModel* dan juga *ordersModel* sebagai jembatan antara *website E-commerce Chesspedia* dan *payment gateway* midtrans. *OngkirController* adalah *class* yang berfungsi untuk melakukan penghitungan ongkos kirim. *Class* ini mengakses *class shippingModel* untuk

mengambil dan menyimpan data. Dan terakhir, shippingController berfungsi untuk menangani proses pengiriman. *Class* ini mengakses *class ordersModel* dan shippingModel untuk mengambil dan menyimpan data.

Selain *class* controller, terdapat juga 6 *class* model yang terdiri dari *authModel, userModel, ordersModel, productsModel, shippingModel,* dan *PaymentModel.* Ke enam *class* ini memiliki relasi *inheritance* dengan *class Laravel_Model.*



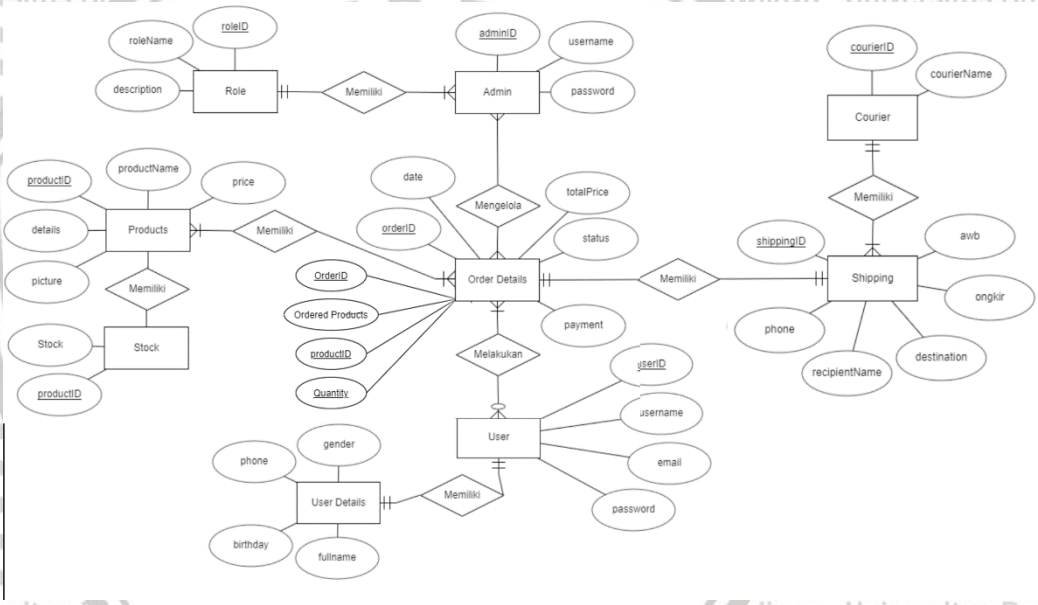
Gambar 5.16 *Class* Diagram Website *E-Commerce* Chesspedia

5.4 Perancangan Basis Data

5.4.1 Entity Relationship Diagram

Gambar 5.11 merupakan Entity Relationship Diagram (ERD) dari Website *E-Commerce* Chesspedia. Terdapat 9 entitas yang masing – masing memiliki attribute yakni pelanggan, info pelanggan, admin, *role*, Transaksi, Produk, Info Pengiriman, kurir, dan pembayaran.

Entitas pelanggan memiliki relasi dengan info pelanggan yang memiliki atribut mengenai informasi dasar pelanggan seperti nama lengkap, nomor telepon, dan sejenisnya. Entitas admin memiliki relasi dengan entitas *role*, karena setiap admin yang ada memiliki *role* atau peran yang berbeda - beda. Entitas transaksi memiliki relasi dengan 4 entitas lain yaitu entitas admin, pelanggan, produk, dan info pengiriman. Entitas admin yang berperan sebagai pengelola transaksi, entitas pelanggan yang berperan sebagai pelaku transaksi, entitas produk, dan entitas info pengiriman yang memiliki relasi dengan entitas kurir.



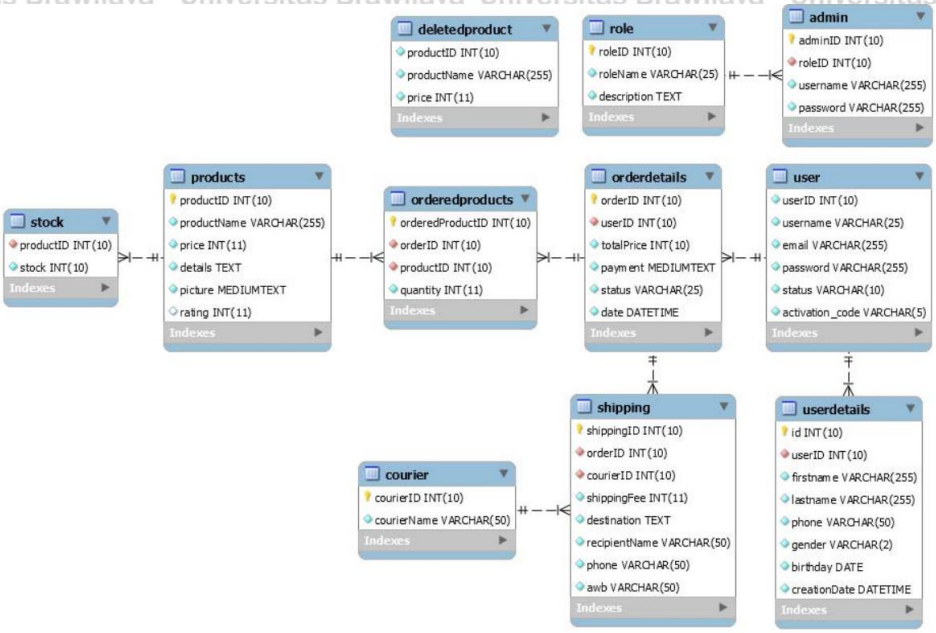
Gambar 5.17 *Entity Relationship Diagram Website E-Commerce* Chesspedia

Sumber : (Penulis, 2021)

5.4.2 Physical Data Model

Website E-Commerce Chesspedia menggunakan 11 tabel dalam database untuk menyimpan data. Tabel admin, digunakan untuk menyimpan data terkait kredensial *login* sebagai admin. Tabel ini memiliki relasi dengan tabel *role* yang menyimpan *role* admin dan deskripsi dari *role* tersebut. Tabel userdetails menyimpan data pelanggan seperti nama dan nomor telepon. Tabel ini memiliki relasi dengan tabel user yang berisi data terkait kredensial *login* pelanggan terdaftar. Tabel orderdetails, digunakan untuk menyimpan data pesanan pelanggan. Tabel ini memiliki relasi dengan tabel shipping yang berisi data pengiriman yang juga memiliki relasi dengan tabel courier yang berisi data

mengenai kurir ekspedisi. Selain itu, tabel orderdetails juga memiliki relasi dengan tabel orderedproducts yang berisi produk apa saja yang dibeli oleh pelanggan. Tabel orderedproducts memilki relasi dengan tabel product yang berisi informasi mengenai produk yang memiliki relasi dengan tabel stock yang berisi informasi stok.

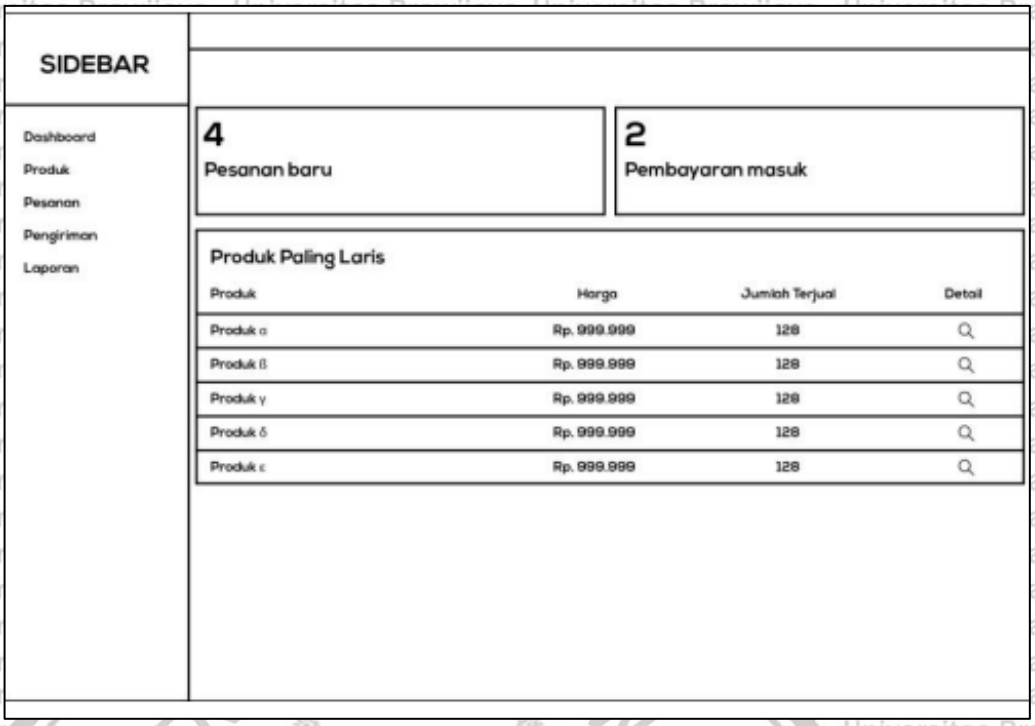


Gambar 5.18 Physical Data Model Website *E-commerce* Chesspedia

5.5 Perancangan Antarmuka Sistem

5.5.1 Tampilan Halaman *Dashboard* Admin

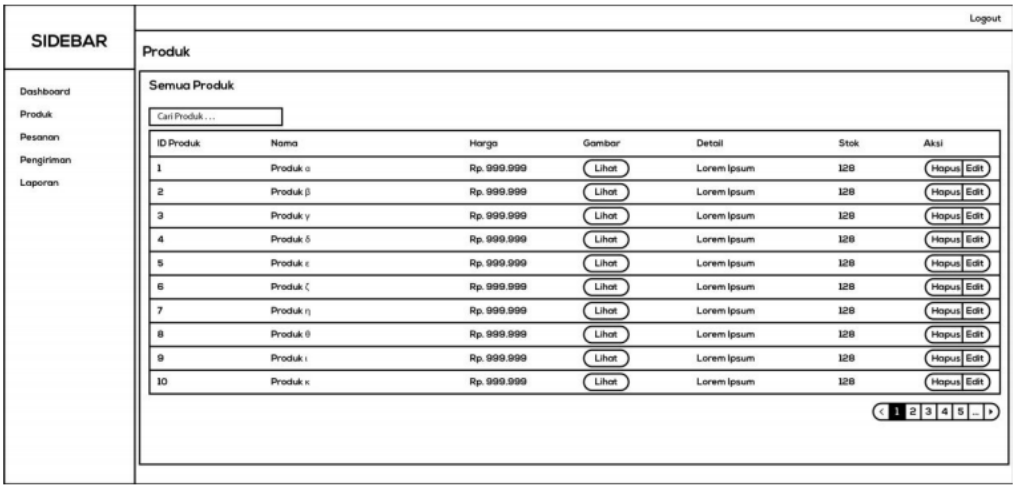
Dashboard pada gambar 5.19 merupakan halaman yang pertama kali muncul saat admin mengakses halaman admin. Pada halaman ini admin dapat mengecek informasi seperti pesanan baru, dan produk apa yang laris. Selain itu terdapat sidebar yang berisi beberapa menu yakni produk, pesanan, laporan dan pengiriman.



Gambar 5.19 *Dashboard* Halaman Admin

5.5.2 Tampilan Halaman Pengaturan Produk

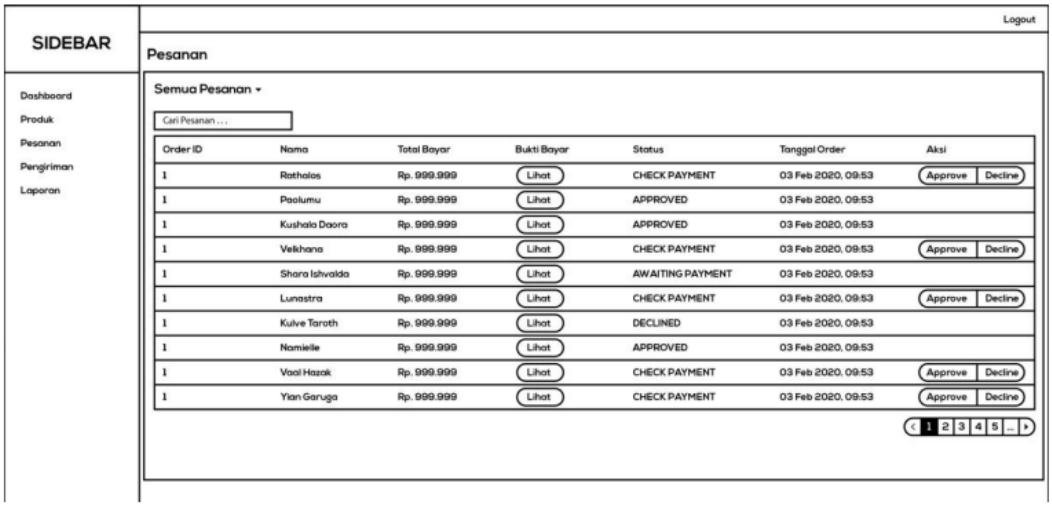
Gambar 5.20 merupakan tampilan dari halaman pengaturan produk. Pada halaman ini terdapat 3 menu pada side bar, yakni produk, atrribut dan categori. Produk merupakan daftar dari seluruh produk yang ada. Kategori adalah pengelompokan produk berdasarkan jenisnya seperti kategori papan catur, bidak catur dan jam catur. Sedangkan attribute merupakan pengelompokan spesifik pada suatu produk seperti ukuran produk maupun warna produk



Gambar 5.20 Halaman Pengaturan Produk

5.5.3 Tampilan Halaman Pesanan

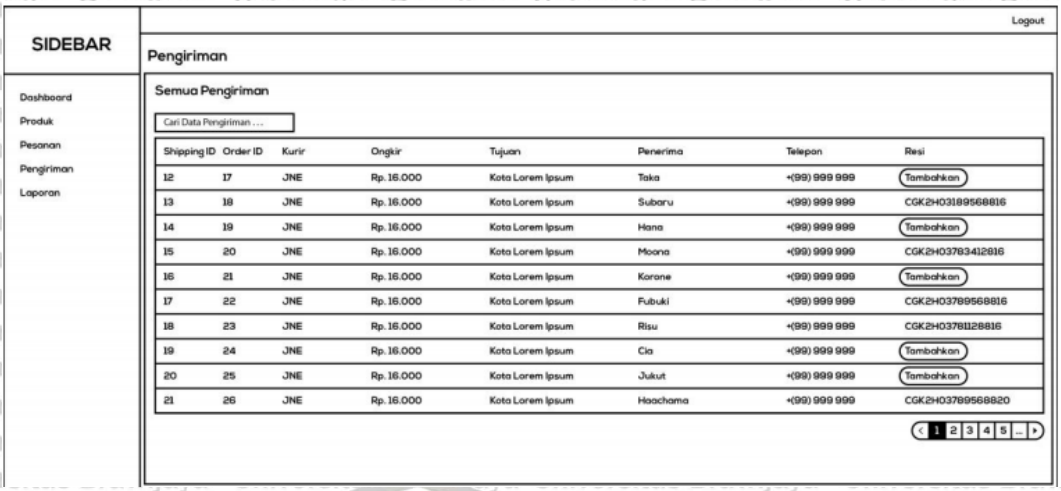
Halaman pesanan pada gambar 5.21 berisi data terkait daftar seluruh pesanan yang masuk di *website ecommerce chesspedia.* Admin dapat melihat detail dari pesanan melalui tombol yang tersedia, juga status pembayaran dari suatu pesanan. Admin dapat menyetujui pesanan yang masuk maupun menolaknya.



Gambar 5.21 Tampilan Halaman Pesanan

5.5.4 Tampilan Halaman Pengiriman

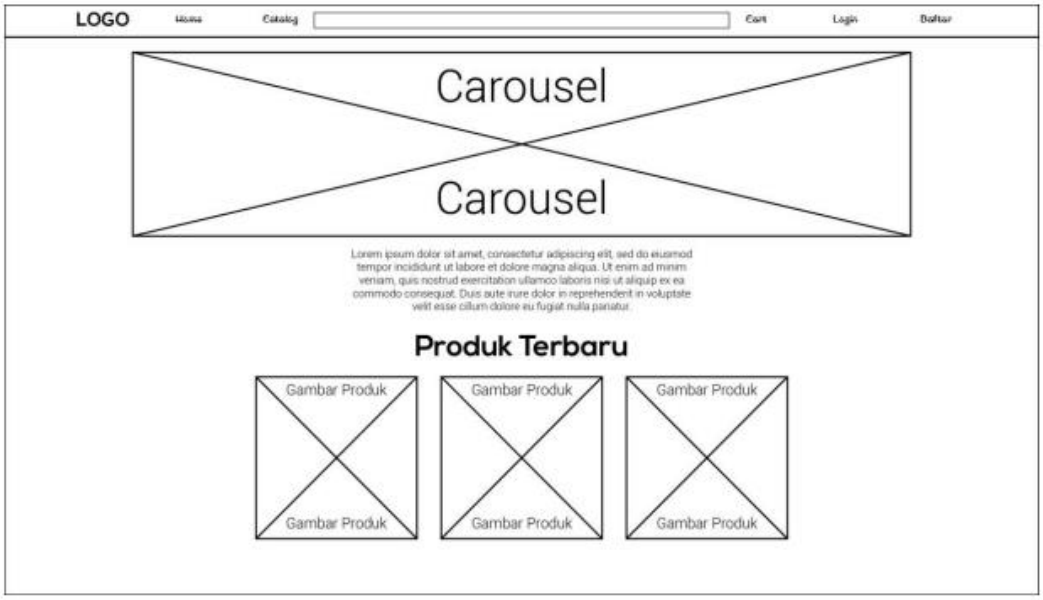
Pada gambar 5.22 merupakan gambar dari perancangan halaman pengiriman yang mana berisi terkait pesanan yang sudah siap dan siap dikirim. Admin dapat melihat detail pesanan dengan melakukan klik pada tombol lihat order, jika resi sudah didapatkan, dapat dimasukan dengan klik tombol tambah resi.



