LEMBAGA PENELITIAN DAN PENGABDIAN KEPADA MASYARAKAT
UNIVERSITAS BINA SARANA INFORMATIKA
Alamat : Jl. Kramat Raya No. 98, Senen. Jakarta Pusat 10450
Telp. (021) 23231170 Fax (021) 21236158 Ext. 704, 705 e-mail: lppm@bsi.ac.id

SURAT TUGAS

1068/B.01/LPPM-UBSI/VII/2025

Tentang

Pelindungan Ciptaan di Bidang Ilmu Pengetahuan, Seni dan Sastra Berdasarkan Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta
Nomor dan Tanggal Permohonan : EC002025101640, 30 Juli 2025
Nomor Pencatatan : 000941901

PADA SURAT PENCATATAN CIPTAAN KEMENTERIAN HUKUM DAN HAK ASASI MANUSIA REPUBLIK INDONESIA

Program Komputer

Judul Ciptaan :

RANCANG BANGUN PROGRAM EVENT ORGANIZER BERBASIS DEKSTOP (Studi Kasus Pada BAWOR NETWORK)

MEMUTUSKAN

Pertama : Menugaskan kepada saudara

1. 200904432 Melyani
2. 201007224 Al Ghazali
3. 201803006 Fitri Apriyanti
4. 201809210 Hendra Lesmana
5. 201008302 Popon Rabia Adawia
6. 201203313 Eka Dyah Setyaningsih
7. 202109221 Didin Solehudin
8. 200909601 Wahyu Indrarti

Sebagai Pencipta yang mempublikasikan karyanya.

Kedua : Mempunyai tugas sbb:
Melaksanakan Tugas yang diberikan dengan penuh rasa tanggung jawab.

Ketiga : Keputusan ini berlaku sejak tanggal ditetapkan, dengan ketentuan apabila dikemudian hari terdapat kekeliruan akan diubah dan diperbaiki sebagaimana mestinya.

Jakarta, 23 Juli 2025
LPPM Universitas Bina Sarana Informatika
Ketua

Agus Junaidi, M. Kom

Tembusan
- Ka. LPPM Universitas Bina Sarana Informatika
- Arsip
- Ybs

REPUBLIK INDONESIA
KEMENTERIAN HUKUM

SURAT PENCATATAN CIPTAAN

Dalam rangka pelindungan ciptaan di bidang ilmu pengetahuan, seni dan sastra berdasarkan Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta, dengan ini menerangkan:

Nomor dan tanggal permohonan	:	EC002025101640, 30 Juli 2025
Pencipta		
Nama	:	**Melyani, Al ghazali dkk**
Alamat	:	Jl. Bunga Rampai VII gg.7 Rt.010 Rw.006 N0.20 Perumnas Klender, Duren Sawit, Kota Adm. Jakarta Timur, DKI Jakarta, 13860
Kewarganegaraan	:	Indonesia
Pemegang Hak Cipta		
Nama	:	**Melyani, Al ghazali dkk**
Alamat	:	Jl. Bunga Rampai VII gg.7 Rt.010 Rw.006 N0.20 Perumnas Klender, Duren Sawit, Kota Adm. Jakarta Timur, DKI Jakarta, 13860
Kewarganegaraan	:	Indonesia
Jenis Ciptaan	:	**Program Komputer**
Judul Ciptaan	:	**RANCANG BANGUN PROGRAM EVENT ORGANIZER BERBASIS DEKSTOP (Studi Kasus Pada BAWOR NETWORK)**
Tanggal dan tempat diumumkan untuk pertama kali di wilayah Indonesia atau di luar wilayah Indonesia	:	30 Juli 2025, di Kota Adm. Jakarta Timur
Jangka waktu pelindungan	:	Berlaku selama 50 (lima puluh) tahun sejak Ciptaan tersebut pertama kali dilakukan Pengumuman.
Nomor Pencatatan	:	000941901

adalah benar berdasarkan keterangan yang diberikan oleh Pemohon.
Surat Pencatatan Hak Cipta atau produk Hak terkait ini sesuai dengan Pasal 72 Undang-Undang Nomor 28 Tahun 2014 tentang Hak Cipta.

a.n. MENTERI HUKUM
DIREKTUR JENDERAL KEKAYAAN INTELEKTUAL
u.b
Direktur Hak Cipta dan Desain Industri

Agung Damarsasongko,SH.,MH.
NIP. 196912261994031001

Balai Besar Sertifikasi Elektronik

Disclaimer:
1. Dalam hal pemohon memberikan keterangan tidak sesuai dengan surat pernyataan, Menteri berwenang untuk mencabut surat pencatatan permohonan.
2. Surat Pencatatan ini telah disegel secara elektronik menggunakan segel elektronik yang ditertibkan oleh Balai Besar Sertifikasi Elektronik, Badan Siber dan Sandi Negara.
3. Surat Pencatatan ini dapat dibuktikan keasliannya dengan memindai kode QR pada dokumen ini dan informasi akan ditampilkan dalam browser.

LAMPIRAN PENCIPTA

No	Nama	Alamat
1	Melyani	Jl. Bunga Rampai VII gg.7 Rt.010 Rw.006 N0.20 Perumnas Klender Duren Sawit, Kota Adm. Jakarta Timur
2	Al ghazali	Kp. Kaliulu rt 01 rw 01 karangraharja cikarang utara bekasi 17530 Cikarang Utara, Kab. Bekasi
3	Fitri Apriyanti	Kp. Slipi jl E1 No. 36 RT 001/02 kelurahan slipi kecamatan palmerah jakarta barat 11410 Pal Merah, Kota Adm. Jakarta Barat
4	Hendra Lesmana	Ds/Kel Kedunguter RT/RW 02/01 Kec/Kab Brebes Kode pos 52219 Jawa Tengah Brebes, Kab. Brebes
5	Popon Rabia Adawia	jl. Kedasih 1 B1/66, Cikarang Baru, Mekarmukti, Cikarang Utara. Cikarang Utara, Kab. Bekasi
6	Eka Dyah Setyaningsih	Jl.Pepaya Raya No 19A RT03/RW 05 Jagakarsa Jakarta Selatan 12620. Jagakarsa, Kota Adm. Jakarta Selatan
7	Didin Solehudin	Perumahan graha selaras blok D7/1. Jl.M sanun RT 07/08, Harapan jaya, Cibinong Bogor 16914 Cibinong, Kab. Bogor

LAMPIRAN PEMEGANG

No	Nama	Alamat
1	Melyani	Jl. Bunga Rampai VII gg.7 Rt.010 Rw.006 N0.20 Perumnas Klender Duren Sawit, Kota Adm. Jakarta Timur
2	Al ghazali	Kp. Kaliulu rt 01 rw 01 karangraharja cikarang utara bekasi 17530 Cikarang Utara, Kab. Bekasi
3	Fitri Apriyanti	Kp. Slipi jl E1 No. 36 RT 001/02 kelurahan slipi kecamatan palmerah jakarta barat 11410 Pal Merah, Kota Adm. Jakarta Barat
4	Hendra Lesmana	Ds/Kel Kedunguter RT/RW 02/01 Kec/Kab Brebes Kode pos 52219 Jawa Tengah Brebes, Kab. Brebes
5	Popon Rabia Adawia	jl. Kedasih 1 B1/66, Cikarang Baru, Mekarmukti, Cikarang Utara. Cikarang Utara, Kab. Bekasi
6	Eka Dyah Setyaningsih	Jl.Pepaya Raya No 19A RT03/RW 05 Jagakarsa Jakarta Selatan 12620. Jagakarsa, Kota Adm. Jakarta Selatan
7	Didin Solehudin	Perumahan graha selaras blok D7/1. Jl.M sanun RT 07/08, Harapan jaya, Cibinong Bogor 16914 Cibinong, Kab. Bogor

Menu dan Paduan Penggunaan
RANCANG BANGUN PROGRAM EVENT ORGANIZER BERBASIS DEKSTOP

(Studi Kasus Pada BAWOR NETWORK)

BAB I

PENDAHULUAN

1.1 Latar Belakang

Proses penginputan data pesanan paket pada Event Organizer masih menghadapi sejumlah tantangan yang signifikan. Saat ini, penginputan data dilakukan secara manual dengan menulis tangan di buku pesanan, yang sering kali mengakibatkan tingkat kesalahan yang cukup tinggi. Kesalahan ini dapat terjadi akibat tulisan yang kurang jelas, kesalahan pencatatan, atau kehilangan catatan penting. Selain itu, metode manual ini memakan waktu yang lama, karena staf harus mencatat setiap detail pesanan secara individual. Proses yang tidak efisien ini tidak hanya memperlambat operasional, tetapi juga meningkatkan risiko terjadinya kesalahan yang dapat berdampak pada kepuasan klien.

Salah satu kelemahan yang perlu dipertimbangkan. adalah ketergantungan yang tinggi pada konektivitas internet. Jika terjadi gangguan pada jaringan internet, akses terhadap sistem dapat terganggu, yang dapat mempengaruhi operasional Event Organizer, terutama ketika membutuhkan akses data secara real-time. Selain itu, keamanan data menjadi isu penting, karena data klien dan informasi sensitif lainnya disimpan secara online, yang rentan terhadap ancaman siber seperti peretasan dan pencurian data.

Oleh karen itu, Sistem *Event Organizer* berbasis *web* merupakan langkah strategis untuk meningkatkan efisiensi dan akurasi dalam

pengelolaan data pesanan. Sistem berbasis *web* memungkinkan penginputan

data dilakukan secara digital, yang tidak hanya mengurangi risiko kesalahan manusia, tetapi juga mempercepat proses pencatatan dan akses informasi. Dengan sistem ini, data pesanan dapat disimpan dan dikelola secara terpusat, memungkinkan akses *real-time* oleh seluruh tim *Event Organizer* kapan saja dan di mana saja, selama terkoneksi dengan internet.

Sistem berbasis web dirancang untuk mengatasi masalah-masalah ini dengan menyediakan platform yang lebih efisien dan terintegrasi, yang dapat mengurangi risiko kesalahan pencatatan dan kehilangan data. Selain itu, sistem ini menawarkan solusi modern yang sesuai dengan tren teknologi saat ini, memungkinkan Bawor Netwrokuntuk meningkatkan layanan mereka dengan memberikan akses cepat dan mudah ke informasi. Penggunaan teknologi ini diharapkan tidak hanya memperbaiki operasional internal, tetapi juga meningkatkan kepuasan pelanggan melalui layanan yang lebih responsif dan terpersonalisasi. Dengan demikian, proyek ini juga memberikan peluang untuk inovasi dan pengembangan lebih lanjut, yang dapat menjadi model bagi Event Organizer lainnya di industri.

Berdasarkan uraian di atas maka penulis membangun aplikasi *web Event Organizer*. Dari penelitian ini, penulis mengangkat judul "Perancangan Sistem Informasi *wedding organazer* berbasis *web* pada Bawor Network".

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan, rumusan masalah dalam penelitian ini adalah: "Bagaimana merancang Sistem informasi *Event Organizer* berbasis *web*?"

1.3 Batasan Masalah

Berdasarkan rumusan masalah diatas, penulis membatasi permasalahan pada *Sistem* ini akan difokuskan pada layanan pemesanan untuk acara pernikahan. Ini termasuk desain dekorasi, penentuan tema, dan proses pemesanan, Perancangan Sistem informasi *Event Organizer* yang dirancang berbasis *web*, memungkinkan akses melalui berbagai perangkat dengan koneksi internet, seperti komputer desktop, laptop, dan perangkat mobile, Sistem ini ditujukan untuk digunakan oleh pemilik usaha dekorasi, tim desain, dan pelanggan yang mencari layanan dekorasi untuk acara mereka.

1.4 Tujuan Penelitian

Tujuan dari penelitian ini adalah dengan Sistem informasi *Event Organizer* berbasis *web* ini dapat memberikan kontribusi yang signifikan dalam mempermudah proses perencanaan dan pelaksanaan serta meningkatkan pengalaman pengguna dalam mengatur acara spesial mereka.

1.5 Manfaat Penelitian

Adapun manfaat dari penelitian ini adalah sebagai berikut:

1. Pelanggan dapat dengan mudah mengakses informasi tentang layanan yang ditawarkan oleh pemilik usaha melalui platform *web*, yang dapat diakses kapan saja dan dari mana saja melalui perangkat yang terhubung ke internet.
2. Pelanggan memiliki kesempatan untuk berinteraksi langsung dengan tim desain untuk menyesuaikan desain dekorasi sesuai dengan preferensi dan kebutuhan acara mereka, sehingga mereka dapat memiliki pengalaman yang lebih personal dan unik.
3. Keseluruhan, penggunaan Sistem informasi *Event Organizer* berbasis *web* ini dapat meningkatkan pengalaman pelanggan dengan menyediakan layanan yang lebih mudah diakses, personalisasi yang lebih baik, dan proses yang lebih efisien dan transparan.

1.6 Sistematika Penulisan

Sistematika penulisan dalam laporan Praktek Kerja Lapangan ini terbagimenjadi beberapa bab yang akan dibahas sebagai berikut:

BAB I PENDAHULUAN

Bab ini membahas cara penulisan laporan, yang terdiri dari beberapa sub bab: latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan Sistematika penulisan.

BAB II TINJAUAN PUSTAKA

Bab ini menjelaskan konsep dasar dan pengertian yang terkait dengan penelitian yang dilakukan, serta teori-teori dasar baik secara umum maupun khusus.

BAB III METODOLOGI PENELITIAN

Bab ini menjelaskan deskripsi organisasi, visi dan misi,serta analisis Sistem yang sedang berjalan, metode pengumpulan data, Sistem usulan, kerangka berfikir.

BAB IV HASIL DAN PEMBAHASAN

Bab ini menjelaskan hasil yang diperoleh dari penelitian serta membahas Sistem yang telah dibangun.

BAB V KESIMPULAN DAN SARAN

Bab ini menyajikan beberapa kesimpulan dari pembahasan masalah di bab-bab sebelumnya dan memberikan saran.

BAB II

TINJAUAN PUSTAKA

2.1 Tinjauan Pustaka

Tinjauan pustaka atau penelitian terdahulu tentang *Sistem* informasi *Event Organizer* seperti pada penelitian Riyani & Hendradi (2016) dalam Jurnal Ilmiah Fakultas Teknik LIMIT'S Vol.12 No.2, 67-74 dengan judul "Sistem Informasi Pemesanan Paket Pernikahan Pada *Inspiration Beauty* Sanggar Rias Indah" ingin membangun Sistem berbasis web, dimana *Sistem* ini berfungsi untuk memberikan layanan informasi paket pernikahan. Penulis mengembangkan Sistem berbasis web menggunakan database MySql, Codding PHP dan bahasa pemograman HTML. Penelitian Chafid & Mulyawan (2017) jurnal Satya Informatika Vol. 2 No. 1, Mei 2017 Halaman 36-51 dengan judul "Aplikasi *Location Based Service Event Organizer* di Kota Tanggerang Selatan Berbasis Android" Penelitian ini membangun aplikasi berbasis Android yang dapat menentukan *Event Organizer* dan jarak yang terdekat berdasarkan lokasi pengguna. Penelitian Aulianita (2019) dengan judul "*User Center Design* dalam Membangun *Event Organizer* Berbasis *Web*site" *Web*site Event Organizer yang akan memudahkan user dalam mencari informasi sesuai keinginan mereka dengan fitur yang *userfriendly*. Dengan banyaknya vendor yang terlibat dalam portal wedding ini, maka akan membuatdaya saing meningkat dan kompetitif. Penelitian Agustin et al., (2020) Vol.2, No.2, Vol 2020, pp. 15 – 24, dengan judul "Perancangan Sistem Informasi Jasa Event Organizer

pada CV. Boganesia Jaya Berbasis *Web*" Sistem ini dirancang dengan menggunakan pemodelan UML. Dengan pengujian yang bertipe memperlakukansebuah perangkat lunak yang tidak diketahuikinerja dan spesifikasi fungsional didalamnya Sedangkan bahasa pemrograman yang digunakan adalah PHP, serta database My SQL.

Bab ini membandingkan beberapa penelitian mengenai *Sistem* informasi Event Organizer dengan Sistem informasi yang telah dikembangkan oleh peneliti sebelumnya. Perbedaan dalam penelitian ini adalah pembuatan *Sistem* Informasi *Event Organizer* berbasis *web* menggunakan metode pengembangan *prototype*. Sistem yang akan dibangun terdiri dari fitur untuk pelanggan, yang memungkinkan mereka melihat paket-paket yang ditawarkan dan memesan layanan Event Organizer lainnya melalui web. Sedangkan dari sisi Admin, Sistem ini dapat mengelola data pemesanan, data pelanggan, serta data transaksi pemesanan.

2.2 Teori yang Berkaitan dengan Penelitian

2.2.1 Sistem

Menurut Widarti et al., (2024, p. 3), "Sistem adalah kombinasi antara komputer dan pengguna yang bekerja bersama untuk melakukan operasi, manajemen, analisis, dan pengambilan keputusan dalam sebuah organisasi guna mencapai tujuan tertentu."

Menurut Ridwan et al., (2021, p. 19), "Sistem adalah landasan bagi semua aktivitas. Kehadiran Sistem sangat diperlukan di setiap bidang,

karena tanpa konsep Sistem, kegiatan atau pekerjaan akan berlangsung tanpa kendali."

Berdasarkan penjelasan dari para ahli di atas, dapat disimpulkan bahwa Sistem adalah jaringan yang saling berhubungan dan bekerja sama dalam melaksanakan suatu operasi serta menjadi dasar bagi semua aktivitas.

2.2.2 Informasi

Menurut Prehanto (2020, p. 12), "Informasi adalah hasil dari pengolahan data dengan cara tertentu, sehingga menjadi lebih bermakna dan bermanfaat bagi penerimanya."

Menurut Ruthven dan Kelly (dalam Sugiyanto et al., 2022, p. 1), "Informasi didefinisikan sebagai data yang telah diolah menjadi bentuk yang memiliki arti bagi penerimanya dan berguna dalam pengambilan keputusan baik saat ini maupun di masa depan."

2.2.3 Sistem Informasi

Menurut Andi “Sistem informasi merupakan suatu kombinasi teratur dari orang-orang, *hardware,software,* jaringan komunikasi dan sumber daya data yang mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi.

Menurut (Arifin et al., 2022, p. 12) "Sistem informasi dapat diartikan sebagai Sistem yang dirancang oleh manusia, yang mencakup berbagai komponen dalam organisasi untuk mencapai tujuan, yaitu menghasilkan informasi."

2.2.4 Event Organizer

Menurut Sumarsono (dalam Salsabila et al., 2024) *Event Organizer* merupakan sebuah lembaga atau badan yang secara khusus melayani jasa. Secara pribadi atau tim membantu mempersiapkan segala sesuatu yang behubungan dengan acara agar berjalan lancar sesuai dengan yang diinginkan.

Sistem informasi Event Organizer sangat memudahkan bagi mereka yang tidak ingin repot dengan perencanaan pernikahan. Dengan kemajuan teknologi saat ini, Event Organizer mulai beralih dari proses manual ke proses online, yaitu melalui Sistem informasi Event Organizer berbasis web.

2.2.5 Konsep Dasar Internet

Menurut Syafrizal (2020, p. 98), “Internet sendiri didefinisikan sebagai sebuah jaringan komputer yang menggunakan *Protocol* Internet (TCP/IP) yang digunakan untuk berkomunikasi dan berbagi informasi dalam lingkup tertentu.”

Menurut Comer (2019, p. 4), "Internet adalah Sistem komunikasi komputer global yang memungkinkan berbagai layanan. Secara singkat, internet telah memicu revolusi yang mengubah cara kita hidup, bekerja, dan bersenang-senang."

*2.2.6 Web*site

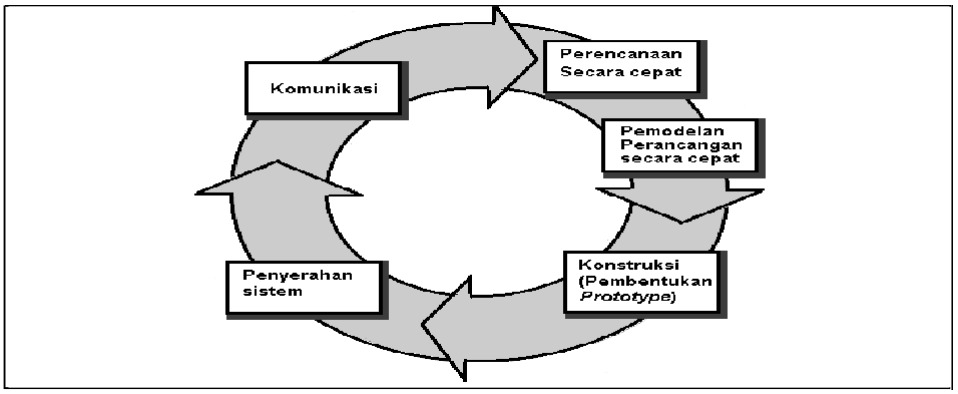
Menurut Saad (2020, p. 5) "*Web*site adalah merupakan kumpulan file yang terletak pada komputer yang terhubung ke internet"

Dalam definisi lain yang dikemukakan oleh Abdullah dalam Sa'ad (2020, p. 3), "website atau web adalah kumpulan halaman yang terdiri dari beberapa laman yang berisi informasi dalam format digital, seperti teks, gambar, video, audio, dan animasi lainnya, yang dapat diakses melalui koneksi internet."

2.2.7 Metode Pengembangan Sistem (*Prototype*)

Menurut Hendri et al., (2022) Model *prototype* ialah sebuah metode yang mengharuskan pengembang perangkat lunak membuat sebuah mockup berupa model aplikasi, sangat cocok pada kondisi dimana pengguna tidak bisa menyajikan informasi secara jelas mengenai kebutuhan yang sesuai dengan keinginannya.

Bagan mengenai *prototype* model dapat dilihat pada gambar 2.1 berikut :



Sumber : Pressman, 2020:51

Gambar 2. 1 Model *Prototype*

Dalam penelitian dengan judul "Perancangan Sistem Informasi Event Organizer Berbasis Web pada PT Vijendra Project," penggunaan metode prototyping sangat relevan karena beberapa alasan:

1. **Pemahaman yang Lebih Baik Terhadap Kebutuhan Pengguna:** Dalam konteks perancangan sistem informasi Event Organizer, kebutuhan pengguna bisa sangat spesifik dan bervariasi. Dengan menggunakan prototyping, tim pengembang bisa mendapatkan gambaran yang lebih jelas tentang apa yang diinginkan oleh pengguna, termasuk fitur-fitur yang paling penting.
2. **Umpan Balik yang Cepat dan Iteratif:** Event Organizer memiliki proses kerja yang dinamis, dan kebutuhan bisa berubah seiring waktu. Prototyping memungkinkan pengguna untuk melihat bentuk awal dari sistem dan memberikan umpan balik yang cepat. Umpan balik ini bisa digunakan untuk melakukan penyesuaian dan peningkatan secara iteratif.
3. **Pengujian Fungsionalitas Khusus:** Fitur-fitur khusus seperti manajemen jadwal, pengelolaan vendor, atau pelacakan anggaran dapat diuji lebih awal melalui prototipe. Ini membantu memastikan bahwa sistem memenuhi kebutuhan operasional Event Organizer dengan efektif sebelum pengembangan penuh dilakukan.
4. **Mengurangi Risiko Kegagalan:** Dengan adanya prototipe, risiko bahwa sistem yang dibangun tidak sesuai dengan harapan pengguna bisa diminimalkan. Setiap potensi masalah bisa diidentifikasi dan diperbaiki pada tahap awal.