Gambar 5.22 Tampilan Halaman Pengiriman

5.5.5 Tampilan Halaman Awal

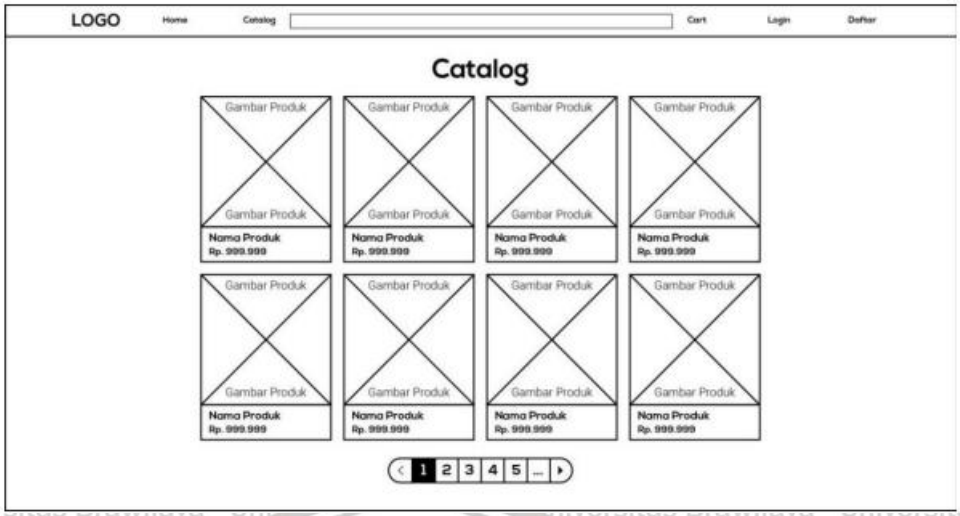
Gambar 5.23 merupakan tampilan halaman awal dari *website* ecommerce chesspedia. Halaman ini adalah halaman yang akan pertama kali dilihat oleh pengguna Ketika mengakses *website*. Pada halaman ini terdapat slider yang berisi promo yang tersedia, keunggulan toko maupun slogan dari toko. Selain itu produk terlaris akan ditampilkan pada halaman ini.



Gambar 5.23 Tampilan Halaman Awal

5.5.6 Tampilan Halaman Seluruh Daftar Produk

Gambar 5.24 merupakan tampilan dari seluruh daftar produk. Pada halaman ini, seluruh produk yang dijual akan ditampilkan.



Gambar 5.24 Tampilan Seluruh Produk

5.5.7 Tampilan Halaman Detail Produk

Gambar 5.25 merupakan tampilan dari halaman detail produk. Halaman ini menampilkan seluruh informasi terkait produk, seperti foto produk, deskripsi produk, stok produk yang ingin dibeli hingga harga produk.

Gambar 5.25 Tampilan Halaman Detail Produk

5.5.8 Tampilan Halaman Keranjang Belanja

Pada gambar 5.26 merupakan tampilan dari halaman keranjang belanja. Halaman ini menampilkan seluruh produk yang telah dimasukan oleh user kedalam keranjang belanja. Pengguna dapat melihat total harga dan kuantitas barang apa saja yang telah dibeli.

Gambar 5.26 Tampilan Halaman Keranjang Belanja

BAB 6 IMPLEMENTASI

6.1 Spesifikasi Lingkungan Implementasi

Spesifikasi lingkungan implementasi yang digunakan untuk pengembangan *Website E-Commerce* Chesspedia dibagi menjadi 2, yaitu spesifikasi perangkat keras dan spesifikasi perangkat lunak.

6.1.1 Spesifikasi Perangkat keras

Spesifikasi perangkat keras yang digunakan dalam pengembangan Website *E-commerce* Chesspedia ditunjukan pada tabel 6.1.

Tabel 6.1 Spesifikasi Perangkat Keras

Processor	Intel Core i5 10210U
Memory (RAM)	8128MB DDR3L
Graphics Card	Nvidia MX250

Sumber : (Penulis, 2021)

6.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam pengembangan Website *E-Commerce* Chesspedia ditunjukan pada Tabel 6.2

Sistem Operasi	Windows 10 *Home* 1909 64-bit
Text Editor	Visual Studio Code 1.52.1
Web Server	XAMPP Version 7.4.1
Bahasa Pemrograman	PHP dan Javascript

Sumber: (Penulis, 2021)

6.2 Implementasi Basis Data

Website *E-commerce* Chesspedia menggunakan MySQL sebagai DBMS (Database Management System) untuk implementasi pada basis data. Pada Gambar 6.2 merupakan daftar dari table DBMS ChesspediaDB yang digunakan oleh Website *E-Commerce* Chesspedia.

Tabel	Tindakan	Baris	Jenis	Penyortiran	Ukuran	Beban
attributes	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	1	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
attribute_options	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	3	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
categories	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	6	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
failed_jobs	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
favorites	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	3	InnoDB	utf8mb4_unicode_ci	48.0 KB	-
jobs	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
migrations	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	32	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
model_has_permissions	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
model_has_roles	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	2	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
orders	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	14	InnoDB	utf8mb4_unicode_ci	144.0 KB	-
order_items	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	15	InnoDB	utf8mb4_unicode_ci	64.0 KB	-
password_resets	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
payments	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	5	InnoDB	utf8mb4_unicode_ci	128.0 KB	-
permissions	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	32	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
products	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	5	InnoDB	utf8mb4_unicode_ci	64.0 KB	-
product_attribute_values	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	0	InnoDB	utf8mb4_unicode_ci	64.0 KB	-
product_categories	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	3	InnoDB	utf8mb4_unicode_ci	48.0 KB	-
product_images	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	8	InnoDB	utf8mb4_unicode_ci	48.0 KB	-
product_inventories	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	5	InnoDB	utf8mb4_unicode_ci	48.0 KB	-
roles	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	2	InnoDB	utf8mb4_unicode_ci	16.0 KB	-
role_has_permissions	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	40	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
shipments	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	14	InnoDB	utf8mb4_unicode_ci	80.0 KB	-
slides	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	5	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
users	Jelajahi Struktur Cari Tambahkan Kosongkan Hapus	6	InnoDB	utf8mb4_unicode_ci	32.0 KB	-
24 tabel	Jumlah	201	InnoDB	utf8mb4_general_ci	1.1 MB	0 B

Gambar 6.1 Daftar Tabel Pada DBMS

Tabel *products* adalah tabel yang berisi informasi terkait data data suatu produk seperti *attribute, categories,* harga, judul, deskripsi produk, foto produk, status, dan informasi dimensi produk (berat, panjang, lebar, tinggi).

Tabel *attributes,* adalah table yang berisi informasi terkait varian yang dimiliki oleh suatu produk. Tabel attribute_options, adalah table yang berisi informasi terkait opsi apa saja yang bsa di masukan ke dalam varian. Misalnya ukuran baju, memiliki varian ukuran S. M, L, dan XL. Data tersebut dimasukan ke dalam table attribute_options

Tabel Categories, berisikan informasi categori produk seperti pada produk bidak catur kayu model bandung memiliki kategori bidak catur. Tabel Product berisi informasi tentang produk yang akan ditampilkan di *website* chesspedia. Terhubung juga dengan 4 tabel lainnya yakni product_attribute_value, product_categories, product_images, product_inventories.

Tabel order, berisikan informasi terkait pesanan yang dilakukan oleh user yang telah melakukan proses *checkout* pesanan dan belum melakukan ke proses pembayaran. Berisikan data terkait nomor invoice, status pesanan, data user, alamat user, jenis jasa kirim yang digunakan dan total pembayaran pesanan. Tabel orders_item berisikan data terkait detail produk yang dipesan, mulai dari harga, pajak hingga biaya pengiriman.

Tabel payments, berisikan informasi terkait pembayaran yang dilakukan oleh user menggunakan *payment gateway* midtrans. Yang mana tabel ini menangkat data dari api *payment gateway* berupa json, dan mengkonversinya kedalam database. Seperti data token, order id, method, payment type yang datanya di ambil dari data json yang diberikan oleh API.

Tabel shipment, berisikan informasi terkait status pengiriman pesanan dari user mulai dari pesanan yang sudah terkonfirmasi pembayarannya maupun yang belum terkonfirmasi pembayarannya.

6.3 Implementasi Kode Program

Pada sub bab implementasi kode program berisi mengenai implementasi kode program yang digunakan untuk mengimplementasikan sistem pada penelitian ini. Implementasi kode program *website e-commerce* Chesspedia. Kode program yang akan dijelaskan pada sub bab kali ini meliputi kode program pembayaran, kode program integrasi antara *API* midtrans dengan Website *E-commerce* Chesspedia.

6.3.1 Integrasi *API* Midtrans

Tabel 6.2 Kode Program *PaymentController.php*

Controller/controllerPayment.php

```
1           $payment = Payment::create($paymentParams);
2
3           if ($paymentStatus && $payment) {
4               \DB::transaction(
5                   function () use ($order, $payment) {
6                       if (in_array($payment-
>status, [Payment::SUCCESS, Payment::SETTLEMENT])) {
7                           $order->payment_status = Order::PAID;
8                           $order->status = Order::CONFIRMED;
9                           $order->save();
10                      }
11                  }
12              );
13          }
14
15          $message = 'Payment status is : '. $paymentStatus;
16
17          $response = [
18              'code' => 200,
19              'message' => $message,
20          ];
21
22          return response($response, 200);
```

Controller/controllerPayment.php

```
    }

    /**
     * Show completed payment status
     *
     * @param Request $request payment data
     *
     * @return void
     */
    public function completed(Request $request)
    {
        $code = $request->query('order_id');
        $order = Order::where('code', $code)->firstOrFail();

        if ($order->payment_status == Order::UNPAID) {
            return redirect('payments/failed?order_id=' . $code);
        }

        \Session::flash('success', "Thank you for completing the payment process!");

        return redirect('orders/received/'. $order->id);
    }

    /**
     * Show unfinish payment page
     *
     * @param Request $request payment data
     *
     * @return void
     */
    public function unfinish(Request $request)
    {
        $code = $request->query('order_id');
        $order = Order::where('code', $code)->firstOrFail();
```

Controller/controllerPayment.php

```
57
58          \Session::flash('error', "Sorry, we couldn't process your payment.");
59
60          return redirect('orders/received/'. $order->id);
61      }
62
63      /**
64       * Show failed payment page
65       *
66       * @param Request $request payment data
67       *
68       * @return void
69       */
70      public function failed(Request $request)
71      {
72          $code = $request->query('order_id');
73          $order = Order::where('code', $code)->firstOrFail();
74
75          \Session::flash('error', "Sorry, we couldn't process your payment.");
76
77          return redirect('orders/received/'. $order->id);
78      }
79  }
```

Tabel 6.3 Kode Program *Library* Midtrans.php

Vendor/midtrans/snap.php

```
1   public static function createTransaction($params)
2   {
3       $payloads = array(
4       'credit_card' => array(
5           // 'enabled_payments' => array('credit_card'),
6           'secure' => Config::$is3ds
7       )
8       );
9
10      if (array_key_exists('item_details', $params)) {
11          $gross_amount = 0;
```

```
12                foreach ($params['item_details'] as $item) {
                      $gross_amount += $item['quantity'] *
13 $item['price'];
14                }
                  $params['transaction_details']['gross_amount'] =
15 $gross_amount;
16            }
17
18            if (Config::$isSanitized) {
19                Sanitizer::jsonRequest($params);
20            }
21
              $params = array_replace_recursive($payloads,
22 $params);
23
24            $result = SnapApiRequestor::post(
25                Config::getSnapBaseUrl() . '/transactions',
26                Config::$serverKey,
27                $params
28            );
29
30            return $result;
31        }
32 }
```

Sumber : (Penulis, 2021)

Integrasi *API Payment Gateway* Midtrans pada *website e-commerce* Chesspedia (*WEC*) dilakukan dengan menginstal *library* pada Laravel melalui *composer* yang berisi fungsi dari *API Payment Gateway Midtrans.* Maksud dari menginstall ini ialah untuk mempermudah melakukan *request API* dengan hanya memanggil nama berserta fungsi dan parameter yang dibutuhkan pada *PaymentController*.

Penjelasan dari kode program integrasi *API Payment Gateway* ialah sebagai berikut :

1. Pada baris 1 – 10 *PaymentController* merupakan awalan dari fungsi *notification* dari koneksi antara *backend WEC* dengan *API* Midtrans. Penghubung antar kedua sistem ini menggunakan *Server API KEY* yang diberikan pada *website* Midtrans. Jika MIDTRANS *SERVER KEY VALID,* maka sistem akan memberikan akses kepada WEC untuk mengakses *Payment Gateway* Midtrans
2. Pada baris 11 – 92 *PaymentController* merupakan fungsi untuk melakukan akses pada *Snap* Midtrans yang berada pada */Vendor/midtrans*
3. Pada baris 1 – 17 SnapMidtrans merupakan fungsi untuk melakukan proses pembayaran menggunakan *payment gateway* midtrans

6.3.2 Integrasi *API* Raja Ongkir

Tabel 6.4 Kode Program *Shipping Controller*

```
Controller/admin/shippingController.php
1       public function update(Request $request, $id)
2       {
3           $request->validate(
4               [
5                   'track_number' => 'required|max:255',
6               ]
7           );
8
9           $shipment = Shipment::findOrFail($id);
10
11          $order = \DB::transaction(
12              function () use ($shipment, $request) {
13                  $shipment->track_number = $request-
>input('track_number');
14                  $shipment->status = Shipment::SHIPPED;
15                  $shipment->shipped_at = now();
16                  $shipment->shipped_by = \Auth::user()->id;
17
18                  if ($shipment->save()) {
19                      $shipment->order-
>status = Order::DELIVERED;
20                      $shipment->order->save();
21                  }
22
23                  return $shipment->order;
24              }
25          );
26
27          if ($order) {
28              $this->_sendEmailOrderShipped($shipment-
>order);
29          }
30
31          \Session::flash('success', 'The shipment has been u
pdated');
32          return redirect('admin/orders/'. $order->id);
33      }
```

Sumber : Penulis (2021)

Tabel 6.5 Potongan Kode Fungsi rajaOngkirRequest

```
Controller/Controller.php
1       protected function rajaOngkirRequest($resource,
$params = [], $method = 'GET')
```

```
Controller/Controller.php
    {
        $client = new \GuzzleHttp\Client();

        $headers = ['key' => $this->rajaOngkirApiKey];
        $requestParams = [
            'headers' => $headers,
        ];

        $url = $this->rajaOngkirBaseUrl . $resource;
        if ($params && $method == 'POST') {
            $requestParams['form_params'] = $params;
        } else if ($params && $method == 'GET') {
            $query = is_array($params) ? '?'.
http_build_query($params) : '';
            $url = $this->rajaOngkirBaseUrl . $resource .
$query;
        }

        $response = $client->request($method, $url,
$requestParams);

        return json_decode($response->getBody(), true);
    }
```

Sumber : Penulis (2021)

Tabel 6.6 Potongan Fungsi *getProvinces* dan *getCity*

```
Controller/Controller.php
    protected function getProvinces()
    {
        $provinceFile = 'provinces.txt';
        $provinceFilePath = $this-
>uploadsFolder. 'files/' . $provinceFile;

        $isExistProvinceJson = \Storage::disk('local')-
>exists($provinceFilePath);

        if (!$isExistProvinceJson) {
            $response = $this->rajaOngkirRequest('province');
```

```
            \Storage::disk('local')-
>put($provinceFilePath, serialize($response['rajaongkir']['re
sults']));
        }

        $province = unserialize(\Storage::get($provinceFilePa
th));

        $provinces = [];
        if (!empty($province)) {
            foreach ($province as $province) {
                $provinces[$province['province_id']] = strtou
pper($province['province']);
            }
        }

        return $provinces;
    }

    /*****************************************************
     * Get cities by province ID
     *
     * @param int $provinceId province id
     *
     * @return array
     *****************************************************/
    protected function getCities($provinceId)
    {
        $cityFile = 'cities_at_'. $provinceId .'.txt';
        $cityFilePath = $this-
>uploadsFolder. 'files/' .$cityFile;

        $isExistCitiesJson = \Storage::disk('local')-
>exists($cityFilePath);

        if (!$isExistCitiesJson) {
            $response = $this-
>rajaOngkirRequest('city', ['province' => $provinceId]);
            \Storage::disk('local')-
>put($cityFilePath, serialize($response['rajaongkir']['result
s']));
        }

        $cityList = unserialize(\Storage::get($cityFilePath))
;

        $cities = [];
        if (!empty($cityList)) {
            foreach ($cityList as $city) {
```

```
49          $cities[$city['city_id']] = strtoupper($city['type'].' '.$city['city_name']);
50          }
51      }
52
53      return $cities;
54  }
```

Sumber : Penulis (2021)

Integrasi *API Raja Ongkir* pada Website Ecommerce Chesspedia (*WEC*) dilakukan dengan menginstal *library* pada Laravel melalui *composer* yang berisi fungsi dari *API Raja Ongkir.* Maksud dari menginstall ini adalah untuk mempermudah melakukan request API dengan hanya memanggil nama berserta fungsi dan parameter yang dibutuhkan pada *ShippingController.*

Penjelasan dari kode program integrasi *API* Raja Ongkir ialah sebagai berikut :

1. Pada baris 1 – 21 *rajaOngkirRequest* merupakan awalan dari fungsi request koneksi antara *backend WEC* dengan API Raja Ongkir. Penghubung antar kedua sistem ini menggunakan *Raja Ongkir API KEY* yang diberikan pada *website* Raja Ongkir. Jika Raja Ongkir *API Key Valid*, maka sistem akan memberikan akses kepada WEC untuk mengakses *API* Raja Ongkir
2. Pada baris 1 – 23 *Controller*.php merupakan fungsi untuk melakukan akses untuk mendapatkan data provinsi pada *API* Raja Ongkir
3. Pada baris 32 – 54 *Controller*.php merupakan fungsi untuk melakukan akses untuk mendapatkan data kota pada *API* Raja Ongkir
4. Pada baris 1 – 33 *shippingController*.php merupakan fungsi untuk melakukan *update* data resi pada suatu pesanan.

6.3.3 Implementasi Fungsi *Register*, *Login* dan *Logout*

Tabel 6.7 Potongan Fungsi *Register* pada *register*Controller.php

```
Controller/auth/registerController.php
1  protected function create(array $data)
2  {
3      return User::create(
4          [
5              'first_name' => $data['first_name'],
6              'last_name' => $data['last_name'],
7              'email' => $data['email'],
8              'password' => Hash::make($data['password']),
9          ]
10     );
11 }
```

```
12
13      /**
14       * Show register form
15       *
16       * @return void
17       */
18      public function showRegistrationForm()
19      {
20          if (property_exists($this, 'registerView')) {
21              return view($this->registerView);
22          }
23          return $this->loadTheme('auth.register');
24      }
25  }
```

Tabel 6.8 Potongan Kode Fungsi *Login* dan *Logout* pada *login*Controller.php

```
loginController.php
1       protected function redirectTo()
2       {
3           if (auth()->user()->role == 'admin') {
4               return '/admin/dashboard';
5           }
6           return '/home';
7       }
8
9       /**
10       * Create a new controller instance.
11       *
12       * @return void
13       */
14      public function __construct()
15      {
16          $this->middleware('guest')->except('logout');
17      }
18
19      /**
20       * Show login form
21       *
22       * @return void
23       */
24      public function showLoginForm()
25      {
26          if (view()->exists('auth.authenticate')) {
27              return view('auth.authenticate');
28          }
```

Impelentasi fungsi *register*, *login* dan *logout website* e-*commerce* Chesspedia (*WEC*) dilakukan dengan menggunakan fitur dari Laravel yang bernama *authy.* Dengan menggunakan fitur *authy* akan mempermudah melakukan fungsi *register*, *login* dan *logout*

Penjelasan dari kode program implementasi *register*, *login* dan *logout* ialah sebagai berikut :

1. Pada baris 1 – 25 *register*Controller merupakan fungsi untuk user dapat melakukan *register* pada sistem WEC. Yang mana setelah user melakukan input data, WEC akan memanggil *authenticate* milik Laravel Authy untuk digunakan sebagai jembatan sistem dalam melakukan pendaftaran data – data user pada *database*.
2. Pada baris 1 – 28 *loginController*.php merupakan fungsi untuk melakukan *login* pada sistem WEC. Sama seperti fungsi *register*, sistem akan Kembali memanggil *authenticate authy* untuk melakukan verifikasi terhadap data yang di input. Apakah data tersebut sudah sesuai dengan database atau tidak.
3. Pada baris 13-17 *login*Controller.php bertugas untuk menghapus *session* pengguna yang telah melakukan *login* ke dalam sistem.

6.3.4 Implementasi Tampilkan Seluruh Produk

Tabel 6.9 Potongan Kode Fungsi Tampilkan Produk

ProductController.php

```
1     public function show($slug)
2     {
3         $product = Product::active()->where('slug', $slug)->first();
4
5         if (!$product) {
6             return redirect('products');
7         }
8
9         if ($product->configurable()) {
10            $this->data['colors'] = ProductAttributeValue::getAttributeOptions($product, 'color')->pluck('text_value', 'text_value');
11            $this->data['sizes'] = ProductAttributeValue::getAttributeOptions($product, 'size')->pluck('text_value', 'text_value');
12        }
13
14        $this->data['product'] = $product;
15
16        return $this->loadTheme('products.show', $this->data);
17    }
18
19    /**
20     * Quick view product.
21     *
```

```
22      * @param string $slug product slug
23      *
24      * @return \Illuminate\Http\Response
25      */
26     public function quickView($slug)
27     {
28         $product = Product::active()->where('slug', $slug)->firstOrFail();
29         if ($product->configurable()) {
30             $this->data['colors'] = ProductAttributeValue::getAttributeOptions($product, 'color')->pluck('text_value', 'text_value');
31             $this->data['sizes'] = ProductAttributeValue::getAttributeOptions($product, 'size')->pluck('text_value', 'text_value');
32         }
33
34         $this->data['product'] = $product;
35
36         return $this->loadTheme('products.quick_view', $this->data);
37     }
38 }
```

Sumber : Penulis (2021)

Kode program pada Tabel 6.9 merupakan implementasi kode program dari fungsi *show*() yang mana untuk menampilkan daftar produk pada halaman katalog. Halaman ini menggunakan fungsi *pagination* yang membatasi jumlah produk yangmuncul pada tiap halaman

Penjelasan dari implementasi kode program tampilan produk ialah sebagai berikut :

1. Baris 1 – 17 merupakan implementasi fungsi show() untuk menampilkan sejumlah produk pada halaman tampilan produk. Yang mana tiap halaman akan dibatasi jumlah produknya menggunakan pagination.
2. Baris 26 – 31 merupakan implementasi fungsi *quickView*() untuk menampilkan sekilas detail dari masing – masing produk.

6.3.5 Implementasi Fungsi Keranjang Belanja

Tabel 6.10 Implementasi Kode Program Fungsi Keranjang Belanja

```
CartController.php
1  public function store(Request $request)
2      {
3          $params = $request->except('_token');
4
5          $product = Product::findOrFail($params['product_id']);
6          $slug = $product->slug;
```

```
7
8           $attributes = [];
9           if ($product->configurable()) {
10              $product = Product::from('products as p')
11                  ->whereRaw(
12                      "p.parent_id = :parent_product_id
13                  and (select pav.text_value
14                          from product_attribute_values pav
15                          join attributes a on a.id = pav.att
    ribute_id
16                          where a.code = :size_code
17                          and pav.product_id = p.id
18                          limit 1
19                      ) = :size_value
20                  and (select pav.text_value
21                          from product_attribute_values pav
22                          join attributes a on a.id = pav.att
    ribute_id
23                          where a.code = :color_code
24                          and pav.product_id = p.id
25                          limit 1
26                      ) = :color_value
27                      ",
28                      [
29                          'parent_product_id' => $product-
    >id,
30                          'size_code' => 'size',
31                          'size_value' => $params['size'],
32                          'color_code' => 'color',
33                          'color_value' => $params['color'],
34                      ]
35                  )->firstOrFail();
36
37              $attributes['size'] = $params['size'];
38              $attributes['color'] = $params['color'];
39          }
40
41          $itemQuantity =  $this-
    >_getItemQuantity(md5($product->id)) + $params['qty'];
```

```
42          $this->_checkProductInventory($product, $itemQuantity);
43
44          $item = [
45              'id' => md5($product->id),
46              'name' => $product->name,
47              'price' => $product->price,
48              'quantity' => $params['qty'],
49              'attributes' => $attributes,
50              'associatedModel' => $product,
51          ];
52
53          \Cart::add($item);
54
55          \Session::flash('success', 'Product '. $item['name'] .' has been added to cart');
56          return redirect('/product/'. $slug);
57      }
```

Sumber : Penulis (2021)

Kode program pada Tabel 6.10 adalah kode program yang berfungsi menambahkan produk ke dalam *shopping cart*. Pengguna dapat menambahkan produk kedalam *shopping cart* pada halaman detail produk maupun halaman seluruh produk. Produk yang ditambahkan ke *shopping cart* terlebih dahulu disimpan pada *session* dan ditampilkan pada halaman *shopping cart*.

6.3.6 Implementasi Fungsi *Checkout*

Tabel 6.11 Implementasi Kode Fungsi *Checkout*

```
1   public function checkout()
2       {
3           if (\Cart::isEmpty()) {
4               return redirect('carts');
5           }
6
7           \Cart::removeConditionsByType('shipping');
8           $this->_updateTax();
9
10          $items = \Cart::getContent();
11          $this->data['items'] = $items;
12          $this->data['totalWeight'] = $this->_getTotalWeight() / 1000;
13
14          $this->data['provinces'] = $this->getProvinces();
15          $this->data['cities'] = isset(\Auth::user()->province_id) ? $this->getCities(\Auth::user()->province_id) : [];
```

```
16          $this->data['user'] = \Auth::user();
17
18
19
20          return $this->loadTheme('orders.checkout', $this->data);
21      }
22
23      /**
24       * Get cities by province ID
25       *
26       * @param Request $request province id
27       *
28       * @return json
29       */
30      public function cities(Request $request)
31      {
32          $cities = $this->getCities($request-
>query('province_id'));
33          return response()->json(['cities' => $cities]);
34      }
```

Sumber : Penulis (2021)

Tabel 6.11 merupakan implementasi dari kode program fungsi *checkout*() yang berfungsi untuk menampilkan halaman *checkout* pada WEC. Selain itu, fungsi ini juga menampilkan daftar provinsi yang dipanggil melalui api raja ongkir yang nantinya akan digunakan sebagai parameter untuk menampilkan daftar kota sesuai provinsi. Daftar kota ini juga digunakan sebagai parameter ongkos kirim.

6.3.7 Implementasi Fungsi Order

Tabel 6.12 Implementasi Potongan Kode Fungsi Order

```
Controller/orderController.php
1       private function _saveOrderItems($order)
2       {
3           $cartItems = \Cart::getContent();
4
5           if ($order && $cartItems) {
6               foreach ($cartItems as $item) {
7                   $itemTaxAmount = 0;
8                   $itemTaxPercent = 0;
9                   $itemDiscountAmount = 0;
10                  $itemDiscountPercent = 0;
11                  $itemBaseTotal = $item->quantity *
$item->price;
12                  $itemSubTotal = $itemBaseTotal +
$itemTaxAmount - $itemDiscountAmount;
```

Controller/*orderController*.php

```
13
14                  $product = isset($item->associatedModel-
>parent) ? $item->associatedModel->parent :
$item->associatedModel;
15
16                  $orderItemParams = [
17                      'order_id' => $order->id,
18                      'product_id' => $item->associatedModel->id,
19                      'qty' => $item->quantity,
20                      'base_price' => $item->price,
21                      'base_total' => $itemBaseTotal,
22                      'tax_amount' => $itemTaxAmount,
23                      'tax_percent' => $itemTaxPercent,
24                      'discount_amount' => $itemDiscountAmount,
25                      'discount_percent' => $itemDiscountPercent,
26                      'sub_total' => $itemSubTotal,
27                      'sku' => $item->associatedModel->sku,
28                      'type' => $product->type,
29                      'name' => $item->name,
30                      'weight' => $item->associatedModel->weight,
31                      'attributes' => json_encode($item-
>attributes),
32                  ];
33
34                  $orderItem = OrderItem::create($orderItemParams);
35
36                  if ($orderItem) {
37                      ProductInventory::reduceStock($orderItem-
>product_id, $orderItem->qty);
38                  }
39              }
40          }
41      }
```

Sumber : Penulis (2021)

Kode program pada Tabel 6.12 merupakan kode program yang berfungsi untuk menyimpan pesanan setelah pengguna menyelesaikan *checkout*. Data pesanan meliputi data produk yang dipesan seperti ID produk dan jumlah yang produk dipesan, dan data untuk pengiriman yang berisi tujuan, nama penerima, ongkos kirim, dan nomor telepon penerima. Kedua data ini disimpan secara terpisah. Data produk yang dipesan disimpan menggunakan fungsi *orderedProducts*(), sementara data untuk pengiriman disimpan menggunakan

fungsi *shippingInfo*(). Setelah data pesanan disimpan, pengguna dialihkan ke halaman pembayaran.

6.3.8 Implementasi Fungsi Pembayaran

Tabel 6.13 Implementasi Kode Program Fungsi Pembayaran

Controller/controllerPayment.php

```
1           $payment = Payment::create($paymentParams);
2
3           if ($paymentStatus && $payment) {
4               \DB::transaction(
5                   function () use ($order, $payment) {
6                       if (in_array($payment-
>status, [Payment::SUCCESS, Payment::SETTLEMENT])) {
7                           $order->payment_status = Order::PAID;
8                           $order->status = Order::CONFIRMED;
9                           $order->save();
10                      }
11                  }
12              );
13          }
14
15          $message = 'Payment status is : '. $paymentStatus;
16
17          $response = [
18              'code' => 200,
19              'message' => $message,
20          ];
21
22          return response($response, 200);
23      }
24
25      /**
26       * Show completed payment status
27       *
28       * @param Request $request payment data
29       *
30       * @return void
31       */
32      public function completed(Request $request)
33      {
34          $code = $request->query('order_id');
```

Controller/controllerPayment.php

```
        $order = Order::where('code', $code)->firstOrFail();

        if ($order->payment_status == Order::UNPAID) {
            return redirect('payments/failed?order_id=' . $code);
        }

        \Session::flash('success', "Thank you for completing the payment process!");

        return redirect('orders/received/' . $order->id);
    }

    /**
     * Show unfinish payment page
     *
     * @param Request $request payment data
     *
     * @return void
     */
    public function unfinish(Request $request)
    {
        $code = $request->query('order_id');
        $order = Order::where('code', $code)->firstOrFail();

        \Session::flash('error', "Sorry, we couldn't process your payment.");

        return redirect('orders/received/' . $order->id);
    }

    /**
     * Show failed payment page
     *
     * @param Request $request payment data
     *
     * @return void
     */
    public function failed(Request $request)
```

```
Controller/controllerPayment.php
71      {
72          $code = $request->query('order_id');
73          $order = Order::where('code', $code)->firstOrFail();
74
75          \Session::flash('error', "Sorry, we couldn't process
your payment.");
76
77          return redirect('orders/received/'. $order->id);
78      }
79  }
```

Sumber : Penulis (2021)

Implementasi fungsi pembayaran pada *website e-commerce* Chesspedia (*WEC*) dilakukan dengan menginstal *library* pada Laravel melalui *composer* yang berisi fungsi dari *API Payment Gateway Midtrans.* Maksud dari menginstall ini ialah untuk mempermudah melakukan request API dengan hanya memanggil nama berserta fungsi dan parameter yang dibutuhkan pada PaymentController.

Penjelasan dari kode program integrasi API *Payment Gateway* ialah sebagai berikut :

1. Pada baris 1 – 10 PaymentController merupakan awalan dari fungsi *notification* dari koneksi antara *backend WEC* dengan API Midtrans. Penghubung antar kedua sistem ini menggunakan *Server API KEY* yang diberikan pada *website* Midtrans. Jika *MIDTRANS SERVER KEY VALID,* maka sistem akan memberikan akses kepada WEC untuk mengakses *Payment Gateway* Midtrans
2. Pada baris 11 – 32 *PaymentController* merupakan fungsi untuk melakukan akses pada *Snap* Midtrans yang berada pada */Vendor/midtrans*

6.4 Implementasi Antarmuka

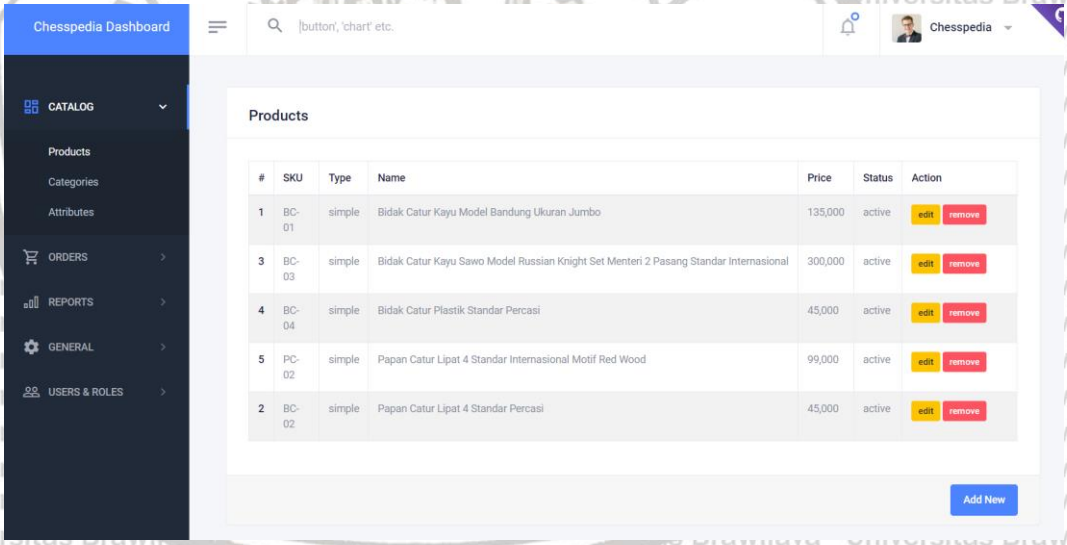
6.4.1 *Dashboard* Admin

Pada Gambar 6.2 merupakan implementasi dari tampilan *dashboard* panel admin. Terdapat beberapa informasi seperti data yang menunjukan informasi seperti pesanan masuk, total pemasukan, hingga total produk terjual.