5. **Fleksibilitas dalam Pengembangan:** Jika ada perubahan dalam kebutuhan atau tujuan bisnis, metode prototyping memungkinkan tim untuk menyesuaikan desain dan fungsionalitas sistem tanpa harus mengubah seluruh sistem dari awal.

2.3 Konsep Dasar Bahasa Pemrograman

2.3.1 PHP (Hypertext Perprocessor)

Menurut Oetomo & Mahargiono (2020, p. 1), "*PHP* merupakan bahasa pemrograman yang digunakan secara luas untuk penanganan, pembuatan, dan pengembangan sebuah situs *web* dan biasanya digunakan bersama dengan *HTML*".

Menurut Miftachudin (2022, p. 5), "*Hypertext Preprocessor* adalah Bahasa server-*side scripting* yang menyatu dengan *HTML* untuk membuat halaman *web* yang dinamis".

2.3.2 HTML

Menurut Devi (dalam Arthantio, 2022, p. 25), "*Hypertext Markup Language* (*HTML*) adalah sebuah bahasa markah yang digunakan untuk membuat sebuah halaman *web*, menampilkan berbagai informasi dalam sebuah penjelajah *web* internet dan memformat hiperteks sederhana yang ditulis dalam berkas format ASCII agar dapat menghasilkan tampilan wujud yang terintegrasi".

Menurut Muhardian (dalam Limbong, 2021, p. 19), "*Hypertext Mark Up Language* adalah bahasa standar pemrograman untuk membuat suatu *web*site yang bisa diakses dengan internet. Dengan kata lain halaman *web*site yang dilihat dan dibaca disusun dengan menggunakan bahasa ini dan kemudian diterjemahkan oleh komputer agar dapat dipahami oleh penggunanya".

2.3.3 CSS

Menurut Saad (2020, p. 29), "*CSS* adalah bahasa *style sheet* yang digunakan untuk mengatur tampilan dokumen. Dengan adanya *CSS*, programmer dapat menampilkan halaman yang sama dengan format yang berbeda".

Menurut Faisal & Abadi (2020, p. 116), "*Cascading Style Sheets* (*CSS*) adalah bahasa yang digunakan (untuk) memberikan *style* pada halaman *web* yang ditulis dengan kode *HTML. CSS* menentukan bagaimana elemen *HTML* ditampilkan yang meliputi bentuk, warna, dan posisi suatu tag atau elemen *HTML*".

2.3.4 Javascript

Menurut Kadir (dalam Sa'ad, 2020, p. 31), "*Javascript* adalah kode untuk menyusun halaman *web* yang memungkinkan pada sisi klien. *Javascript* adalah bahasa yang digunakan agar dokumen *HTML* yang ditampilkan dalam browser menjadi lebih interaktif".

Menurut Azis (2019, p. 15), “Javascript adalah bahasa yang digunakan untuk membuat program yang digunakan agar dokumen *HTML* yang ditampilkan dalam browser menjadi lebih interaktif, tidak hanya indah saja. *Javascript* memberikan fungsionalitas ke dalam suatu halaman *web* sehingga dapat menjadi sebuah program yang disajikan dengan antarmuka pada *web*”.

2.3.5 UML

Menurut Andi: “*Unified Modeling Language (UML)* adalah bahasa pemodelan yang digunakan untuk menspesifikasikan, memvisualisasikan, membuat, dan mendokumentasi artefak Sistem perangkat lunak baik yang sedang dirancang ataupun dikembangkan”.

Menurut Munawar (2021, p. 49): “*UML* adalah salah satu alat bantu yang sangat handal di dunia pengembangan Sistem yang berorientasi obyek, karena *UML* menyediakan bahasa pemodelan visual yang memungkinkan bagi pengembang Sistem untuk membuat cetak biru atas visi mereka dalam bentuk yang baku, mudah dimengerti, serta dilengkapi dengan mekanisme yang efektif untuk berbagi (*sharing*) dan mengkomunikasikan rancangan mereka dengan yang lain.

Berdasarkan definisi di atas, maka dapat disimpulkan bahwa *UML* (*Unified Modelling Languange*) merupakan suatu bahasa yang sudah menjadi standar pada visualisasi, perancangan dan juga pendokumentasian

Sistem software. Jenis – jenis atau komponen dalam membuat suatu diagram *UML*, yaitu:

A. *Use Case Diagram*

Menurut Ahmad (2020) Use Case diagram adalah suatu urutan interaksi yang saling berkaitan antara Sistem dan aktor. Use case dijalankan melalui cara menggambarkan tipe interaksi antara user suatu program (Sistem) dengan Sistemnya sendiri. Use case melalui sebuah cerita yang mana sebuah Sistem itu dipakai. Use case juga dipakai untuk membentuk perilaku (*behaviour*) Sistem yang akan dibuat. Sebuah use case menggambarkan sebuah interakasi antara pengguna (aktor) dengan Sistem yang sudah ada.

B. *Activity Diagram*

Menurut Simatupang & Sianturi (2019) Diagram aktivitas atau activity diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah Sistem atau proses bisnis atau menu yang ada pada perangkat lunak.

C. *Class Diagram*

Menurut Nugroho & Primadewi (2020) Class Diagram adalah gambar yang menjelaskan struktur dari program yang akan dibuat menggunakan konsep *OOP* (*Object Oriented Programming*). Class diagram menggambarkan bagaimana objek pada dunia nyata digambarkan pada struktur yang biasa memiliki atribut dan method.

D. Sequence Diagram

Menurut Nugroho (2020) Sequence diagram adalah grafik dua dimensi dimana obyek ditunjukkan dalam dimensi horizontal, sedangkan lifeline ditunjukkan dalam dimensi vertikal.

2.4 Perangkat Lunak yang Digunakan

2.4.1 XAMPP

Menurut Devi (dalam Arthantio, 2022) "*XAMPP* adalah sebuah *software* (perangkat lunak) untuk menginstal atau memasang localhost pada PC atau Laptop".

Menurut Fathoroni et al., (2020, p. 50), "*XAMPP* adalah perangkat lunak bebas, yang mendukung banyak Sistem operasi, merupakan kompilasi dari beberapa program. *XAMPP* merupakan tool yang menyediakan paket perangkat lunak ke dalam satu buah paket. Dengan menginstal *XAMPP*, maka user tidak perlu lagi untuk menginstal *web server Apache*, *PHP*, dan *MySQL* secara manual".

2.4.2 Fitur *XAMPP*

Menurut Fathoroni et al., (2020, p. 51–52) *XAMPP* merupakan aplikasi yang memiliki banyak fitur yang berguna untuk membantu user dalam mengembangkan aplikasi *web*. Beberapa fitur tersebut diantaranya:

a. *Apache*

Apache adalah perangkat lunak *open source* yang menjadi alternatif untuk server *web Netscape*. Server ini beroperasi pada berbagai *Sistem* operasi dan berfungsi untuk melayani serta menjalankan situs *web*.

b. *MySQL*

Ini adalah singkatan dari "*My Structured Query Language*." Program ini berperan sebagai server yang menawarkan akses multi-pengguna ke berbagai database. Biasanya, digunakan oleh perangkat lunak gratis yang membutuhkan fitur lengkap untuk manajemen database, seperti *WordPress, phpBB*, dan lain-lain.

2.4.3 Microsoft Visual Studio Code

Menurut Salamah (2021, p. 1), "*Visual Studio Code* adalah sebuah teks editor ringan dan andal yang dibuat oleh Microsoft untuk *Sistem* operasi *multiplatform*, artinya tersedia juga untuk versi Linux, Mac, dan Windows. Teks editor ini secara langsung mendukung *Javascript, Typescript*, dan *Node.js* serta bahasa lain dengan bantuan plugin yang dapat dipasang via marketplace seperti C++, C#, *Python*, GO, Java, dll.)".

Menurut Speight (2021, p. 2), "Visual Studio *Code* merupakan kode editor bersifat gratis, *open-source*, dan *cross-platform* yang dikembangkan oleh Microsoft. Aplikasi ini tidak hanya baik untuk mengedit *source code*,

tapi juga terdapat dukungan bawaan untuk berkolaborasi dengan sesama pengguna dan lingkungan yang *cloudhosted*".

BAB III

METODOLOGI PENELITIAN

3.1 Lokasi Penelitian

Dalam proses penyusunan skripsi ini penulis melakukan penelitian di kantor PT.Vijendra Project yang dilakukan pada:

Nama Institusi : PT.Vijendra Project

Alamat Perusahaan : 88 Office Tower Kota Kasablanka, Lantai 26

JL. Casablanka Raya Kav. 88 Jakarta Selatan 12870

Telepon Perusahaan : 0812 1857 7511

E-mail Perusahaan : marketing.vijendra@gmail.com

marketing@vijendraacademy.com

3.1.1 Deskripsi Perusahaan

PT.Vijendra Project merupakan suatu bentuk usaha keluarga yang bergerak di bidang jasa penyediaan pelaksanaan. Usaha ini didirikan pertama kali oleh Ibu Vivi Bahagianti yang merupakan pemilik usaha saat ini. Ada juga PT.Vijendra Academy yang bergerak di bidang Training & Consulting yang sekarang dikelola oleh suami dari Ibu Vivi Bahagianti yaitu Bapak Afifudin. 2 bisnis yang dijalankan dimulai dari tahun 2022 yang dimana sudah berjalan 2 tahun.

Dengan keahlian dan pengalaman yang dimiliki oleh tim kami, kami siap untuk menghadirkan acara-acara yang tak terlupakan dan sesuai dengan harapan klien kami. Kami memahami betapa pentingnya setiap momen dalam sebuah acara, dan itulah mengapa kami bekerja keras untuk merancang dan melaksanakan setiap detail dengan teliti dan penuh perhatian. Keunggulan kami terletak pada dedikasi kami untuk memberikan pelayanan yang ramah, responsif, dan profesional kepada setiap klien. Kami percaya bahwa kolaborasi yang baik antara kami dan klien adalah kunci keberhasilan dalam penyelenggaraan acara yang sukses dan berkesan.

Dengan semangat yang segar dan tekad untuk menjadi mitra terpercaya dalam penyelenggaraan acara, Bawor Netwroksiap untuk membantu mewujudkan setiap impian acara klien kami dengan cemerlang dan sempurna.

Gambar 3. 1 Logo Bawor Netwrok

3.1.2 Visi & Misi Perusahaan

a. Visi

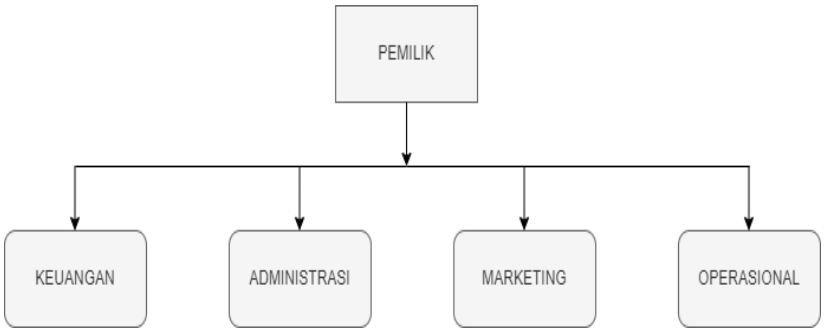
Kami bermimpi untuk mengubah setiap acara menjadi pengalaman yang luar biasa bagi setiap individu dan perusahaan yang kami layani. Melalui kreativitas, inovasi, dan pelayanan yang unggul,

kami bertujuan untuk memenuhi setiap kebutuhan dan harapan klien kami dengan melebihi ekspektasi mereka.

b. Misi

1. Senantiasa berupaya untuk selalu berkomitmen terhadap prinsip-prinsip Etika dalam seluruh Aspek Operasional Perusahaan.
2. Memiliki Sistem Perusahaan yang BerTakwa, Loyalitas dan Totalitas terhadap perusahaan, Amanah, Professional.
3. Inovatif serta Progresif dalam bekerja untuk menghasilkan Pelayanan sempurna disetiap Kegiatan.

3.1.3 Struktur Organisasi



Gambar 3. 2 Struktur Organisasi Bawor Network

Tugas dan wewenang pada struktur organisasi Bawor Network sebagai berikut:

1) Pemilik

Pemilik *Event Organizer* harus memiliki kreativitas untuk membantu memilih tema dan konsep acara yang sesuai dengan keinginan mereka.

mengawasi jalannya bisnis *Event Organizer* dan mengambil keputusan strategis yang diperlukan untuk meningkatkan usaha.

mengelola operasional usaha *Event Organizer*, termasuk menentukan jadwal, menyusun tim, dan mengatur peralatan dan lokasi.

2) Keuangan

a. membuat rencana anggaran yang menyediakan biaya untuk mengelola usaha *Event Organizer*, termasuk biaya untuk menyediakan jasa , peralatan, lokasi, dan karyawan.

b. harus mengatur penerimaan dari klien, termasuk membuat *invoice*, mengirimkan faktur, dan menyusun laporan pendapatan.

c. harus membuat laporan keuangan yang mencakup pendapatan, pengeluaran, dan keseimbangan keuangan.

3) Administrasi

a. membuat jadwal acara yang rincih, termasuk resepsi, acara makan-makan, dan acara lainnya.

b. harus berkoordinasi dengan pihak yang terlibat dalam acara, seperti toko bunga, restoran untuk konsumsi, fotografer, musisi, dan lainnya.

c. menjadwalkan pertemuan dengan vendor dan klien untuk membahas detail spesifik tentang acara.

4) Marketing

a. Membuat strategi pemasaran yang efektif untuk menarik klien ke perusahaan anda. Strategi ini mungkin termasuk pemasaran online, pemasaran offline, atau kombinasi dari kedua.

b. arus membuat iklan yang menarik dan efektif untuk menarik klien ke perusahaan anda. Iklan ini mungkin termasuk iklan online, iklan offline, atau kombinasi dari kedua.

c. mengatur pemasaran secara offline, termasuk membuat iklan offline, membuat brosur, dan menyusun event promosi.

5) Operasional

a. Menyusun tema dan konsep acara yang sesuai dengan keinginan klien.

b. harus berkoordinasi dengan pihak vendor yang dipesan oleh klien.

c. Menyiapkan susunan acara, termasuk pengonsepan, konsumsi, stopper, penerima tamu, perlengkapan, dan lainnya.

3.2 Metode Pengumpulan Data

Pengumpulan data merupakan Langkah awal untuk merancang Solusi dari permasalahan yang relevan dengan tema penelitian, metode pengumpulan data yang digunakan pada penelitian ini antara lain :

A. Studi Literatur

Merupakan metode pengumpulan data dengan menggali informasi dari berbagai sumber referensi maupun literasi seperti

buku, karya ilmiah dan jurnal digital lainnya. Melalui metode ini diharapkan penulis memperoleh pengetahuan dan pemahaman mengenai gambaran penelitian yang akan dilakukan.

B. Studi Lapangan

1. Observasi

   Dalam perancangan ini peneliti melakukan pengamatan langsung terhadap sistem *Event Organizer* yang berlaku di Bawor Network.

2. Wawancara

   Mengadakan wawancara dengan staf dan bagian administrasi tentang masalah sistem *Event Organizer*.

3. Studi Pustaka

   Dalam proses pengumpulan data, penulis mengakses informasi dari studi pustaka atau referensi yang mencakup teori dan pengetahuan terkait dengan skripsi atau tugas akhir yang membahas tentang sistem *Event Organizer* berbasis *web*. Data ini diperoleh melalui pembacaan jurnal penelitian, skripsi, dan sumber-sumber internet.

Adapun data daftar jenis dan nama produk yang diperoleh yaitu:

Tabel 3. 1 Paket Event Organizer

No	Nama	Harga	Deskripsi
1.	Paket Ekslusif	Rp 80.000.000 - Rp 100.000.000	Semua Layanan Paket Premium, ditambah: - Manajer acara yang berdedikasi, - Koordinator tamu, - Pengaturan akomodasi untuk tamu, - Layanan purna acara (misalnya, koordinasi honeymoon).
2.	Paket Premium	Rp 40.000.000 - Rp 60.000.000	Konsultasi Pernikahan: - Sesi konsultasi tidak terbatas, - Panduan perencanaan dan - jadwal pernikahan, - Konsultasi dengan vendor. Manajemen Anggaran: - Penyusunan dan - pengelolaan anggaran pernikahan. Koordinasi Hari-H: - Koordinasi upacara pernikahan, - Koordinasi resepsi. Dekorasi: - Tema dekorasi kustom,

			- Bunga untuk acara, - Dekorasi eksklusif sesuai permintaan. Logistik: - Koordinasi transportasi tamu, - Penanganan tamu. Catering: - Konsultasi menu, - Uji coba makanan. Hiburan: - Penyediaan band atau DJ, - Pengaturan panggung dan suara. Staf: - 1 koordinator utama, - 6 asisten.
3.	Paket Lengkap	Rp 20.000.000 - Rp 30.000.000	Konsultasi Pernikahan: - 5 sesi konsultasi, - Panduan perencanaan dan - jadwal pernikahan, Konsultasi dengan vendor. Manajemen Anggaran: - Penyusunan dan - pengelolaan anggaran pernikahan. Koordinasi Hari-H:

			- Koordinasi upacara pernikahan, - Koordinasi resepsi. Dekorasi: - Tema dekorasi dasar, - Bunga untuk acara. Staf: - 1 koordinator utama, - 4 asisten.
4.	Paket Dasar	Rp 10.000.000 - Rp 15.000.000	Konsultasi Pernikahan: - 3 sesi konsultasi, - Panduan perencanaan, - dan jadwal pernikahan. Koordinasi Hari-H: - Koordinasi upacara pernikahan Koordinasi resepsi. Dekorasi : - Penataan dekorasi. Staf: - 1 koordinator utama, - 2 asisten.
5.	Fotografi dan Videografi	Rp 5.000.000 - Rp 20.000.000	- Sesi foto 4 jam (Sunrise/Sore), - Maksimal 2 lokasi, - 50 foto editan beresolusi tinggi,

			- Semua file dalam format JPEG dengan flashdisk, - MUA & Hair Do + 1 kali retouch, - 1 kanvas tanpa bingkai ukuran 40x60 cm, - Gratis 1 menit video sinematik, - Gratis 15 foto cetak 3R dengan tanda tangan kami, - Transportasi & akomodasi kru sudah termasuk - PROMO GRATIS 1 GAUN PREWEDDING.
6.	Makeup dan Hairstyling	Rp 2.000.000 - Rp 7.000.000	- Termasuk 1 kali makeup dan hair do, - Termasuk retouch makeup dan hair do, - Layanan ini disediakan oleh Wila Beauty dengan 1 tim hairstylist dari Wila Beauty menggunakan produk premium.
7.	Photobooth	Rp 3.000.000 - Rp 7.000.000	- Photobooth dengan konsep selfie box yang dilengkapi dengan LCD touch screen, - Serta pencahayaan yang baik, - Menggunakan printer khusus photobooth dengan kecepatan cetak 12 detik per foto,

			- Mencetak tanpa batas sesuai durasi jam yang dipilih. - Sudah termasuk backdrop kain glitter, fun props, bingkai foto, dan soft file yang diberikan dalam flashdisk, - Template kustom, barcode, GIF, & email.
8.	Undangan Pernikahan	Rp 500.000 – Rp 5.000.000	- Undangan STANDAR single hardcover, - Bahan KERTAS FANCY, - Ukuran 20x14, - Termasuk: amplop undangan single hardcover, 2 sisi kartu maps, kartu souvenir, kartu tukar souvenir plastik, desain, dan stiker nama.
9.	Souvenir Pernikahan	Rp 500.000. - Rp 10.000.000	- Bisa di emboss, - Kemasan tile, - Kemasan mika + Pita, - FREE : Kemasan plastik & Paper brown.

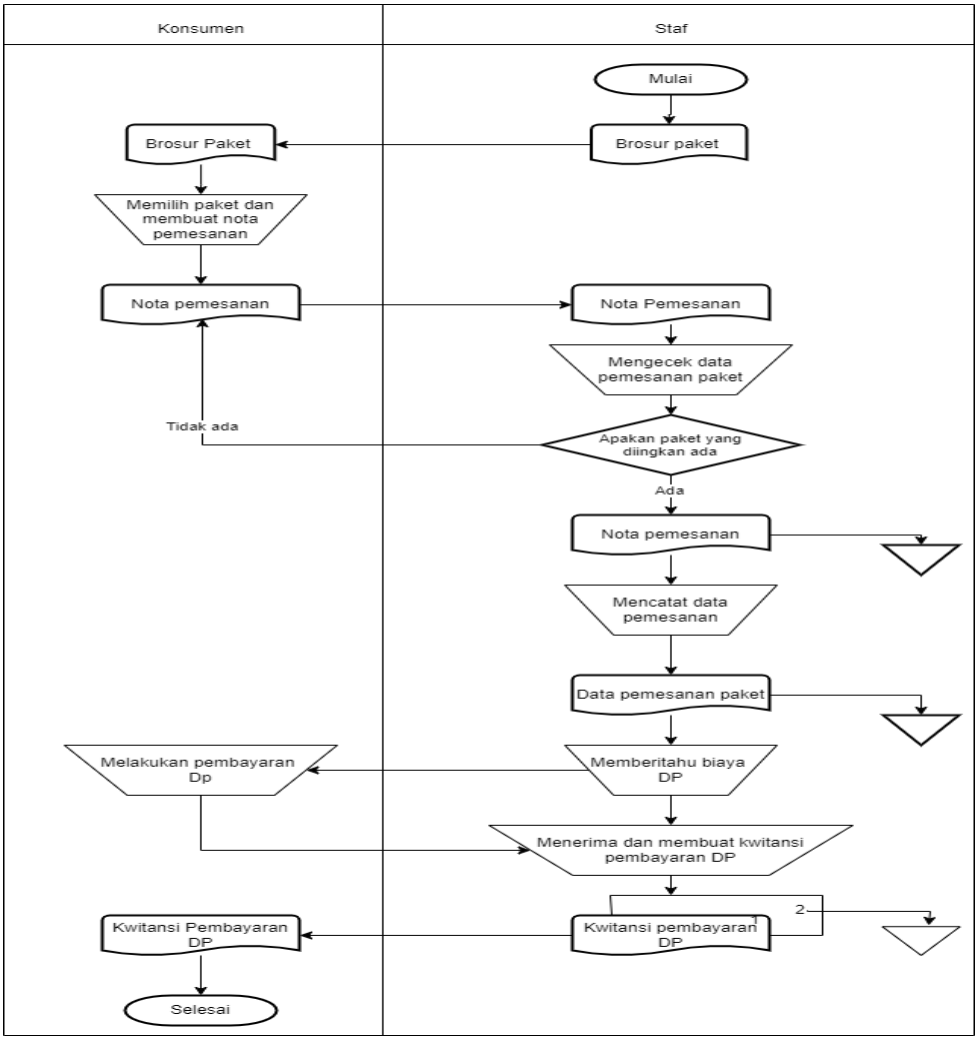
3.3 Analisis Sistem yang Sedang Berjalan

Berdasarkan hasil wawancara penulis dengan Staf Administrasi mengenai berbagai aktivitas Staf Bawor Network, termasuk pengelolaan laporan pemesanan paket dan masalah terkait cara pemesanan paket saat ini di Bawor Network.

Berikut ini adalah proses pemesanan paket di Bawor Network:

1. Pelanggan datang langsung ke lokasi Bawor Network.
2. Staf Administrasi akan memberikan brosur tentang paket dan menjelaskan detail paket tersebut.
3. Pelanggan memilih paket, kemudian admin akan mencatat pemesanan tersebut.
4. Admin memeriksa jadwal pemesanan paket; jika jadwal sudah penuh, mereka akan memberitahu pelanggan bahwa jadwal tersebut tidak tersedia. Namun, jika jadwal masih kosong, admin akan mencatat data pemesanan tersebut.
5. Selanjutnya, admin akan memberitahu pelanggan mengenai biaya Uang Muka (DP).
6. Pelanggan melakukan pembayaran untuk Uang Muka (DP).
7. Admin membuat kwitansi untuk pembayaran Uang Muka (DP) dan menyerahkan kwitansi tersebut kepada pelanggan.
8. Pelanggan menerima kwitansi untuk pembayaran Uang Muka (DP).