Gambar 6.2 Tampilan *Dashboard* Panel Admin

6.4.2 Menu Produk

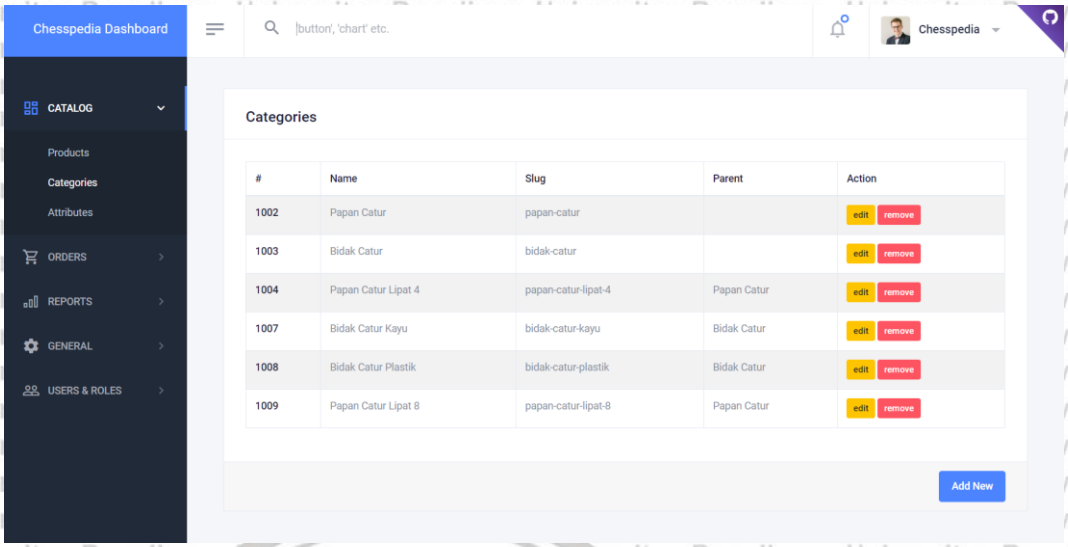
Pada Gambar 6.3 merupakan implementasi dari tampilan panel produk pada admin. Admin dapat melihat seluruh data produk, menambahkan produk, mengubah detail produk, melakukan *update* stok produk, menambahkan gambar hingga menghapus produk.



Gambar 6.3 Menu Produk

6.4.3 Menu Kategori

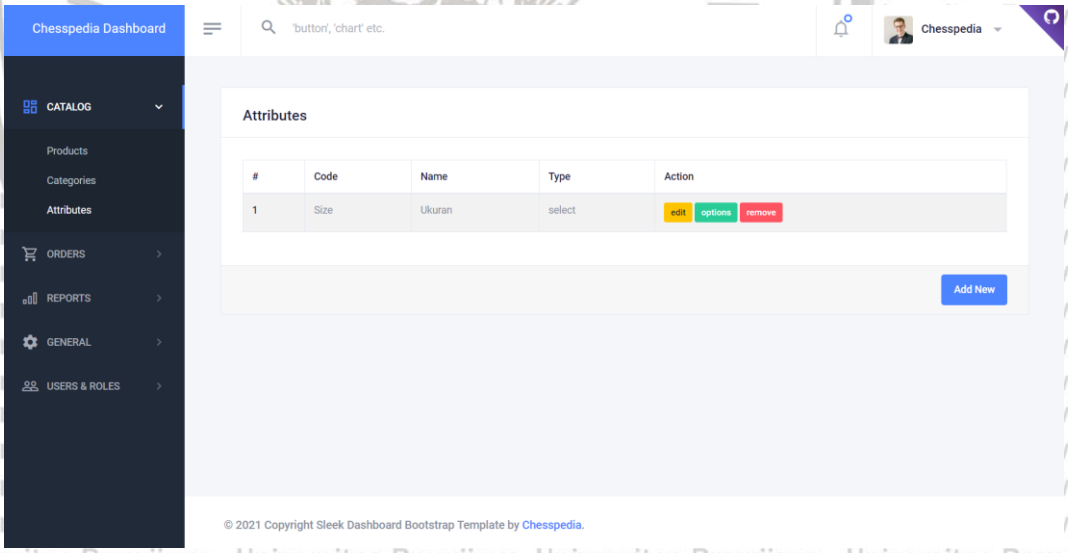
Pada Gambar 6.4 merupakan implementasi dari tampilan panel kategori pada produk. Admin dapat menambahkan kategori produk, mengedit kategori produk, maupun menghapus kategori produk.



Gambar 6.4 Menu Kategori

6.4.4 Menu Attribut

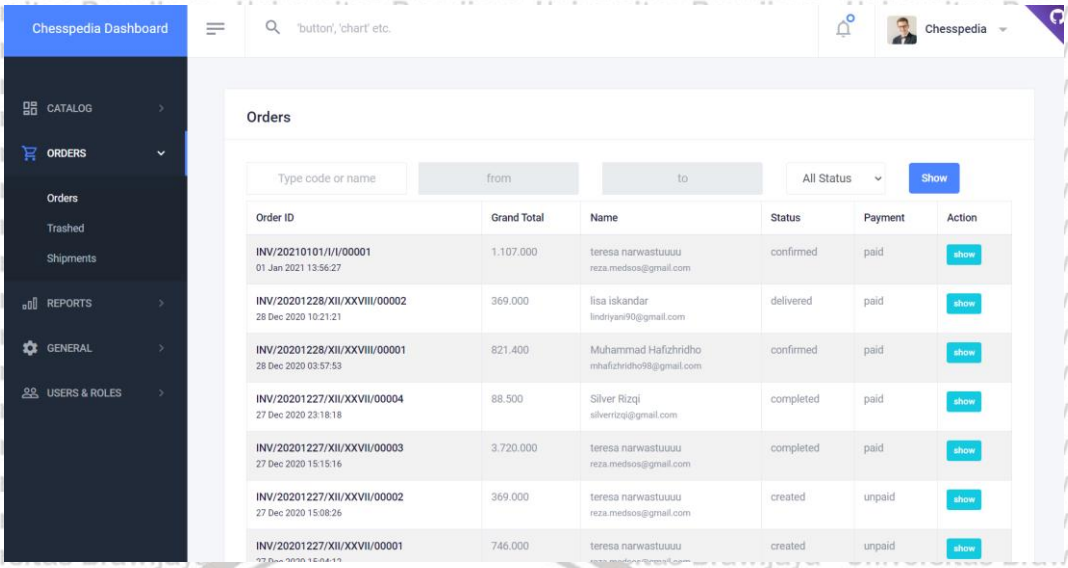
Pada Gambar 6.5 merupakan implementasi dari tampilan panel attribute pada produk. Admin dapat melihat seluruh attribute produk, menambahkan attribute, menghapus attribute, hingga mengedit attribute.



Gambar 6.5 Menu Attribut

6.4.5 Menu Pesanan

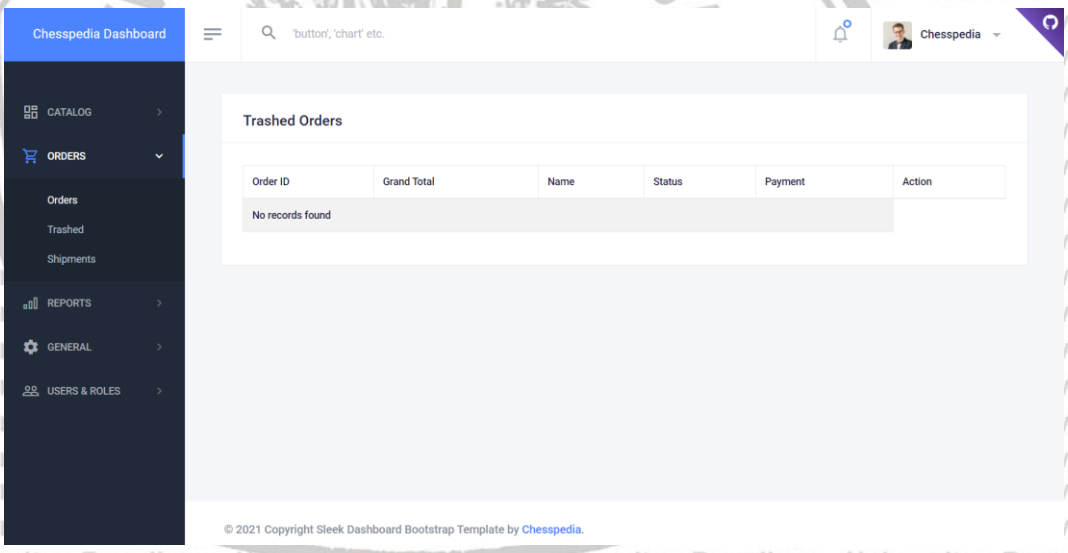
Pada Gambar 6.6 di bawah ini adalah implementasi menu pesanan. Disini admin dapat melihat semua pesanan yang masuk, memilah pesanan berdasarkan statusnya, menolak, atau menghapus dan menyetujui pesanan.



Gambar 6.6 Menu Pesanan

6.4.6 Menu *Trash*

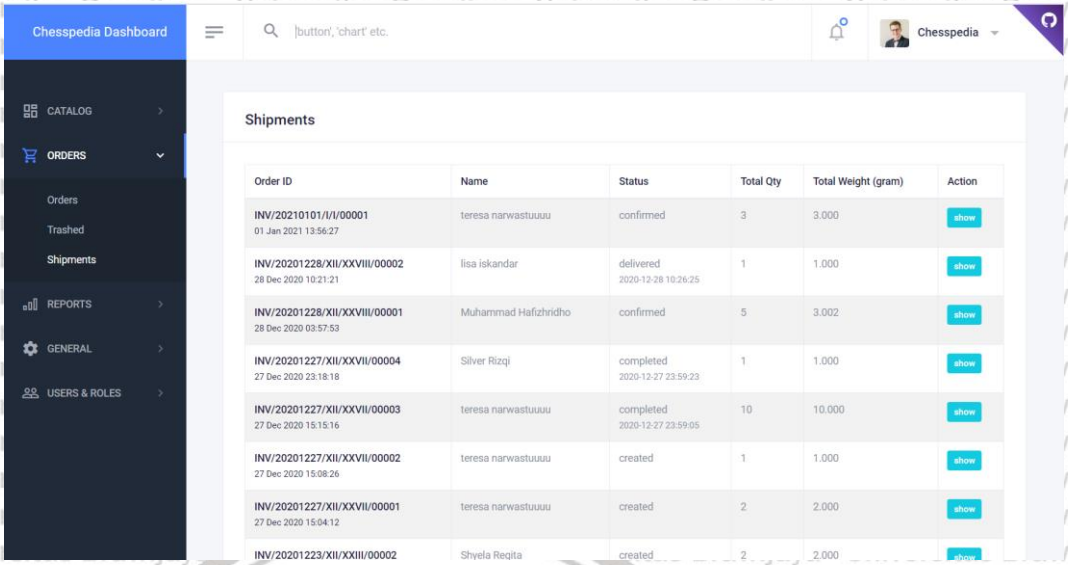
Pada gambar 6.7 dibawah ini merupakan impelementasi dari menu Sampah. Yang berisi Riwayat orderan yang di hapus oleh admin.



Gambar 6.7 Menu *Trash*

6.4.7 Menu Pengiriman Pesanan (*Shipment)*

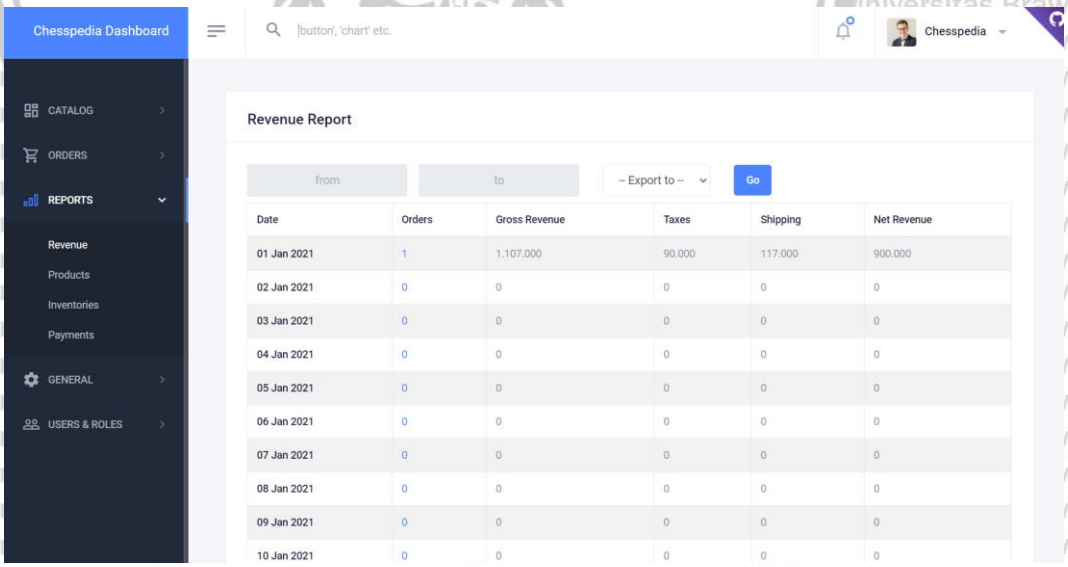
Pada gambar 6.8 dibawah ini merupakan impelementasi dari menu Pengiriman Pesanan (Shipment). Pada panel ini, admin dapat melihat pesanan yang telah memiliki status dikirimkan, dan memeriksa status pengiriman dari pesanan pelanggan.



Gambar 6.8 Menu Pengiriman Pesanan (*Shipment*)

6.4.8 Menu Laporan Pendapatan (*Revenue Reports*)

Pada gambar 6.9 dibawah ini merupakan impelementasi dari menu Laporan Pendapatan *(Revenue Reports).* Pada panel ini, admin dapat melihat laporan pendapatan dari pesanan yang telah selesai dilakukan. Secara *default* data pesanan yang muncul di laporan pendapatan merupakan seluruh pesanan yang telah selesai didalam satu bulan berjalan. Admin juga dapat melakukan filter terhadap tanggal berapa saja pesanan dibuat serta melakukan cetak laporan pendapatan

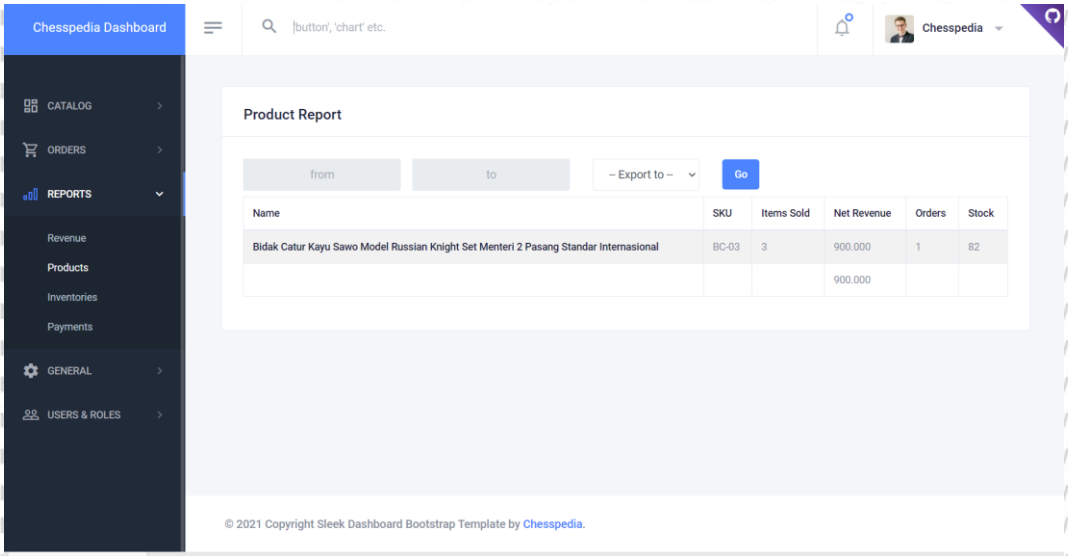


Gambar 6.9 Menu Laporan Pendapatan (*Revenue Reports*)

6.4.9 Menu Laporan Produk (Products Reports)

Pada gambar 6.10 dibawah ini merupakan impelementasi dari menu Laporan Produk *(Products Reports).* Pada panel ini, admin dapat melihat laporan

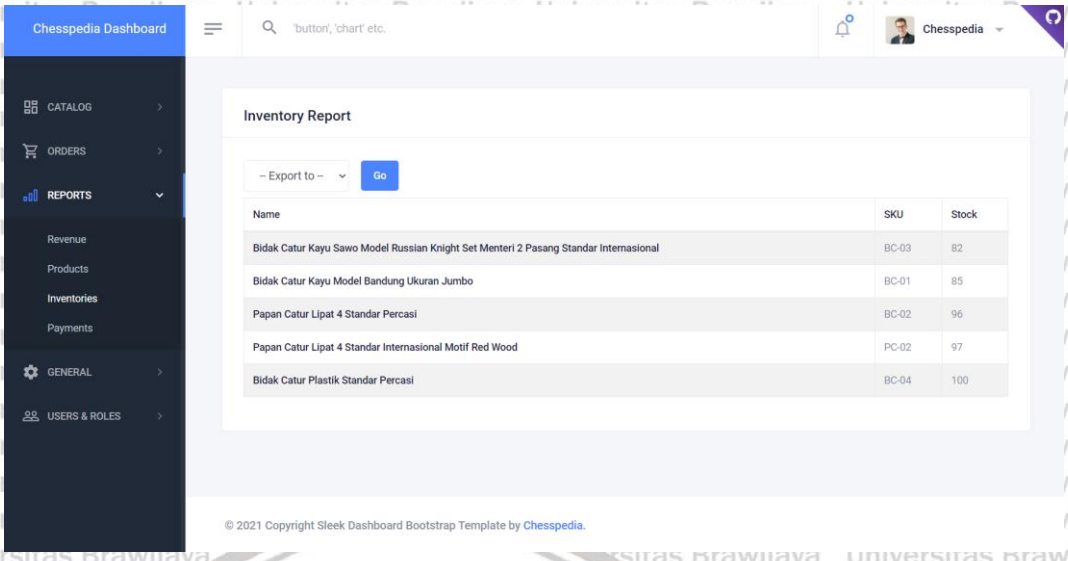
produk mana saja yang telah terjual. Secara *default* data produk yang muncul di laporan produk merupakan seluruh produk yang telah selesai didalam satu bulan berjalan. Admin juga dapat melakukan filter terhadap tanggal berapa saja produk terjual serta melakukan export maupun cetak laporan produk terjual.



Gambar 6.10 Menu Laporan Produk (Products Reports)

6.4.10 Menu Laporan Inventori *(Inventory Report)*

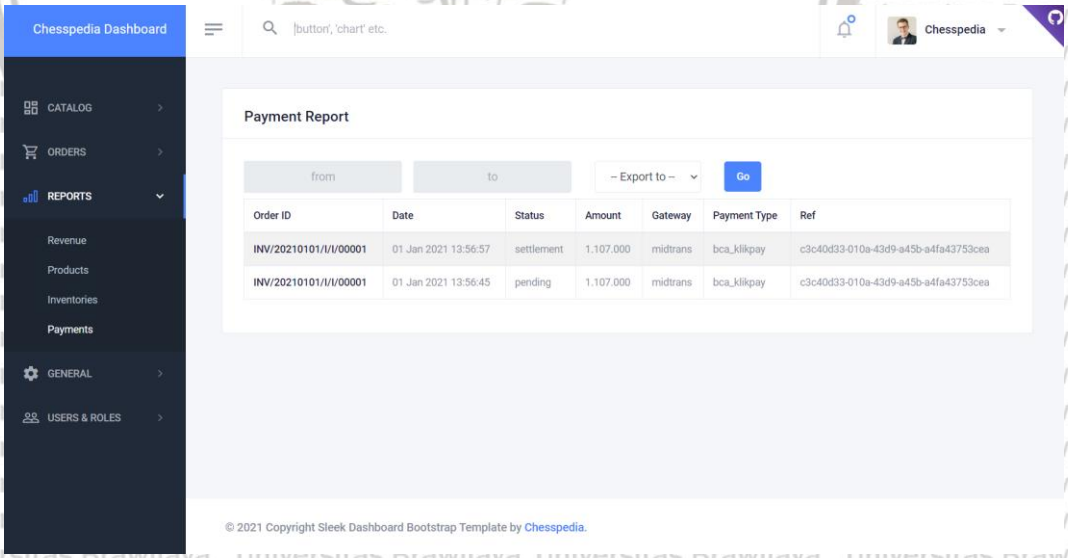
Pada gambar 6.11 dibawah ini merupakan implementasi dari menu Laporan Inventori Produk *(Inventories Reports).* Pada panel ini, admin dapat melihat laporan banyaknya stok produk yang dimiliki. Admin juga dapat melakukan export maupun cetak laporan stok produk yang dimiliki.



Gambar 6.11 Menu Laporan Inventori *(Inventory Report)*

6.4.11 Menu Laporan Pembayaran *(Payments Reports)*

Pada gambar 6.12 dibawah ini merupakan implementasi dari menu Laporan Pembayaran *(Payments Reports).* Pada panel ini, admin dapat melihat laporan pembayaran yang sudah dilakukan oleh buyer melalui *payment gateway* midtrans. Secara *default* data pembayaran yang muncul di laporan pembayaran merupakan data pembayaran yang dilakukan dalam satu bulan berjalan. Admin juga dapat melakukan filter terhadap tanggal berapa saja pembayaran dilakukan serta melakukan *export* maupun cetak laporan pembayaran

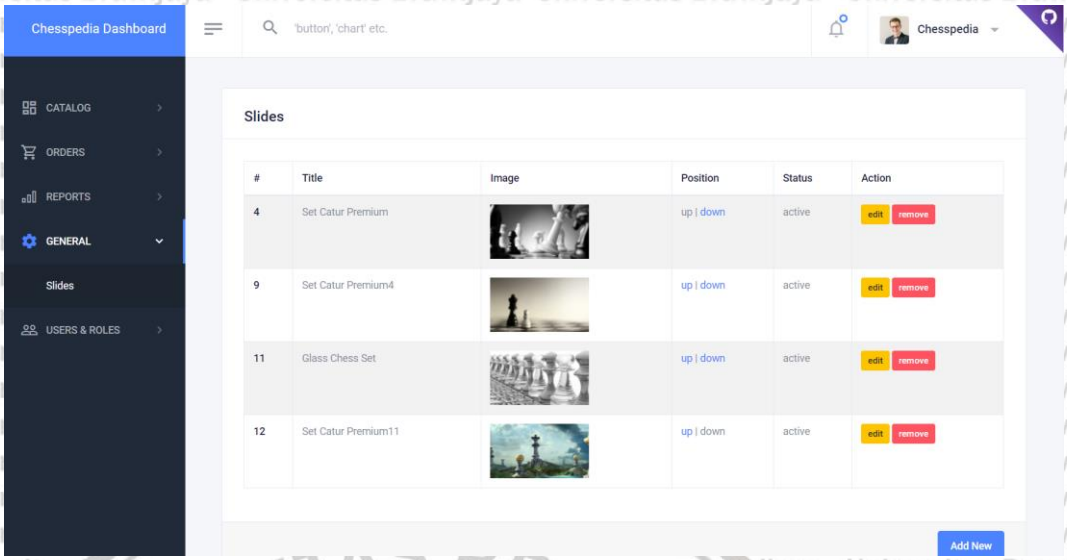


Gambar 6.12 Menu Laporan Pembayaran *(Payments Reports)*

6.4.12 Menu Pengaturan *Slide*

Pada gambar 6.13 dibawah ini merupakan implementasi dari menu pengaturan slide. Admin dapat mengatur gambar mana saja yang akan

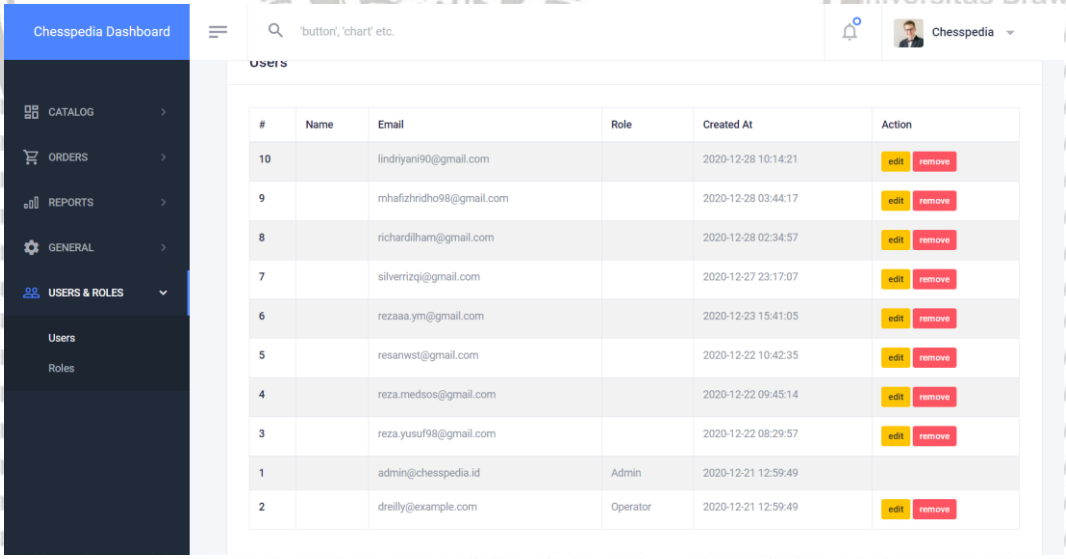
ditampilkan dalam halaman utama *website* ecommerce chesspedia, mengedit slide, dan menghapus slide.



Gambar 6.13 Menu Pengaturan Slide

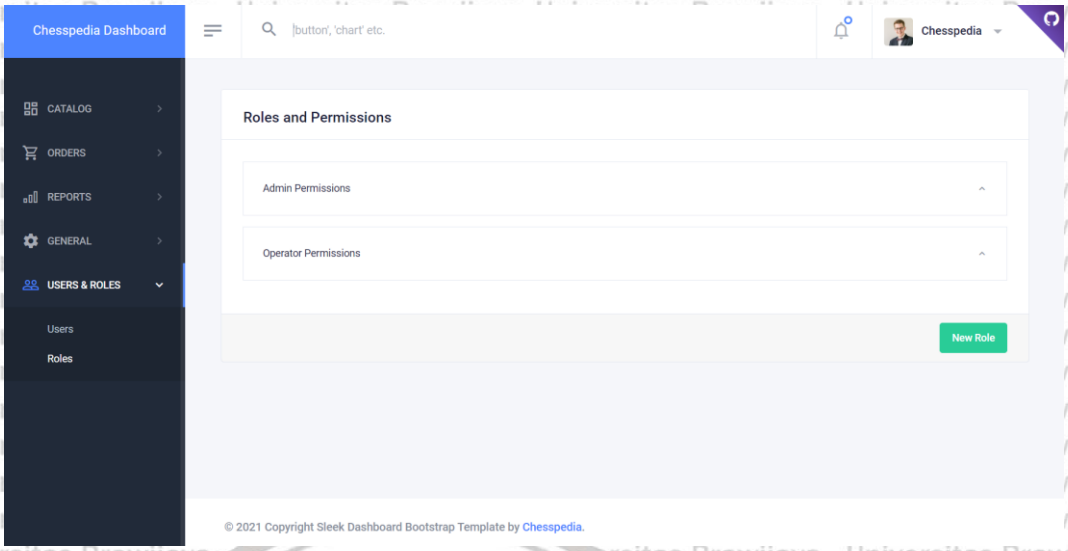
6.4.13 Menu Pengaturan User

Pada gambar 6.14 dibawah ini merupakan implementasi dari menu pengaturan user. Admin dapat melihat user mana saja yang telah terdaftar di dalam *website* ecommerce chesspedia, serta melakukan hapus serta mengedit informasi user.



6.4.14 Menu Pengaturan *Role*

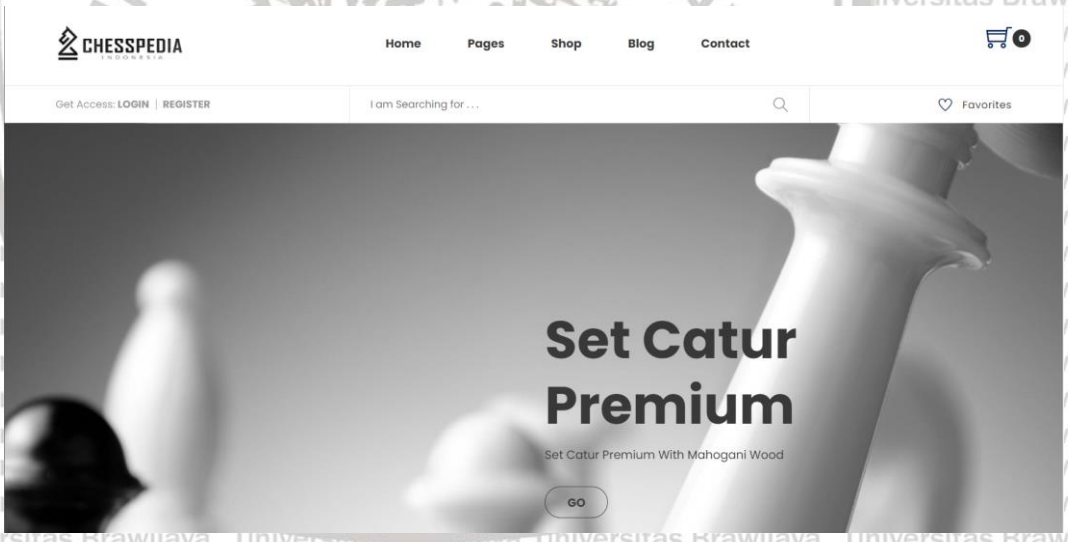
Pada gambar 6.14 dibawah ini merupakan implementasi dari menu pengaturan *role* user. Admin dapat mengatur apakah suatu user merupakan *role* Admin, Operator maupun user biasa.

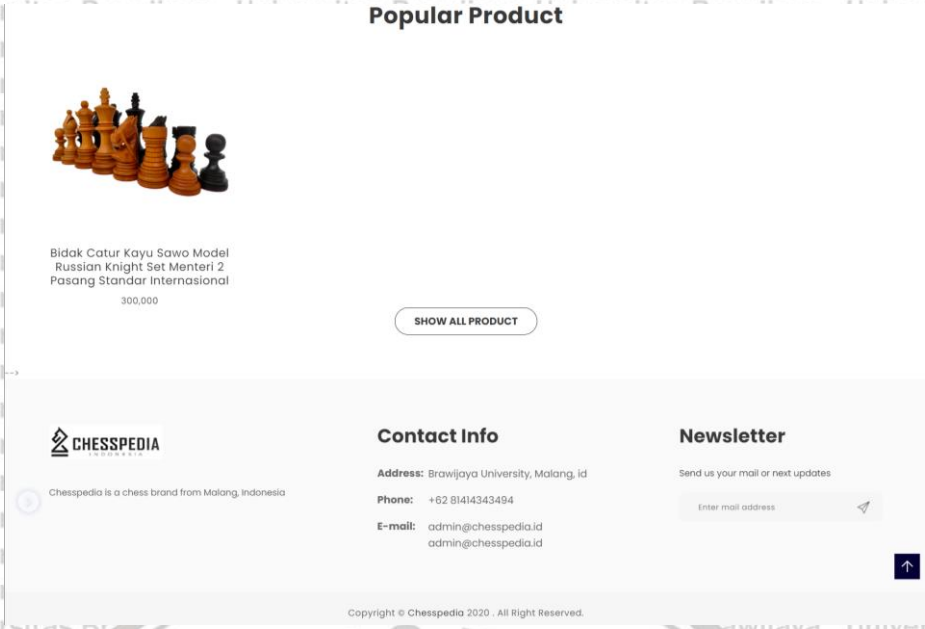


Gambar 6.14 Menu Pengaturan *Role*

6.4.15 Halaman Utama Chesspedia

Pada gambar 6.15 merupakan hasil dari implementasi dari tampilan utama *website* ecommerce chesspedia. Pengunjung maupun pelanggan dapat melihat foto produk pada slide dan juga produk favorit pada halaman utama.