   Berikut adalah flowmap yang ditunjukkan pada (Gambar 3.3) di bawah ini:



(Sumber : Wawancara dengan Bapak fadilah)

Gambar 3. 3 Flowmap Sistem berjalan Pemesanan Paket pada Bawor Network

BAB IV

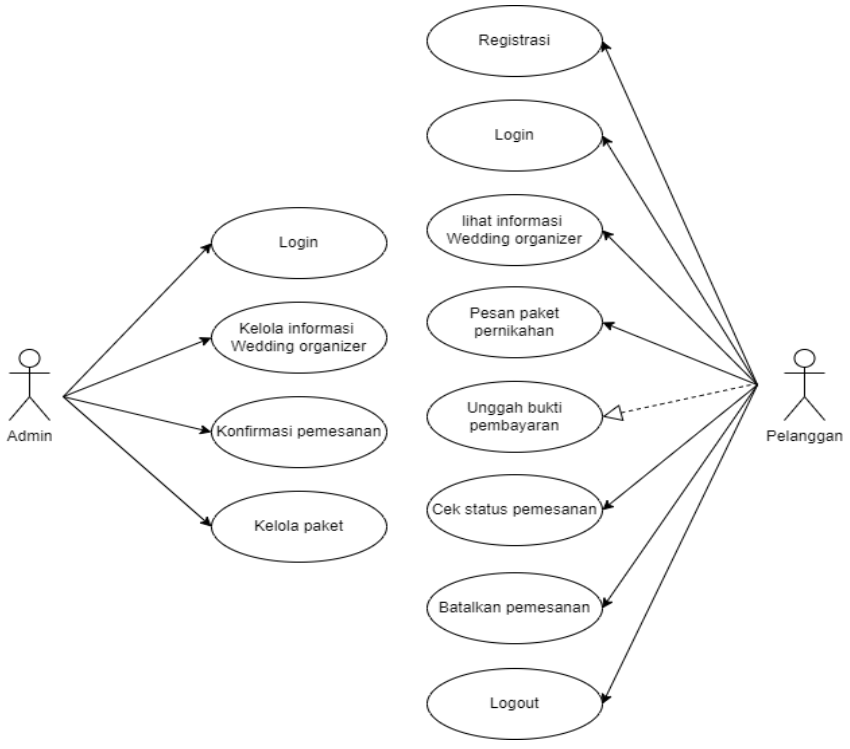
HASIL DAN PEMBAHASAN

4.1 Perancangan Sistem

Penulis akan menggunakan Unified Modeling Language untuk merancang Sistem yang relevan. Rancangan ini akan mencakup Use Case Diagram, Activity Diagram, Sequence Diagram, dan Class Diagram.

4.1.1 Use Case Diagram

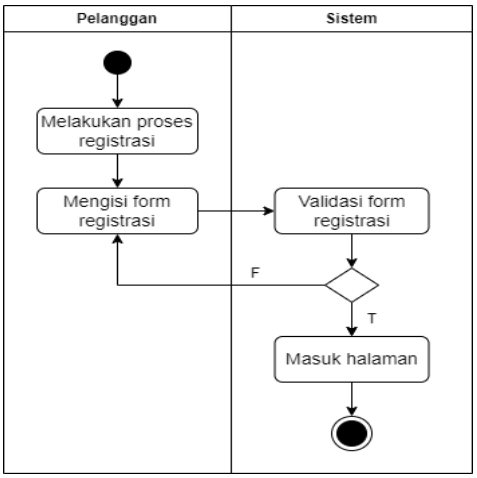
Dalam Sistem ini, aktivitas yang dapat dilakukan oleh aktor (pengguna Sistem) dapat digambarkan melalui use case diagram berikut:



Gambar 4. 1 Use Case Diagram Event Organizer

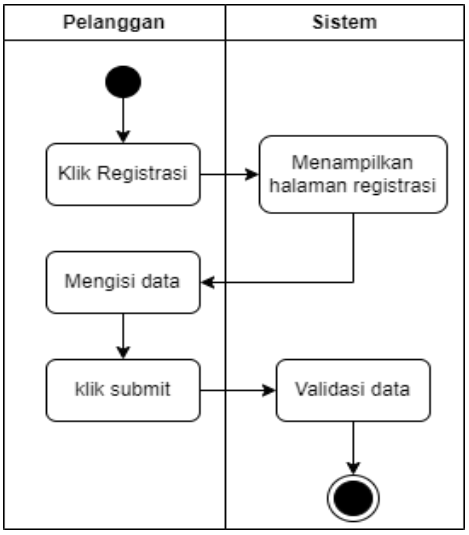
Dari hasil gambar diatas, dapat diuraian deskripsi yang terdapat pada gambar use case diagram yaitu, mencakup proses registrasi, login, pemesanan paket pernikahan, dan pengelolaan layanan oleh admin. Pengguna memulai dengan registrasi dan login, lalu dapat melihat informasi Event Organizer, memesan paket pernikahan, dan mengunggah bukti pembayaran—dengan catatan bahwa pengunggahan bukti pembayaran bergantung pada pemesanan paket pernikahan terlebih dahulu. Setelah pemesanan, pengguna dapat memeriksa status pesanan, membatalkan pesanan jika diperlukan, dan akhirnya logout dari sistem. Admin, di sisi lain, memiliki peran untuk mengelola informasi Event Organizer, mengonfirmasi pemesanan, serta mengelola paket yang tersedia. Diagram ini secara keseluruhan menggambarkan alur kerja sistem yang memastikan interaksi antara pengguna dan admin berjalan sesuai prosedur dalam pengelolaan layanan Event Organizer.

4.1.2 Activity Diagram



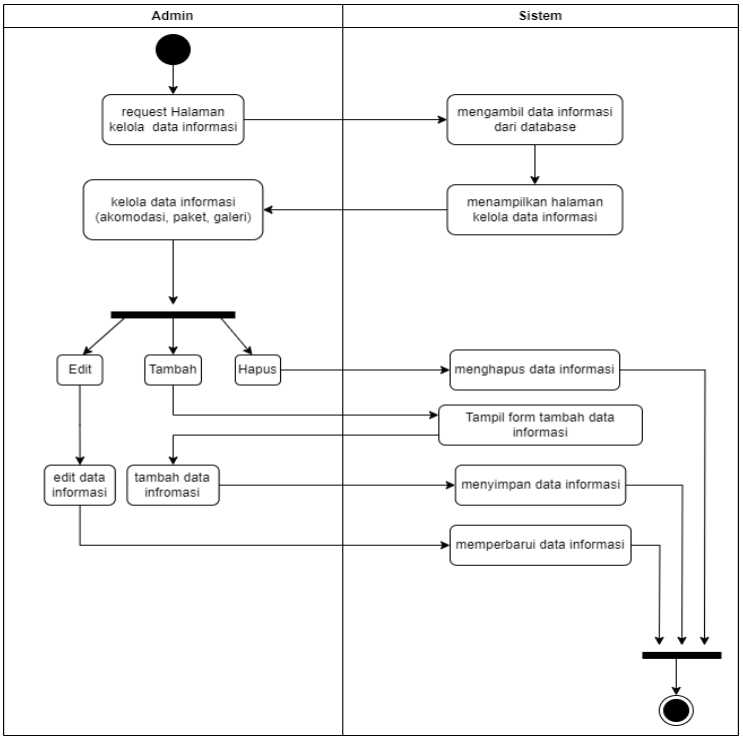
Gambar 4. 2 Aktivity Diagram Registrasi

Proses registrasi pengguna dimulai ketika user memilih opsi registrasi di halaman utama dan mengisi form yang berisi informasi pribadi, Setelah user mengirimkan form, sistem melakukan validasi data untuk memastikan bahwa semua informasi diisi dengan benar dan sesuai dengan format yang ditetapkan. Jika terjadi kesalahan, seperti email yang sudah terdaftar atau format data yang salah, sistem akan mengarahkan user kembali ke halaman form dengan pesan kesalahan yang relevan untuk diperbaiki. Sebaliknya, jika data valid, sistem akan menyimpan informasi tersebut ke dalam database dan mengarahkan user ke halaman beranda atau halaman konfirmasi registrasi yang sukses. Proses ini memastikan bahwa hanya data yang benar dan valid yang diterima oleh sistem, memberikan keamanan dan integritas dalam pengelolaan data pengguna.



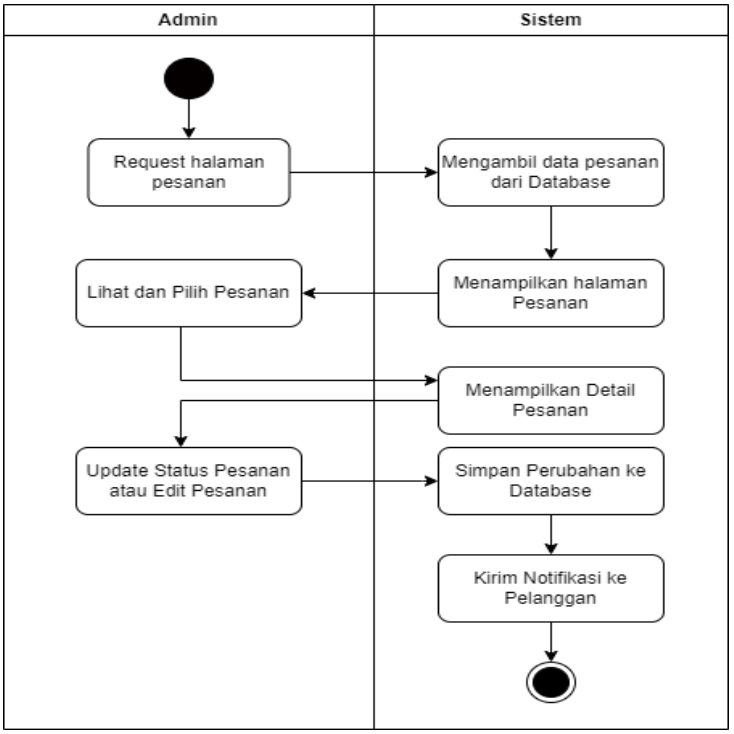
Gambar 4. 3 Aktivity Diagram Login Admin dan User

Activity Diagram di atas memberikan gambaran tentang proses login di website. Untuk melakukan login, pengguna diminta memasukkan username dan password. Setelah mengklik tombol login, Sistem akan memverifikasi apakah username dan password yang dimasukkan benar. Jika benar, pengguna akan diarahkan ke halaman utama, sedangkan jika salah, akan muncul pesan kesalahan yang meminta mereka untuk memasukkan kembali username dan password.



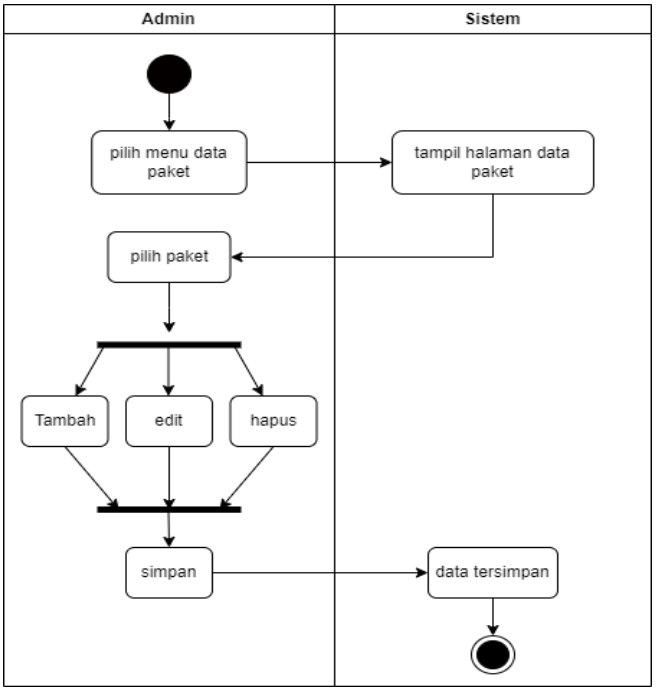
Gambar 4. 4 Aktivity Diagram Kelola Informasi Event Organizer

Aktivity Diagram di atas memberikan gambaran tentang bagaimana proses Kelola informasi. Proses dimulai dengan permintaan halaman pengelolaan data oleh Admin, di mana Sistem kemudian menampilkan data dari database. Admin dapat mengelola data yang mencakup berbagai kategori seperti akomodasi, paket, dan galeri dengan opsi untuk mengedit, menambah, atau menghapus data. Sistem merespons setiap tindakan Admin dengan menghapus data dari database, menampilkan formulir untuk data baru, dan menyimpan atau memperbarui data sesuai perintah Admin. Alur ini menunjukkan bagaimana Admin dan Sistem berinteraksi untuk menjaga agar data informasi tetap terorganisir dan up-to-date.



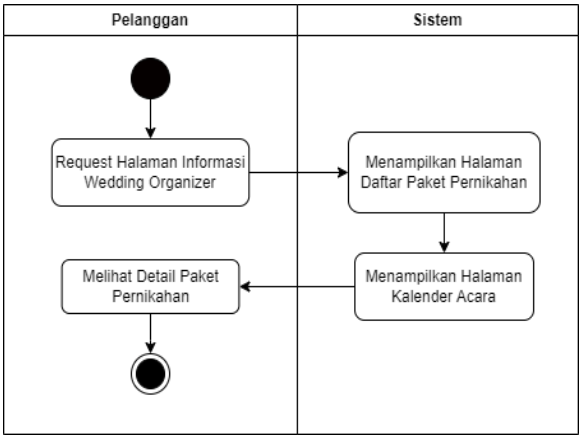
Gambar 4. 5 Aktivity Diagram Konfirmasi Pemesanan

Aktivity Diagram di atas memberikan gambaran tentang bagaimana proses konfirmasi pemesanan. Proses dimulai dari permintaan halaman pesanan oleh Admin, yang direspons oleh Sistem dengan menampilkan data pesanan dari database. Admin kemudian dapat melihat dan memilih pesanan untuk diperiksa atau diedit. Setelah melakukan perubahan, Sistem akan menyimpan update ke database dan mengirimkan notifikasi kepada pelanggan. Secara keseluruhan, diagram ini menunjukkan langkah-langkah yang jelas dan terstruktur untuk memastikan pengelolaan pesanan berjalan efisien dan informatif bagi pelanggan.



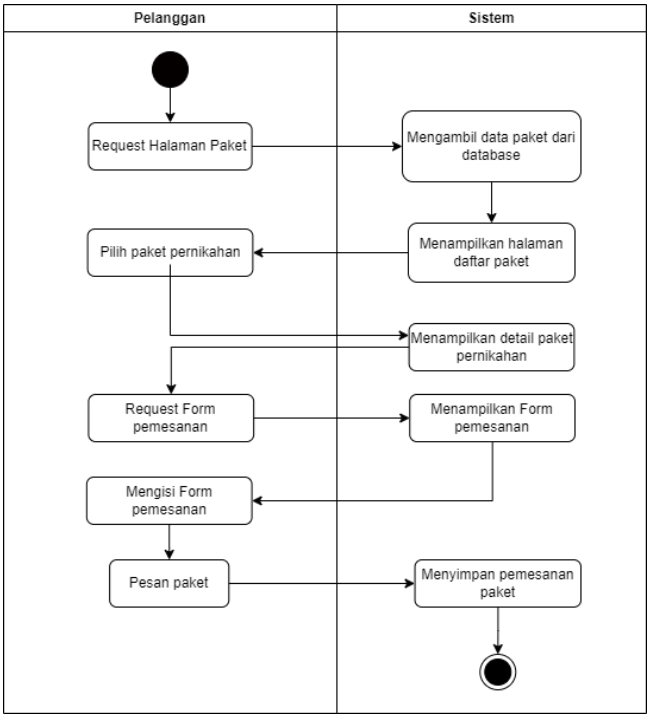
Gambar 4. 6 Aktivity Diagram Kelola Paket

Diagram ini menunjukkan alur pengelolaan data paket oleh Admin melalui Sistem. Proses dimulai dengan pemilihan menu data paket, dilanjutkan dengan menampilkan halaman data paket oleh Sistem. Admin kemudian memilih paket untuk dikelola, dengan opsi untuk menambah, mengedit, atau menghapus paket. Setelah perubahan dilakukan, Admin menyimpan data, yang kemudian tersimpan dalam database oleh Sistem. Diagram ini menggambarkan prosedur yang sistematis dan terstruktur untuk memastikan bahwa pengelolaan data paket dilakukan secara efisien dan akurat.



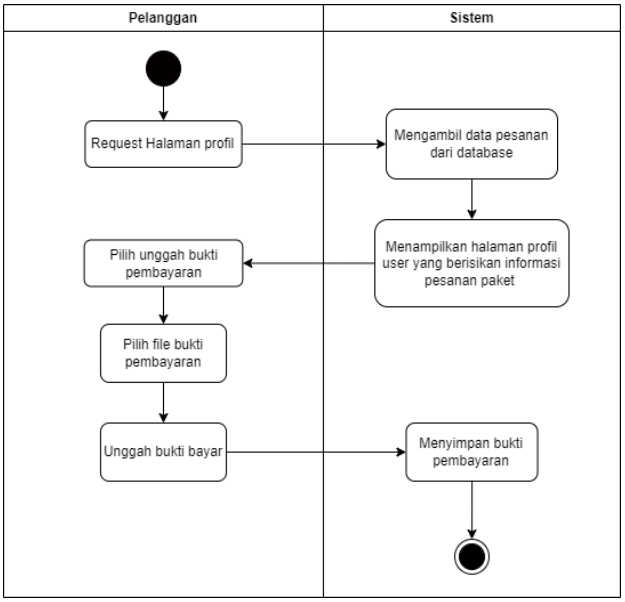
Gambar 4. 7 Lihat Informasi Event Organizer

Diagram ini lebih sederhana karena hanya berfokus pada langkah-langkah inti untuk melihat informasi paket Event Organizer tanpa melibatkan proses pemesanan atau pengelolaan detail lainnya.



Gambar 4. 8 Pesan Paket Pernikahan

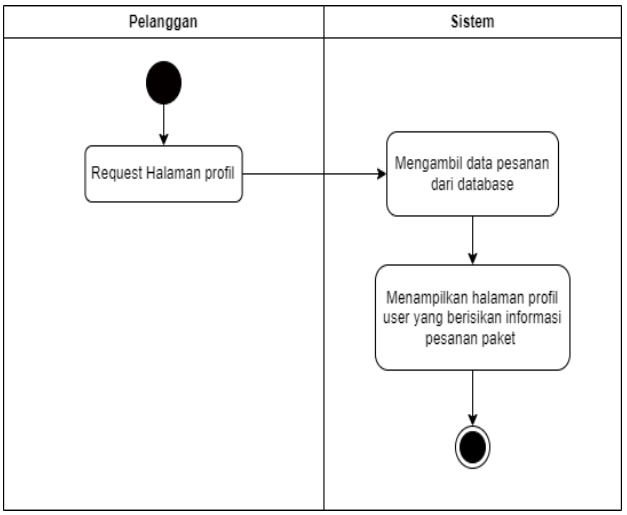
Dalam proses pemesanan paket pernikahan, pengguna memulai dengan meminta halaman yang menampilkan daftar paket pernikahan yang tersedia. Sistem kemudian mengambil data paket dari database dan menampilkan daftar tersebut kepada pengguna. Setelah melihat daftar, pengguna memilih salah satu paket, dan sistem menampilkan detail lengkap dari paket yang dipilih. Pengguna kemudian meminta untuk mengisi formulir pemesanan, yang ditampilkan oleh sistem. Setelah mengisi formulir dengan informasi yang diperlukan, pengguna mengkonfirmasi pemesanan, dan sistem menyimpan pemesanan tersebut ke dalam database, menyelesaikan proses pemesanan.



Gambar 4. 9 Unggah Bukti Pembayaran

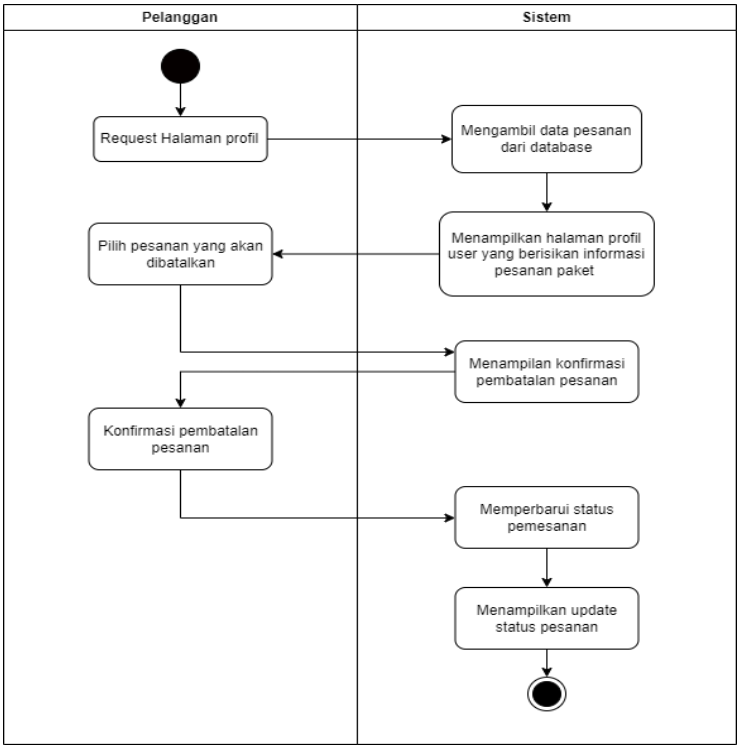
Dalam proses ini, pengguna memulai dengan meminta akses ke halaman profil mereka, di mana sistem mengambil data pesanan dari database dan menampilkan informasi terkait pesanan paket. Setelah melihat

detail pesanan, pengguna memilih opsi untuk mengunggah bukti pembayaran, kemudian memilih dan mengunggah file bukti pembayaran dari perangkat mereka. Sistem kemudian memproses dan menyimpan bukti pembayaran tersebut dalam database, memastikan bahwa informasi pembayaran tersimpan dengan aman dan dapat diverifikasi.



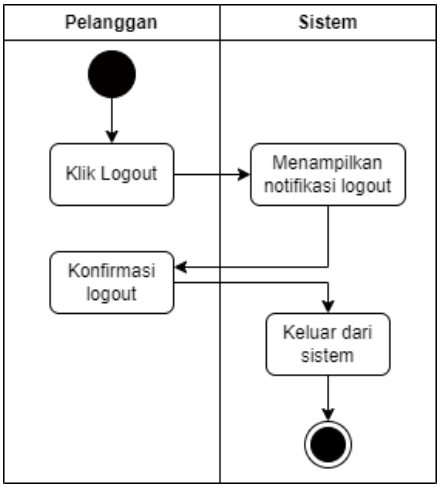
Gambar 4. 10 Cek Status Pemesanan

User memulai dengan meminta akses ke halaman profil mereka untuk melihat informasi terkait pesanan. Sistem merespons permintaan tersebut dengan mengambil data pesanan dari database yang terkait dengan akun pengguna. Setelah data pesanan diambil, sistem menampilkan halaman profil yang berisi detail lengkap tentang pesanan paket yang telah dilakukan, memungkinkan pengguna untuk memeriksa status dan rincian pesanan mereka.