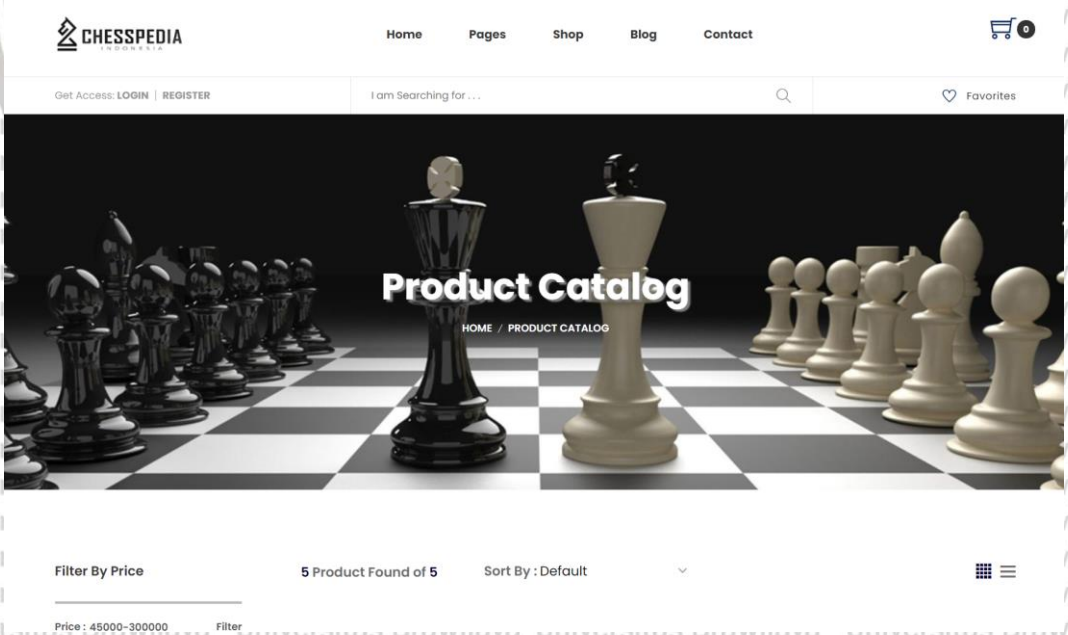


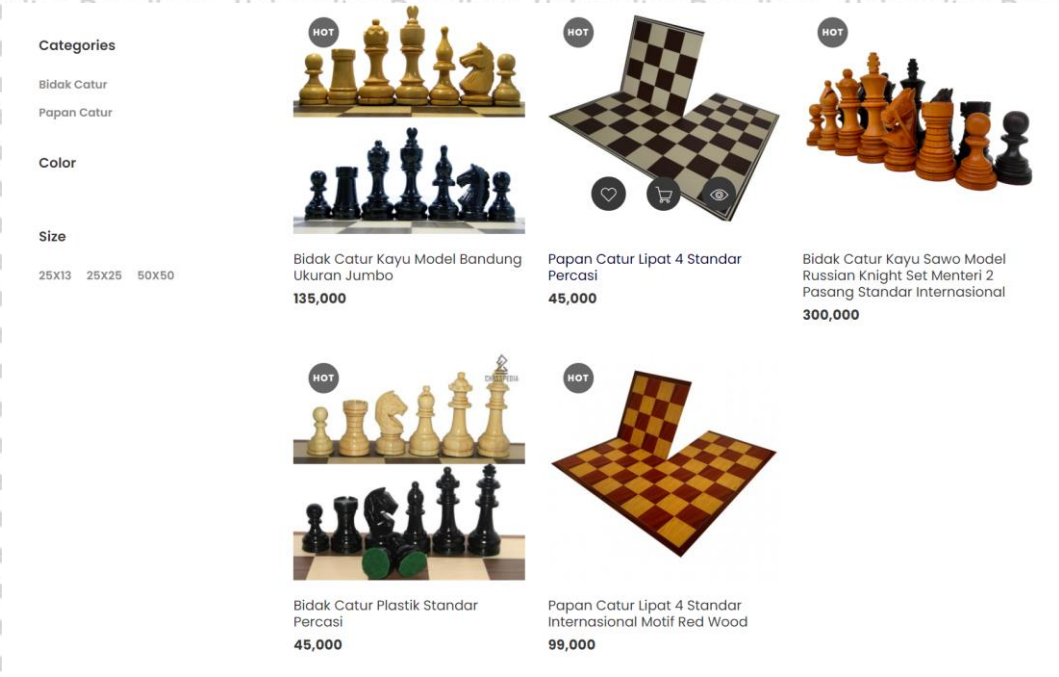


Gambar 6.15 Tampilan Halaman Utama Chesspedia

6.4.16 Halaman Seluruh Produk

Pada gambar 6.16 merupakan hasil dari implementasi dari tampilan seluruh produk *website* ecommerce chesspedia. Pengunjung maupun pelanggan dapat melihat daftar seluruh produk yang di sediakan serta dapat melakukan filter pada daftar produk yang diinginkan.

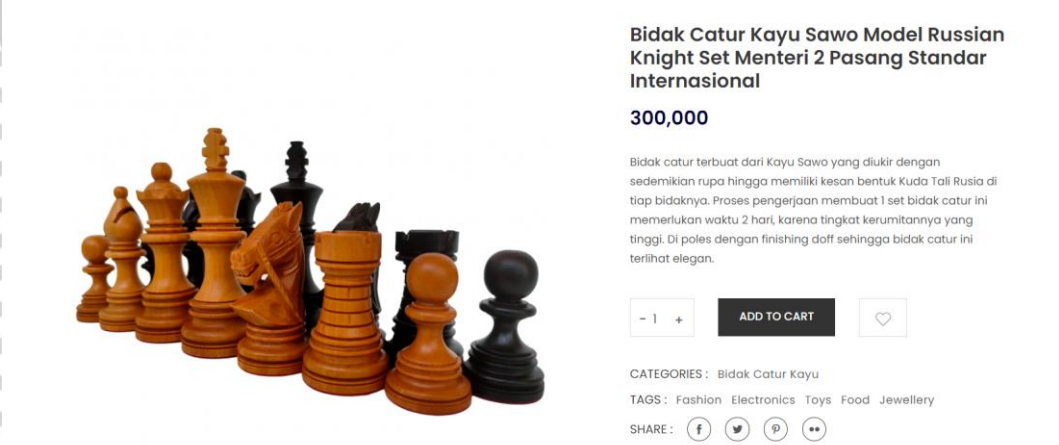




Gambar 6.16 Halaman Seluruh Produk

6.4.17 Halaman Detail Produk

Pada gambar 6.17 merupakan hasil dari implementasi dari tampilan detail produk *website* ecommerce chesspedia. Pengunjung maupun pelanggan dapat melihat detail dari produk yang disediakan.



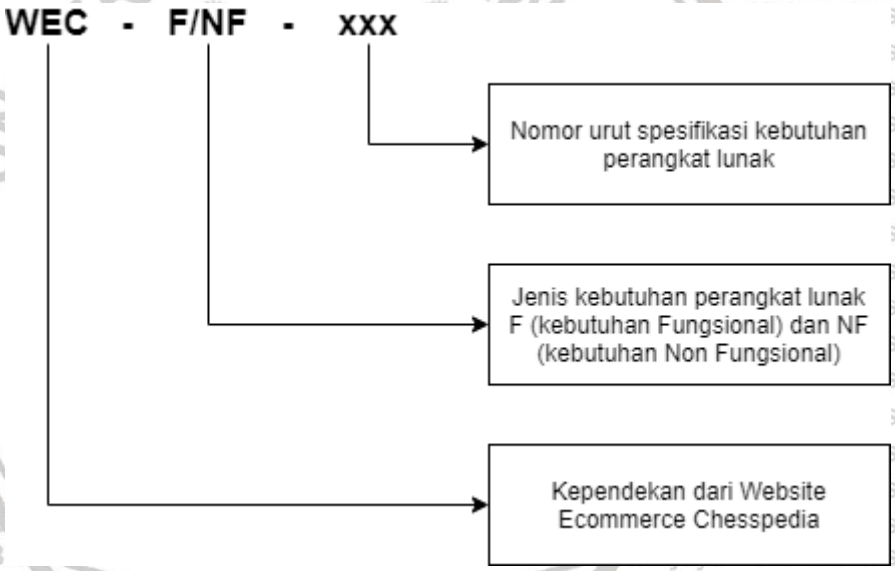
Gambar 6.17 Halaman Detail Produk

BAB 7 PENGUJIAN SISTEM

Pada bab ini dilakukan pengujian pada *website e-commerce* Chesspedia. Pengujian ini dilakukan untuk memastikan apakah hasil perancangan dan implementasi sesuai dengan spesifikasi. Pengujian ini dibagi menjadi dua, yaitu pengujian fungsional dengan menggunakan metode *black-box*, dan pengujian non-fungsional menggunakan *usability testing.*

7.1 Pengujian Fungsional

Tahap pengujian fungsional akan melakukan pengujian terkait website *E-commerce* Chesspedia untuk memastikan apakah sistem dapat berjalan sesuai dengan kebutuhan fungsional yang sudah peneliti ajukan sebelumnya pada bab perancangan sistem. Pengujian fungsional kali ini menggunakan metode black-box. Pada setiap tabel pengujian akan diberikan penomoran dengan format seperti pada Gambar 7.1. Yang mana penomoran terdiri atas singkatan nama sistem, singkatan pengujian, dan aktor yang melakukan pengujian fungsional beserta nomor urutnya.



Gambar 7.1 Aturan Penomoran Pengujian Fungsional

Hasil pengujian fungsional website *e-commerce* Chesspedia akan dijelaskan pada Tabel 7.1 – 7.20.

7.1.1 Uji Validasi Melakukan Proses *Register*

Pada Tabel 7.1 menjelaskan mengenai Langkah-langkah pengujian validasi melakukan *register* pada sistem dengan mendaftarkan data user seperti nama, email, dan *password*. Kondisi WEC-P-001 bertujuan memastikan sistem dapat digunakan untuk melakukan *register* user baru ke dalam sistem. Pengujian dilakukan terhadap penguji yakni pengunjung *website e-commerce* chesspedia.

Tabel 7.1 Uji Validasi Melakukan Proses *Register*

Kode Pengujian	WEC-P-001
Kasus Pengujian	Melakukan *Register*
Kode Kebutuhan Fungsional	WEC-F-001
Tujuan Pengujian	Sistem berhasil menyimpan data user baru
Prosedur Uji	Aktor memilih menu *register* dan mengisi data diri yang diminta oleh sistem yakni nama, email dan *password*
Hasil yang diharapkan	Sistem menyimpan data user baru dan mengalihkan ke dalam halaman *home* dengan tampilan *login* yang telah ditampilkan
Keluaran	Sistem menyimpan data user baru dan menampilkan halaman *home* dengan halaman *login* yang telah ditampilkan
Status Uji	Valid

7.1.2 Uji Validasi Melihat Daftar Seluruh Produk

Pada tabel 7.2 dilakukan uji validasi melihat daftar seluruh produk. Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan dan pengunjung *website e-commerce* chesspedia. Kondisi WEC-P-002 bertujuan untuk memastikan apakah sistem dapat menampilkan daftar seluruh produk dengan baik

Tabel 7.2 Uji Validasi Melihat Daftar Seluruh Produk

Kode Pengujian	WEC-P-002
Kasus Pengujian	Melihat Daftar Seluruh Produk
Kode Kebutuhan Fungsional	WEC-F-002
Tujuan Pengujian	Sistem menampilkan halaman daftar seluruh produk
Prosedur Uji	Memilih button tampilkan seluruh produk
Hasil yang diharapkan	Sistem dapat menampilkan halaman daftar seluruh produk
Keluaran	Sistem berhasil menampilkan halaman daftar seluruh produk
Status Uji	Valid

7.1.3 Uji Validasi Melihat Halaman Detail Produk

Pada tabel 7.3 dilakukan uji validasi melihat detail produk. Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan dan pengunjung *website e-commerce* chesspedia. Kondisi WEC-P-003 bertujuan untuk memastikan apakah sistem dapat menampilkan halaman detail produk dengan baik

Tabel 7.3 Validasi Melihat Halaman Detail Produk

Kode Pengujian	WEC-P-003
Kasus Pengujian	Melihat Halaman Detail Produk
Kode Kebutuhan Fungsional	WEC-F-003
Tujuan Pengujian	Sistem menampilkan halaman detail produk
Prosedur Uji	Memilih salah satu produk dari dalam daftar seluruh produk
Hasil yang diharapkan	Sistem dapat menampilkan halaman detail produk
Keluaran	Sistem berhasil menampilkan halaman detail produk
Status Uji	Valid

7.1.4 Uji Validasi Melakukan *Search* Produk

Pada tabel 7.4 dilakukan uji validasi melakukan *search* produk. Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan dan pengunjung *website e-commerce* chesspedia. Kondisi WEC-P-004 bertujuan untuk memastikan apakah sistem dapat menampilkan halaman pencarian produk dengan baik

Tabel 7.4 Validasi Melakukan *Search* Produk

Kode Pengujian	WEC-P-004
Kasus Pengujian	Melakukan *Search* Produk
Kode Kebutuhan Fungsional	WEC-F-004
Tujuan Pengujian	Sistem menampilkan halaman produk yang dicari
Prosedur Uji	Memilih kata kunci untuk melakukan pencarian produk
Hasil yang diharapkan	Sistem dapat menampilkan halaman produk yang dicari

Keluaran	Sistem berhasil menampilkan halaman produk yang dicari
Status Uji	Valid

7.1.5 Uji Validasi Proses Login

Pada tabel 7.5 dilakukan uji validasi melakukan proses *login*. Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan dan admin *website e-commerce* chesspedia. Kondisi WEC-P-005 bertujuan untuk memastikan apakah user dapat masuk ke dalam sistem (*login*) dengan baik

Tabel 7.5 Uji Validasi Proses *Login*

Kode Pengujian	WEC-P-005
Kasus Pengujian	Melakukan Proses *Login*
Kode Kebutuhan Fungsional	WEC-F-005
Tujuan Pengujian	Aktor berhasil melakukan *login* dan masuk ke dalam sistem
Prosedur Uji	Memilih menu *login* dan menginputkan kolom email dan *password*
Hasil yang diharapkan	Sistem memvalidasi inputan, dan mengalihkan Kembali kehalaman dimana *login* dilakukan.
Keluaran	Sistem berhasil memvalidasi inputan, dan mengalihkan Kembali user ke halaman terakhir sebelum *login*.
Status Uji	Valid

7.1.6 Uji Validasi Melakukan Proses Logout

Pada tabel 7.6 dilakukan uji validasi melakukan proses *logout*. Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan dan admin *website e-commerce* chesspedia. Kondisi WEC-P-006 bertujuan untuk memastikan apakah user dapat melakukan proses *logout* dengan baik.

Tabel 7.6 Uji Validasi Melakukan Proses *Logout*

Kode Pengujian	WEC-P-006
Kasus Pengujian	Melakukan Proses *Logout*
Kode Kebutuhan Fungsional	WEC-F-006
Tujuan Pengujian	Sistem menghapus *session* dan menampilkan halaman *home*.
Prosedur Uji	Memilih menu *logout*

Hasil yang diharapkan	Sistem dapat menghapus *session* dan menampilkan halaman *home*.
Keluaran	Sistem berhasil menghapus *session* dan menampilkan halaman *home*.
Status Uji	Valid

7.1.7 Uji Validasi Menambahkan Produk ke Keranjang

Pada tabel 7.7 dilakukan uji validasi menambahkan produk ke keranjang. Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan *website e-commerce* chesspedia. Kondisi WEC-P-007 bertujuan untuk memastikan apakah user dapat menambahkan produk ke dalam keranjang dengan baik.

Tabel 7.7 Uji Validasi Menambahkan Produk ke Keranjang

Kode Pengujian	WEC-P-007
Kasus Pengujian	Menambahkan produk ke keranjang
Kode Kebutuhan Fungsional	WEC-F-007
Tujuan Pengujian	Sistem menambahkan produk ke dalam keranjang belanja dan jumlah barang yang terdapat dalam keranjang di tampilkan pada menuju menu keranjang yang berada pada pojok kanan atas *website*
Prosedur Uji	Klik logo keranjang pada halaman seluruh produk maupun detail produk
Hasil yang diharapkan	Sistem dapat menambahkan produk ke dalam keranjang belanja dan jumlah barang yang terdapat dalam keranjang di tampilkan pada menuju menu keranjang yang berada pada pojok kanan atas *website*
Keluaran	Sistem berhasil menambahkan produk ke dalam keranjang belanja dan jumlah barang yang terdapat dalam keranjang di tampilkan pada menuju menu keranjang yang berada pada pojok kanan atas *website*
Status Uji	Valid

7.1.8 Uji Validasi Melakukan Chekcout

Pada tabel 7.8 dilakukan uji validasi melakukan *checkout* Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan *website e-*

commerce chesspedia. Kondisi WEC-P-008 bertujuan untuk memastikan apakah user dapat melakukan proses *checkout* dengan baik.

Tabel 7.8 Uji Validasi Melakukan *Checkout*

Kode Pengujian	WEC-P-008
Kasus Pengujian	Melakukan *Checkout*
Kode Kebutuhan Fungsional	WEC-F-008
Tujuan Pengujian	Sistem menyimpan pesanan, dan menampilkan halaman pembayaran
Prosedur Uji	Mengisi data pengiriman yang diminta dan klik tombol bayar
Hasil yang diharapkan	Sistem menyimpan pesanan, dan berhasil menampilkan halaman pembayaran
Keluaran	Sistem menyimpan pesanan, dan menampilkan halaman pembayaran
Status Uji	Valid

7.1.9 Uji Validasi Melakukan Pembayaran

Pada tabel 7.9 dilakukan uji validasi melakukan pembayaran. Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan *website e-commerce* chesspedia. Kondisi WEC-P-009 bertujuan untuk memastikan apakah user dapat melakukan proses pembayaran dengan baik.

Tabel 7.9 Uji Validasi Melakukan Pembayaran

Kode Pengujian	WEC-P-009
Kasus Pengujian	Melakukan *Checkout*
Kode Kebutuhan Fungsional	WEC-F-009
Tujuan Pengujian	Sistem dapat mengakses halaman pembayaran milik midtrans, dan berhasil melakukan proses pembayaran transaksi pelanggan
Prosedur Uji	Memilih pembayaran yang ingin digunakan, sistem melakukan verifikasi pesanan yang telah dibayar.
Hasil yang diharapkan	Sistem dapat mengakses halaman pembayaran milik midtrans, dan berhasil melakukan proses pembayaran transaksi pelanggan
Keluaran	Sistem berhasil mengakses halaman pembayaran milik midtrans, dan berhasil melakukan proses pembayaran transaksi pelanggan

Status Uji	Valid

7.1.10 Uji Validasi Melihat Riwayat Transaksi

Pada tabel 7.10 dilakukan uji validasi melihat Riwayat transaksi. Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan *website e-commerce* chesspedia. Kondisi WEC-P-010 bertujuan untuk memastikan apakah user dapat melihat Riwayat transaksi yang pernah dilakukan.