Gambar 4. 11 Batalkan Pemesanan

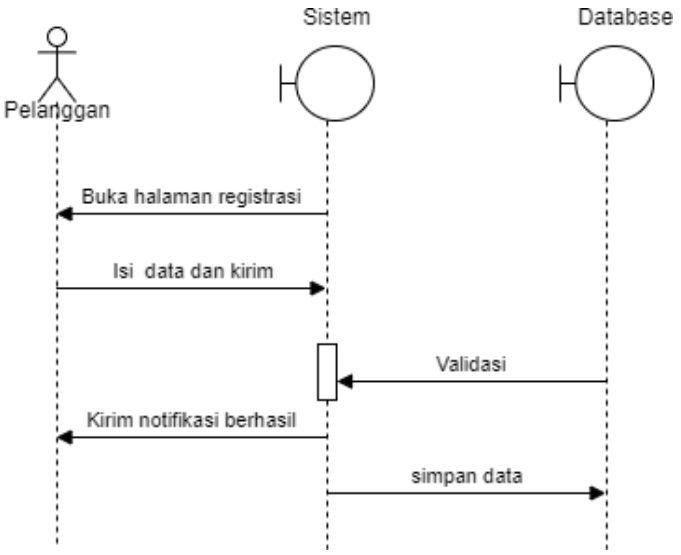
User memulai dengan meminta akses ke halaman profil mereka untuk melihat detail pesanan yang telah dibuat. Sistem merespons dengan mengambil data pesanan dari database dan menampilkan halaman profil yang mencakup informasi pesanan tersebut. Pengguna kemudian memilih pesanan yang ingin dibatalkan dan sistem menampilkan konfirmasi untuk memastikan keputusan pembatalan. Setelah pengguna mengonfirmasi pembatalan, sistem memperbarui status pesanan di database menjadi "dibatalkan" dan menampilkan status terbaru kepada pengguna, mengonfirmasi bahwa pembatalan telah berhasil diproses.



Gambar 4. 12 Logout

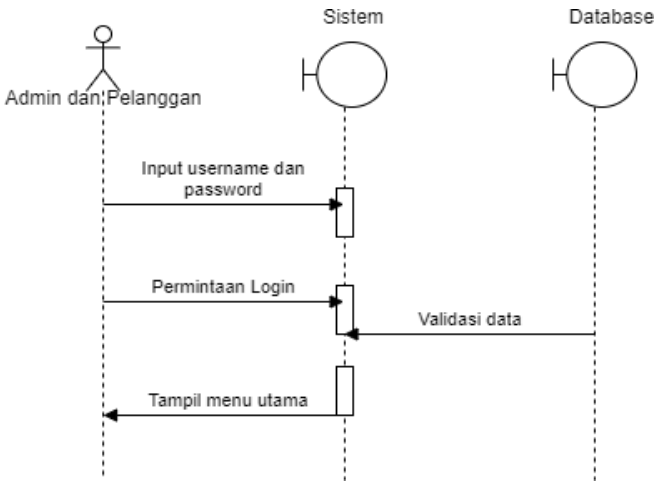
Aktivity diagram ini menggambarkan interaksi antara pengguna dan Sistem dalam proses Logout. Pengguna mengklik tombol "Logout" di aplikasi. Tindakan ini menandakan bahwa pengguna ingin keluar dari sesi aktif di aplikasi, Sistem merespons dengan mengakhiri sesi pengguna, Sistem menyelesaikan proses logout dan mengonfirmasi bahwa pengguna telah berhasil keluar dari Sistem, pengguna akan diarahkan ke halaman halaman utama aplikasi yang dapat diakses tanpa login.

4.1.3 Sequence Diagram



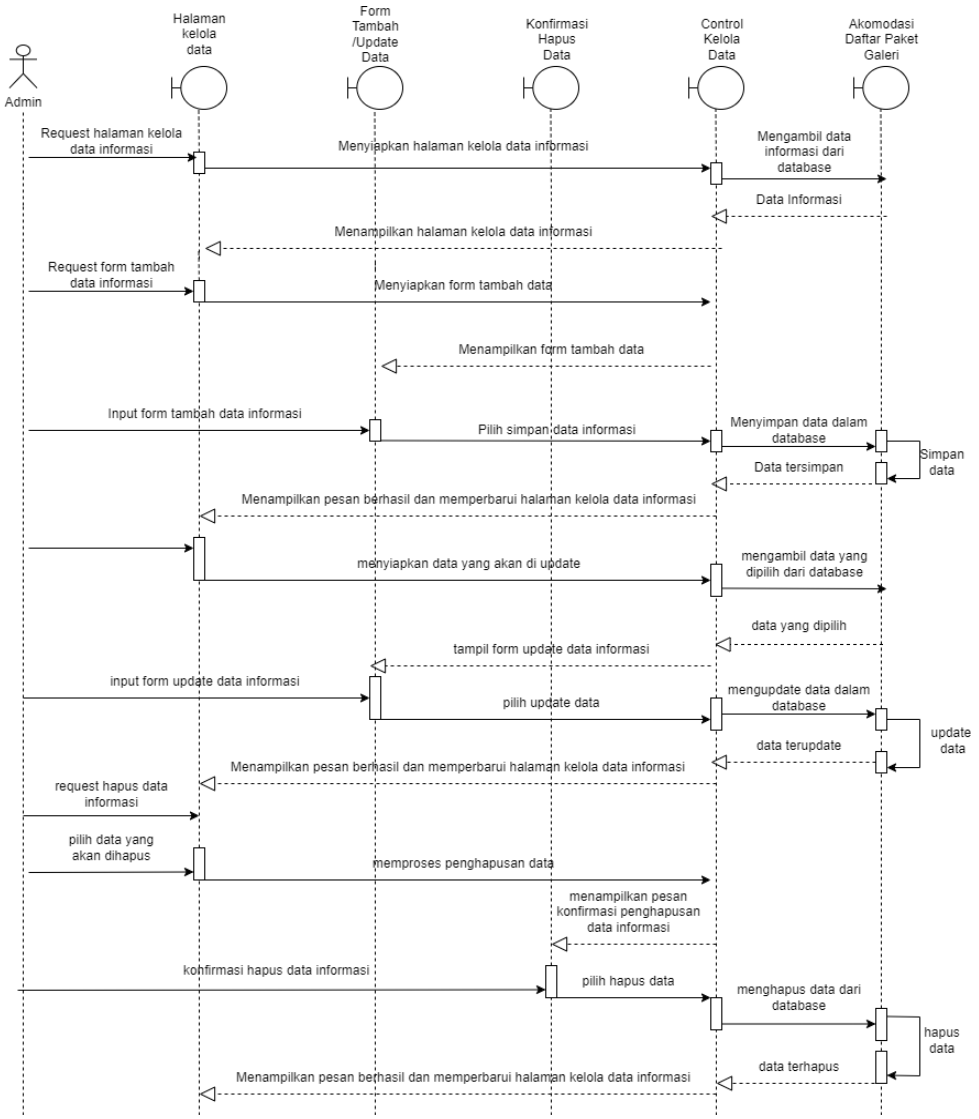
Gambar 4. 13 Sequence Diagram Registrasi

Diagram ini menggambarkan langkah-langkah utama yang terlibat dalam proses registrasi, mulai dari user membuka halaman registrasi hingga sistem mengonfirmasi bahwa registrasi telah berhasil.



Gambar 4. 14 Sequence Diagram Login Admin dan Pelanggan

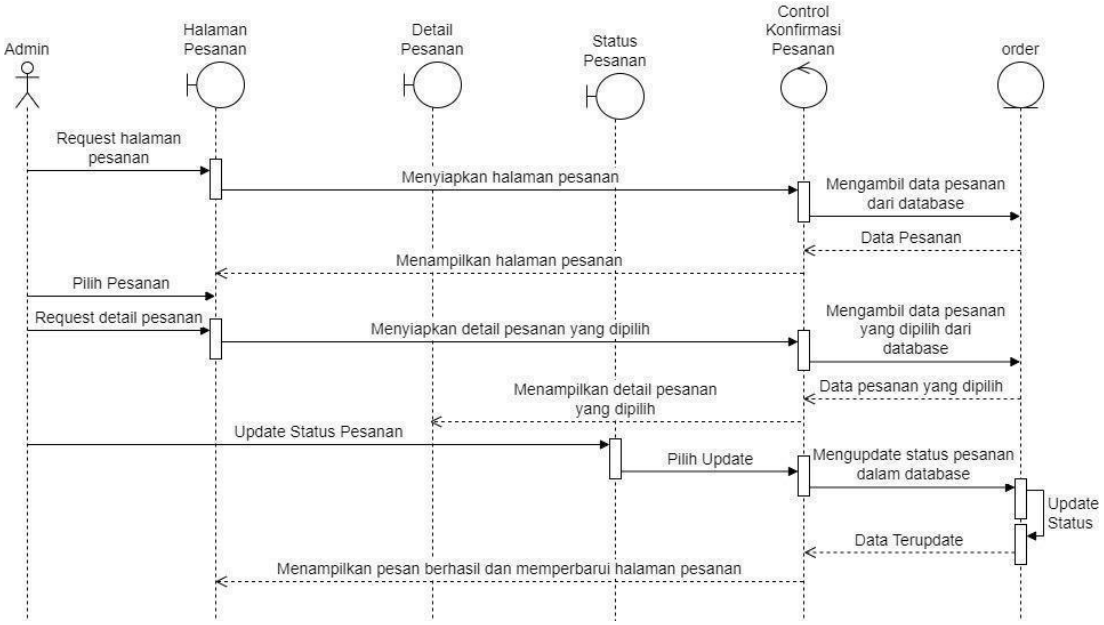
Pengguna memasukkan username dan password pada formulir login di aplikasi web Event Organizer, Browser mengirim permintaan login yang berisi data login (username dan password) ke web server, Web server menerima permintaan login dari browser dan mengirim permintaan validasi data login ke database, Database memeriksa keabsahan username dan password yang diterima dari web server dan mengirimkan hasil validasi kembali ke web server, Jika data login valid, web server mengirimkan respons sukses ke browser. Jika data login tidak valid, web server mengirimkan pesan kesalahan ke browser, Browser menerima respons dari web server dan menampilkan menu utama.



Gambar 4. 15 Sequence Diagram Kelola Informasi Event Organizer

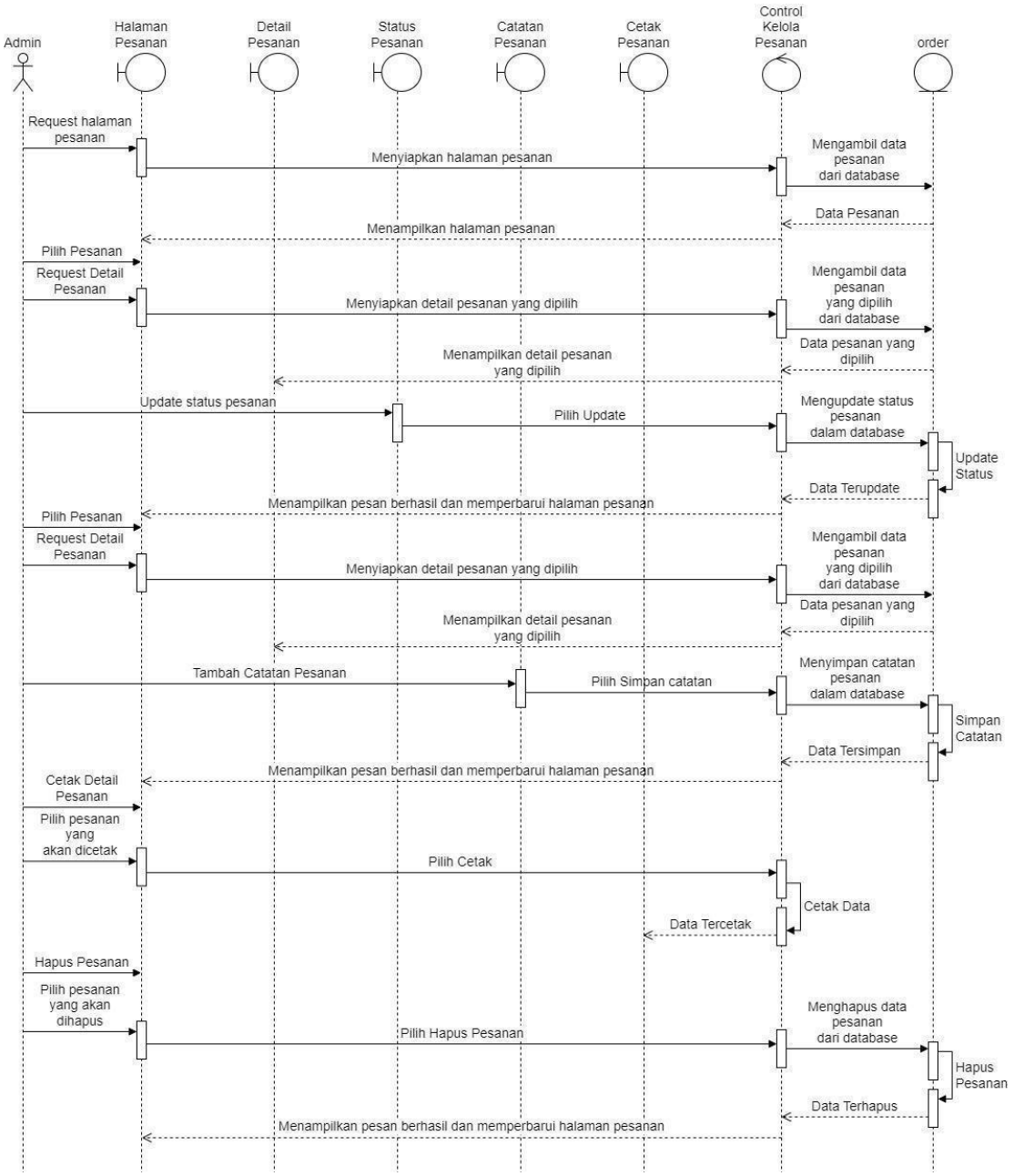
Dijelaskan interaksi dalam proses kelola informasi Event Organizer yang dilakukan oleh Admin. Interaksi melibatkan beberapa langkah, seperti request halaman kelola data informasi, tambah data informasi, ubah data informasi, dan hapus data informasi. Beberapa objek

yang terlibat dalam interaksi ini, seperti halaman kelola data, form tambah/update data dan konfirmasi hapus data, control kelola data sebagai control. Kemudian akomodasi, daftar paket dan galeri sebagai entity pada database.



Gambar 4. 16 Sequence Diagram Konfirmasi Pemesanan

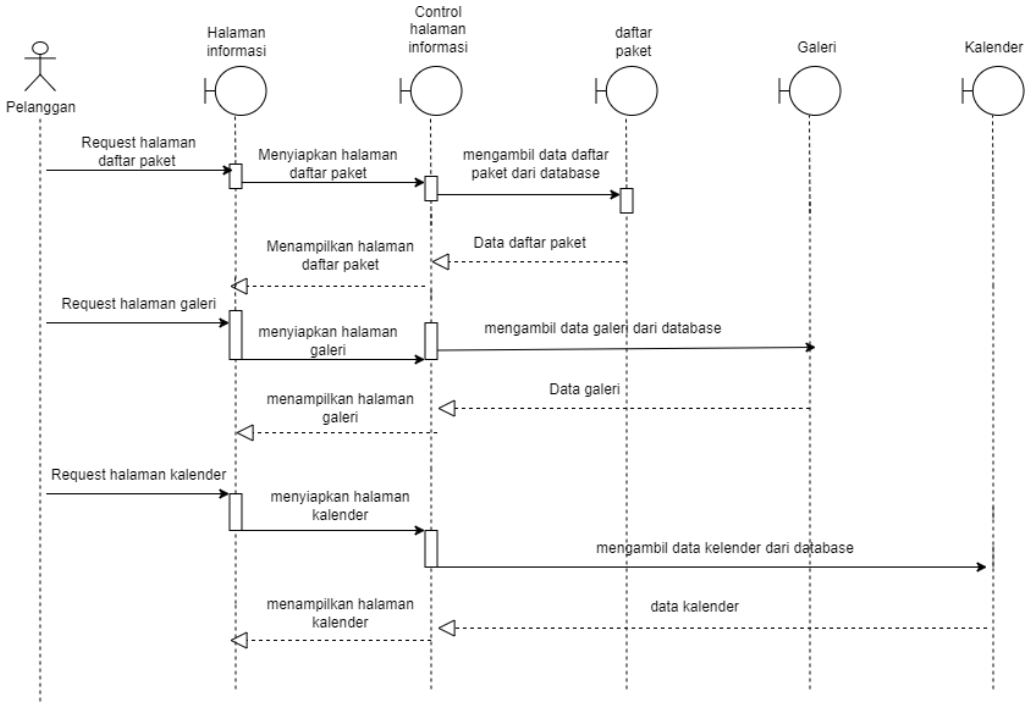
Dijelaskan interaksi dalam proses konfirmasi pemesanan yang dilakukan oleh Admin. Interaksi melibatkan beberapa langkah, seperti request halaman pesanan, request detail pesanan dan update status pesanan. Beberapa objek yang terlibat dalam interaksi ini, seperti halaman pesanan, detail pesanan dan status pesanan sebagai boundary, control konfirmasi pesanan sebagai control dan order sebagai entity pada database.



Gambar 4. 17 Sequence Kelola Paket

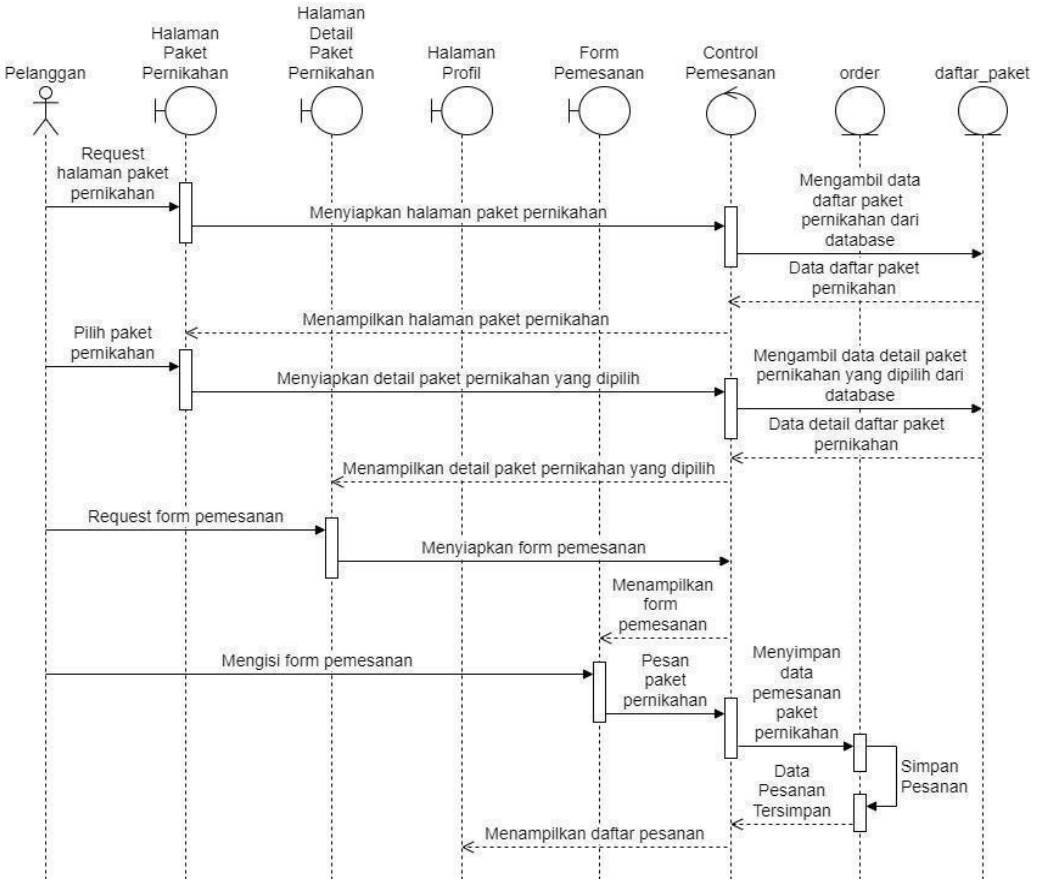
Dijelaskan interaksi dalam proses kelola Paket yang dilakukan oleh Admin. Interaksi melibatkan beberapa langkah, seperti request halaman pesanan, request detail pesanan, update status pesanan, tambah catatan pesanan, cetak detail pesanan dan hapus pesanan. Beberapa objek yang

terlibat dalam interaksi ini, seperti halaman pesanan, detailnpesanan, status pesanan dan catatan pesanan sebagai boundary, control kelola pesanan sebagai control dan order sebagai entity pada database.



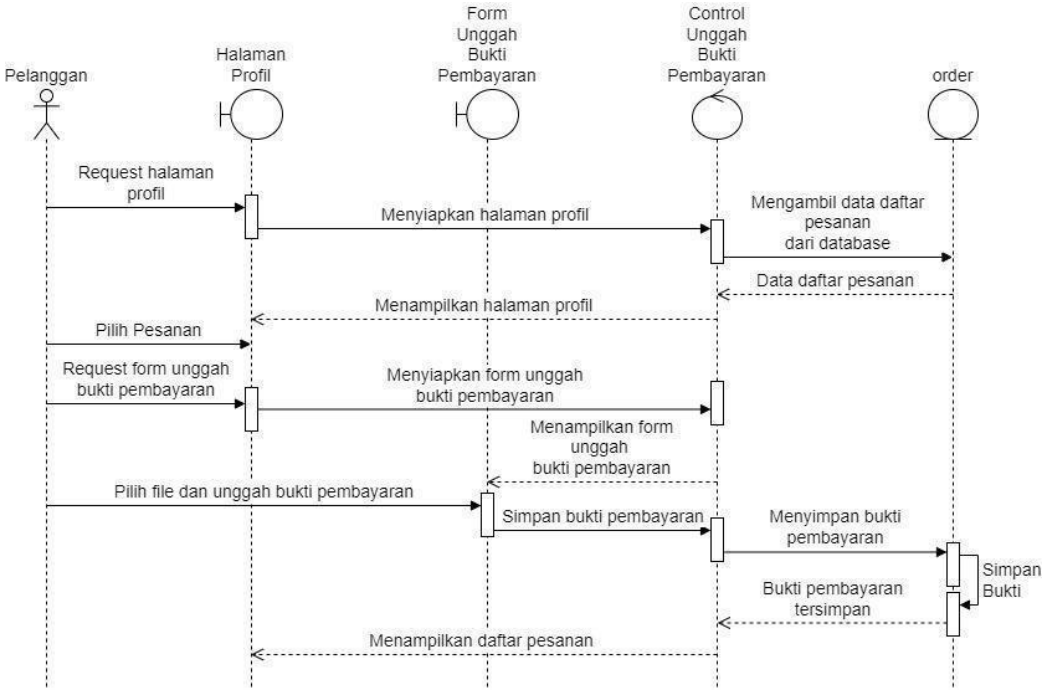
Gambar 4. 18 Lihat Informasi Event Organizer

Dijelaskan interaksi dalam proses lihat informasi Event Organizer yang dilakukan oleh Pelanggan. Interaksi melibatkan beberapa langkah, seperti request halaman daftar paket pernikahan, request halaman galeri, request halaman kalender. Beberapa objek yang terlibat dalam interaksi ini, seperti halaman informasi sebagai boundary, control halaman informasi sebagai control dan daftar_paket, galeri, kalender_aura dan system_info sebagai *entity* pada database.



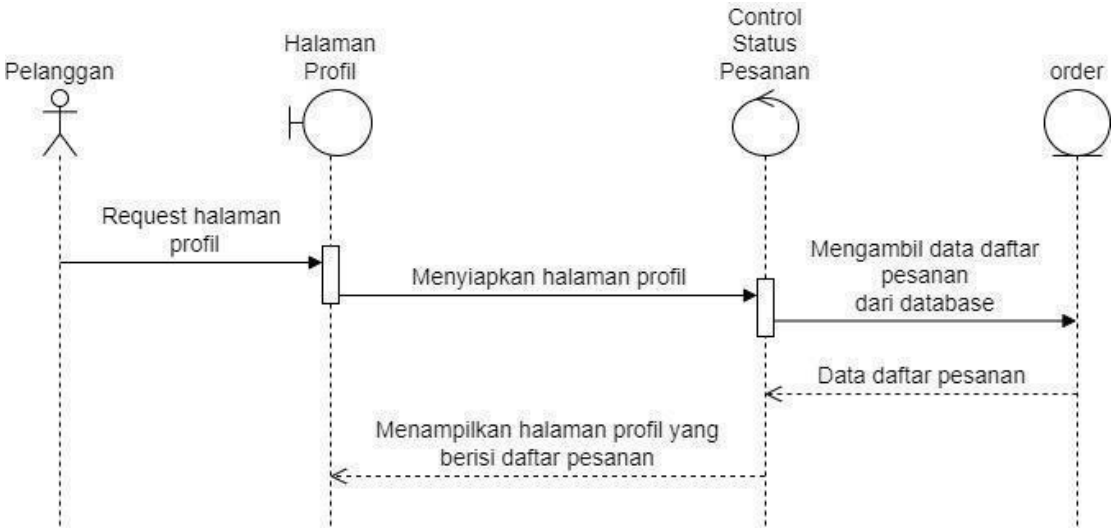
Gambar 4. 19 Pesan Paket Pernikahan

Dijelaskan interaksi dalam proses pesan paket pernikahan yang dilakukan oleh Pelanggan. Interaksi melibatkan beberapa langkah, seperti request halaman paket pernikahan, pilih paket pernikahan, request form pemesanan dan mengisi form pemesanan. Beberapa objek yang terlibat dalam interaksi ini, seperti halaman paket pernikahan, halaman detail paket pernikahan, halaman profil dan form pemesanan sebagai *boundary, control* pemesanan sebagai *control* serta order dan daftar_paket sebagai *entity* pada database.



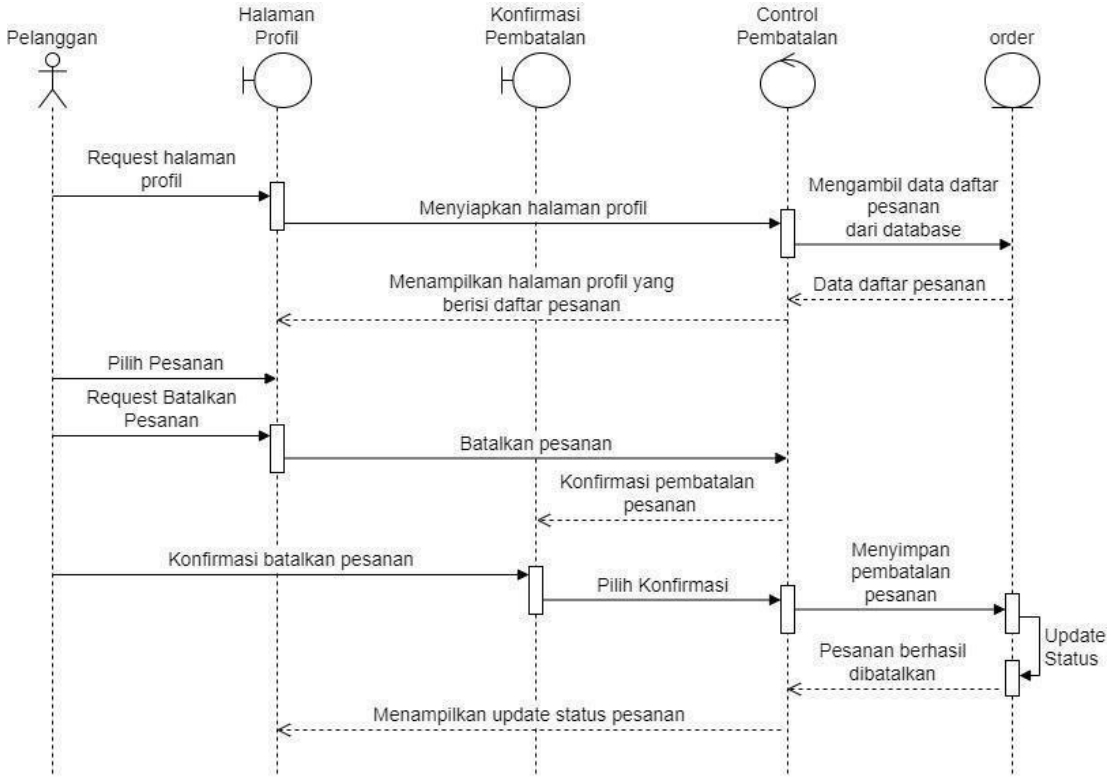
Gambar 4. 20 Unggah Bukti Pembayaran

Dijelaskan interaksi dalam proses unggah bukti pembayaran yang dilakukan oleh Pelanggan. Interaksi melibatkan beberapa langkah, seperti request halaman profil, request form unggah bukti pembayaran dan unggah bukti pembayaran. Beberapa objek yang terlibat dalam interaksi ini, seperti halaman profil dan form unggah bukti pembayaran sebagai boundary, control unggah bukti pembayaran sebagai control dan order sebagai entity pada database.



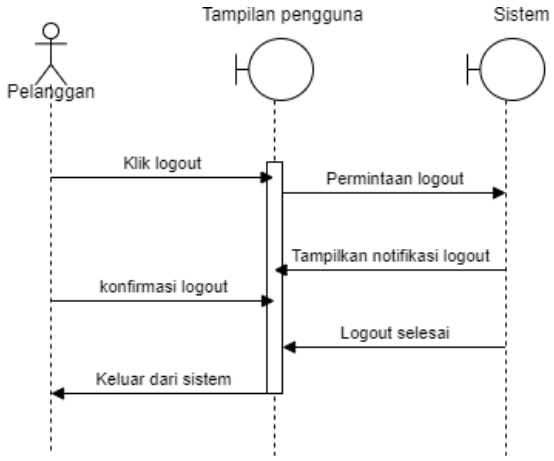
Gambar 4. 21 Cek Status Pemesanan

Dijelaskan interaksi dalam proses pengecekan status pemesanan yang dilakukan oleh Pelanggan. Interaksi melibatkan proses request halaman profil. Beberapa objek yang terlibat dalam interaksi ini, seperti halaman profil sebagai boundary, control status pesanan sebagai control dan order sebagai entity pada database.



Gambar 4. 22 Batalkan Pemesanan

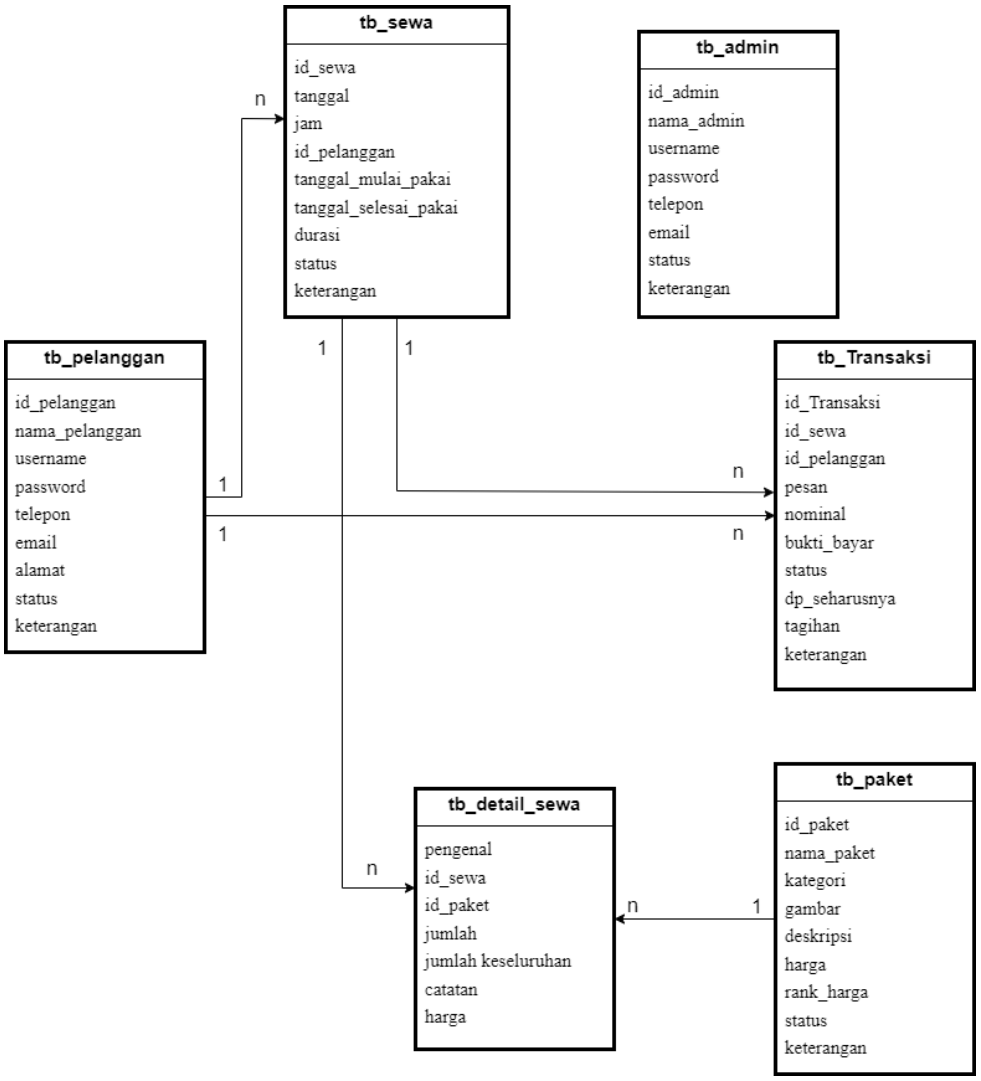
Dijelaskan interaksi dalam proses pembatalan pesanan yang dilakukan oleh Pelanggan. Interaksi melibatkan beberapa langkah, seperti request halaman profil, pilih pesanan dan konfirmasi batalkan pesanan. Beberapa objek yang terlibat dalam interaksi ini, seperti halaman profil dan konfirmasi pembatalan sebagai boundary, control pembatalan sebagai control dan order sebagai entity pada database.



Gambar 4. 23 Logout

Pelanggan mengklik tombol logout di antarmuka pengguna, yang kemudian mengirimkan permintaan logout ke sistem. Sistem merespons dengan menampilkan notifikasi konfirmasi kepada pelanggan, meminta mereka untuk mengonfirmasi tindakan logout. Setelah pelanggan memberikan konfirmasi, sistem memproses logout dengan menghapus sesi pengguna dan menyelesaikan prosedur logout. Akhirnya, tampilan pengguna mengarahkan pelanggan kembali ke halaman login atau halaman utama, menandakan bahwa proses logout telah berhasil. Diagram ini menunjukkan bagaimana setiap komponen berinteraksi untuk memastikan logout dilakukan dengan benar.

4.1.4 Class Diagram



***Gambar 4. 24* Class Diagram**

Merupakan class diagram yang digunakan sebagai referensi untuk hubungan antar tabel dalam database.

4.1.5 Perancangan Basis Data

Dalam Sistem informasi Pemesanan paket ini menggunakan basis data MySQL. Adapun hasil dari implementasi basi data tersebut adalah sebagai berikut:

1. Tabel Admin

   Nama Tabel : tb_admin

   Keterangan : Data Admin

   Primary key : id_admin

Tabel 4. 1 Admin

No	Name Field	Type Field	Collation
1.	Id_admin	Varchar(15)	Latin1_general_ci
2.	Nama_admin	Varchar(25)	Latin1_general_ci
3.	Username	Varchar(10)	Latin1_general_ci
4.	Password	Varchar(10)	Latin1_general_ci
5.	Telepon	Varchar(15)	Latin1_general_ci
6.	Email	Varchar(30)	Latin1_general_ci
7.	Status	Enum('aktif','tidak')	Latin1_general_ci
8.	Keterangan	text	Latin1_general_ci

2. Tabel paket

   Nama Tabel : tb_paket

   Keterangan : Data paket

   Primary key : id_paket

Tabel 4. 2 paket

No	Name Field	Type Field	Collation	Default
1.	Id_paket	Varchar(15)	Utf8mb4_general_ci	None
2.	Nama_paket	Varchar(20)	Utf8mb4_general_ci	None
3.	Kategori	Varchar(20)	Utf8mb4_general_ci	None
4.	Gambar	Varchar(100)	Utf8mb4_general_ci	None
5.	Deskripsi	Text	Utf8mb4_general_ci	None
6.	Harga	Int(10)		None
7.	Rank_harga	Varchar(20)	Utf8mb4_general_ci	None
8.	Status	Varchar(20)	Utf8mb4_general_ci	None
9.	keterangan	Text	Utf8mb4_general_ci	None

3. Tabel Transaksi

   Nama Tabel : tb_transaksi

   Keterangan : Data transaksi

   Primary key : id_transaksi

Tabel 4. 3 Transaksi

No	Nama Field	Type Field	Collation	Default
1.	Id_transaksi	Varchar(15)	Latin1_swedish_ci	None
2.	Id_sewa	Varchar(15)	Latin1_swedish_ci	None
3.	Id_pelanggan	Varchar(15)	Latin1_swedish_ci	None
4.	Pesan	Text	Latin1_swedish_ci	None
5.	Nominal	Int(8)		None
6.	Bukti_bayar	Varchar(100)	Latin1_swedish_ci	None
7.	Status	Varchar(15)	Latin1_swedish_ci	None
8.	Dp_seharusnya	Int(12)		None
9.	Tagihan	Int(12)		None
10.	Keterangan	Text	Latin1_swedish_ci	None

4. Tabel Pelanggan

   Nama Tabel : tb_pelanggan

   Keterangan : Data Pelanggan

   Primary key : id_pelanggan

Tabel 4. 4 Pelanggan

No	Nama Field	Type Field	Collation	Default
1.	Id_pelanggan	Varchar(15)	Latin1_general_ci	None
2.	Nama_pelanggan	Varchar(25)	Latin1_general_ci	None
3.	Username	Varchar(10)	Latin1_general_ci	None
4.	Password	Varchar(10)	Latin1_general_ci	None
5.	Telepon	Varchar(15)	Latin1_general_ci	None
6.	Email	Varchar(30)	Latin1_general_ci	None
7.	Alamat	Text	Latin1_general_ci	None
8.	Status	Enum('Aktif',Tidak Aktif)	Latin1_general_ci	Aktif
9.	Keterangan	Text	Latin1_general_ci	None

5. Tabel Sewa

   Nama Tabel : tb_sewa

   Keterangan : Data Sewa

   Primary key : id_sewa

Tabel 4. 5 Sewa

No	Nama Field	Type Field	Collation	Default
1.	Id_sewa	Varchar(15)	Utf8mb4_general_ci	None
2.	Tanggal	Date		None
3.	Jam	Time		None
4.	Id_pelanggan	Varchar(15)	Utf8mb4_general_ci	None
5.	Tanggal_mulai_pakai	Date		None
6.	Tanggal_selesai_pakai	Date		None
7.	Durasi	Varchar(15)	Utf8mb4_general_ci	None
8.	Status	Varchar(15)	Utf8mb4_general_ci	None
9.	Keterangan	Text	Utf8mb4_general_ci	None

6. Tabel Sewadetail

   Nama Tabel : tb_sewadetail

   Keterangan : Data Sewadetail

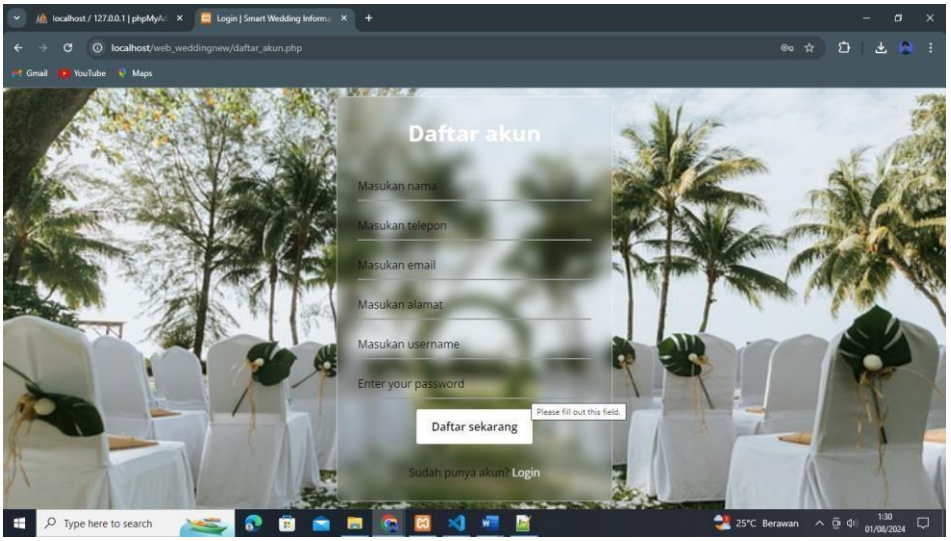
   Primary key : id

Tabel 4. 6 Sewadetail

No	Nama Field	Type Field	Collation	Default
1.	Id	Int(8)		None
2.	Id_sewa	Varchar(15)	Latin1_swedish_ci	None
3.	Id_paket	Varchar(15)	Latin1_swedish_ci	None
4.	Jumlah	Int(8)		None
5.	Subtotal	Varchar(15)	Latin1_swedish_ci	None
6.	Catatan	Text	Latin1_swedish_ci	None
7.	Harga	Int(8)		None

4.2 Tampilan Sistem

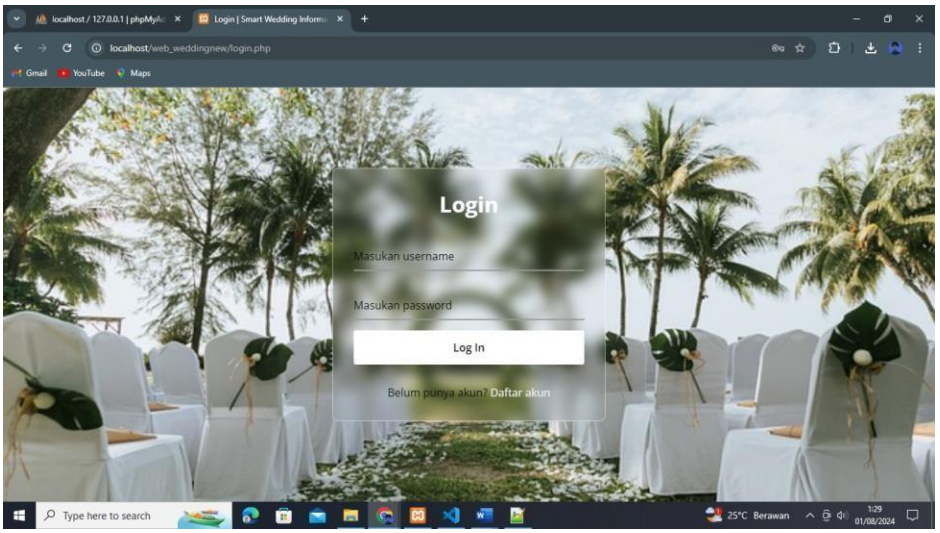
4.2.1 Halaman Registrasi User



Gambar 4. 25 Halaman Registrasi

Gambar 4.25 menampilkan halaman pendaftaran akun, di tengah halaman terdapat formulir pendaftaran yang terdiri dari beberapa kolom input, seperti nama, telepon, email, alamat, username, dan kata sandi. Pengguna diminta untuk mengisi informasi ini sebelum menekan tombol "Daftar sekarang" untuk membuat akun baru.

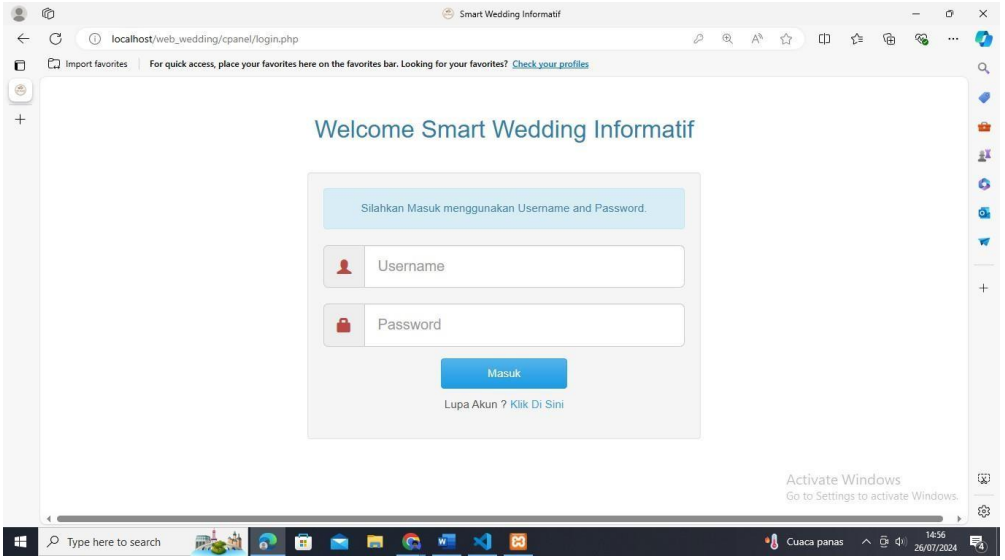
4.2.2 Halaman Login Pelanggan



Gambar 4. 26 Login Pelanggan

Gambar 4.26 menampilkan halaman login, di tengah halaman terdapat form login yang terdiri dari username, password, tombol login, dan daftar akun jika belum mempunyai akun.

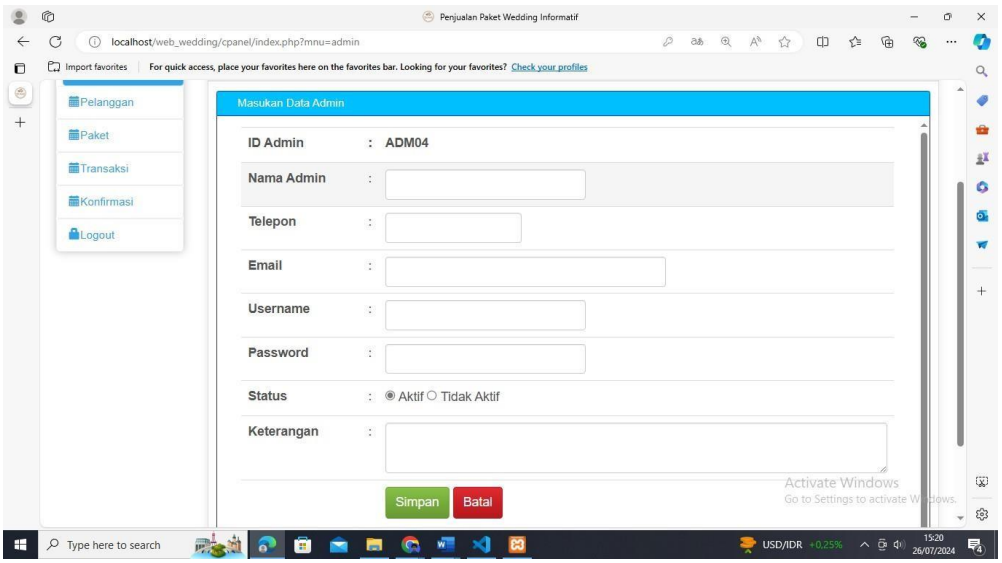
4.2.3 Halaman Login Admin



Gambar 4. 27 Login Admin

Gambar 4,27 menampilkan halaman login admin, di tengah halaman terdapat form login yang terdiri dari username, password, tombol masuk dan lupa akun.

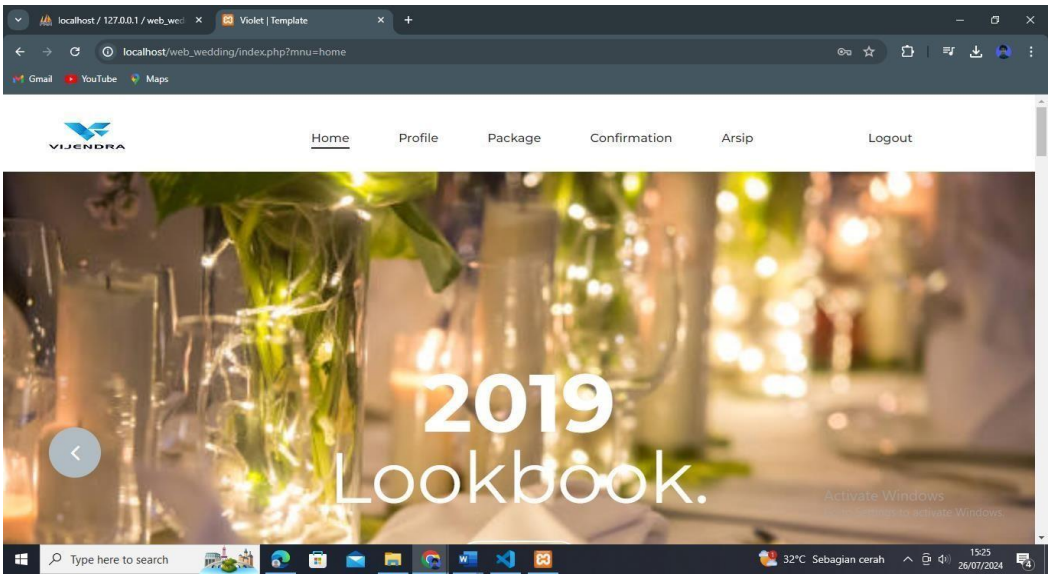
4.2.4 Halaman Daftar Admin



Gambar 4. 28 Daftar Admin

Gambar 4.28 menunjukkan halaman daftar admin, di sisi kiri terdapat menu navigasi dengan beberapa opsi seperti "Pelanggan," "Paket," "Transaksi," "Konfirmasi," dan "Logout." Di tengah halaman, terdapat formulir untuk memasukkan data admin baru dengan beberapa kolom input, seperti "Nama Admin," "Telepon," "Email," "Username," dan "Password." Selain itu, ada opsi untuk memilih status admin, apakah "Aktif" atau "Tidak Aktif," serta kolom "Keterangan" untuk menambahkan informasi tambahan. Di bagian bawah, terdapat dua tombol aksi: "Simpan" untuk menyimpan data yang telah dimasukkan dan "Batal" untuk membatalkan pengisian data.

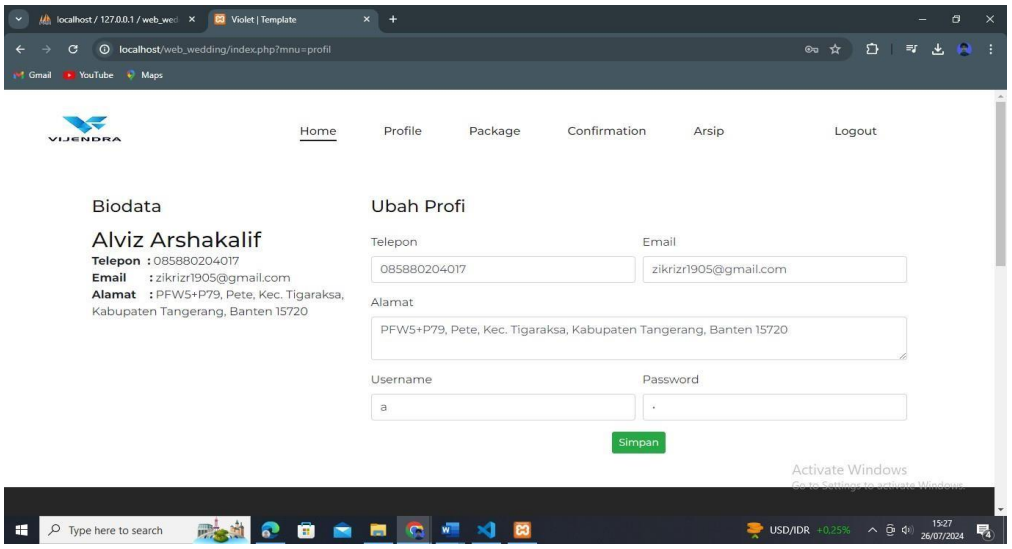
4.2.5 Halaman Utama



Gambar 4. 29 Halaman Utama

Gambar 4.29 menunjukkan halaman utama, di bagian atas, terdapat menu navigasi dengan beberapa pilihan seperti "Home", "Profile", "Package", "Confirmation", "Arsip", dan "Logout".

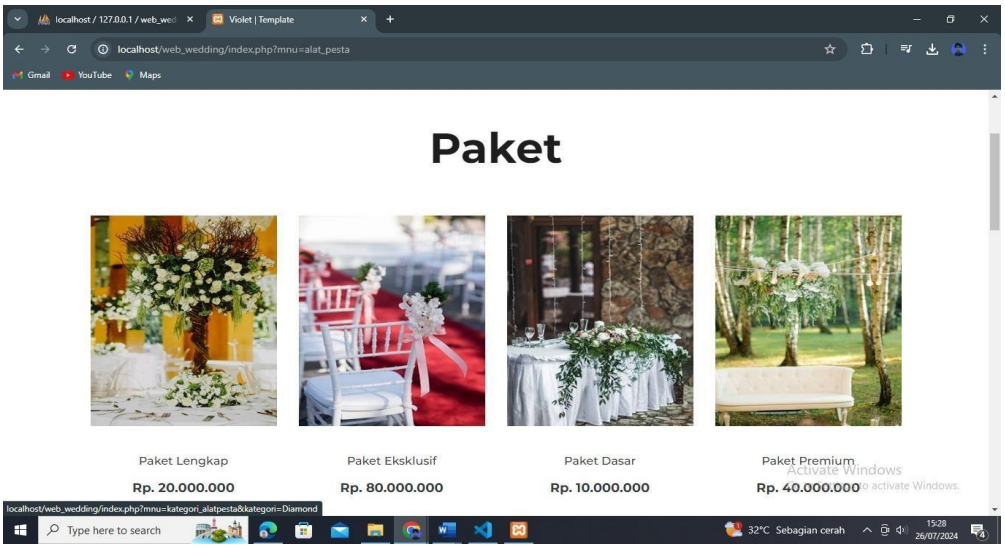
4.2.6 Halaman Profil User



Gambar 4. 30 Halaman Profil User

Gambar 4.30 menunjukkan halaman antarmuka profil pengguna, di bagian kiri, terdapat informasi pengguna dengan nama, termasuk nomor telepon, email, dan alamat lengkap, di bagian kanan, terdapat formulir untuk mengubah profil pengguna, yang mencakup kolom isian untuk nomor telepon, email, alamat, nama pengguna (username), dan kata sandi (password). Terdapat tombol "Simpan" di bawah formulir tersebut untuk mengonfirmasi perubahan data yang dilakukan oleh pengguna. Navigasi menu di bagian atas halaman menyediakan opsi seperti "Home", "Profile", "Package", "Confirmation", "Arsip", dan "Logout".

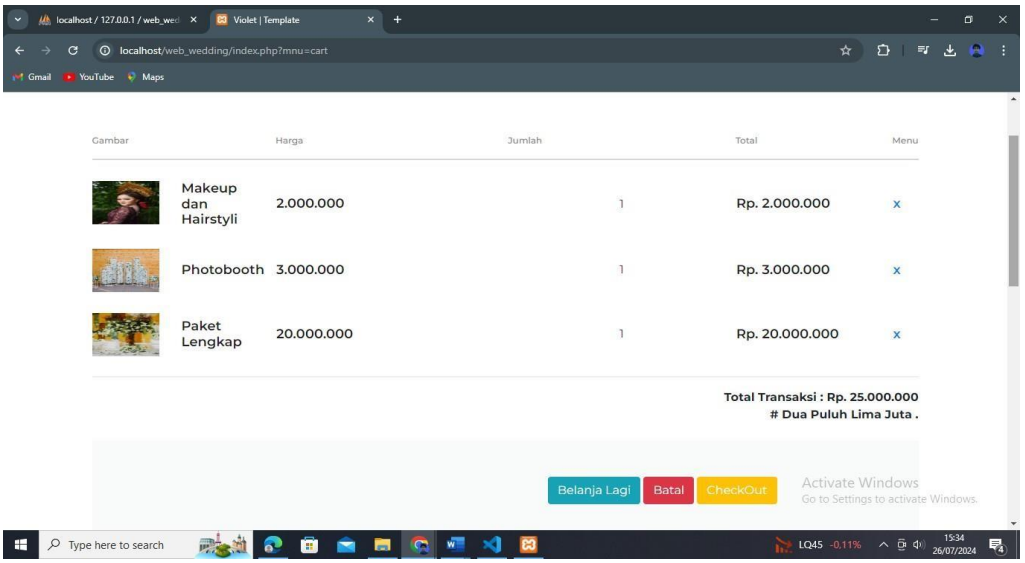
4.2.7 Halaman Paket



Gambar 4. 31 Halaman Paket

Gambar 4.31 menampilkan halaman paket yang menawarkan berbagai paket acara, di bagian atas halaman terdapat judul "Paket" yang mengindikasikan bahwa ini adalah daftar pilihan paket yang tersedia. Ada beberapa paket yang ditampilkan dengan gambar masing-masing dan deskripsi singkat serta harga di bawahnya.

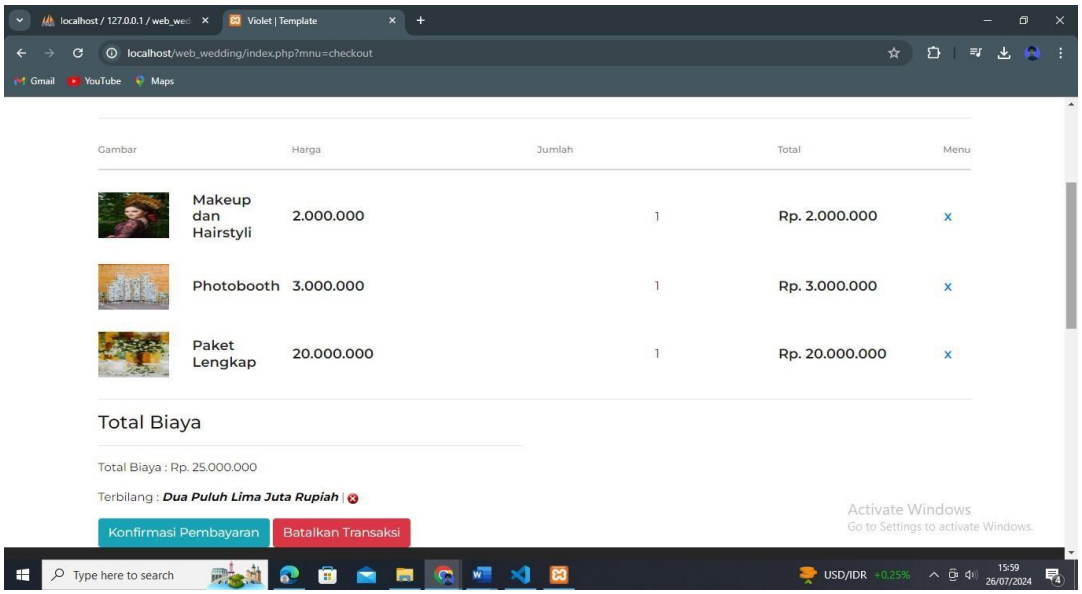
4.2.8 Halaman Keranjang



Gambar 4. 32 Halaman Keranjang

Gambar 4.32 menampilkan halaman keranjang, yang menampilkan setelah pelanggan memilih paket, disertai tombol belanja lagi, batal, dan Checkout.

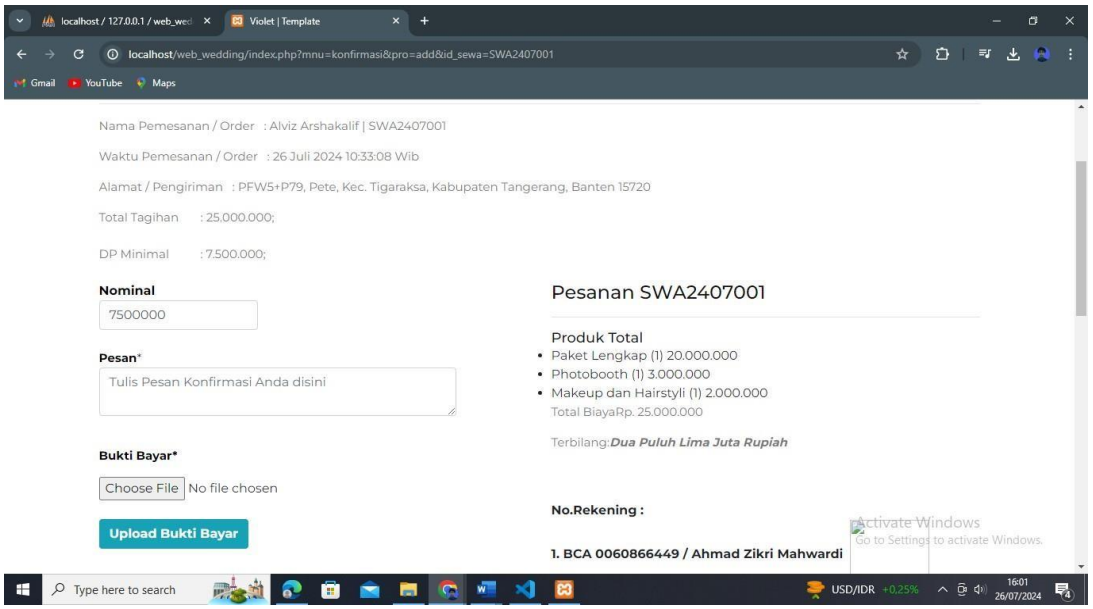
4.2.9 Halaman Check Out



Gambar 4. 33 Halaman Check out

Gambar 4.33 menampilkan halaman Check out, di mana pengguna dapat melihat rincian pesanan, total biaya, dan opsi untuk melanjutkan ke pembayaran atau membatalkan transaksi.

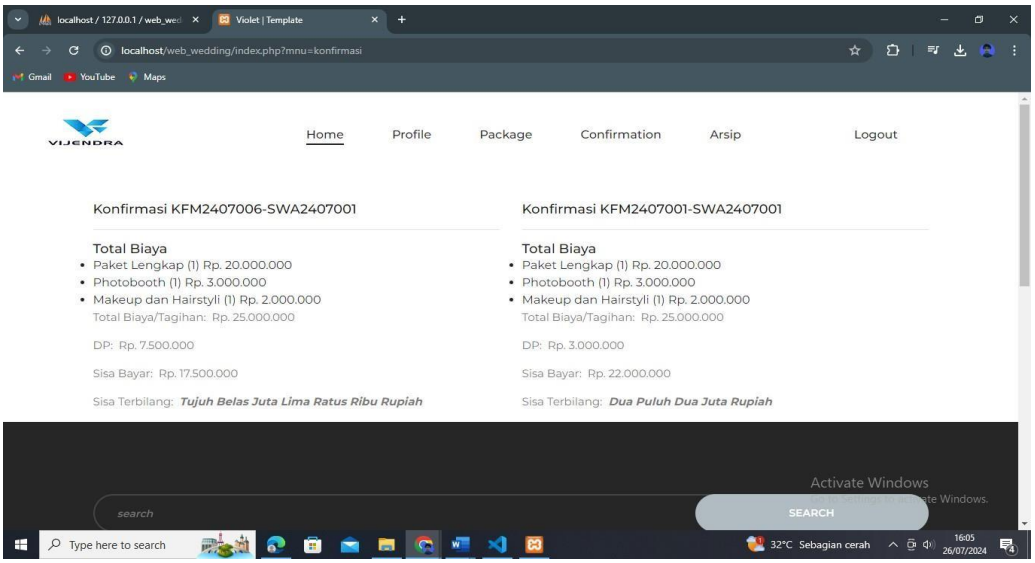
4.2.10 Halaman Pembayaran



Gambar 4. 34 Halaman Pembayaran

Gambar 4.34 menampilkan halaman pembayaran, di mana pelanggan dapat melihat rincian pesanan, total biaya, DP minimal, dan alamat. Dimana pengguna di menginput form nominal dan pesan yang ingin disampaikan, setelah itu pelanggan melakukan pembayaran dengan mengupload bukti.

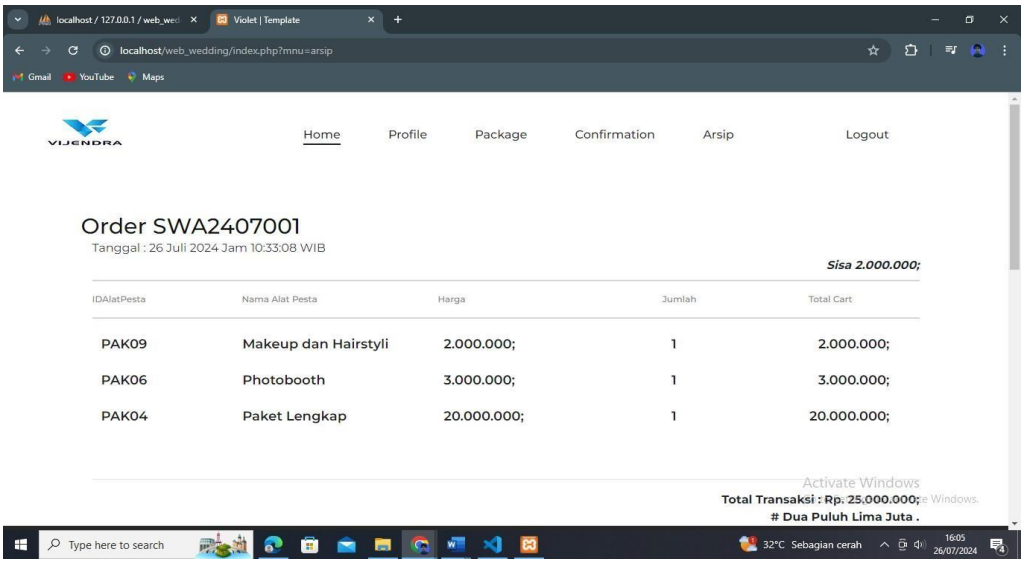
4.2.11 Halaman Konfirmasi



Gambar 4. 35 Halaman Konfirmasi

Gambar 4.35 menampilkan halaman konfirmasi, dimana pelanggan dapat melihat detail informasi setelah melakukan pembayaran, terdiri dari nama paket, total paket, dp yang sudah dibayarkan, dan sisa bayar.

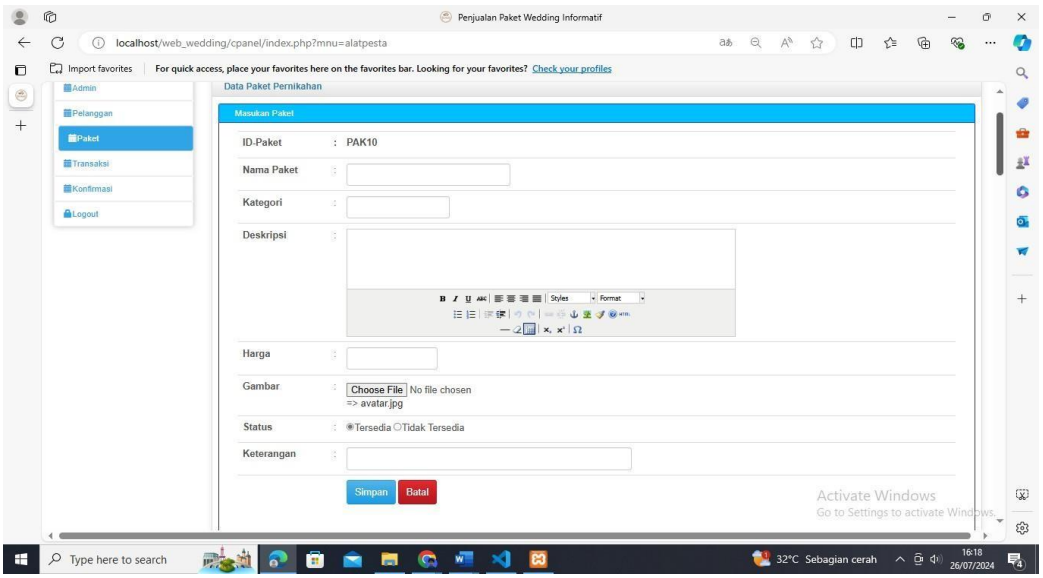
4.2.12 Halaman Arsip



Gambar 4. 36 Halaman Arsip

Gambar 4.36 menampilkan halaman arsip, dimana pengguna dapat melihat table yang berisi id paket, nama paket, harga paket, jumlah paket yang di pesan dari masing masing paket, dan total dari paket yang dipesan.

4.2.13 Halaman Admin Masukan Paket

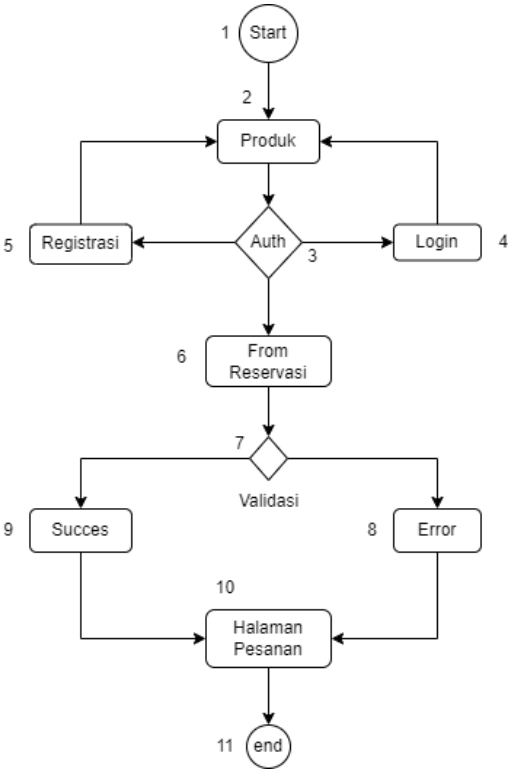


Gambar 4. 37 Halaman Admin Masukan Paket

Gambar 4.37 menampilkan halaman admin masukan paket, terdapat formulir untuk memasukkan data paket baru dengan beberapa kolom input terdiri dari nama paket, kategori, deskripsi paket, harga paket, "tombol choose file=untuk memasukan gambar", status, dan keterangan. Dibagian bawah terdapat tombol simpan dan batal.

4.2.14 Pengujian *White Box*

Pengujian *white box* digunakan untuk memeriksa detail program secara mendalam. Pengujian ini berfokus pada pendeteksian kondisi-kondisi dalam *Sistem* yang tidak sesuai atau mengalami kesalahan logika dalam penulisan program. Berikut adalah hasil pengujian Sistem:



Gambar 4. 38 Flow Chart

1. Listing Program Pesanan

Tabel 4. 7 Listing Program Pesanan

No	Code yang diuji	Buat Tes Code	Hasil

1.	# start.
	class Database:
	def___init_(self):
	self.connected = False
	def connect(self):
	self.connected = True
		# test_start.
		import unittest
		from start import Database, Application
		class TestStart(unittest.TestCase):
		def setUp(self):
		self.db = Database()
		self.app = Application(self.db)
			Ran 4 tests in 0.001s
			OK

```
        return
self.connected
    def disconnect(self):
        self.connected =
False
        return
self.connected
class Application:
    def__init_(self, db):
        self.db = db
        self.initialized =
False
    def initialize(self):
        if
self.db.connect():
            self.initialized =
True
            return
self.initialized
        return False
    def
load_initial_data(self):
        if not
self.initialized:
            raise
Exception("Application
not initialized")
        return ["package1",
"package2",
"package3"]
    def shutdown(self):
```

```
    def
test_database_connection(self):self.assertF
alse(self.db.connected)
        self.db.connect()
      self.assertTrue(self.db.connected)
      self.db.disconnect()
      self.assertFalse(self.db.connected)
    def test_application_initialization(self):
    self.assertFalse(self.app.initialized)
    self.assertTrue(self.app.initialize())
    self.assertTrue(self.app.initialized)
    def test_load_initial_data(self):
        with self.assertRaises(Exception):
            self.app.load_initial_data()
        self.app.initialize()
        data = self.app.load_initial_data()
        self.assertEqual(data, ["package1",
"package2", "package3"])
    def test_application_shutdown(self):
        self.app.initialize()
   self.assertTrue(self.app.shutdown())
    self.assertFalse(self.app.initialized)
    self.assertFalse(self.db.connected)
if__name__==    '__main__':
    unittest.main()
```

	self.db.disconnect() self.initialized = False return True		
2.	<?php function get_sql_query($kat = null, $kategori = null) { $table = 'tbalatpesta'; if ($kat) { return "SELECT * FROM `$table` WHERE `kategori` LIKE '$kat' ORDER BY `nama_alatpesta` ASC"; } elseif ($kategori) { return "SELECT * FROM `$table` WHERE `kategori` LIKE '$kategori' ORDER BY `nama_alatpesta` ASC"; } else { return "SELECT * FROM `$table` ORDER BY `nama_alatpesta` DESC"; } } ?>	<?php use PHPUnit\Framework\TestCase; require 'functions.php'; class FunctionsTest extends TestCase { public function test_get_sql_query_without_parameters() { $expected = "SELECT * FROM `tbalatpesta` ORDER BY `nama_alatpesta` DESC"; $this->assertEquals($expected, get_sql_query()); } public function test_get_sql_query_with_kat() { $kat = "some_category"; $expected = "SELECT * FROM `tbalatpesta` WHERE `kategori` LIKE '$kat' ORDER BY `nama_alatpesta` ASC"; $this->assertEquals($expected, get_sql_query($kat)); } public function test_get_sql_query_with_kategori() { $kategori = "another_category"; $expected = "SELECT * FROM `tbalatpesta` WHERE `kategori` LIKE '$kategori' ORDER BY `nama_alatpesta` ASC"; $this->assertEquals($expected, get_sql_query(null, $kategori)); } } ?>	PHPUnit 9.5.10 by Sebastian Bergmann and contributors. ... Time: 00:00.001, Memory: 6.00 MB OK (3 tests, 3 assertions)
3.	# auth. class User: def__init__(self, username, password):	# test_auth. import unittest from auth import authenticate, register, users_db, User	Ran 4 tests in 0.001s OK

	self.username = username
	self.password = password
	# Simulasi database pengguna
	users_db = [
	User("user1", "password1"),
	User("user2", "password2"),
	]
	def authenticate(username, password):
	for user in users_db:
	if user.username == username and user.password == password:
	return True
	return False
	
	def register(username, password):
	if any(user.username == username for user in users_db):
	return False
		class TestAuth(unittest.TestCase):
		
		def setUp(self):
		# Reset the users_db before each test
		global users_db
		users_db.clear()
		users_db.append(User("user1", "password1"))
		users_db.append(User("user2", "password2"))
		
		def test_authenticate_success(self):
		self.assertTrue(authenticate("user1", "password1"))
		self.assertTrue(authenticate("user2", "password2"))
		
		def test_authenticate_failure(self):
		self.assertFalse(authenticate("user1", "wrongpassword"))
		self.assertFalse(authenticate("user3", "password3"))
		
		def test_register_success(self):
		self.assertTrue(register("newuser", "newpassword"))
		self.assertEqual(len(users_db), 3)
		
		def test_register_failure_existing_user(self):	

	new_user = User(username, password) users_db.append(new_user) return True	self.assertFalse(register("user1", "password1")) self.assertEqual(len(users_db), 2) if __name___== '_main_': unittest.main()	

4.	# login. class User: def __init_(self, username, password):
	self.username = username self.password = password # Simulasi database pengguna
	users_db = [
	User("user1", "password1"),
	User("user2", "password2"),] def login(username, password): for user in users_db: if user.username == username and user.password == password: return "Login successful" return "Login failed"
		# test_login.
		
		import unittest
		from login import login
		
		class TestLogin(unittest.TestCase):
		def test_login_successful(self):
		self.assertEqual(login("user1", "password1"), "Login successful")
		self.assertEqual(login("user2", "password2"), "Login successful")
		
		def test_login_failed_incorrect_password(self):
		self.assertEqual(login("user1", "wrongpassword"), "Login failed")
		self.assertEqual(login("user2", "wrongpassword"), "Login failed")
		def test_login_failed_nonexistent_user(self):
		self.assertEqual(login("nonexistent", "password"), "Login failed")
			Ran 3 tests in 0.001s
			OK

		self.assertEqual(login("user3", "password3"), "Login failed") if __name__ == '__main__': unittest.main()	

5.	<?php
	session_start();
	$users_db = [];
	function is_username_taken($username) {
	global $users_db;
	return in_array($username, array_column($users_db, 'username'));
	}
	function is_email_taken($email) {
	global $users_db;
	return in_array($email, array_column($users_db, 'email'));
	}
	function register($nama_pelanggan, $telepon, $email, $username, $password, $alamat) {
	global $users_db;
		import unittest
		from registration import register, is_username_taken, is_email_taken, clear_users_db
		class TestRegistration(unittest.TestCase):
		def setUp(self):
		clear_users_db()
		def test_register_success(self):
		result = register("John Doe", "123456789", "john@example.com", "johndoe", "password", "123 Street")
		self.assertEqual(result, "Registration successful")
		self.assertEqual(len(users_db), 1)
		def test_register_username_taken(self):
		register("John Doe", "123456789", "john@example.com", "johndoe", "password", "123 Street")
		result = register("Jane Doe", "987654321", "jane@example.com", "johndoe", "password", "456 Avenue")
		self.assertEqual(result, "Username is already taken")
		self.assertEqual(len(users_db), 1)
		def test_register_email_taken(self):
			Ran 3 tests in 0.001s
			OK

	if (is_username_taken($username)) {
	return "Username is already taken";
	}
	if (is_email_taken($email)) {
	return "Email is already registered";
	}
	$new_user = [
	'nama_pelanggan' => $nama_pelanggan,
	'telepon' => $telepon,
	'email' => $email,
	'username' => $username,
	'password' => password_hash($password, PASSWORD_DEFAULT),
	'alamat' => $alamat,
	];
	$users_db[] = $new_user;
		register("John Doe", "123456789", "john@example.com", "johndoe", "password", "123 Street")
		result = register("Jane Doe", "987654321", "john@example.com", "janedoe", "password", "456 Avenue")
		self.assertEqual(result, "Email is already registered")
		self.assertEqual(len(users_db), 1)
		if __name___== '_main_':
		unittest.main()	

	return "Registration successful";
	}
	if ($_SERVER["REQUEST_METHOD"] == "POST") {
	$nama_pelanggan = $_POST['nama_pelanggan'];
	$telepon = $_POST['telepon'];
	$email = $_POST['email'];
	$username = $_POST['username'];
	$password = $_POST['password'];
	$alamat = $_POST['alamat'];
	$result = register($nama_pelanggan, $telepon, $email, $username, $password, $alamat);
	echo $result;
	}
	?>		

6.	# reservation. class Reservation: def __init__(self, name,	# test_reservation. import unittest from reservation import create_reservation, clear_reservations_db, reservations_db	Ran 3 tests in

	date, guests, package): self.name = name self.date = date self.guests = guests self.package = package reservations_db = [] def is_date_available(date): return not any(reservation.date == date for reservation in reservations_db) def create_reservation(nam e, date, guests, package): if not is_date_available(date): return "Date is not available" if guests <= 0: return "Number of guests must be greater than zero" new_reservation = Reservation(name, date, guests, package) reservations_db.append(new_reservation) return "Reservation successful" def clear_reservations_db(): global reservations_db reservations_db = []	class TestReservation(unittest.TestCase): def setUp(self): clear_reservations_db() def test_create_reservation_success(self): result = create_reservation("John Doe", "2024-08-10", 5, "Gold Package") self.assertEqual(result, "Reservation successful") self.assertEqual(len(reservations_db), 1) self.assertEqual(reservations_db[0].name, "John Doe") self.assertEqual(reservations_db[0].date, "2024-08-10") self.assertEqual(reservations_db[0].guests, 5) def test_create_reservation_date_not_available (self): create_reservation("John Doe", "2024-08-10", 5, "Gold Package") result = create_reservation("Jane Doe", "2024-08-10", 3, "Silver Package") self.assertEqual(result, "Date is not available") self.assertEqual(len(reservations_db), 1) def test_create_reservation_invalid_guests(self): result = create_reservation("John Doe", "2024-08-10", 0, "Gold Package") self.assertEqual(result, "Number of guests must be greater than zero") self.assertEqual(len(reservations_db), 0) if __name__ == '__main__': unittest.main()	0.001s OK

7.	class User: def __init (self, name, email): self.name = name self.email = email self.registration_success = False self.reservation_success = False def register_user(name, email): if not name or not email: return "Invalid input" user = User(name, email) user.registration_success = True return user def make_reservation(user, date, package): if not user.registration_success: return "User not registered" if not date or not package: return "Invalid reservation details" user.reservation_success = True return user def success_message(user): if user.registration_success and user.reservation_success: return	import unittest from success import register_user, make_reservation, success_message, User class TestSuccess(unittest.TestCase): def test_register_user_success(self): user = register_user("John Doe", "john@example.com") self.assertIsInstance(user, User) self.assertTrue(user.registration_success) self.assertEqual(user.name, "John Doe") self.assertEqual(user.email, "john@example.com") def test_register_user_invalid_input(self): result = register_user("", "john@example.com") self.assertEqual(result, "Invalid input") result = register_user("John Doe", "") self.assertEqual(result, "Invalid input") def test_make_reservation_success(self): user = register_user("John Doe", "john@example.com") user = make_reservation(user, "2024-08-10", "Gold Package") self.assertTrue(user.reservation_success) def test_make_reservation_user_not_registered (self): user = User("John Doe", "john@example.com") result = make_reservation(user, "2024-08-10", "Gold Package") self.assertEqual(result, "User not registered") def	Ran 7 tests in 0.001s OK

	f"Congratulations {user.name}, your reservation is confirmed!" elif user.registration_success: return f"Congratulations {user.name}, your registration is successful!" return "No success status found"	test_make_reservation_invalid_details(self): user = register_user("John Doe", "john@example.com") result = make_reservation(user, "", "Gold Package") self.assertEqual(result, "Invalid reservation details") result = make_reservation(user, "2024-08-10", "") self.assertEqual(result, "Invalid reservation details") def test_success_message(self): user = register_user("John Doe", "john@example.com") user = make_reservation(user, "2024-08-10", "Gold Package") message = success_message(user) self.assertEqual(message, "Congratulations John Doe, your reservation is confirmed!") user = register_user("Jane Doe", "jane@example.com") message = success_message(user) self.assertEqual(message, "Congratulations Jane Doe, your registration is successful!") user = User("No Success", "no@success.com") message = success_message(user) self.assertEqual(message, "No success status found") if__name__== '_main_': unittest.main()	
8.	def find_user(user_id): users = {"1": "John Doe", "2": "Jane Doe"} if user_id not in users:	import unittest from error_handling import find_user, create_reservation, validate_input, UserNotFoundError, ReservationError, InputValidationError	Ran 10 tests in 0.001s OK

	raise UserNotFoundError("User not found") return users[user_id] def create_reservation(user_id, date, guests): if guests <= 0: raise ReservationError("Number of guests must be greater than zero") user = find_user(user_id) if not date: raise ReservationError("Reservation date is required") return f"Reservation created for {user} on {date} for {guests} guests" def validate_input(name, email): if not name: raise InputValidationError("Name is required") if not email: raise InputValidationError("Email is required") if "@" not in email: raise InputValidationError("Email is invalid") return "Valid input"	class TestErrorHandling(unittest.TestCase): def test_find_user_success(self): user = find_user("1") self.assertEqual(user, "John Doe") def test_find_user_not_found(self): with self.assertRaises(UserNotFoundError): find_user("3") def test_create_reservation_success(self): result = create_reservation("1", "2024-08-10", 5) self.assertEqual(result, "Reservation created for John Doe on 2024-08-10 for 5 guests") def test_create_reservation_invalid_guests(self): with self.assertRaises(ReservationError): create_reservation("1", "2024-08-10", 0) def test_create_reservation_user_not_found(self): with self.assertRaises(UserNotFoundError): create_reservation("3", "2024-08-10", 5) def test_create_reservation_no_date(self): with self.assertRaises(ReservationError): create_reservation("1", "", 5) def test_validate_input_success(self): result = validate_input("John Doe", "john@example.com") self.assertEqual(result, "Valid input") def test_validate_input_no_name(self): with self.assertRaises(InputValidationError): validate_input("", "john@example.com")	

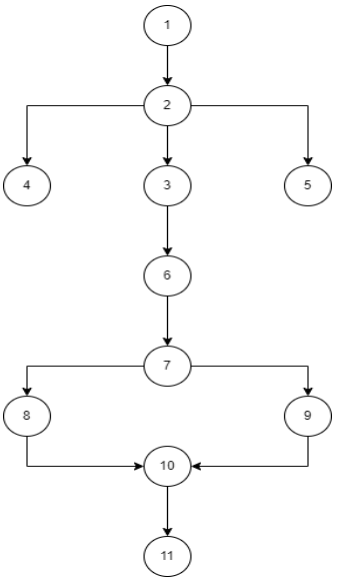
		def test_validate_input_no_email(self): with self.assertRaises(InputValidationError): validate_input("John Doe", "") def test_validate_input_invalid_email(self): with self.assertRaises(InputValidationError): validate_input("John Doe", "johnexample.com") if _name___== '_main__': unittest.main()	

9.	<?php class Product { private $conn; private $table; public function __construct($conn, $table) { $this->conn = $conn; $this->table = $table; } public function getRandomProducts() { $sqlc = "SELECT * FROM `$this->table` ORDER BY RAND()"; return getData($this->conn, $sqlc); } public function formatProduct($product) { $kategori = $product["kategori"]; $id_alatpesta = $product["id_alatpesta"]; $nama_alatpesta = $product["nama_alatpes	<?php
		use PHPUnit\Framework\TestCase;
		
		class ProductTest extends TestCase
		{
		private $conn;
		private $product;
		
		protected function setUp(): void
		{
		$this->conn = $this->createMock(PDO::class);
		$this->product = new Product($this->conn, 'tbalatpesta');
		}
		
		public function testGetRandomProducts()
		{
		$expected = [
			Time: 00:00.012, Memory: 4.00 MB OK (2 tests, 2 assertions)

	ta"]; $rank_harga = $product["rank_harga"]; $harga = RP($product["harga"]); $gambar = $product["gambar"]; return " <figure> <img src='cpanel/ypathfile/$gambar' alt='$gambar' title='$nama_alatpesta'> </figure> $nama_alatpesta Rp $harga "; } }	["id_alatpesta" => 1, "nama_alatpesta" => "Product 1", "kategori" => "Category 1", "rank_harga" => "Rank 1", "harga" => 1000, "gambar" => "image1.jpg"], ["id_alatpesta" => 2, "nama_alatpesta" => "Product 2", "kategori" => "Category 2", "rank_harga" => "Rank 2", "harga" => 2000, "gambar" => "image2.jpg"]]; $this->conn->method('query') ->willReturn($expected); $result = $this->product->getRandomProducts(); $this->assertEquals($expected, $result); } public function testFormatProduct() { $product = ["id_alatpesta" => 1, "nama_alatpesta" => "Product 1", "kategori" => "Category 1", "rank_harga" => "Rank 1", "harga" => 1000, "gambar" => "image1.jpg"]; $formattedProduct = $this->product->formatProduct($product);	

		$expected = " <figure> <img src='cpanel/ypathfile/image1.jpg' alt='image1.jpg' title='Product 1'> </figure> Product 1 Rp 1000 "; $this->assertEquals($expected, $formattedProduct); } }	
10.	}		

2. Flow Graph Pesanan



Gambar 4. 39 Flow Graph

Berdasarkan urutan alurnya, di dapatkan suatu kelompok basis flowgraph:

Jalur 1: 1 - 2 - 3 - 5 - 6 - 8 - 9 - 10 - 11

Jalur 2: 1 - 2 - 3 - 4 - 2 - 3 - 5 - 6 - 8 - 9 - 10 - 11

Jalur 3: 1 - 2 - 3 - 4 - 2 - 3 - 5 - 6 - 7 - 9 - 10 – 11

3. Pengujian Basic Path

No	Input	Proses	Keterangan
1.	Klik Order Now	Tampil Form Pemesanan lali klik send order	Pelanggan Melakukan Pemesanan Paket
2.	Klik Order Now	Tampil Halaman Login	Pelanggan Yang Belum Melakukan Login Akan di arahkan ke halaman login terlebih dahulu untuk melakukan pemesanan.
3.	Klik Order Now	Tampil Form Pemesanan lali klik send order	Pelanggan Melakukan Pemesanan Paket tapi pesanan di tolak karena masih ada pemesanan yang masih di proses

4.2.15 Pengujian Black Box

Penulis mendapatkan evaluasi dari responden yang telah melakukan uji coba Sistem. Berikut ini adalah hasil tes dan evaluasi dari responden.

Tabel 4. 8 Pengujian Black Box

No	Fungsi yang Diuji	Cara yang Dilakukan	Hasil yang Diharapkan	Hasil Pengujian
1.	Fungsi halaman utama	Membuka halaman utama Bawor Netwrokpada web.	Menampilkan halaman utama website, beserta menu-menu seperti menu profil, menu paket, keranjang pemesanan, checkout, konfirmasi, dan arsip.	[✓]Diterima []Ditolak
2.	Fungsi Register	Pilih menu login pada tampilan halaman utama , setelah itu klik *registrasi*	Menampilkan formulir pendaftaran, di mana pelanggan dapat mengisi formulir dan mendaftar sebagai pelanggan.	[✓]Diterima []Ditolak
3.	Fungsi Login pada web	pelanggan memasukkan username dan password dengan benar	pelanggan masuk ke dalam Sistem menampilkan halaman utama	[✓]Diterima []Ditolak
4.	Fungsi Login pada web	Pelanggan memasukkan username dan password yang salah	Menampilkan pemberitahuanLogin salah, pelanggan tetap berada dalam halaman login	[✓]Diterima []Ditolak
5.	Fungsi Profile	Pilih menu profile setelah melakukan login.	Menampilkan form profile pelanggan berupa nama lengkap, Alamat, nomor, dan email	[✓]Diterima []Ditolak

6.	Fungsi Keranjang	Tambahkan paket yang tersedia ke dalam keranjang.	Paket ditambahkan ke keranjang, jumlah item di keranjang bertambah.	[✓]Diterima []Ditolak
7.	Fungsi Belanja lagi pada Keranjang	Tambahkan paket yang sudah ada di keranjang.	Jumlah item bertambah sesuai dengan penambahan.	[✓]Diterima []Ditolak
8.	Fungsi cancel Pada Keranjang	Hapus paket dari keranjang.	Item berhasil dihapus dari keranjang, jumlah item berkurang.	[✓]Diterima []Ditolak
9.	Fungsi CheckOut pada Keranjang	Lakukan proses checkout dengan item yang ada di keranjang.	Proses checkout berhasil, dan pengguna diarahkan ke halaman konfirmasi atau pembayaran.	[✓]Diterima []Ditolak
10.	Fungsi Pembayaran pada Check out	Melakukan pembayaran melalui bank yang tertera	Formulir berhasil disubmit, dan pengguna diarahkan ke halaman konfirmasi pembayaran.	[✓]Diterima []Ditolak
11.	Fungsi Batal pada Check out	Klik tombol batal	Sistem membatalkan proses checkout dan pengguna diarahkan kembali ke halaman sebelumnya atau halaman keranjang belanja, tanpa menyimpan data yang diisi.	[✓]Diterima []Ditolak
12.	Fungsi Choose File pada Pembayaran	Pilih file dengan format yang valid (misalnya, .jpg, .png, .pdf) dan ukuran file yang diperbolehkan.	File berhasil dipilih, dan nama file tampil di form upload.	[✓]Diterima []Ditolak
13.	Fungsi Button Upload Bukti Bayar pada Pembayaran	Periksa apakah tombol "Upload Bukti Bayar" terlihat jelas di	Tombol terlihat jelas dan mudah diakses oleh pengguna.	[✓]Diterima []Ditolak

		halaman pembayaran.		
14.	Fungsi menu Confirmation	Periksa apakah menu "Confirmation" terlihat jelas dan dapat diakses dari halaman utama atau halaman terkait lainnya.	Menu "Confirmation" terlihat jelas dan dapat diakses dengan satu klik.	[✓]Diterima []Ditolak
15.	Fungsi menu Arsip	Klik menu "Arsip" dan periksa apakah pengguna diarahkan ke halaman arsip yang benar.	Pengguna diarahkan ke halaman arsip yang sesuai tanpa adanya error atau masalah navigasi.	[✓]Diterima []Ditolak
16.	Fungsi Logout	Klik tombol/logout dan periksa apakah pengguna berhasil dikeluarkan dari sesi.	Pengguna berhasil keluar dari aplikasi dan diarahkan ke halaman login atau halaman utama yang tidak memerlukan autentikasi.	[✓]Diterima []Ditolak

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil penelitian yang telah dilakukan mengenai perancangan Sistem informasi Event Organizer berbasis web pada PT Vijendra Project, dapat disimpulkan, sebagai berikut:

1. Sistem Informasi Event Organizer berbasis website berhasil tercipta, memfasilitasi interaksi antara admin dan pelanggan dalam pemesanan paket pernikahan.
2. Fitur-fitur yang disediakan pada sistem ini, seperti kemudahan akses informasi tentang paket pernikahan, pengecekan ketersediaan tanggal, dan proses pemesanan. Memudahkan pelanggan dalam melakukan pemesanan paket pernikahan.

5.2 Saran

Berdasarkan kesimpulan yang telah disampaikan, terdapat beberapa saran yang dapat diberikan untuk pengembangan Sistem ke depan:

1. Penambahan fitur chat atau obrolan langsung untuk komunikasi cepat antara pelanggan dan Vijendra project Event Organizer, untuk lebih meningkatkan layanan Pelanggan.
2. Penerapan virtual account pada proses pembayaran akan memudahkan Pelanggan dalam melakukan pembayaran tanpa perlu mengirim bukti transfer secara manual.

DAFTAR PUSTAKA

Agustin, D., Bani, A. U., & Fauziyah. (2020). Perancangan Sistem Informasi Jasa Event Organizer pada CV. Boganesia Jaya Berbasis Web. *Jurnal Jaring SainTek (JJST)*, *2*(2), 15–26.

Ahmad. (2020). *Pengertian Sequence Diagram : Tujuan, Simbol, dan Contohnya*.

Arifin, N. Y., Tyas, S. S., & Sulistiani, H. (2022). *Analisa Perancangan Sistem Informasi*. Cendikia Mulia Mandiri.

Arthantio, A. (2022). *Aplikasi Buku Tamu Berbasis Website Pada Dinas Pemberdayaan Masyarakat Dan Desa Provinsi Sumatera Selatan*. Politeknik Negeri Sriwijaya.

Aulianita, R. (2019). User Center Design Dalam Membangun Event Organizer Berbasis Website. *Information Management For Educators And Professionals: Journal of Information Management*, *4*(1), 31–40.

Azis, A., Setiawan, I., Krisbiantoro, D., & Riyanto. (2019). *Panduan Pemilu Desa Berbasis Website(Teknologi Sistem Cerdas dan Implementasi di Masyarakat)*. Deepublish.

Analisis dan Perancangan Sistem Informasi Menggunakan Model Terstruktur dan UML. (n.d.). (n.p.): Penerbit Andi.

Chafid, N., & Mulyawan, A. (2023). Aplikasi Location Based Service Event Organizer Di Kota Tangerang Selatan Berbasis Android. *Jurnal Satya Informatika*, *2*(1), 36–51. https://doi.org/10.59134/jsk.v2i1.369

Comer, D. E. (2019). *The Internet book: everything you need to know about computer networking and how the Internet works*. Chapman and Hall/CRC.

Faisal, M. R., & Abadi, F. (2020). *Pemrograman Web Dasar I: Belajar HTML*. Scripta Cendekia.

Fathoroni, A., Fatonah, R. N. S., Andarsyah, R., & Riza, N. (2020). *Buku tutorial sistem pendukung keputusan penilaian kinerja dosen menggunakan metode 360 degree feedback*. CV. Kreatif Industri Nusantara.

Hendri, H., Meisak, D., & Rianti, S. (2022). Workshop Dampak Perkembangan Teknologi Informasi Dalam Masa Pandemic Covid 19 Di Indonesia. *Jurnal Pengabdian Masyarakat UNAMA*, *1*(2), 83–92. https://doi.org/10.33998/jpmu.2022.1.2.75

Kaya DarI BIsnIs Event Organizer. (2022). (n.p.): Pustaka Ananda Srva .

Limbong, T. (2021). *Pemrograman Web Dasar*. Unimed.

Miftachudin, M. (2022). Penerapan Sistem Ujian Online Terhadap Kemampuan Dasar Pemrograman PHP Berbasis Website. *Teknologipintar*, 1–12.

Munawar. (2021). *Analisis Perancangan Sistem Berorientasi Objek dengan (Unified Modeling Language)*. Bandung Informatika.

Nugroho, A. (2020). *Rancang Bangun Sistem Informasi Geografis Pencarian Rute Tercepat Lokasi Penjualan Oleh-Oleh Khas Lombok dengan Menggunakan Fungsi Waypoint dan Metode Heuristik Greedy di Kota Mataram Berbasis Web*. Universitas Mataram.

Nugroho, S., & Primadewi, A. (2020). Penerapan Web Service pada Integrasi Database Simperpus dan SIAK Menggunakan Rest API. *Jurnal Komtika (Komputasi Dan Informatika)*, *4*(2), 71–81.

Oetomo, I. H. W., & Mahargiono, P. B. (2020). *E-Commerce Aplikasi PHP dan MySQL pada Bidang Manajemen: Program Studi S-1 Manajemen Tahun Ajaran 2019/2020*. Andi.

Prehanto. (2020). *Buku Ajar Konsep Sistem Informasi*. Scopindo Media Pustaka.

Ridwan, M., Widiastiwi, Y., Zaidiah, A., Purabaya, R. H., Isnainiyah, I. N., Ardilla, Y., & Rahayu, T. (2021). *Sistem informasi manajemen*. Penerbit Widina.

Riyani, D., & Hendradi, P. (2016). Sistem Informasi Pemesanan Paket Pernikahan Pada Inspiration Beauty Sanggar Rias Indah. *Jurnal Ilmiah Fakultas Teknik LIMIT'S*, *12*(2), 67–74.

Saad, M. I. (2020). *Otodidak Web Programming: Membuat Website Edutainment*. Elex Media Komputindo.

Salamah, U. G. (2021). *Tutorial Visual Studio Code*. Media Sains Indonesia.

Salsabila, A., Herawati, E., & Atmanto, D. (2024). Analisis Faktor-Faktor Yang Mempengaruhi Kepuasan Pelanggan Shofi Event Organizer di Bekasi. *Jurnal Adijaya Multidisplin*, *2*(03), 640–663.

Sari, I. R. F., & Utami, A. (2021). *Rekayasa Perangkat Lunak Berorientasi Objek Menggunakan PHP*. Andi.

Simatupang, J., & Sianturi, S. (2019). Perancangan sistem informasi pemesanan tiket bus pada po. Handoyo berbasis online. *Jurnal Intra-Tech*, *3*(2), 11–25.

Speight, A. (2021). *Visual studio code for python programmers*. John Wiley & Sons.

Pengantar Sistem Informasi. (n.d.). (n.p.): Penerbit Andi.

Syafrizal, M. (2020). *Pengantar jaringan komputer*. Andi.

Widarti, E., Joosten, J., Pratiwi, P. Y., Pradnyana, G. A Indradewi, I. G. A. A. D., Kamilah, N., & Sepriano, S. (2024). *Buku Ajar Pengantar Sistem Informasi*. PT. Sonpedia Publishing Indonesia.

Listing Program Pesanan

No	Code yang diuji	Buat Tes Code	Hasil
1.	*# start.* *class Database:* *def___init_(self):* *self.connected = False* *def connect(self):* *self.connected = True*	*# test_start.* *import unittest* *from start import Database, Application* *class TestStart(unittest.TestCase):* *def setUp(self):* *self.db = Database()* *self.app = Application(self.db)*	*Ran 4 tests in 0.001s* *OK*

```
        return
self.connected
    def disconnect(self):
        self.connected =
False
        return
self.connected
class Application:
    def __init (self, db):
        self.db = db
        self.initialized =
False
    def initialize(self):
        if
self.db.connect():
            self.initialized =
True
            return
self.initialized
        return False
    def
load_initial_data(self):
        if not
self.initialized:
            raise
Exception("Application
not initialized")
        return
["package1",
"package2",
"package3"]
    def shutdown(self):
```

```
    def
test_database_connection(self):self.assertF
alse(self.db.connected)
        self.db.connect()
      self.assertTrue(self.db.connected)
      self.db.disconnect()
      self.assertFalse(self.db.connected)
    def test_application_initialization(self):
    self.assertFalse(self.app.initialized)
    self.assertTrue(self.app.initialize())
    self.assertTrue(self.app.initialized)
    def test_load_initial_data(self):
        with self.assertRaises(Exception):
            self.app.load_initial_data()
        self.app.initialize()
        data = self.app.load_initial_data()
        self.assertEqual(data,
        ["package1",
"package2", "package3"])
    def test_application_shutdown(self):
        self.app.initialize()
   self.assertTrue(self.app.shutdown())
    self.assertFalse(self.app.initialized)
    self.assertFalse(self.db.connected)
if__name__==    '__main__':
    unittest.main()
```

	self.db.disconnect() *self.initialized = False* *return True*		
2.	*<?php function get_sql_query($kat = null, $kategori = null) { $table = 'tbalatpesta'; if ($kat) { return "SELECT * FROM `$table` WHERE `kategori` LIKE '$kat' ORDER BY `nama_alatpesta` ASC"; } elseif ($kategori) { return "SELECT * FROM `$table` WHERE `kategori` LIKE '$kategori' ORDER BY `nama_alatpesta` ASC"; } else { return "SELECT * FROM `$table` ORDER BY `nama_alatpesta` DESC"; } } ?>*	*<?php use PHPUnit\Framework\TestCase; require 'functions.php'; class FunctionsTest extends TestCase { public function test_get_sql_query_without_parameters() { $expected = "SELECT * FROM `tbalatpesta` ORDER BY `nama_alatpesta` DESC"; $this->assertEquals($expected, get_sql_query()); } public function test_get_sql_query_with_kat() { $kat = "some_category"; $expected = "SELECT * FROM `tbalatpesta` WHERE `kategori` LIKE '$kat' ORDER BY `nama_alatpesta` ASC"; $this->assertEquals($expected, get_sql_query($kat)); } public function test_get_sql_query_with_kategori() { $kategori = "another_category"; $expected = "SELECT * FROM `tbalatpesta` WHERE `kategori` LIKE '$kategori' ORDER BY `nama_alatpesta` ASC"; $this->assertEquals($expected, get_sql_query(null, $kategori)); } } ?>*	*PHPUnit 9.5.10 by Sebastian Bergmann and contributors. ... Time: 00:00.001, Memory: 6.00 MB OK (3 tests, 3 assertio ns)*
3.	*# auth.* *class User:* *def___init__(self, username, password):*	*# test_auth.* *import unittest* *from auth import authenticate, register, users_db, User*	*Ran 4 tests in 0.001s OK*

	self.username = username
	self.password = password
	# Simulasi database pengguna
	users_db = [
	User("user1", "password1"),
	User("user2", "password2"),
	]
	def authenticate(username, password):
	for user in users_db:
	if user.username == username and user.password == password:
	return True
	return False
	
	def register(username, password):
	if any(user.username == username for user in users_db):
	return False
		class TestAuth(unittest.TestCase):
		
		def setUp(self):
		# Reset the users_db before each test
		global users_db
		users_db.clear()
		users_db.append(User("user1", "password1"))
		users_db.append(User("user2", "password2"))
		
		def test_authenticate_success(self):
		self.assertTrue(authenticate("user1", "password1"))
		self.assertTrue(authenticate("user2", "password2"))
		
		def test_authenticate_failure(self):
		self.assertFalse(authenticate("user1", "wrongpassword"))
		self.assertFalse(authenticate("user3", "password3"))
		
		def test_register_success(self):
		self.assertTrue(register("newuser", "newpassword"))
		self.assertEqual(len(users_db), 3)
		
		def test_register_failure_existing_user(self):	

	new_user = User(username, password) *users_db.append(new_user)* *return True*	*self.assertFalse(register("user1", "password1"))* *self.assertEqual(len(users_db), 2) if __name__ == '__main__': unittest.main()*	
4.	*# login. class User: def __init__(self, username, password): self.username = username self.password = password # Simulasi database pengguna users_db = [User("user1", "password1"), User("user2", "password2"),] def login(username, password): for user in users_db: if user.username == username and user.password == password: return "Login successful" return "Login failed"*	*# test_login.* *import unittest* *from login import login* *class TestLogin(unittest.TestCase):* *def test_login_successful(self):* *self.assertEqual(login("user1", "password1"), "Login successful")* *self.assertEqual(login("user2", "password2"), "Login successful")* *def test_login_failed_incorrect_password(self):* *self.assertEqual(login("user1", "wrongpassword"), "Login failed")* *self.assertEqual(login("user2", "wrongpassword"), "Login failed")* *def test_login_failed_nonexistent_user(self):* *self.assertEqual(login("nonexistent", "password"), "Login failed")*	*Ran 3 tests in 0.001s OK*

		self.assertEqual(login("user3", "password3"), "Login failed") if __name__ == '__main__': unittest.main()	

5.	<?php
	session_start()
	;
	$users_db = [];
	function
	is_username_taken($username) {
	global
	$users_db;
	return
	in_array($username, array_column($users_db, 'username'));
	}
	function
	is_email_taken($email)
	{
	global
	$users_db;
	return
	in_array($email, array_column($users_db, 'email'));
	}
	function
	register($nama_pelanggan, $telepon, $email, $username, $password, $alamat) {
	global $users_db;
		import unittest
		from registration import register, is_username_taken, is_email_taken, clear_users_db
		class TestRegistration(unittest.TestCase):
		def setUp(self):
		clear_users_db()
		def test_register_success(self):
		result = register("John Doe", "123456789", "john@example.com", "johndoe", "password", "123 Street")
		self.assertEqual(result, "Registration successful")
		self.assertEqual(len(users_db), 1)
		def test_register_username_taken(self):
		register("John Doe", "123456789", "john@example.com", "johndoe", "password", "123 Street")
		result = register("Jane Doe", "987654321", "jane@example.com", "johndoe", "password", "456 Avenue")
		self.assertEqual(result, "Username is already taken")
		self.assertEqual(len(users_db), 1)
		def test_register_email_taken(self):
			Ran 3 tests in 0.001s
			OK

	if (is_username_taken($username)) {* *return "Username is already taken";* *}* *if (is_email_taken($email)) { *return "Email is already registered";* *}* *$new_user = [* *'nama_pelanggan' =>* *$nama_pelanggan, 'telepon' => $telepon,* *'email' => $email,* *'username' => $username,* *'password' => password_hash($password, PASSWORD_DEFAULT),* *'alamat' => $alamat,* *];* *$users_db[] = $new_user;*	*register("John Doe", "123456789", "john@example.com", "johndoe", "password", "123 Street")* *result = register("Jane Doe", "987654321", "john@example.com", "janedoe", "password", "456 Avenue")* *self.assertEqual(result, "Email is already registered")* *self.assertEqual(len(users_db), 1)* *if __name___== '_main_':* *unittest.main()*	

	return "Registration successful";
	}
	if ($_SERVER["REQUEST_METHOD"] == "POST") {
	$nama_pelanggan = $_POST['nama_pelanggan'];
	$telepon = $_POST['telepon'];
	$email = $_POST['email'];
	$username = $_POST['username'];
	$password = $_POST['password'];
	$alamat = $_POST['alamat'];
	$result = register($nama_pelanggan, $telepon, $email, $username, $password, $alamat);
	echo $result;
	}
	?>		

6.	#reservation. class Reservation: def __init__(self, name,	#test_reservation. import unittest from reservation import create_reservation, clear_reservations_db, reservations_db	Ran 3 tests in

	date, guests, package): self.name = name self.date = date self.guests = guests self.package = package reservations_db = [] def is_date_available(date): return not any(reservation.date == date for reservation in reservations_db) def create_reservation(name, date, guests, package): if not is_date_available(date): return "Date is not available" if guests <= 0: return "Number of guests must be greater than zero" new_reservation = Reservation(name, date, guests, package) reservations_db.append(new_reservation) return "Reservation successful" def clear_reservations_db() : global reservations_db reservations_db = []	*class TestReservation(unittest.TestCase): def setUp(self): clear_reservations_db() def test_create_reservation_success(self): result = create_reservation("John Doe", "2024-08-10", 5, "Gold Package") self.assertEqual(result, "Reservation successful") self.assertEqual(len(reservations_db), 1) self.assertEqual(reservations_db[0].name , "John Doe") self.assertEqual(reservations_db[0].date, "2024-08-10") self.assertEqual(reservations_db[0].guests, 5) def test_create_reservation_date_not_availabl e (self): create_reservation("John Doe", "2024-08-10", 5, "Gold Package") result = create_reservation("Jane Doe", "2024-08-10", 3, "Silver Package") self.assertEqual(result, "Date is not available") self.assertEqual(len(reservations_db), 1) def test_create_reservation_invalid_guests(self): result = create_reservation("John Doe", "2024-08-10", 0, "Gold Package") self.assertEqual(result, "Number of guests must be greater than zero") self.assertEqual(len(reservations_db), 0) if __name__== '__main__': unittest.main()*	*0.001s OK*

7.	*class User: def __init (self, name, email): self.name = name self.email = email self.registration_success = False self.reservation_success = False def register_user(name, email): if not name or not email: return "Invalid input" user = User(name, email) user.registration_success = True return user def make_reservation(user, date, package): if not user.registration_success: return "User not registered" if not date or not package: return "Invalid reservation details" user.reservation_success = True return user def success_message(user): if user.registration_success and user.reservation_success : return*	*import unittest from success import register_user, make_reservation, success_message, User class TestSuccess(unittest.TestCase): def test_register_user_success(self): user = register_user("John Doe", "john@example.com") self.assertIsInstance(user, User) self.assertTrue(user.registration_success) self.assertEqual(user.name, "John Doe") self.assertEqual(user.email, "john@example.com") def test_register_user_invalid_input(self): result = register_user("", "john@example.com") self.assertEqual(result, "Invalid input") result = register_user("John Doe", "") self.assertEqual(result, "Invalid input") def test_make_reservation_success(self): user = register_user("John Doe", "john@example.com") user = make_reservation(user, "2024-08-10", "Gold Package") self.assertTrue(user.reservation_success) def test_make_reservation_user_not_registered (self): user = User("John Doe", "john@example.com") result = make_reservation(user, "2024-08-10", "Gold Package") self.assertEqual(result, "User not registered") def*	*Ran 7 tests in 0.001s OK*

	f"Congratulations {user.name}, your reservation is confirmed!" elif user.registration_success: return f"Congratulations {user.name}, your registration is successful!" return "No success status found"	*test_make_reservation_invalid_details(self): user = register_user("John Doe", "john@example.com") result = make_reservation(user, "", "Gold Package") self.assertEqual(result, "Invalid reservation details") result = make_reservation(user, "2024-08-10", "") self.assertEqual(result, "Invalid reservation details") def test_success_message(self): user = register_user("John Doe", "john@example.com") user = make_reservation(user, "2024-08-10", "Gold Package") message = success_message(user) self.assertEqual(message, "Congratulations John Doe, your reservation is confirmed!") user = register_user("Jane Doe", "jane@example.com") message = success_message(user) self.assertEqual(message, "Congratulations Jane Doe, your registration is successful!") user = User("No Success", "no@success.com") message = success_message(user) self.assertEqual(message, "No success status found") if __name__ == '_main_': unittest.main()*	
8.	*def find_user(user_id): users = {"1": "John Doe", "2": "Jane Doe"} if user_id not in users:*	*import unittest from error_handling import find_user, create_reservation, validate_input, UserNotFoundError, ReservationError, InputValidationError*	*Ran 10 tests in 0.001s OK*

	raise UserNotFoundError("User not found") return users[user_id] def create_reservation(user_id, date, guests): if guests <= 0: raise ReservationError("Number of guests must be greater than zero") user = find_user(user_id) if not date: raise ReservationError("Reservation date is required") return f"Reservation created for {user} on {date} for {guests} guests" def validate_input(name, email): if not name: raise InputValidationError("Name is required") if not email: raise InputValidationError("E mail is required") if "@" not in email: raise InputValidationError("E mail is invalid") return "Valid input"	*class TestErrorHandling(unittest.TestCase): def test_find_user_success(self): user = find_user("1") self.assertEqual(user, "John Doe") def test_find_user_not_found(self): with self.assertRaises(UserNotFoundError): find_user("3") def test_create_reservation_success(self): result = create_reservation("1", "2024-08- 10", 5) self.assertEqual(result, "Reservation created for John Doe on 2024-08-10 for 5 guests") def test_create_reservation_invalid_guests(self): with self.assertRaises(ReservationError): create_reservation("1", "2024-08-10", 0) def test_create_reservation_user_not_found(self): with self.assertRaises(UserNotFoundError): create_reservation("3", "2024-08-10", 5) def test_create_reservation_no_date(self): with self.assertRaises(ReservationError): create_reservation("1", "", 5) def test_validate_input_success(self): result = validate_input("John Doe", "john@example.com") self.assertEqual(result, "Valid input") def test_validate_input_no_name(self): with self.assertRaises(InputValidationError): validate_input("", "john@example.com")*	

		def test_validate_input_no_email(self): with self.assertRaises(InputValidationError) : validate_input("John Doe", "") def test_validate_input_invalid_email(self): with self.assertRaises(InputValidationError) : validate_input("John Doe", "johnexample.com") if __name___== '__main__': unittest.main()	

9.	*<?php class Product { private $conn; private $table; public function __construct($conn, $table) { $this->conn = $conn; $this->table = $table; } public function getRandomProducts() { $sqlc = "SELECT * FROM `$this->table` ORDER BY RAND()"; return getData($this->conn, $sqlc); } public function formatProduct($product) { $kategori = $product["kategori"]; $id_alatpesta = $product["id_alatpesta"] ; $nama_alatpesta = $product["nama_alatpes*	*<?php use PHPUnit\Framework\TestCase;
		
		class ProductTest extends TestCase
		{
		private $conn;
		private $product;
		
		protected function setUp(): void
		{
		$this->conn = $this->createMock(PDO::class);
		$this->product = new Product($this->conn, 'tbalatpesta');
		}
		
		public function testGetRandomProducts()
		{
		$expected = [*
			Time: 00:00.012, Memory: 4.00 MB OK (2 tests, 2 assertions)

	ta"]; $rank_harga = $product["rank_harga"]; $harga = RP($product["harga"]); $gambar = $product["gambar"]; return " <figure> <img src='cpanel/ypathfile/$gambar' alt='$gambar' title='$nama_alatpesta' > </figure> $nama_alatpesta < /a> Rp $harga "; } }	*["id_alatpesta" => 1, "nama_alatpesta" => "Product 1", "kategori" => "Category 1", "rank_harga" => "Rank 1", "harga" => 1000, "gambar" => "image1.jpg"], ["id_alatpesta" => 2, "nama_alatpesta" => "Product 2", "kategori" => "Category 2", "rank_harga" => "Rank 2", "harga" => 2000, "gambar" => "image2.jpg"]];* *$this->conn->method('query')* *->willReturn($expected);* *$result = $this->product->getRandomProducts();* *$this->assertEquals($expected, $result);* *}* *public function testFormatProduct()* *{* *$product = [* *"id_alatpesta" => 1,* *"nama_alatpesta" => "Product 1", "kategori" => "Category 1",* *"rank_harga" => "Rank 1",* *"harga" => 1000,* *"gambar" => "image1.jpg"* *];* *$formattedProduct = $this->product->formatProduct($product);*	

		$expected = " <figure> <img src='cpanel/ypathfile/image1.jpg' alt='image1.jpg' title='Product 1'> </figure> Product 1 Rp 1000 "; $this->assertEquals($expected, $formattedProduct); } }	
10.	*}*		

```
<?php
include"../konmysqli.php";

echo"<link href='../$PATH/$css' rel='stylesheet' type='text/css' />";
$sql="SELECT `bukti_bayar` FROM `$tbkonfirmasi` WHERE
`id_konfirmasi`='".$_GET["id"]."'";
if(getJum($conn,$sql)>0){
    $d = getField($conn,$sql);
    $bukti_bayar=$d["bukti_bayar"];
}
else{$bukti_bayar="avatar.jpg"; }
echo "<p align=center><img src='../$YPATH/$bukti_bayar' border='0'
width='100%' height='100%'></p>";
/*++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
+++++++
++++++++++++++++++++++++++++*/

function getJum($conn,$sql){
  $rs=$conn->query($sql);
  $jum= $rs->num_rows;
    $rs->free();
    return
```

```
}

function getField($conn,$sql){
    $rs=$conn->query($sql);
    $rs->data_seek(0);
    $d= $rs->fetch_assoc();
    $rs-
    >free();
    return $d;
}

function getData($conn,$sql){
    $rs=$conn->query($sql);
    $rs->data_seek(0);
    $arr = $rs->fetch_all(MYSQLI_ASSOC);

    $rs-
    >free();
    return $arr;
}
```