Tabel 7.10 Uji Validasi Melihat Riwayat Transaksi

Kode Pengujian	WEC-P-010
Kasus Pengujian	Melihat Riwayat Transaksi
Kode Kebutuhan Fungsional	WEC-F-010
Tujuan Pengujian	Sistem dapat menampilkan halaman user dengan daftar Riwayat transaksi yang pernah dilakukan
Prosedur Uji	Memilih menu akun dan dilanjutkan dengan memilik sub-menu transaksi
Hasil yang diharapkan	Sistem dapat menampilkan halaman user dengan daftar Riwayat transaksi yang pernah dilakukan
Keluaran	Sistem berhasil menampilkan halaman user area dengan daftar Riwayat transaksi yang pernah dilakukan
Status Uji	Valid

7.1.11 Uji Validasi Melihat Status Transaksi

Pada tabel 7.11 dilakukan uji validasi melihat status transaksi. Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai pelanggan *website e-commerce* chesspedia. Kondisi WEC-P-011 bertujuan untuk memastikan apakah user dapat melihat status transaksi yang pernah dilakukan.

Tabel 7.11 Uji Validasi Melihat Riwayat Transaksi

Kode Pengujian	WEC-P-011
Kasus Pengujian	Melihat Riwayat Transaksi
Kode Kebutuhan Fungsional	WEC-F-011
Tujuan Pengujian	Sistem dapat menampilkan halaman user dengan status transaksi yang pernah dilakukan

Prosedur Uji	Memilih menu akun, dilanjutkan dengan memilih sub-menu transaksi, dan memilih transaksi mana yang ingin dilihat statusnya
Hasil yang diharapkan	Sistem dapat menampilkan halaman user dengan daftar status transaksi yang dipilih
Keluaran	Sistem berhasil menampilkan halaman user area dengan daftar status transaksi yang dipilih
Status Uji	Valid

7.1.12 Uji Validasi Melakukan *Login* Admin

Pada tabel 7.12 dilakukan uji validasi Melakukan *Login* Admin Pengujian dilakukan terhadap penguji yang teridentifikasi sebagai admin *website e-commerce* chesspedia. Kondisi WEC-P-012 bertujuan untuk memastikan apakah admin dapat melakukan *login* kedalam sistem WEC.

Tabel 7.12 Uji Validasi Melakukan *Login* Admin

Kode Pengujian	WEC-P-012
Kasus Pengujian	Melakukan *login* admin
Kode Kebutuhan Fungsional	WEC-F-012
Tujuan Pengujian	Sistem memvalidasi masukan dan mengalihkan ke halaman panel admin
Prosedur Uji	Memilih menu *login* dan mengisi kolom *username* dan *password*
Hasil yang diharapkan	Sistem dapat memvalidasi masukan dan mengalihkan ke halaman panel admin
Keluaran	Sistem berhasil memvalidasi masukan dan mengalihkan ke halaman panel admin.
Status Uji	Valid

7.1.13 Uji Validasi Mengelola Pesanan

Pada tabel 7.13 dilakukan uji validasi mengelola pesanan dilakukan terhadap penguji yang teridentifikasi sebagai admin *website e-commerce* chesspedia. Kondisi WEC-P-01 bertujuan untuk memastikan apakah admin dapat melakukan pengelolaan terhadap pesanan yang masuk didalam sistem WEC.

Tabel 7.13 Uji Validasi Mengelola Pesanan

Kode Pengujian	WEC-P-013
Kasus Pengujian	Mengelola Pesanan
Kode Kebutuhan Fungsional	WEC-F-013
Tujuan Pengujian	Sistem berhasil menjalankan aktivitas pengelolaan pesanan
Prosedur Uji	Memilih menu pesanan dan melakukan filter pesanan berdasarkan status, melihat status pembayaran dan menyetujui atau menolak pesanan
Hasil yang diharapkan	Sistem dapat menjalankan aktivitas pengelolaan pesanan
Keluaran	Sistem berhasil menjalankan aktivitas pengelolaan pesanan
Status Uji	Valid

7.1.14 Uji Validasi Mengelola Produk

Pada tabel 7.14 dilakukan uji validasi mengelola peroduk dilakukan terhadap penguji yang teridentifikasi sebagai admin *website e-commerce* chesspedia. Kondisi WEC-P-014 bertujuan untuk memastikan apakah admin dapat melakukan pengelolaan terhadap produk yang masuk didalam sistem WEC.

Tabel 7.14 Uji Validasi Mengelola Produk

Kode Pengujian	WEC-P-014
Kasus Pengujian	Mengelola Produk
Kode Kebutuhan Fungsional	WEC-F-014
Tujuan Pengujian	Sistem berhasil menjalankan aktivitas pengelolaan produk seperti proses *create, read, update,* dan *delete* pada produk
Prosedur Uji	Memilih menu produk, dan melakukan proses *create, read, update,* dan *delete* pada produk
Hasil yang diharapkan	Sistem dapat menjalankan aktivitas pengelolaan produk
Keluaran	Sistem berhasil menjalankan aktivitas pengelolaan produk
Status Uji	Valid

7.1.15 Uji Validasi Mengelola Pengiriman

Pada tabel 7.15 dilakukan uji validasi mengelola pengiriman dilakukan terhadap penguji yang teridentifikasi sebagai admin *website e-commerce* chesspedia. Kondisi WEC-P-015 bertujuan untuk memastikan apakah admin dapat melakukan pengelolaan terhadap pengiriman dari transaksi yang berjalan didalam sistem WEC.

Tabel 7.15 Uji Validasi Mengelola Pengiriman

Kode Pengujian	WEC-P-015
Kasus Pengujian	Mengelola Pengiriman
Kode Kebutuhan Fungsional	WEC-F-015
Tujuan Pengujian	Sistem berhasil menjalankan aktivitas pengelolaan pengiriman
Prosedur Uji	Memilih menu pengiriman, dan melakukan penambahan atau mengubah nomor resi pengiriman.
Hasil yang diharapkan	Sistem dapat menjalankan aktivitas pengelolaan pengiriman
Keluaran	Sistem berhasil menjalankan aktivitas pengelolaan pengiriman
Status Uji	Valid

7.1.16 Uji Validasi Mencetak Laporan Penjualan

Pada tabel 7.16 dilakukan uji validasi mencetak laporan penjualan dilakukan terhadap penguji yang teridentifikasi sebagai admin *website e-commerce* chesspedia. Kondisi WEC-P-016 bertujuan untuk memastikan apakah admin dapat melakukan pencetakan terhadap laporan dari transaksi yang berjalan didalam sistem WEC.

Tabel 7.16 Uji Validasi Mencetak Laporan Penjualan

Kode Pengujian	WEC-P-016
Kasus Pengujian	Mencetak Laporan Penjualan
Kode Kebutuhan Fungsional	WEC-F-016
Tujuan Pengujian	Sistem berhasil menjalankan aktivitas Mencetak Laporan Penjualan
Prosedur Uji	Memilih menu laporan, dan mengisi filter tanggal lalu klik tombol lihat laporan

Hasil yang diharapkan	Sistem dapat menjalankan aktivitas mencetak laporan penjualan
Keluaran	Sistem berhasil menjalankan aktivitas cetak lapoan penjualan
Status Uji	Valid

7.1.17 Uji Validasi Mencetak Laporan Stok Produk

Pada tabel 7.17 dilakukan uji validasi mencetak laporan Stok Produk dilakukan terhadap penguji yang teridentifikasi sebagai admin *website e-commerce* chesspedia. Kondisi WEC-P-017 bertujuan untuk memastikan apakah admin dapat melakukan pencetakan laporan stok produk dari produk yang terdapat didalam sistem WEC.

Tabel 7.17 Uji Validasi Mencetak Laporan Stok Produk

Kode Pengujian	WEC-P-017
Kasus Pengujian	Mencetak Laporan Stok Produk
Kode Kebutuhan Fungsional	WEC-F-017
Tujuan Pengujian	Sistem berhasil menjalankan aktivitas Mencetak Laporan Stok Produk
Prosedur Uji	Memilih menu laporan, dan mengisi filter tanggal lalu klik tombol lihat laporan
Hasil yang diharapkan	Sistem dapat menjalankan aktivitas mencetak laporan stok produk
Keluaran	Sistem berhasil menjalankan aktivitas cetak lapoan stok produk
Status Uji	Valid

7.2 Pengujian Non-Fungsional

Pada pengujian non-fungsional akan diuji dengan menggunakan dua metode pengujian yakni *usability testing dan compatibility testing*. Pada pengujian usability yang akan diuji pada sistem menggunakan metode *System Usability Scale* (SUS). Lalu untuk pengujian kompabilitas dilakukan dengan memanfaatkan *tools* bernama *sortsite* dengan mengecek apakah website yang dibangun didukung oleh berbagai macam jenis browse.

7.2.1 Usability Testing

Pada pengujian non-fungsional akan diuji dengan menggunakan usability testing dengan melibatkan beberapa responden yakni admin dan customer yang pernah berbelanja di *website e-commerce chesspedia.* Pengujian usability yang akan diuji pada sistem menggunakan metode *System Usability Scale* (SUS).

7.2.1.1 Prosedur *Usability Testing*

Pada proses usability testing digunakan metode SUS dilakukan dengan memberikan beberapa pertanyaan kepada beberapa responden yang bersangkutan yang sebelumnya diberikan kesempatan untuk mencoba mengoperasikan sistem. Pada proses pengujian yang dilakukan terdapat lima respoden yang berpatisipasi. Responden akan dipersilahkan untuk menjalankan sistem sesuai dengan *role*-nya. Kemudian akan diberi waktu untuk menjawab pertanyaan-pertanyaan yang berkaitan sistem yang dicoba gunakan. Daftar pertanyaan menggunakan metode SUS dapat dilihat pada Tabel 7.15.

Tabel 7.18 Daftar Pertanyaan Pengujian *Usability* menggunakan metode SUS.

No.	Pertanyaan
1.	Saya berpikir akan menggunakan sistem ini lagi.
2.	Saya merasa sistem ini rumit untuk digunakan.
3.	Saya merasa sistem ini mudah digunakan.
4.	Saya membutuhkan bantuan dari orang lain atau teknisi dalam menggunakan sistem ini.
5.	Saya merasa fitur-fitur sistem ini berjalan dengan semestinya.
6.	Saya merasa ada banyak hal yang tidak konsisten (tidak serasi pada sistem ini).
7.	Saya merasa orang lain akan memahami cara menggunakan sistem ini dengan cepat.
8.	Saya merasa sistem ini membingungkan.
9.	Saya merasa tidak ada hambatan dalam menggunakan sistem ini.
10.	Saya perlu membiasakan diri terlebih dahulu sebelum menggunakan sistem ini.

7.2.1.2 Analisis dan Hasil Usability Testing

Setelah prosedur proses *usability* testing selesai dilakukan, maka didapatkan nilai respon yang diberikan oleh responden. Selanjutnya, dilakukan proses analisa hasil pengujian yang telah didapat. Hasil pengujian berdasarkan pertanyaan yang diberikan kepada responden dapat dilihat pada Tabel 7.16.

Tabel 7.19 Hasil Kuisioner Responden terhadap pertanyaan Metode SUS.

No	Nama	Pertanyaan									
		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10
1	Edi Tanjung	4	2	4	3	4	2	4	2	4	4
2	Liska Novita	3	3	5	2	3	2	5	3	4	4
3	Luqmanul Hakim	5	2	5	1	5	3	5	2	5	1
4	Lisa Iskandar	4	3	4	2	3	1	4	2	4	2
5	M Hafizh Ridho	4	3	5	3	4	2	4	2	5	2

Setelah responden mengisi kuisioner, dilakukan proses perhitungan nilai akhir yang didasari oleh penilaian yang telah diberikan responden dalam kuisioner tersebut. Perhitungan total nilai akhir pada tabel 7.16 diproses dengan ketentuan nilai yakni pertanyaan ganjil akan dikurangi satu (1) nilai dan nilai lima (5) akan dikurangi dari pertanyaan genap. Hasil perhitungan total nilai akhir dapat dilihat pada tabel 7.17.

Tabel 7.20 Hasil Perhitungan Skor Akhir Penilaian Kuisioner.

No	Nama	Pertanyaan										Total Nilai	Dikali 2,5
		Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9	Q10		
1	Edi Tanjung	3	2	3	3	3	1	3	3	3	3	27	67,5
2	Liska Novita	4	2	2	3	3	2	2	3	4	1	26	65
3	Luqmanul Hakim	4	2	4	3	4	4	4	4	4	3	36	90
4	Lisa Iskandar	3	2	3	3	2	4	3	3	3	3	29	72.5
5	M Hafizh Ridho	3	2	4	2	3	3	3	3	4	3	30	75
Rata-Rata													74

Berdasarkan hasil nilai akhir yang didapatkan pada Tabel 7.17 nilai akhir yang diberikan dari kelima responden dalam pengujian usability sistem adalah 74. Berdasarkan nilai rata-rata tersebut dapat disimpulkan bahwa sistem termasuk kedalam kategori *acceptable* yang berarti sistem yang diimplementasikan telah sesuai dengan kebutuhan pengguna.

7.2.2 Compatibility Testing

Pengujian non-fungsional pada *Website E-commerce* Chesspedia dilakukan juga dengan menggunakan *compatibility testing*. *Compatibility testing* ini dilakukan untuk memeriksa apkah *Website E-Commerce* Chesspedia dapat berjalan pada beberapa jenis browser yang berbeda - beda. Pengujian dilakukan menggunakan *tools* yang bernama *sortsite*. Gambar 7.2 merupakan hasil dari pengujian menggunakan *sortsite*.

Browser	IE	Edge	Firefox	Safari	Opera	Chrome	iOS		Android
Version	11	88	84	14	73	88	≤ 13	14	88
Critical Issues	✓	✓	✓	✓	✓	✓	✓	✓	✓
Major Issues	◔	✓	✓	✓	✓	✓	✓	✓	✓
Minor Issues	◐	✓	◐	◐	✓	✓	✓	✓	✓

Key

- ● Missing content or functionality
- ◔ Major layout or performance problems
- ◐ Minor layout or performance problems

Gambar 7.2 Hasil Pengujian *Compatibility Testing*

Pada Gambar 7.2 dapat dilihat bahwa terdapat *major issues* pada *website e-commerce* Chesspedia di satu browser, yaitu *Internet Explorer 11*, serta 3 *minor issues* pada tiga (3) browser yakni *Internet Explorer 11, Firefox v.84* dan *Safari v.14.*

Terdapat 1 permasalahan pada pengujian *compatibility testing website* chesspedia. Seperti dapat dilihat pada Gambar 7.3, permasalahan tersebut adalah masalah keamanan pada *website* karena tidak menggunakan protokol HTTPS. Ini terjadi terutama pada halaman yang memiliki *password field*, seperti pada halaman *login* dan *register*. Permasalahan ini terjadi akibat telah berakhirnya lisensi SSL yang dimiliki oleh Chesspedia.

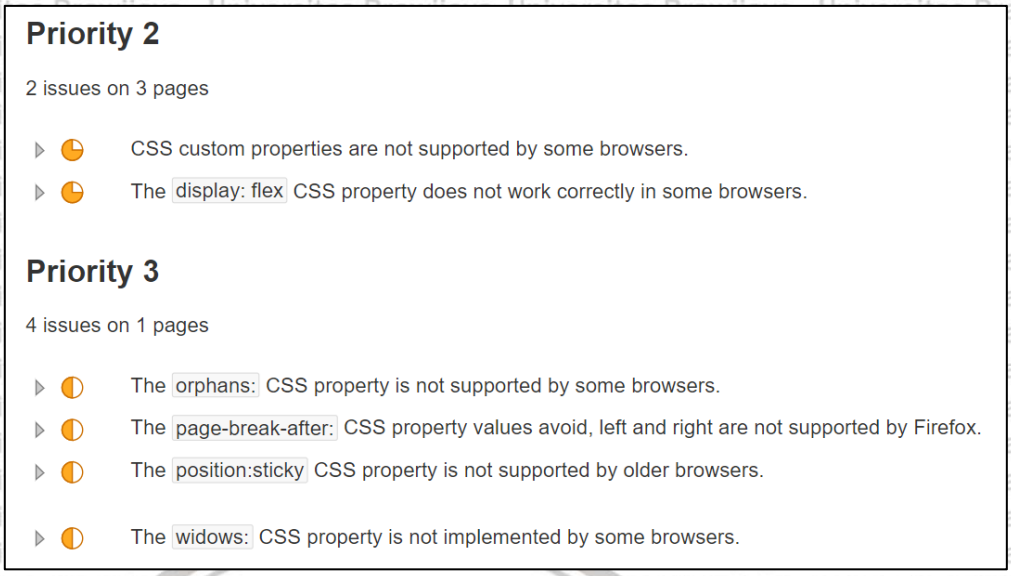
Priority 1

1 issues on 3 pages

● Browsers display multiple security warnings for password fields that don't use HTTPS.

Gambar 7.3 *Critical Issues* Pada Hasil Pengujian Compatibility Testing

Selain *critical issues*, juga terdapat beberapa *major* dan *minor issues*. Pada Gambar 7.4 menunjukan *major* dan *minor issues* yang dialami oleh *website e-commerce* Chesspedia. Kedua permasalahan ini disebabkan oleh beberapa properti CSS yang digunakan untuk antarmuka pengguna tidak didukung oleh beberapa browser. Serta pada beberapa jenis browser lama, property CSS yang digunakan tidak dapat di implementasikan oleh browser.



Gambar 7.4 *Major & Minor Issues* Pada Hasil Pengujian Compatibility Testing

Dari hasil pengujian *compatibility testing* yang telah dilakukan, dapat disimpulkan bahwa *website* dapat dijalankan seluruh browser. Dan untuk hasil yang optimal, penggunaan *browser* versi terbaru sangat disarankan.

BAB 8 KESIMPULAN DAN SARAN

8.1 Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan maka didapat kesimpulan sebagai berikut.

1. Hasil dari analisis kebutuhan *Website E-commerce Chesspedia* memiliki 16 kebutuhan fungsional serta 1 kebutuhan non-fungsional. Terdapat 3 aktor pada sistem yaitu pengunjung, pelanggan dan admin. Analisis kebutuhan diper*oleh* melalui hasil wawancara dengan pemilik. Hasil tersebut selanjutnya diilustrasikan dalam bentuk *use case diagram* dan *use case scenario*

2. Hasil yang diper*oleh* pada tahap analisis kebutuhan dilanjutkan pada tahap perancangan. Pada tahapan ini hasil analisis kebutuhan dijadikan sebagai acuan dan dasar. Perancangan yang dilakukan adalah perancangan arsitektur sistem, *sequence diagram*, *class* diagram, struktur data penyimpanan serta perancangan antarmuka sistem.

   Proses implementasi berisi proses penyusunan kode program untuk mengimplementasikan sebuah *website e-commerce Chesspedia* yang didasari oleh hasil dari perancangan. Proses implementasi dilakukan dengan menggunakan teknologi web yaitu dengan *framework* laravel dengan bahasa pemrograman php. Proses implementasi dilanjutkan dengan pengimplementasian antarmuka sistem berdasarkan hasil perancangan.

3. Aplikasi yang telah dibangun diuji menggunakan metode *black-box* testing serta usability testing. Hasil dari *black-box testing* didapat dari pengujian fungsional menyatakan bahwa sistem telah 100% memenuhi seluruh kebutuhan fungsional. Dan untuk hasil pengujian usability testing nilai yang diper*oleh* 74 dengan kategori *acceptable*. Lalu untuk pengujian *compability testing* dengan menggunakan aplikasi *sortsite* didapatkan hasil bahwa *website* dapat berjalan di berbagai macam browser keluaran terbaru. Namun, terdapat beberapa *major* dan *minor issues* yang berkaitan dengan properti dari CSS untuk tampilan antarmuka tidak didukung oleh beberapa browser versi lama.

8.2 Saran

Saran yang dapat diberikan oleh penulis untuk penelitian lebih lanjut mengenai *Website E-commerce Chesspedia* adalah sebagai berikut.

1. Perlu dilakukan evaluasi terhadap antarmuka *Website E-commerce Chesspedia* agar dapat lebih disesuaikan dengan kebutuhan pengguna. Hal ini berguna dalam rangka optimalisasi *Website E-commerce Chesspedia* serta pelayanan maksimal yang akan dihasilkan.

2. Perlu dilakukannya penambahan fitur rating pada *Website E-commerce Chesspedia* agar pengguna dapat memberikan penilaian terhadap produk yang telah dibeli.

DAFTAR REFERENSI

Aditya, K., Putra, F., & Arwani, I. (2021). *Pemanfaatan API RajaOngkir untuk Cek Ongkos Kirim Otomatis pada Pembangunan Website E-Commerce menggunakan Framework Codeigniter (Studi Kasus : Jingga Hijab)*. *5*(1), 311–318.

Ardi, H. Y., Priyambadha, B., & Soebroto, A. A. (2019). Pengembangan Aplikasi Reservasi dan Manajemen Marketplace Homestay berbasis Web (Studi Kasus : Kota Pekanbaru). *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *3*(9), 8914–8923.

Caldas, M. P. (2003). Management information systems: managing the digital firm. In *Revista de Administração Contemporânea* (Vol. 7, Issue 1). https://doi.org/10.1590/s1415-65552003000100014

DataReportal – Global Digital Insights, 2021. *Ecommerce in Indonesia in 2021*. [online] Available at: https://datareportal.com/reports/digital-2021-indonesia [Diakses 21 Mei. 2021].

Destian Wijaya, B., E.M.A, F. and Fiade, A., 2015. *Implementasi JSON Parsing Pada Aplikasi Mobile E-Commerce*. Jurnal Pseudocode, 2(1).

Febriyanto, E., Rahardja, U., & Alnabawi, N. (2019). Penerapan Midtrans sebagai Sistem Verifikasi Pembayaran pada Website iPanda. *Jurnal Informatika Upgris*, *4*(2), 246–254. https://doi.org/10.26877/jiu.v4i2.2923

Ivanova, M. (2014). Web Services : Features , Technologies and. *PROCEEDINGS of the International Conference InfoTech*, *November*, 156–165.

Iyer, B., Freedman, J., Gaynor, M., & Wyner, G. (2003). Web Services: Enabling Dynamic Business Networks. *Communications of the Association for Information Systems*, *11*(April). https://doi.org/10.17705/1cais.01130

Jardinia Tanuwidjaja, C., & Setiawan, A. (2017). Perancangan dan Pembuatan Website E-Commerce pada Toko Aksesoris Komputer di Surabaya. *Jurnal Infra*, *5*(2), 301–307.

Marthasar, G. I., Aminudin, & Munarko, Y. (2010). Implementasi Web Service Untuk Mendukung Interoperabilitas Pada Aplikasi E-Commerce. *The 12th Industrial Electronics Seminar, Indonesia*, *2010*(Ies), 978–979.

Sidik, M. (2019). Perancangan dan Pengembangan E-commerce dengan Metode Research and Development. *Jtiust*, *04*(1), 99–107.

Statista, 2020. *eCommerce - Indonesia*. [online] Available at: https://www.statista.com/outlook/243/120/ecommerce/indonesia [Diakses 19 March. 2021].

Sommerville, I., 2011. Software Engineering: Ninth Edition. Boston: Pearson.

Smith, J. G., & Clark, F. E. (1928). Principles of Marketing. In *The Economic Journal* (Vol. 38, Issue 151). https://doi.org/10.2307/2224326

Syah, R., Iqbal, M., & Elveny, M. (2014). Pembangunan Aplikasi E-Commerce Dalam Menunjang Penjualan Produk Pada Cv.Riztech. *Jurnal Penelitian Teknik Informatika*, *4*, 1–80. https://www.academia.edu/8663833/PEMBANGUNAN_APLIKASI_E-COMMERCE_DALAM_MENUNJANG_PENJUALAN_PRODUK_PADA_CV.RIZTECH

Tanuwidjaja, CJ. & Setiawan, A., 2017. *Perancangan dan Pembuatan Website E-Commerce pada Toko Aksesoris Komputer di Surabaya*. Jurnal Infra.

Turban, E., King, D., Lee, J. K., Liang, T.-P., & Turban, D. C. (2015). Electronic Commerce - A Managerial and Social Perspective. In *Springer*. http://www.springer.com/gp/book/9783319100906

Nisrina, Y. E., Putra, W. H. N., & Hanggara, B. T. (2019). Pengembangan E - Commerce Dengan Pemanfaatan Sistem Payment Gateway (Studi Kasus: Wisata Kampung Sapi Adventure). *Jurnal Pengembangan Teknologi Informasi Dan Ilmu Komputer*, *Vol. 3*(10), 9419–9425.

Puspasari, T. M. M., & Maulina, D. (2019). Digitalisasi Pembayaran Marketplace Menggunakan Midtrans Payment Gateway. *Jurnal Mobile and Forensics (MF)*, *1*(1), 22–28.

LAMPIRAN A WAWANCARA TERHADAP PEMILIK CHESSPEDIA

Data Narasumber:

a. Nama: Edi Tanjung
b. Jenis Kelamin: Laki - Laki

1. Sejak dibangun, sudah berapa lama toko ini beroperasi?
   - Untuk jualan caturnya sudah dari tahun 1990, namun untuk penamaan chesspedia dari tahun 2018
2. Bagaimana perkembangan chesspedia ini dari pertama dibuka sampai saat ini?
   - Dari pertama buka sampai saat ini ya cukup berkembang, awal-awal yang beli cuman dari kalangan pecatur malang. Tapi sekarang udah banyak costumer langganan baik itu yang ambil secara personal atau untuk instansi. Saat ini juga sudah ada *website* chesspedia untuk promosi catur saya
3. Bagaimana alur pemesanan yang terjadi pada *website* chesspedia ini?
   - Ya seperti *website* lainnya, ada pembeli lihat lihat barang, kalau cocok bisa dimasukan ke keranjang. Terus kalo *deal* mau transaksi diarahkan untuk melakukan pembayaran lewat *whatsapp*. Kalau sudah melakukan pembayaran akan dilakukan pengiriman pesanan.
4. Bagaimana proses pembayaran yang dilakukan oleh pelanggan?
   - Ya untuk pembayarannya sih, saya pakai transfer bank BNI saja mas. Soalnya adanya bni saja
5. Apakah proses transaksi di chesspedia lancar?
   - Lancar, tapi kebanyakan mengeluh saat whatssap nya lama direspon untuk verifikasi pembayaran
6. Apakah ada kendala yang dihadapi pada saat menjalankan proses bisnis?
   - Ada kendalanya mas, banyak pembeli yang protes karena pembayarannya melalui transfer secara langsung ke saya dan bank yang digunakan hanya 1 jenis bank saja. Lalu banyak yang mengeluh karena proses verifikasi pesanannya lama.
7. Apakah akibat yang ditimbulkan dari permasalahan yang dihadapi?
   - Ya banyak pembeli yang akhirnya tidak jadi membeli karena terbatas dalam hal pembayaran.

8. Apa harapan yang ingin dicapai dari mengatasi permasalahan yang dihadapi?

- Harapannya biar proses transaksi berjalan lancar, dan gak merugikan pembeli. Biar tokonya makin ramai lagi yang beli.

LAMPIRAN B RESPONDEN SUS 1

Lembar Kuesioner System Usability Scale (SUS)
Website E-commerce Chesspedia

Nama: Edi Tanjung

Jawablah pertanyaan berikut dengan memberi satu tanda centang (V) pada setiap pertanyaan pada kolom jawaban yang tersedia. Dengan nilai 1 = sangat tidak setuju (STS), nilai 2 = tidak setuju (TS), nilai 3 = ragu-ragu (RG), nilai 4 = setuju (S), nilai 5 = sangat setuju (SS).

No.	PERTANYAAN	STS	TS	RG	S	SS
1	Saya berpikir akan menggunakan sistem ini lagi.				✓	
2	Saya merasa sistem ini rumit untuk digunakan.		✓			
3	Saya merasa sistem ini mudah digunakan.				✓	
4	Saya membutuhkan bantuan dari orang lain atau teknisi dalam menggunakan sistem ini.			✓		
5	Saya merasa fitur-fitur sistem ini berjalan dengan semestinya.				✓	
6	Saya merasa ada banyak hal yang tidak konsisten (tidak serasi pada sistem ini).		✓			
7	Saya merasa orang lain akan memahami cara menggunakan sistem ini dengan cepat.				✓	
8	Saya merasa sistem ini membingungkan.		✓			
9	Saya merasa tidak ada hambatan dalam menggunakan sistem ini.				✓	
10	Saya perlu membiasakan diri terlebih dahulu sebelum menggunakan sistem ini.				✓	

Malang, 20 Maret 2021

Responden

Edi Tanjung

LAMPIRAN C RESPONDEN SUS 2

Lembar Kuesioner System Usability Scale (SUS)
Website E-commerce Chesspedia

Nama: Lisa Indriyani Iskandar

Jawablah pertanyaan berikut dengan memberi satu tanda centang (V) pada setiap pertanyaan pada kolom jawaban yang tersedia. Dengan nilai 1 = sangat tidak setuju (STS), nilai 2 = tidak setuju (TS), nilai 3 = ragu-ragu (RG), nilai 4 = setuju (S), nilai 5 = sangat setuju (SS).

No.	PERTANYAAN	STS	TS	RG	S	SS
1	Saya berpikir akan menggunakan sistem ini lagi.				✓	
2	Saya merasa sistem ini rumit untuk digunakan.			✓		
3	Saya merasa sistem ini mudah digunakan.				✓	
4	Saya membutuhkan bantuan dari orang lain atau teknisi dalam menggunakan sistem ini.		✓			
5	Saya merasa fitur-fitur sistem ini berjalan dengan semestinya.			✓		
6	Saya merasa ada banyak hal yang tidak konsisten (tidak serasi pada sistem ini).	✓				
7	Saya merasa orang lain akan memahami cara menggunakan sistem ini dengan cepat.				✓	
8	Saya merasa sistem ini membingungkan.		✓			
9	Saya merasa tidak ada hambatan dalam menggunakan sistem ini.				✓	
10	Saya perlu membiasakan diri terlebih dahulu sebelum menggunakan sistem ini.		✓			

Malang, 28 April 2021

Responden

Lisa Indriyani Iskandar

LAMPIRAN D RESPONDEN SUS 3

Lembar Kuesioner System Usability Scale (SUS)
Website E-commerce Chesspedia

Nama: Liska Novita

Jawablah pertanyaan berikut dengan memberi satu tanda centang (V) pada setiap pertanyaan pada kolom jawaban yang tersedia. Dengan nilai 1 = sangat tidak setuju (STS), nilai 2 = tidak setuju (TS), nilai 3 = ragu-ragu (RG), nilai 4 = setuju (S), nilai 5 = sangat setuju (SS).

No.	PERTANYAAN	STS	TS	RG	S	SS
1	Saya berpikir akan menggunakan sistem ini lagi.			✓		
2	Saya merasa sistem ini rumit untuk digunakan.			✓		
3	Saya merasa sistem ini mudah digunakan.					✓
4	Saya membutuhkan bantuan dari orang lain atau teknisi dalam menggunakan sistem ini.		✓			
5	Saya merasa fitur-fitur sistem ini berjalan dengan semestinya.			✓		
6	Saya merasa ada banyak hal yang tidak konsisten (tidak serasi pada sistem ini).		✓			
7	Saya merasa orang lain akan memahami cara menggunakan sistem ini dengan cepat.					✓
8	Saya merasa sistem ini membingungkan.			✓		
9	Saya merasa tidak ada hambatan dalam menggunakan sistem ini.				✓	
10	Saya perlu membiasakan diri terlebih dahulu sebelum menggunakan sistem ini.				✓	

Malang, 26 Maret 2021

Responden

Liska Novita

LAMPIRAN E RESPONDEN SUS 4

Lembar Kuesioner System Usability Scale (SUS)
Website E-commerce Chesspedia

Nama: Luqmanul Hakim

Jawablah pertanyaan berikut dengan memberi satu tanda centang (V) pada setiap pertanyaan pada kolom jawaban yang tersedia. Dengan nilai 1 = sangat tidak setuju (STS), nilai 2 = tidak setuju (TS), nilai 3 = ragu-ragu (RG), nilai 4 = setuju (S), nilai 5 = sangat setuju (SS).

No.	PERTANYAAN	STS	TS	RG	S	SS
1	Saya berpikir akan menggunakan sistem ini lagi.					✓
2	Saya merasa sistem ini rumit untuk digunakan.		✓			
3	Saya merasa sistem ini mudah digunakan.					✓
4	Saya membutuhkan bantuan dari orang lain atau teknisi dalam menggunakan sistem ini.	✓				
5	Saya merasa fitur-fitur sistem ini berjalan dengan semestinya.					✓
6	Saya merasa ada banyak hal yang tidak konsisten (tidak serasi pada sistem ini).			✓		
7	Saya merasa orang lain akan memahami cara menggunakan sistem ini dengan cepat.					✓
8	Saya merasa sistem ini membingungkan.		✓			
9	Saya merasa tidak ada hambatan dalam menggunakan sistem ini.					✓
10	Saya perlu membiasakan diri terlebih dahulu sebelum menggunakan sistem ini.	✓				

Malang, 25 April 2021

Responden

Luqmanul Hakim

LAMPIRAN F RESPONDEN SUS 5

Lembar Kuesioner System Usability Scale (SUS)
Website E-commerce Chesspedia

Nama: M Hafizh Ridho

Jawablah pertanyaan berikut dengan memberi satu tanda centang (V) pada setiap pertanyaan pada kolom jawaban yang tersedia. Dengan nilai 1 = sangat tidak setuju (STS), nilai 2 = tidak setuju (TS), nilai 3 = ragu-ragu (RG), nilai 4 = setuju (S), nilai 5 = sangat setuju (SS).

No.	PERTANYAAN	STS	TS	RG	S	SS
1	Saya berpikir akan menggunakan sistem ini lagi.				✓	
2	Saya merasa sistem ini rumit untuk digunakan.			✓		
3	Saya merasa sistem ini mudah digunakan.					✓
4	Saya membutuhkan bantuan dari orang lain atau teknisi dalam menggunakan sistem ini.			✓		
5	Saya merasa fitur-fitur sistem ini berjalan dengan semestinya.				✓	
6	Saya merasa ada banyak hal yang tidak konsisten (tidak serasi pada sistem ini).		✓			
7	Saya merasa orang lain akan memahami cara menggunakan sistem ini dengan cepat.				✓	
8	Saya merasa sistem ini membingungkan.		✓			
9	Saya merasa tidak ada hambatan dalam menggunakan sistem ini.					✓
10	Saya perlu membiasakan diri terlebih dahulu sebelum menggunakan sistem ini.		✓			

Malang, 22 April 2021

Responden

M Hafizh Ridho

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA