PENGEMBANGAN SISTEM INFORMASI PELAYANAN *TOUR TRAVEL* DAN SISTEM REKOMENDASI PAKET WISATA DI YOGYAKARTA MENGGUNAKAN METODE *CONTENT-BASED FILTERING*

Tugas Akhir

Diajukan untuk Memenuhi Salah Satu Persyaratan Mencapai Derajat Sarjana Informatika

Dibuat Oleh:

ANGGUN DWI CAHYADI

130707364

PROGRAM STUDI INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

UNIVERSITAS ATMA JAYA YOGYAKARTA

2020

HALAMAN PENGESAHAN

Tugas Akhir Berjudul

PENGEMBANGAN SISTEM INFORMASI PELAYANAN TOUR TRAVEL DAN SISTEM REKOMENDASI PAKET WISATA DI YOGYAKARTA MENGGUNAKAN METODE CONTENT-BASED FILTERING

yang disusun oleh

ANGGUN DWI CAHYADI

130707364

dinyatakan telah memenuhi syarat pada tanggal 22 April 2020

		Keterangan
Dosen Pembimbing 1	: B. Yudi Dwiandiyanta, ST., MT.	Telah menyetujui
Dosen Pembimbing 2	: Yulius Harjoseputro, ST., MT.	Telah menyetujui
Tim Penguji		
Penguji 1	: B. Yudi Dwiandiyanta, ST., MT.	Telah menyetujui
Penguji 2	: Dr.Alb. Joko Santoso, MT.	Telah menyetujui
Penguji 3	: Prof. Ir. Suyoto, MSc., PhD	Telah menyetujui

Yogyakarta, 22 April 2020

Universitas Atma Jaya Yogyakarta

Fakultas Teknologi Industri

Dekan

ttd

Dr. A. Teguh Siswantoro, M.Sc

PERNYATAAN ORISINALITAS & PUBLIKASI ILMIAH

Saya yang bertanda tangan di bawah ini:

Nama Lengkap	: Anggun Dwi Cahyadi
NPM	: 130707364
Program Studi	: Informatika
Fakultas	: Teknologi Industri
Judul Penelitian	: Pengembangan Sistem Informasi Pelayanan *Tour Travel* dan Sistem Rekomendasi Paket Wisata di Yogyakarta Menggunakan Metode *Content-Based Filtering.*

Menyatakan dengan ini:

1. Tugas Akhir ini adalah benar tidak merupakan salinan sebagian atau keseluruhan dari karya penelitian lain.
2. Memberikan kepada Universitas Atma Jaya Yogyakarta atas penelitian ini, berupa Hak untuk menyimpan, mengelola, mendistribusikan, dan menampilkan hasil penelitian selama tetap mencantumkan nama penulis.
3. Bersedia menanggung secara pribadi segala bentuk tuntutan hukum atas pelanggaran Hak Cipta dalam pembuatan Tugas Akhir ini.

Demikianlah pernyataan ini dibuat dan dapat dipergunakan sebagaimana mestinya.

Yogyakarta, 22 April 2020

Yang menyatakan,

Anggun Dwi Cahyadi

130707364

HALAMAN PERSEMBAHAN

"Impian besar menjadi nyata

bila

bermusuhan dengan rasa malas"

Tugas akhir ini dipersembahkan untuk:

Orang tua dan keluarga besar

semua sahabat dan teman-teman penulis

dan pembaca sekalian

Semua akan indah pada waktu-Nya

KATA PENGANTAR

Puji dan syukur penulis aturkan kepada Tuhan Yang Maha Esa karena berkat rahmat dan karunia-Nya penulis dapat menyelesaikan pembuatan tugas akhir "Pengembangan Sistem Informasi Pelayanan Tour Travel dan Sistem Rekomendasi Paket Wisata di Yogyakarta Menggunakan Metode *Content-Based Filtering*" ini dengan baik.

Penulisan tugas akhir ini bertujuan untuk memenuhi salah satu syarat untuk mencapai derajat sarjana Informatika dari Program Studi Informatika, Fakultas Teknologi Industri di Universitas Atma Jaya Yogyakarta.

Penulis menyadari bahwa dalam pembuatan tugas akhir ini penulis telah mendapatkan bantuan, bimbingan, dan dorongan dari banyak pihak. Untuk itu, pada kesempatan ini penulis ingin mengucapkan terima kasih kepada:

1. Tuhan Yesus Kristus yang selalu membimbing dalam iman-Nya, memberikan berkat-Nya, dan menyertai penulis selalu.
2. Bapak Dr. A. Teguh Siswantoro, selaku Dekan Fakultas Teknologi Industri, Universitas Atma Jaya Yogyakarta.
3. Bapak B. Yudi Dwiandiyanta, S.T., M.T, selaku dosen pembimbing I yang telah membimbing dan memberikan masukan serta motivasi kepada penulis untuk menyelesaikan tugas akhir ini.
4. Bapak Yulius Harjoseputro, S.T., M.T, selaku dosen pembimbing II yang telah membimbing dan memberikan masukan serta motivasi kepada penulis untuk menyelesaikan tugas akhir ini.
5. Seluruh jajaran Dosen dan Staf Fakultas Teknologi Industri Universitas Atma Jaya Yogyakarta.
6. Kedua Orang tua beserta seluruh keluarga yang telah memberikan dukungan doa, dukungan moral, dan dukungan material selama proses pembuatan skripsi.

7. Sahabat komunitas Jogja *Tour Travel* selaku teman seperjuangan di dunia pariwisata, terima kasih untuk segala pengalaman dan informasi yang diberikan kepada penulis untuk menyelesaikan skripsi ini.

8. Semua pihak yang tidak dapat disebutkan satu persatu yang telah membantu dan memberikan dukungan.

Penyusunan laporan tugas akhir dilakukan dengan sebaik-baiknya. Apabila dalam penyusunan laporan tugas akhir penulis masih terdapat kekurangan, saran dan kritik yang bersifat membangun dari semua pihak sangat diharapkan. Demikian laporan tugas akhir ini dibuat, dan penulis mengucapkan terima kasih kepada semua pihak. Semoga laporan ini dapat bermanfaat bagi pembaca.

Yogyakarta, 22 April 2020

Anggun Dwi Cahyadi

130707364

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

INTISARI

PENGEMBANGAN SISTEM INFOMASI PELAYANAN *TOUR TRAVEL* DAN SISTEM REKOMENDASI PAKET WISATA DI YOGYAKARTA MENGGUNAKAN *METODE CONTENT-BASED FILTERING*

Abstrak

Anggun Dwi Cahyadi

130707364

Perkembangan teknologi informasi di sektor wisata membuka kesempatan yang sangat luas untuk masyarakat ikut dalam bisnis jasa pariwisata, salah satunya menjadi penyedia jasa *Tour* dan *Travel* di Yogyakarta. Perkembangan yang pesat di bidang sistem informasi meningkatkan persaingan pemasaran di dunia digital, sehingga banyak sekali fitur-fitur yang ditawarkan oleh berbagai penyedia jasa pariwisata. Ketersediaan *Website* menjadi salah satu fitur wajib yang dibutuhkan oleh penyedia jasa pariwisata untuk memasarkan jasa pariwisata menjadi lebih luas karena banyaknya informasi-informasi obyek wisata yang dapat ditampilkan di *Website*, dengan *Website pertukaran* informasi dapat dilakukan kapan pun dan di mana pun.

Pesatnya perkembangan teknologi informasi dan komunikasi saat ini, maka dibangun sistem informasi pelayanan *Tour Travel* yang lebih interaktif dan bisa diakses dengan baik menggunakan *Desktop* maupun *Smartphone*. Mengingat banyaknya obyek wisata di Yogyakarta maka fitur rekomendasi paket wisata berdasarkan kriteria yang diisikan oleh pengguna akan sangat berguna. Pembangunan sistem informasi pelayanan *Tour Travel* ini menggunakan *framework Laravel* sehingga tampilan akan lebih dinamis dan menarik karena menggunakan *Responsive Design*. Data obyek pariwisata diperoleh dengan wawancara, observasi, dan studi pustaka. Sistem rekomendasi dalam memberikan pelayanan paket wisata di Yogyakarta menggunakan metode *Content-Based Filtering* sehingga menghasilkan rekomendasi paket wisata yang sesuai dengan masukan kriteria oleh pengguna.

Sistem informasi pelayanan *Tour Travel* dapat mempermudah para wisatawan yang ingin berekreasi di Yogyakarta karena dapat memperoleh informasi yang tepat sesuai dengan keinginan wisatawan. Hasil dari penelitian ini sistem bisa mengolah data-data dari wisatawan untuk dapat memberikan informasi rekomendasi yang tepat untuk wisatawan.

Kata Kunci: Pariwisata, *Website, Laravel, Content-Based Filtering.*

Dosen Pembimbing I : B. Yudi Dwiandiyanta, S.T., M.T

Dosen Pembimbing II : Yulius Harjoseputro, S.T., M.T

Jadwal Sidang Tugas Akhir : 22 April 2020

BAB I
PENDAHULUAN

Pada bab ini akan dijelaskan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan penelitian, metode penelitian, dan sistematika penelitian.

1.1. Latar Belakang

Pesatnya perkembangan dan kemajuan teknologi khususnya di bidang informasi sekarang ini membuat peranan suatu sistem informasi sangat penting untuk mendukung dan membantu efektivitas manusia di kehidupan sehari-hari. Majunya era digital sekarang ini semua orang bisa mengakses segala informasi di mana saja dan kapan saja. Sejak berkembangnya dunia *Mobile Smartphone*, semua orang bisa mengakses segala informasi dari genggaman tangan. Teknologi sistem informasi ini data dipakai untuk membantu orang-orang dalam menyelesaikan masalah, seperti sistem informasi pengolahan data jual beli, pengolahan data keuangan, dan lain sebagainya. Perkembangan teknologi juga terjadi di dunia transportasi. Moda transportasi sekarang sudah semakin banyak jenisnya, dan sudah semakin canggih untuk membantu orang-orang dalam bepergian sehingga lebih menghemat waktu dan biaya.

Sektor pariwisata memiliki potensi yang luar biasa untuk dikembangkan sebagai salah satu sumber pendapatan masyarakat dan pemerintah daerah. Perkembangan di sektor pariwisata ini ditandai dengan menjamurnya obyek wisata baru khususnya di Yogyakarta. Supaya obyek wisata dan daya tarik pariwisata dapat dimanfaatkan secara menyeluruh maka perlu dikembangkan teknologi sistem informasi yang tepat dan mendukung sektor pariwisata. Seiring dengan kemajuan teknologi saat ini, bidang pariwisata mengembangkan pemasaran dengan menggunakan *Website* yang tentunya sekarang sudah ramah dengan tampilan di Smartphone. Penetrasi pengguna *Website* menurut Asosiasi Penyedia Jasa Internet Indonesia (APJII) bersama *Polling* Indonesia di tahun 2018 awal sebanyak 171,17

juta pengguna dari total penduduk 264, 16 juta jiwa atau sebesar 64,8% penduduk Indonesia menggunakan Internet [1]. Sedangkan 91% internet diakses melalui perangkat Smartphone [2]. Dari data ini, peluang pemasaran melalui media Internet/*Website* sangat tinggi karena mencakup lebih dari 50% penduduk Indonesia.

Perkembangan ini juga diikuti oleh perusahaan penyedia jasa pariwisata, salah satunya penyedia jasa *Tour* dan *Travel*. Perusahaan dituntut untuk dapat selalu meningkatkan kinerjanya. Penerapan teknologi informasi adalah salah satu cara untuk meningkatkan kualitas pelayanan kepada konsumen dan memberikan keuntungan kepada perusahaan tersebut. Sistem informasi sangat berperan penting di segala sektor perusahaan. Proses pemesanan dan pemasaran dapat dilakukan dengan menggunakan sistem informasi, misalnya melalu *Website*. Dengan dikembangkannya sistem informasi, pelayanan jasa dan informasi pariwisata terhadap pengguna dapat berjalan dengan baik dan semestinya. Penyediaan informasi mengenai pariwisata kepada pelanggan menjadi lebih baik dan lebih efisien serta pemasaran untuk pelanggan baru berjalan lebih cepat dan luas.

Internet merupakan suatu media yang sangat membantu proses pemasaran dan perbaikan layanan kepada pelanggan. Banyaknya persaingan dengan lingkungan bisnis, perusahaan menghadapi tiga tantangan strategis yaitu risiko permintaan, risiko inovasi, dan risiko efektivitas [3]. Dengan internet sekarang sudah menjamur *electronic commerce* atau biasa disingkat *e-commers*. Peranan *e-commerce* dalam perusahaan penyedia layanan jasa pariwisata sangat krusial karena dengan adanya sistem *e-commerce* mempermudah semua proses bisnis dan setiap aktivitas, mulai dari transaksi sampai pemasaran atau periklanan.

Pada penelitian kali ini juga dilakukan untuk membangun sebuah sistem informasi yang dapat menampilkan berbagai obyek wisata di Yogyakarta, serta menampilkan berbagai macam paket jasa pariwisata di Yogyakarta. Paket jasa pariwisata ditawarkan kepada pengguna sistem dalam bentuk rekomendasi. Sistem rekomendasi yang digunakan yaitu sistem rekomendasi *Content-Based Filtering.* Sistem rekomendasi ini merupakan salah satu metode yang digunakan pada suatu

sistem karena metode ini menyaring informasi berdasarkan keinginan pengguna dan berdasarkan konten yang disediakan di sistem [4]. Untuk menghasilkan hasil rekomendasi yang lebih akurat maka digunakan metode penggabungan dengan metode lain, yaitu dengan menggunakan algoritma Klasifikasi *Nearest Neighbor* (NN).

Algoritma Klasifikasi *Nearest Neighbor* (NN) adalah metode atau algoritma yang melakukan pendekatan data untuk mencari kasus. Metode ini melakukan penghitungan antara inputan kasus yang baru dimasukkan pengguna dengan kasus lama yang sudah disimpan di basis data dengan mencocokkan *rating* dari beberapa atribut atau kriteria yang sudah ada. Setelah itu Algoritma *Nearest Neighbor* akan mengklasifikasikan dan jika hanya ada atribut atau kriteria dari kasus baru yang sesuai dengan salah satu atribut yang terdapat di kasus yang lama [5].

Dengan adanya sistem informasi dalam bentuk *Website* ini diharapkan dapat membantu para wisatawan dalam mencari informasi dan meningkatkan pemasaran perusahaan penyedia jasa pariwisata khususnya jasa *Tour* dan *Travel*. Selain itu di *Website* ini dapat memberikan rekomendasi yang tepat untuk pemilihan paket *Tour* dan *Travel* wisata di Yogyakarta berdasarkan kriteria-kriteria yang di inginkan wisatawan.

1.2. Rumusan Masalah

Berdasarkan latar belakang di atas, maka dapat didapatkan rumusan masalah sebagai berikut :

1) Bagaimana membangun sistem informasi berupa *Website* tentang Pariwisata di Yogyakarta.
2) Bagaimana membangun sistem rekomendasi paket perjalanan wisata di Yogyakarta kepada pengguna layanan *Website*.

1.3. Batasan Masalah

Batasan-batasan masalah yang terdapat di aplikasi ini adalah sebagai berikut:

1) Sistem informasi yang dikembangkan adalah sistem untuk menampilkan informasi wisata dan paket pariwisata di Yogyakarta serta proses bisnis dengan pelanggan menggunakan *platform Website*.
2) Metode sistem rekomendasi menggunakan *Content-Based Filtering* dengan pendekatan algoritma *Nearest Neighbor.*
3) Pembangunan *Website* menggunakan *Framework Laravel.*
4) Pengguna dari sistem informasi adalah wisatawan dalam negeri yang ingin berwisata ke Yogyakarta.
5) Hasil dari sistem rekomendasi paket wisata tidak dibatasi pada pilihan kategori paket wisata.

1.4. Tujuan Penelitian

Tujuan penelitian dari tugas akhir ini adalah sebagai berikut:

1) Membangun sistem informasi berupa *Website* tentang pariwisata di Yogyakarta.
2) Membangun sistem rekomendasi paket perjalanan wisata di Yogyakarta kepada pengguna layanan *Website*.

1.5. Metode Penelitian

Metodologi menjelaskan tahapan yang dilakukan untuk membuat sistem informasi *Tour dan Travel* berbasis *Website*. Penelitian yang dilakukan adalah *research and Development.* Penelitian ini ditujukan untuk menghasilkan pengembangan suatu produk yaitu suatu sistem informasi. Metode yang digunakan yaitu metode pengumpulan data dan metode pengembangan perangkat lunak. Berikut penjelasan untuk masing-masing metode :

1.5.1. Metode Pengumpulan Data

Pada penelitian ini penulis menggunakan 3 cara dalam metode pengumpulan data yaitu menggunakan studi pustaka, wawancara, serta observasi.

1) Studi Pustaka

Studi pustaka adalah metode yang dilakukan oleh peneliti dalam mengumpulkan informasi yang relevan dengan topik atau masalah yang sedang diteliti. Informasi dapat diperoleh dari buku ilmiah, laporan penelitian, tesis, ensiklopedia dan sumber-sumber lainya dari media cetak ataupun media elektronik.

2) Wawancara

Wawancara merupakan metode yang digunakan untuk mengumpulkan informasi yang relevan dengan topik atau masalah yang sedang diteliti degan melakukan percakapan. Percakapan berlangsung antara pewawancara dan terwawancara, yang nantinya akan memberikan jawaban atas pertanyaan yang disampaikan.

3) Observasi

Observasi adalah aktivitas penelitian untuk mengumpulkan data yang berhubungan dengan informasi-informasi yang dibutuhkan oleh peneliti melalui proses pengamatan langsung di lapangan. Observasi ini dilakukan oleh penulis untuk mendapatkan informasi-informasi yang lebih lengkap dan detail tentang pariwisata di Yogyakarta.

1.5.2. Metode Pengembangan Perangkat Lunak

Metode pengembangan perangkat lunak dibagi menjadi 4 bagian atau proses meliputi analisis masalah, analisis pengguna, desain sistem informasi, dan metode pengujian.

1) Analisis Masalah

Metode analisis masalah dilakukan dengan beberapa cara yaitu :

a) Mendefinisikan dan menganalisis masalah yang dihadapi dengan mencari pemecahan alternatifnya.

b) Melakukan pengamatan dan mempelajari sistem informasi serupa sehingga bias mengetahui masalah-masalah dari sistem tersebut dan dapat mengembangkan alternatif penyelesaiannya.

2) Analisis Penggunaan

Pengguna di sistem informasi ini adalah wisatawan yang masih kekurangan informasi untuk berwisata di Kawasan wisata Yogyakarta

3) Desain Sistem Informasi

Pengembangan desain sistem informasi dibagi menjadi 4 tahapan yaitu :

a) Perancangan proses

Perancangan proses sistem digunakan untuk menjaga agar proses data lancar dan teratur sehingga menghasilkan sistem yang benar.

b) Perancangan basis data

Perancangan basis data adalah proses dilakukannya perancangan basis data yang diperlukan oleh sistem. Sistem basis data adalah suatu sistem yang di dalamnya terdapat data yang saling terhubung. Perancangan basis data dilakukan dengan menentukan kebutuhan basis data untuk sistem dan menentukan parameter dari basis data tersebut.

c) Perancangan antar muka

Desain antar muka pengguna adalah desain untuk perangkat komputer atau mesin sehingga mesin tersebut dapat berkomunikasi dengan pengguna. Perancangan antar muka bertujuan untuk membuat interaksi antara pengguna dengan mesin atau komputer secara efisien dalam mencapai tujuan pengguna.

d) Tahap pembuatan kode

Pada proses ini adalah proses pembuatan kode untuk membangun sistem yang akan dibangun.

4) Metode Pengujian

Metode pengujian dilakukan dengan melakukan pengujian fungsionalitas program terhadap sistem informasi yang dibangun untuk menentukan sistem ini berjalan dengan baik atau tidak.

1.5.3. Dokumentasi

Metode dokumentasi merupakan pencatatan data yang sudah dikumpulkan oleh pengembang untuk pembangun *Website* ke dalam bentuk dokumen.

1.6. Sistematika Penulisan

Dalam Tugas Akhir ini dipergunakan sistematika penulisan sebagai berikut:

BAB 1 Pendahuluan

Bab ini berisi tentang latar belakang masalah, rumusan masalah, Batasan masalah, tujuan penelitian, metode penelitian yang digunakan selama pembuatan sistem, serta sistematika penulisan yang digunakan.

BAB II Tinjauan Pusaka

Bab ini berisikan uraian dari hasil penelitian yang sudah dilakukan sebelumnya dan memiliki kemiripan permasalahan dengan topik Tugas Akhir. Tinjauan pustaka digunakan sebagai acuan penulis dalam penulisan Tugas Akhir.

BAB III Landasan Teori

Bab ini membahas tentang uraian dasar teori, prinsip, serta pendapat yang mendukung dalam pembangunan sistem informasi ini.

BAB IV Analisis dan Perancangan Perangkat Lunak

Pada bab ini berisikan penjelasan mengenai tahapan dalam analisis dan perancangan perangkat lunak mulai dari analisis sistem, lingkup masalah, perspektif produk, fungsi produk, kebutuhan antarmuka pengguna, serta perancangan arsitektur sistem dan perancangan antarmuka.

BAB V Implementasi dan Pengujian Perangkat Lunak

Bab ini berisikan deskripsi dari hasil sistem yang sudah dibuat serta hasil pengujian perangkat lunak yang sudah dibuat.

BAB VI Kesimpulan dan Saran

Pada bab ini berisikan kesimpulan yang diperoleh dari aplikasi sistem informasi yang telah dibuat, serta saran untuk pengembangan lebih lanjut.

BAB II
TINJAUAN PUSTAKA

Pada bab II ini berisikan tentang penelitian-penelitian terdahulu yang menyangkut dengan penelitian yang dilakukan. Terdapat tabel perbandingan antara penelitian yang dilakukan dengan penelitian-penelitian terdahulu, yang dapat dilihat di tabel 2.1.

Di era digital saat ini perkembangan *Website* begitu cepat sehingga *Website* dapat diakses di mana saja kapan saja asalkan terhubung dengan Internet. Dengan adanya fitur-fitur ringan di *Website* ini dapat mempermudah aktivitas dan proses bisnis dalam kehidupan sehari-hari. Berikut adalah contoh-contoh penelitian yang sudah dilakukan tentang pembangunan *Website* sebagai sistem informasi dan pelayanan yang bermanfaat untuk mempermudah aktivitas manusia.

Penelitian yang dilakukan oleh Maulana pada tahun 2015 yaitu penelitian pembuatan Sistem Informasi ini dibangun menggunakan sistem *Website E-Commerce* menggunakan CMS (*Content Manajement System*), yaitu dengan CMS *WordPress*. Bahasa pemrograman yang digunakan yaitu bahasa pemrograman PHP, HTML, CSS, dan JavaScript. Untuk menyimpan konten dan informasi menggunakan basis data MySQL [6]. Untuk sistem rekomendasi pada sistem ini dilakukan secara manual, yaitu dengan admin memberikan informasi rekomendasi yang sama secara langsung kepada seluruh pengunjung *Website*. Data rekomendasi ini disimpan di basis data sistem.

Selain Maulana penelitian pembangunan sistem rekomendasi juga dilakukan oleh Santosa yang menggunakan suatu metode untuk sistem rekomendasinya. Penelitian yang dilakukan Santosa yaitu merancang dan membangun sistem rekomendasi berbasis web yang menarik dan digunakan oleh wisatawan. Sistem rekomendasi di penelitian ini menggunakan metode *Collaborative Filtering*. Sistem ini dibuat dengan menggunakan framework Laravel sehingga bahasa pemrograman yang digunakan yaitu PHP dan menggunakan basis data MySQL. Platform yang

digunakan yaitu dengan platform *Website*. Dengan *Website* pengguna dapat mengakses informasi dengan mudah serta memanfaatkan fitur-fitur yang dibuat untuk mempermudah aktivitas dalam kehidupan sehari-hari [7].

Selain penelitian yang dilakukan oleh Santosa, penelitian mengenai rekomendasi juga dilakukan oleh Arief. Menurut Arief pada tahun 2012 membuat penelitian membuat sistem rekomendasi dengan menggunakan metode *Collaborative Filtering* serta *Location Based Filtering*. Sistem ini dibuat di *Platform Mobile*. Pada sistem ini akan memasukkan data ke sistem rekomendasi yang dibagi dalam dua modul yaitu *Collaborative Filtering* yang akan menyaring obyek wisata berdasarkan dengan kemiripan kategori. Setelah itu modul kedua yaitu *Location Based System* akan menyaring informasi berdasarkan jarak terdekat antara posisi *user* saat ini dengan jarak posisi obyek wisata. Setelah itu sistem akan mengeluarkan hasil rekomendasi antara modul *Collaborative Filtering* dan *Location Based System* sesuai dengan referensi dan jarak terdekat pengguna [8].

Berbeda dengan Santosa dan Arief yang hanya menggunakan satu metode untuk membuat sistem rekomendasi, Utomo menggunakan metode *Hybrid* untuk membuat sistem rekomendasi. Utomo pada tahun 2015 melakukan pembangunan sistem rekomendasi, sistem rekomendasi tersebut melakukan penyaringan informasi dengan menggunakan metode *Hybrid* yaitu penggabungan metode *Content-Based* dan *Collaborative Filtering*. Untuk mendapatkan nilai kemiripannya peneliti menggunakan algoritma Nearest Neighbor untuk klasifikasi datanya. Informasi yang digunakan untuk diolah sehingga menghasilkan suatu rekomendasi yaitu menggunakan data profil user dan aktivitas *rating* terhadap tempat wisata. Dengan metode *Hybrid* ini wisatawan tidak akan mendapatkan hasil rekomendasi yang tepat sesuai dengan keinginannya [9].

Tabel 2.1 Tabel Perbandingan

Item	Santosa [7]	Assaf, Widyawan, dan Sunafri [8]	Utomo dan Anggriawan [9]	Maulana dan Rispianda [6]	Cahyadi (2020)*
Sasaran Pengguna	Umum	Umum	Umum	Umum	Umum
Bidang	*Online Shop*	Informasi Pariwisata	Informasi Pariwisata	Informasi Pariwisata	Informasi Pariwisata, Paket Wisata, dan pemesanan paket.
Bahasa Pemrograman	PHP, CSS, HTML dan JavaScript.	Java	PHP, CSS, HTML.	PHP, CSS, HTML.	PHP, CSS, HTML dan JavaScript.
Platform	*Website*	*Mobile*	*Website*	*Website*	*Website*
Database	MySQL	MySQL	MySQL	MySQL	MySQL
Metode Sistem Rekomendasi	*Collaborative Filtering*	*Collaborative Filtering* dan *Location Based Filtering*	*Hybrid Content-Based* dan *Collaborative*	Manual	*Content-Based Filtering*
Framework	*Laravel*	JQuery *Mobile*	*CodeIgniter*	CMS *Wordpress*	*Laravel*
WebServer	*Apache*	*Apache*	*Apache*	*Apache*	*Apache*

*Penelitian yang dilakukan.

BAB V
IMPLEMENTASI DAN PENGUJIAN SISTEM

Pada bab ini akan dijelaskan mengenai implementasi perangkat lunak, pengujian fungsionalitas perangkat lunak, dan hasil pengujian terhadap pengguna.

5.1. Pengantar

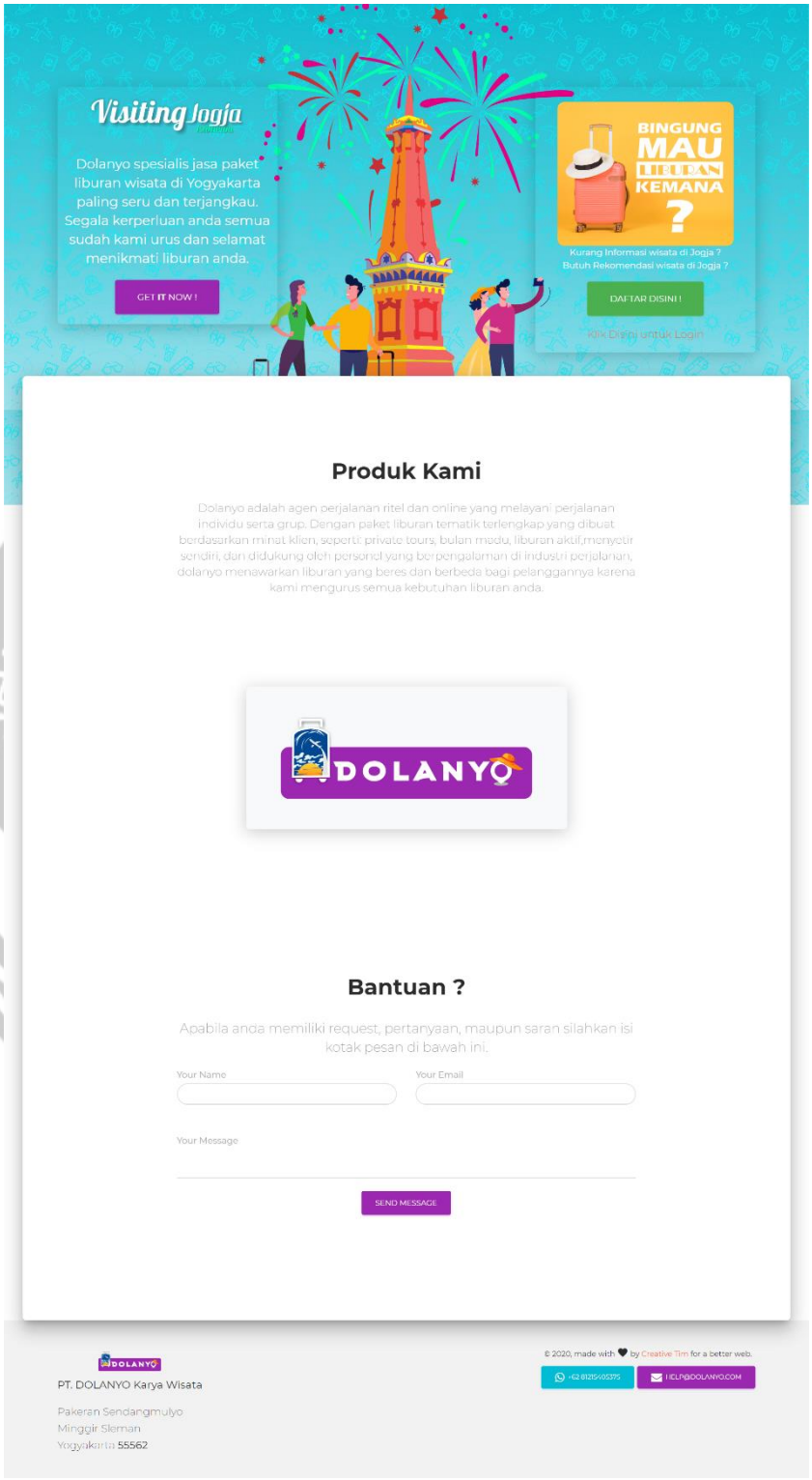
Pada Bab lima akan dibahas tentang hasil implementasi perangkat lunak, pengujian perangkat lunak, serta analisis kelebihan dan kekurangan sistem yang telah dibuat yaitu Sistem Informasi Pelayanan *Tour Travel* dan Sistem Rekomendasi Wisata Berbasis *Website* Menggunakan Metode *Content-Based Filtering.*

5.2. Implementasi perangkat Lunak

5.2.1. Implementasi Halaman *Landing Page*

Halaman *Landing Page* merupakan halaman yang dibuka pertama kali oleh pengguna atau sebagai interaksi pertama antara pengguna dengan Sistem Informasi Dolanyo. Pada halaman *Landing Page* ini terdapat data yang ditampilkan yaitu *banner* informasi yang berisikan tombol-tombol. Tombol *Get It Now!* Ketika diklik akan menuju ke halaman kategori paket wisata, sedangkan tombol Daftar Di sini ! akan menuju ke halaman registrasi, setelah melakukan registrasi pengguna dapat memanfaatkan fitur Rekomendasi paket wisata. Setelah itu ada penjelasan singkat produk jasa yang ditawarkan oleh Sistem Informasi Dolanyo serta keunggulan-keunggulan yang ditawarkan oleh Sistem Informasi Dolanyo. Di bagian bawah terdapat *Form* yang digunakan untuk meminta bantuan atau bertanya kepada Admin Sistem Informasi Dolanyo. *Form* bantuan meminta masukan *Your Name* yang diisikan dengan nama pengguna, *Your Email* yang diisikan dengan *Email* pengguna, serta *Your Message* yang digunakan untuk mengisikan pesan yang akan dikirim ke

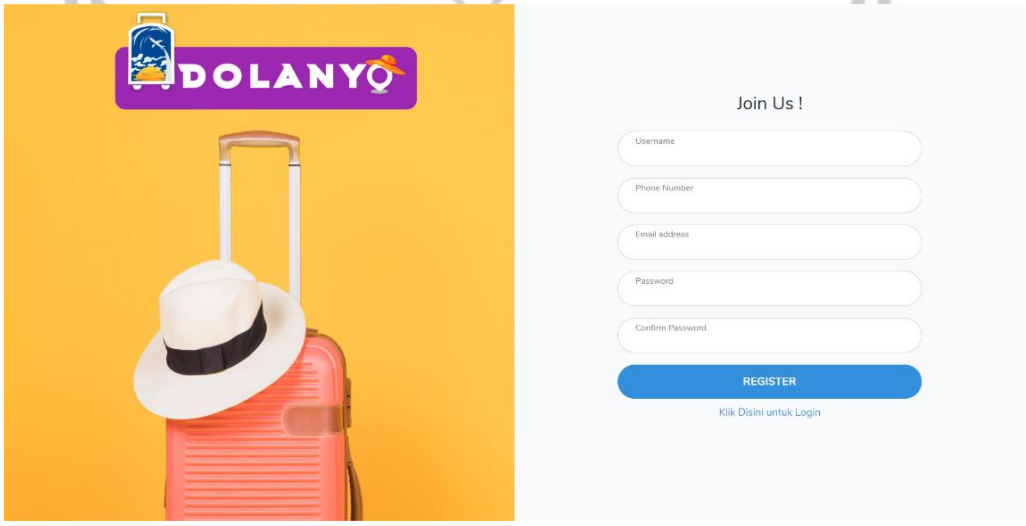
Admin. Setelah itu tombol *Send Message* yang digunakan untuk mengirimkan seluruh data yang sudah diisikan di *Form* ke dalam basis data Sistem Informasi Dolanyo. Gambar 5.1 merupakan implementasi dari halaman *Landing Page.*



Gambar 5.1 Implementasi Halaman Landing Page

5.2.2. Implementasi Halaman Registrasi

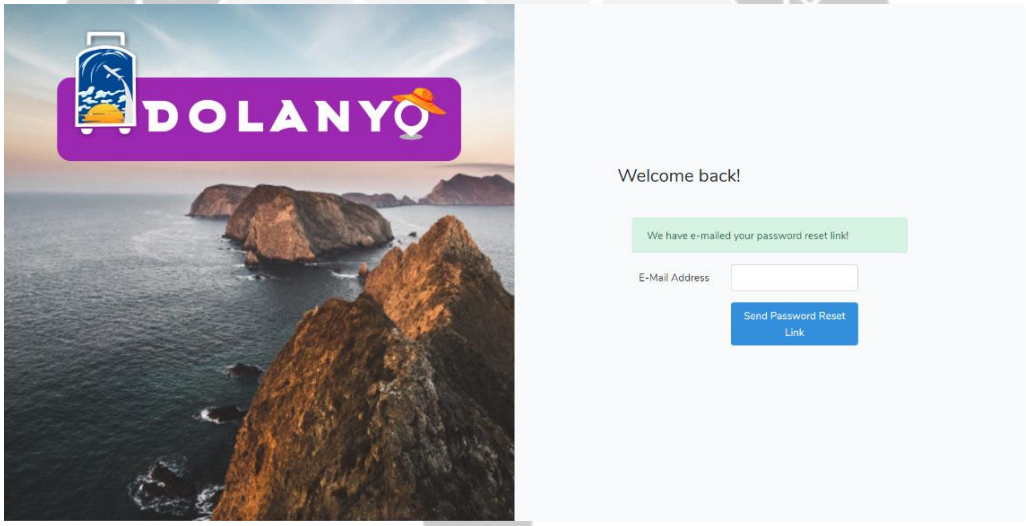
Halaman registrasi merupakan halaman yang digunakan oleh pengguna untuk membuat akun baru, sehingga pengguna dapat menggunakan fitur-fitur yang Sistem Informasi Dolanyo tawarkan antara lain pemesanan paket wisata dan sistem rekomendasi paket wisata. Seperti gambar 5.2, halaman Registrasi berisikan *Form* yang di dalamnya meminta beberapa masukan dari pengguna. Masukan *Username* digunakan untuk memasukkan nama dari pengguna, *Phone Number* digunakan untuk memasukkan nomor *Handphone* pengguna, masukkan *Email Address* digunakan untuk meminta masukan berupa alamat *Email* pengguna, masukan *Password* digunakan untuk memasukkan *Password* pengguna, dan yang terakhir masukan *Confirm Password* digunakan untuk mengkonfirmasi *Password* yang diisikan oleh pengguna. Tombol *Register* digunakan untuk mengkonfirmasi *Form* yang sudah diisikan oleh pengguna dan menyimpan data pengguna tersebut ke dalam basis data Sistem Informasi Dolanyo. Sedangkan *Link* Klik Di sini untuk Login digunakan untuk menuju ke halaman *Login.*



Gambar 5.2 Implementasi Halaman Registrasi

5.2.3. Implementasi Halaman Permintaan Reset *Password*

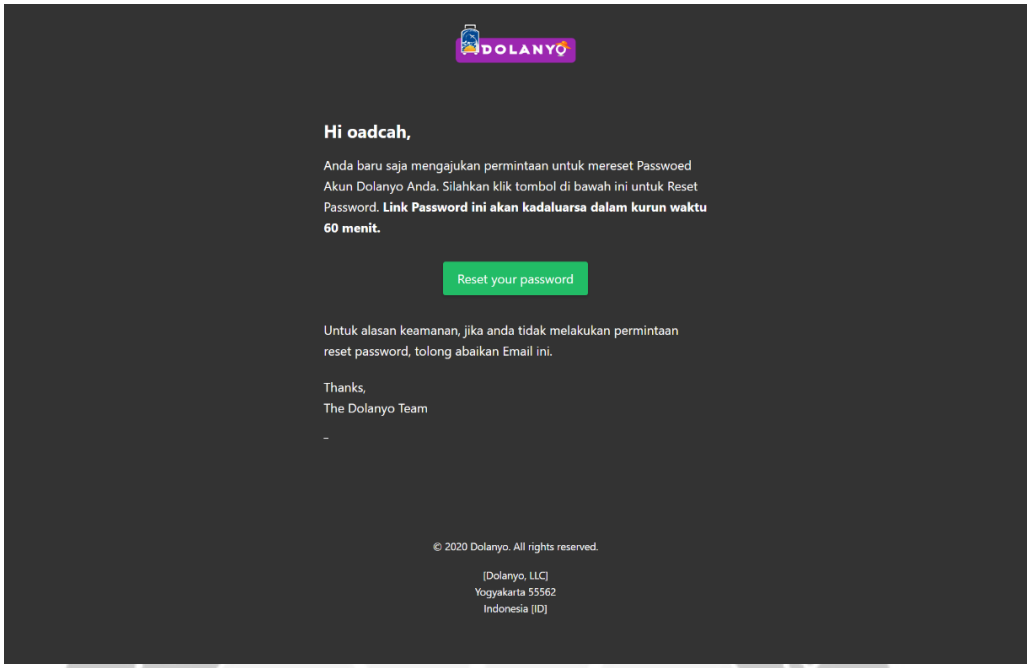
Halaman permintaan reset *Password* merupakan halaman yang digunakan oleh pengguna untuk meminta sistem supaya memasukkan *password* baru. Menurut gambar 5.3 halaman ini terdapat sebuah *Form* yang di dalamnya meminta masukan *Email Address* yang diisikan dengan *Email* pengguna yang akan di reset *Password*-nya. Setelah itu tombol *Send Password Reset Link* ketika diklik akan mengkonfirmasi *Form* yang diisikan tadi dan ketika *Email* yang diisikan benar maka sistem akan mengirimkan *link* reset *Password* untuk pengguna melalui *Email* pengguna. Setelah itu sistem akan memunculkan *Alert We have e-mailed your password reset link* yang berarti *Link* reset *Password* berhasil dikirimkan ke *Email* pengguna.



Gambar 5.3 Implementasi Halaman Permintaan Reset Password

5.2.4. Implementasi Halaman *Email* Reset *Password*

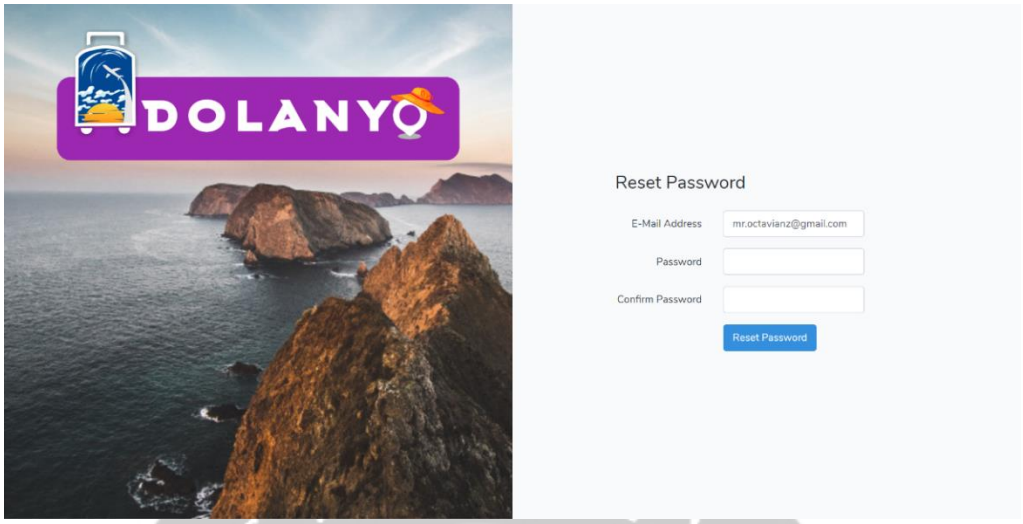
Halaman *email* reset *password* merupakan halaman yang akan ditampilkan ke *email* pengguna jika menerima *email* konfirmasi untuk reset *password.* Menurut gambar 5.4 data yang ditampilkan adalah nama pengguna serta lama *link* reset *password* untuk digunakan yaitu dalam kurun waktu 60 menit. Halaman ini memiliki tombol Reset *Your Password* yang digunakan untuk mengarahkan pengguna ke halaman reset *password.*



Gambar 5.4 Implementasi Halaman Email Reset Password

5.2.5. Implementasi Halaman Reset *Password*

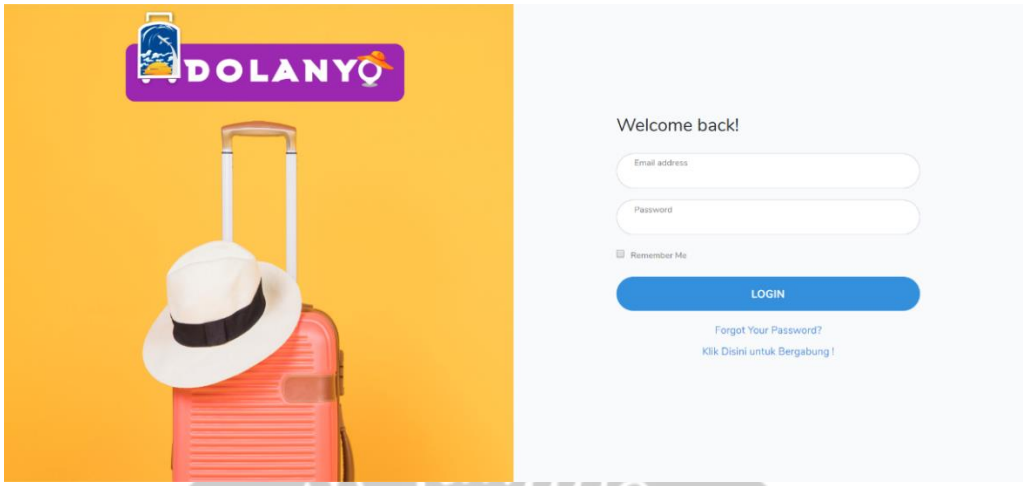
Halaman reset *password* digunakan untuk reset *password* lama pengguna dengan *password* baru. Untuk sampai ke halaman ini pengguna harus melalui *link* yang disediakan di *email* permintaan reset *password.* Menurut gambar 5.5 di halaman ini terdapat *form* yang meminta masukan *Email Address* yang digunakan untuk memasukkan *email* pengguna yang akan di set ulang. *Password* yang digunakan untuk mengisikan *password* baru pengguna, serta *Confirm Password* yang digunakan untuk mengkonfirmasi *password* yang dimasukkan oleh pengguna.



Gambar 5.5 Implementasi Halaman Reset Password

5.2.6. Implementasi Halaman *Login*

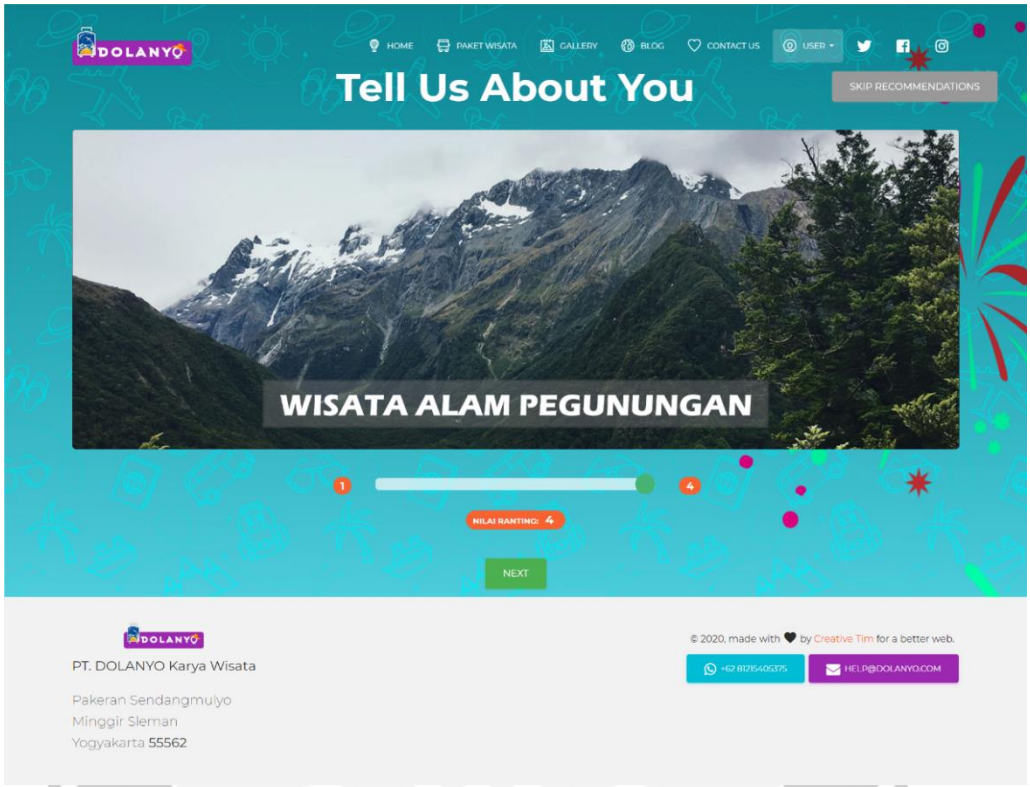
Halaman *Login* digunakan untuk melakukan pengecekan hak akses pengguna. Pengguna dibedakan menjadi dua menurut hak akses. Pengguna biasa atau konsumen ketika *Login* berhasil akan diarahkan ke halaman utama, sedangkan *Administrator* akan dialihkan ke halaman *Dashboard* Admin. Dapat dilihat di gambar 5.6, pengguna akan memasukkan *email* dan *password* untuk nantinya akan di periksa oleh sistem apakah sudah terdaftar atau belum. Di halaman *Login* juga terdapat *Link Forgot Password* dan Klik di sini untuk Bergabung. *Link Forgot Password* digunakan untuk reset *password* pengguna sedangkan *Link* Klik di sini untuk Bergabung akan merujuk ke halaman registrasi.



Gambar 5.6 Implementasi Halaman Login

5.2.7. Implementasi Halaman Rekomendasi

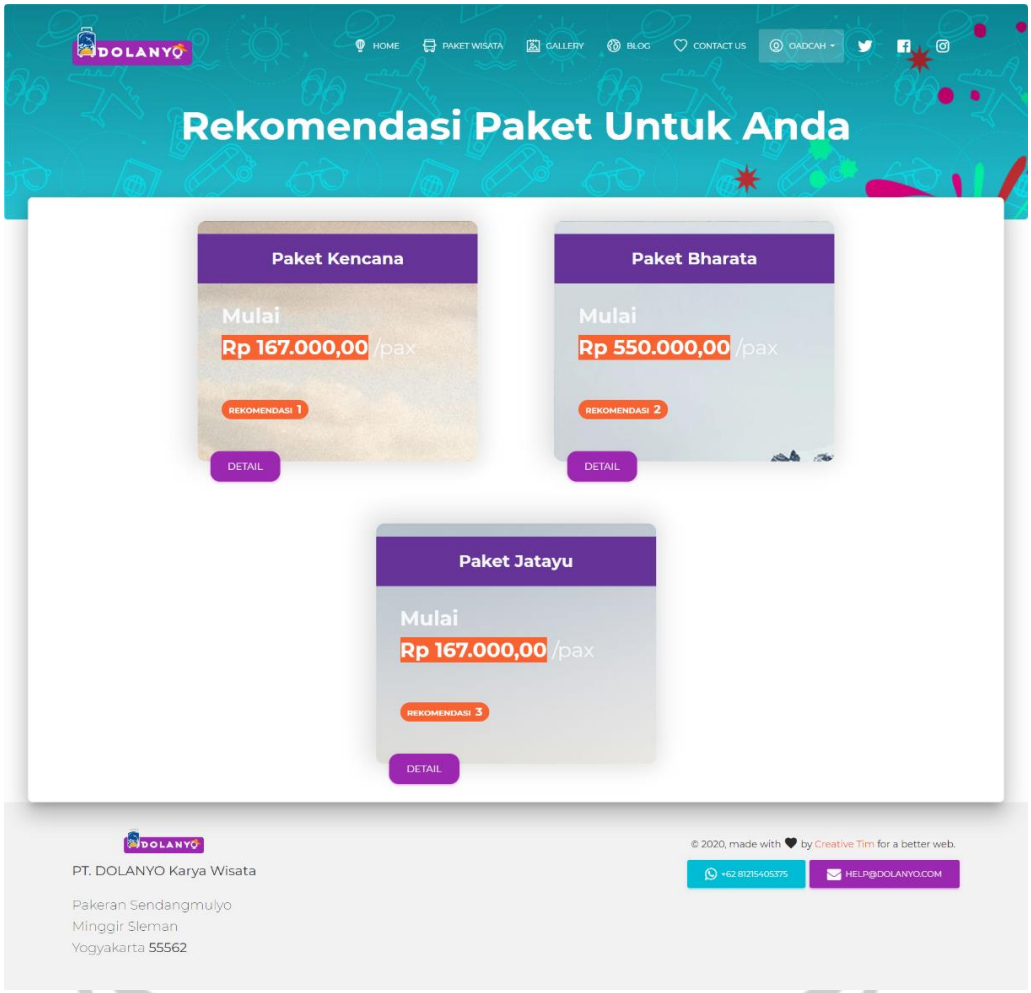
Halaman Rekomendasi digunakan sistem atau *Website* untuk meminta data ketertarikan wisata kepada konsumen. Data tersebut akan diproses oleh sistem sehingga dapat memberikan *Output* berupa data rekomendasi paket wisata kepada pengguna. Menurut gambar 5.7, halaman ini berisikan gambar yang mempresentasikan kategori wisata misalnya alam, kota, dan wisata lainnya. Pengguna diminta untuk memberikan nilai berdasarkan ketertarikannya dengan obyek wisata tersebut. Penilaian dilakukan dengan menggunakan *input* tipe *Slider*. *Slider* berisikan nilai minimal 1 dan maksimal 4 untuk setiap kategori pariwisata.



Gambar 5.7 Implementasi Halaman Rekomendasi

5.2.8. Implementasi Halaman Hasil Rekomendasi

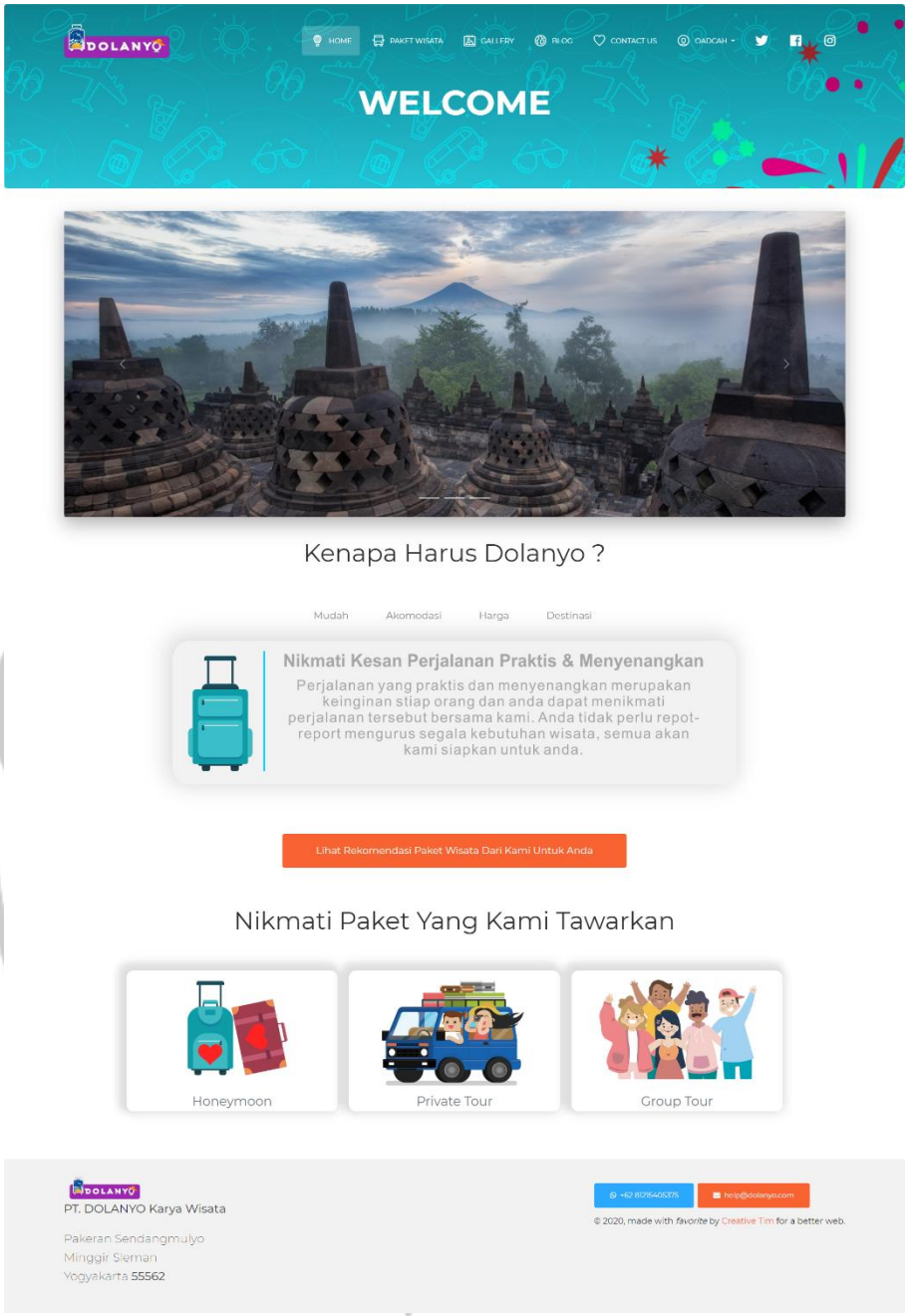
Halaman Hasil Rekomendasi seperti pada gambar 5.8, halaman ini menampilkan hasil dari perhitungan rekomendasi paket wisata yang dilakukan oleh sistem berdasarkan inputan ketertarikan pengguna pada halaman rekomendasi pada gambar 5.7. Hasil rekomendasi diambil dan ditampilkan dalam urutan tiga terbesar dengan perhitungan penilaian pengguna terhadap *rating* yang dimiliki pada setiap paket wisata.



Gambar 5.8 Implementasi Halaman Hasil Rekomendasi

5.2.9. Implementasi Halaman *Home Page*

Halaman Awal atau *Home Page* adalah halaman utama yang diakses oleh pengguna yang beris ikan informasi *Website.* Menurut gambar 5.9, halaman awal berisikan menu-menu dari *link* yang berhubungan ke halaman *Website* lainnya. Halaman awal juga berisikan kategori paket wisata yang ditawarkan. Ketika diklik akan mengarahkan pengguna ke halaman paket wisata yang ditawarkan berdasarkan kategori yang dipilih. Kategori yang disediakan yaitu *Private Tour*, *Group Tour*, dan *Honeymoon.*

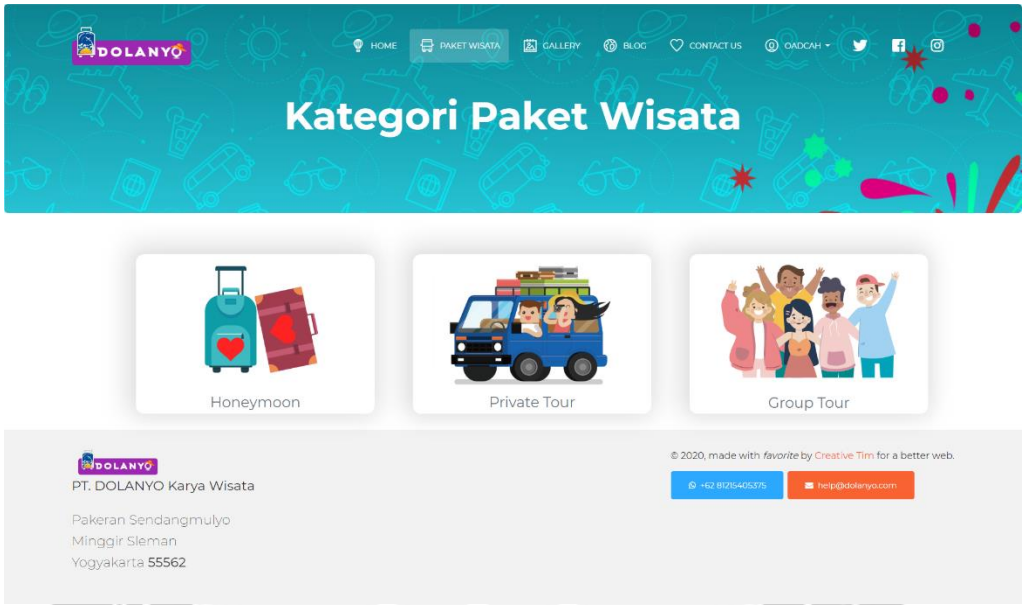


Gambar 5.9 Implementasi Halaman Home Page

5.2.10. Implementasi Halaman Kategori Paket Wisata

Halaman Kategori Paket Wisata merupakan halaman yang digunakan untuk menampilkan kategori-kategori paket wisata. Setiap kategori memiliki banyak paket wisata yang ditawarkan. Seperti pada gambar 5.10 data ditampilkan berupa gambar kategori serta nama kategori. Setiap kategori

adalah *link* yang ketika diklik maka akan mengarahkan pengguna ke halaman paket wisata berdasarkan kategori yang dipilih tadi.



Gambar 5.10 Implementasi Halaman Kategori Paket Wisata.

Untuk menampilkan data kategori seperti pada gambar 5.10 perintah yang dilakukan di dalam kode yaitu dengan memanggil data tabel kategori dan mengirimnya ke halaman kategori. Menurut gambar 5.11, $kategori adalah variabel yang digunakan untuk menyimpan data Kategori yang dipanggil menggunakan metode *Eloquent ORM* yang merupakan fitur *helper framework Laravel* dalam mengakses data dari basis data.

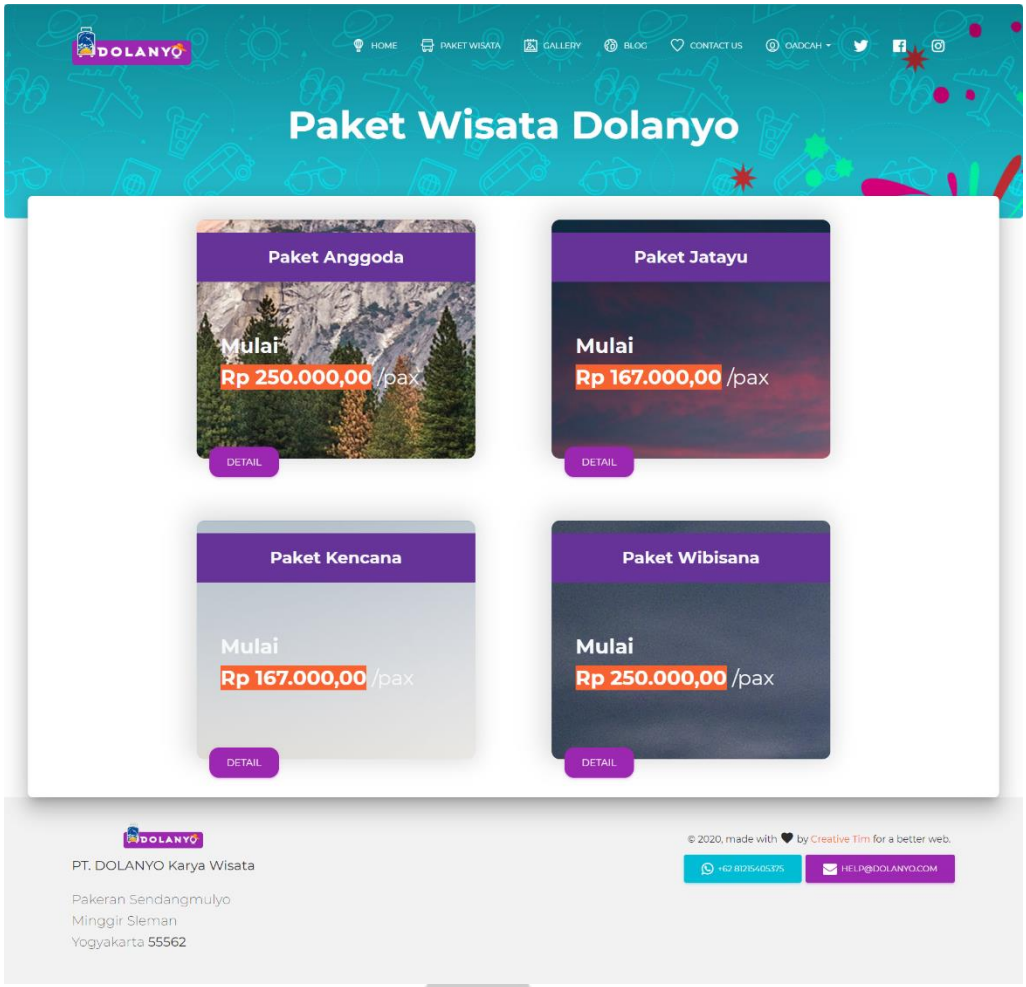
```
public function kategori(){
    $kategori = Kategori::orderBy('created_at', 'DESC')->paginate(5);
    return view('viewpaket')->with('kategori',$kategori);
}
```

Gambar 5.11 Tampilan potongan kode tampil kategori.

5.2.11. Implementasi Halaman Paket Wisata

Halaman Paket Wisata adalah halaman yang memberikan informasi umum mengenai paket-paket wisata yang ditawarkan berdasarkan kategori yang dipilih pengguna. Menurut gambar 5.12, data yang diberikan halaman ini kepada pelanggan adalah data nama paket, harga terendah per pak, serta *popup*

Detail yang memberikan informasi singkat destinasi wisata ketika *pointer mouse* pengguna diarahkan ke *popup* Detail. Setiap paket adalah *link* yang akan mengarahkan pengguna menuju halaman Detail Paket Wisata ketika diklik.



Gambar 5.12 Implementasi Halaman Paket Wisata

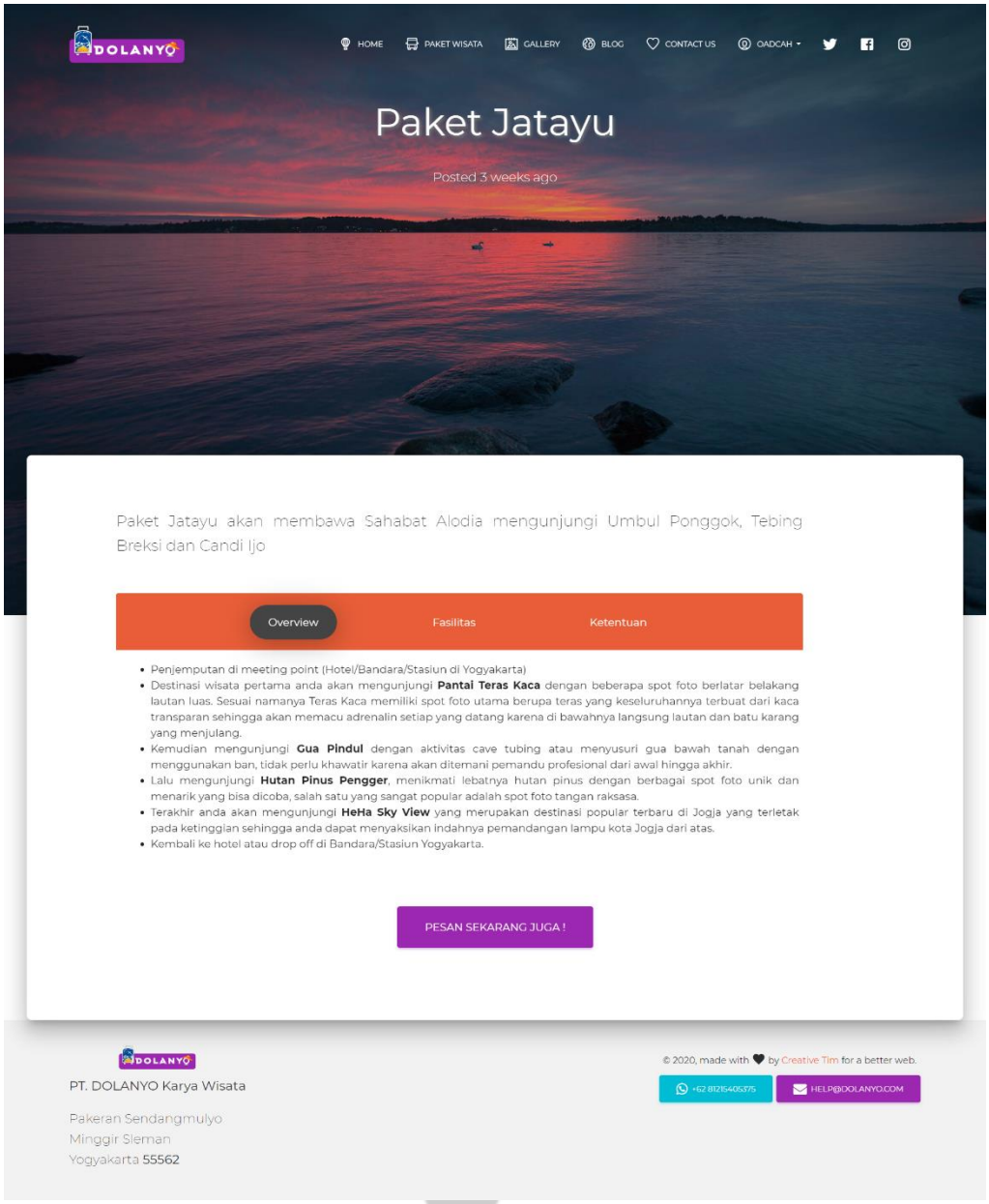
Untuk menampilkan data paket seperti pada gambar 5.12, maka diperlukan kode seperti yang ditampilkan pada gambar 5.13. Fungsi postpaket digunakan untuk menampilkan data paket wisata. Fungsi ini memiliki variabel *request* yang berisikan id kategori paket wisata. Setelah itu kode sistem mengambil data dari tabel paket berdasarkan data id kategori.

```
public function postpaket(Request $request){
    $kategori = DB::table('pakets')->where('kategori',$request->kategori)->get();
    return view('paket')->with('pakets',$kategori);
}
```

Gambar 5.13 Potongan kode untuk menampilkan data kategori.

5.2.12. Implementasi Halaman Detail Paket Wisata

Halaman Detail Paket Wisata merupakan halaman yang berisikan detail informasi dari paket wisata yang dipilih oleh pengguna. Seperti pada gambar 5.14, halaman ini menampilkan judul paket, deskripsi, serta detail paket ditampilkan dalam bentuk menu navigasi *Overview,* Fasilitas, dan Ketentuan. Setiap menu navigasi memiliki detail informasi. Halaman Detail Paket Wisata juga terdapat *form.* Di bagian bawah halaman ini terdapat tombol Pesan Sekarang Juga! yang ketika ditekan akan menampilkan *popup form model* untuk melakukan pemesanan.



Gambar 5.14 Implementasi Halaman Detail Paket Wisata.

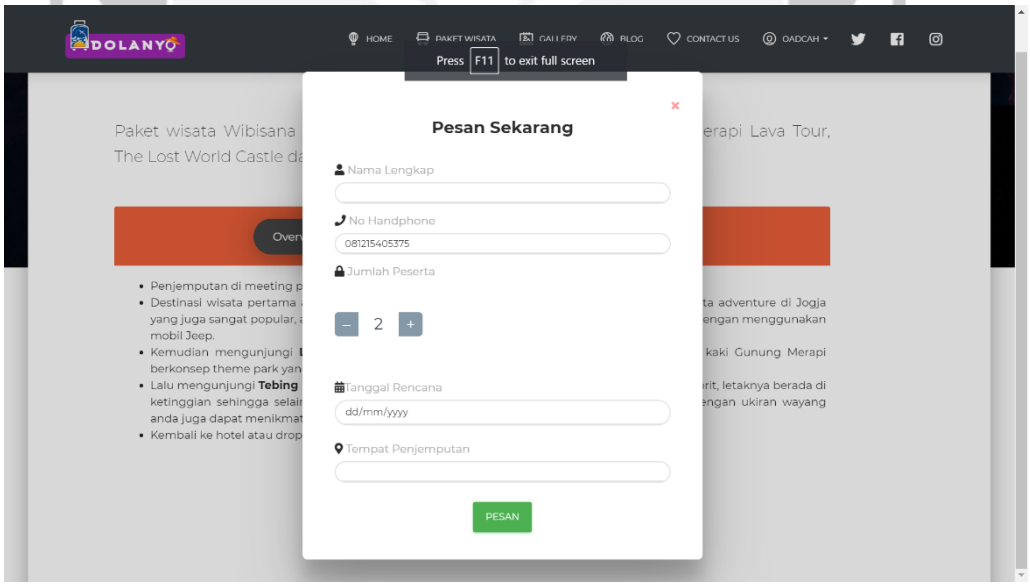
Gambar 5.15 merupakan gambar potongan kode untuk menampilkan data detail paket wisata berdasarkan id paket yang dipilih.

```
public function single (Paket $paket){

    $kategori = DB::table('kategoris')->where('id',$paket->kategori)->first();
    return view('singlepaket', compact('paket'))->with('kategori',$kategori);
}
```

Gambar 5.15 potongan kode untuk menampilkan detail paket wisata.

5.2.13. Implementasi Halaman Pemesanan Transaksi

Halaman pemesanan transaksi merupakan halaman yang digunakan untuk melakukan pemesanan paket wisata. Seperti pada gambar 5.16, halaman ini menggunakan *form model* yang akan muncul ketika tombol Pesan Sekarang Juga! diklik oleh pengguna. *Form* ini meminta masukan nama lengkap untuk memasukkan nama lengkap pengguna atau penanggung jawab, Nomor *Handphone* yang digunakan untuk memasukkan nomor *Handphone* pengguna, Jumlah Peserta yang dimasukkan dengan counter dan memiliki nilai minimal dan maksimal berdasarkan kondisi kategori paket wisata yang dipilih. Masukan Tanggal Pelaksanaan digunakan untuk memasukkan data tanggal pelaksanaan wisata pelanggan, dan masukan tempat penjemputan digunakan untuk memasukkan data tempat penjemputan pelanggan. Bagian bawah halaman ini terdapat tombol Pesan yang ketika ditekan maka sistem akan mengkonfirmasi *form* dan pesanan akan tersimpan di basis data. Setelah itu pelanggan tinggal menunggu balasan konfirmasi email dari Admin.



Gambar 5.16 Implementasi Pemesanan Transaksi.

Proses penyimpanan transaksi dapat dilihat di gambar 5.17. Fungsi transaksi ini memiliki parameter request yang berisikan data paket yang dipilih. Fungsi ini dimulai dengan mengambil data id paket dari basis data berdasarkan judul paket yang dipilih dan disimpan di variabel $Paket_id. Setelah itu

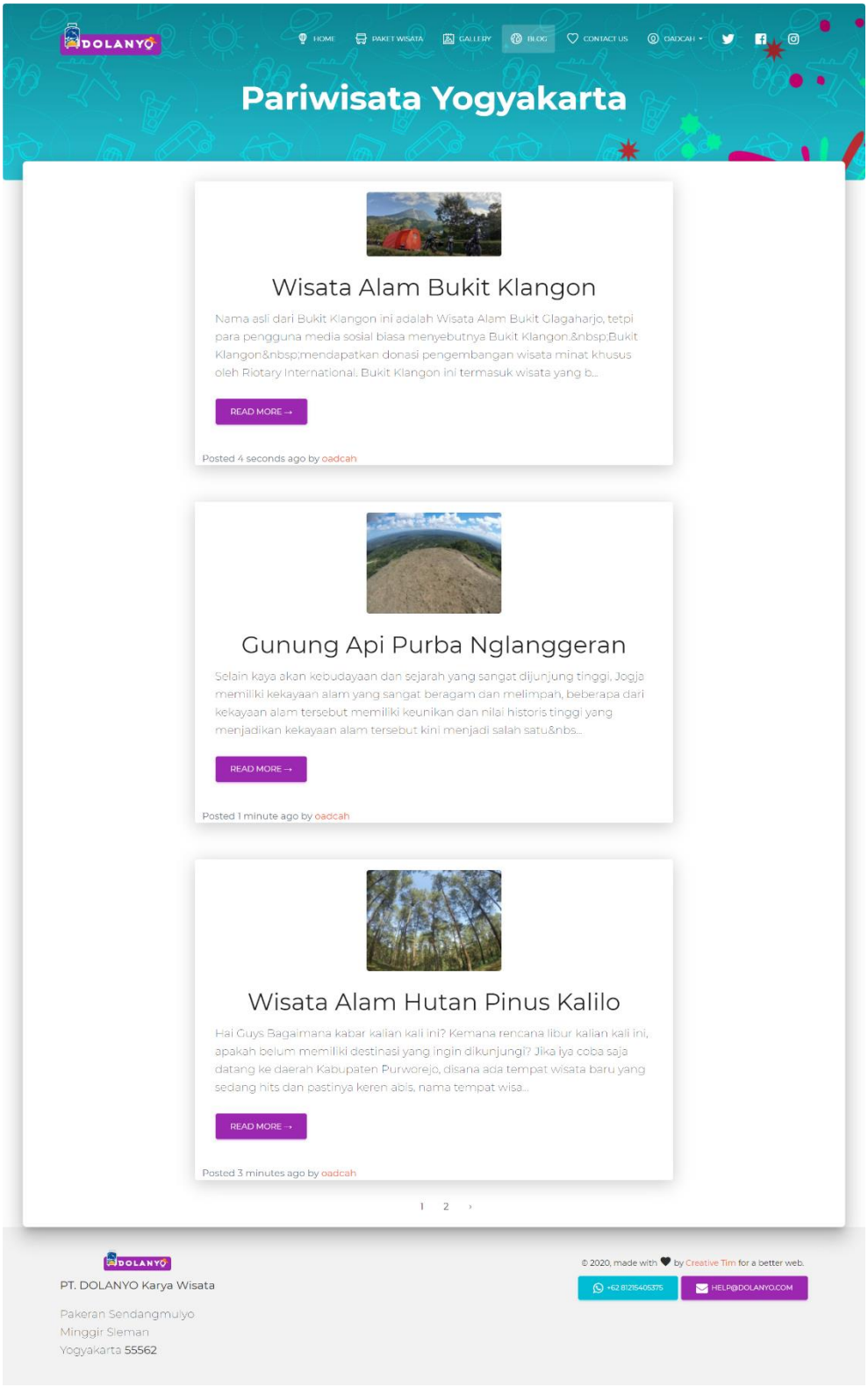
membuat data obyek transaksi baru dan memasukkan data *request* ke dalam obyek transaksi dan menyimpannya di basis data. Setelah itu fungsi akan mengarahkan pengguna ke *route* /Home untuk membuka halaman utama dan muncul pemberitahuan bahwa pesanan berhasil disimpan.

```
public function transaksi(Request $request){
    $paket_id = DB::table('pakets')->where('title', $request->paket)->value('id');
    $post=new Transaksi;
    $post->nama = $request->nama;
    $post->user_id = $request->user_id;
    $post->pakets_id = $paket_id;
    $post->handphone = $request->handphone;
    $post->peserta = $request->peserta;
    $post->harga = $request->harga;
    $post->paket = $request->paket;
    $post->tanggal = $request->tanggal;
    $post->tempat = $request->tempat;
    $post->konfirmasi = 0;

    $post->save();

    return redirect('/home')->with('success','Pesanan Anda sudah kami terima, Silahkan tunggu konfirmasi Email dari kami
}
```

Gambar 5.17 Potongan kode tambah transaksi.

5.2.14. Implementasi Halaman *Content* Blog Wisata

Halaman *Content* Blog Wisata berisikan informasi-informasi tentang pariwisata di Yogyakarta. Menurut gambar 5.18, Halaman ini menampilkan konten berdasarkan waktu unggah. Setiap konten blog memiliki tombol *Read More* yang ketika diklik akan mengarahkan pengguna ke halaman Detail *Content* Blog.



Gambar 5.18 Implementasi Halaman Content Blog Wisata.

Untuk menampilkan seluruh data blog wisata seperti pada gambar

5.18, kode yang dibuat di sistem dapat dilihat pada gambar 5.19. Fungsi all akan memanggil model Post dan menyimpan setiap obyek Post ke dalam variabel posts yang nantinya akan ditampilkan di halaman blog2.

```
public function all(){
    return view('blog2',[
        'posts'=>Post::latest()->paginate(3)
    ]);
}
```

Gambar 5.19 Potongan kode untuk menampilkan seluruh data post blog.

5.2.15. Implementasi Halaman Detail *Content* Blog Wisata

Halaman Detail *Content* Blog Wisata menurut gambar 5.20, halaman ini menampilkan Detail *Content* Blog yang berisikan detail informasi dari konten pariwisata yang dipilih pelanggan.

Gambar 5.20 Implementasi Halaman Detail Content Blog Wisata

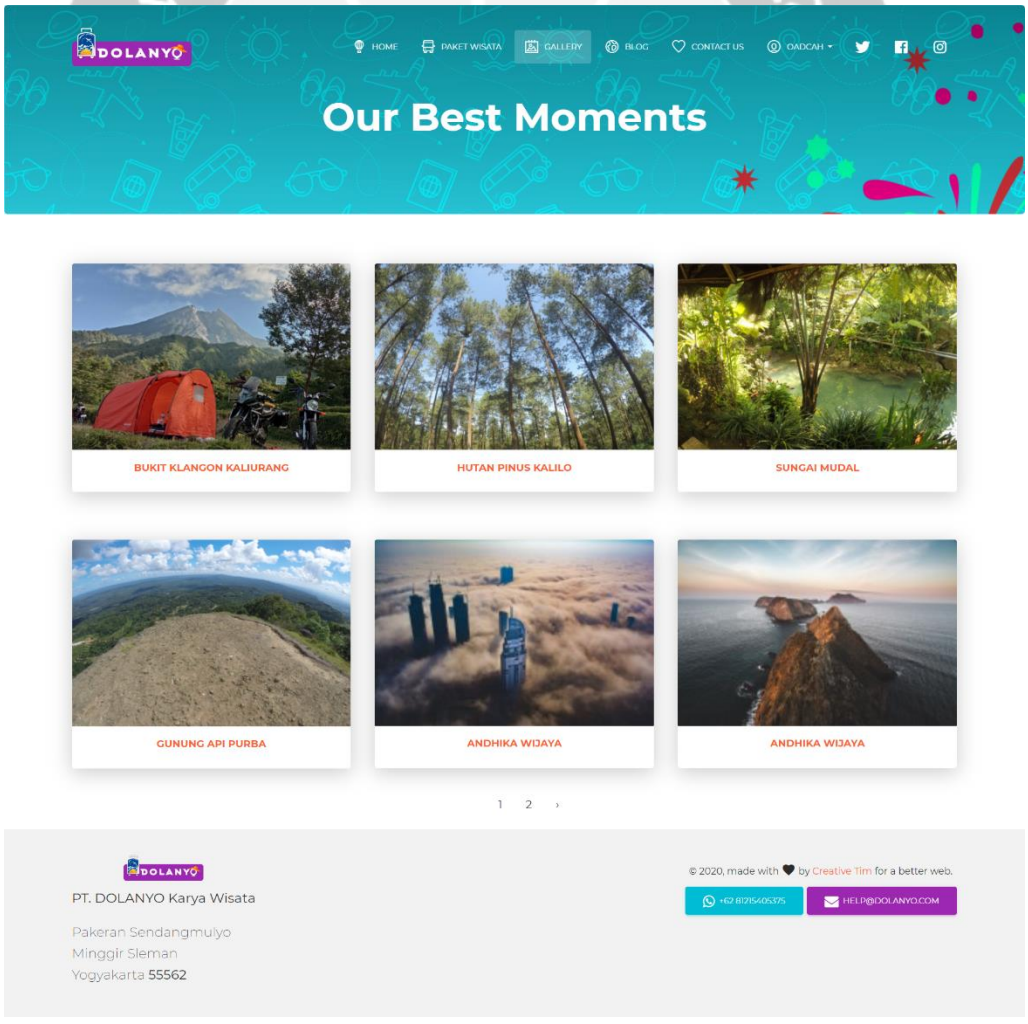
Untuk menampilkan data seperti pada gambar 5.20, maka diperlukan fungsi *Single*(Post $post) seperti pada gambar 5.21. fungsi ini memiliki variabel obyek post yang akan dibawa dan ditampilkan di halaman single2 yang merupakan halaman untuk menampilkan detail post blog.

```php
public function single (Post $post){
    return view('single2', compact('post'));
}
```

Gambar 5.21 Potongan kode untuk menampilkan detail post blog.

5.2.16. Implementasi Halaman *Gallery*

Menurut gambar 5.22, halaman *Gallery* adalah halaman yang berisikan foto-foto yang disimpan di basis data. Setiap foto ketika diklik akan memunculkan *popup modal* yang berisikan gambar foto dengan resolusi yang lebih besar.



Gambar 4.22 Implementasi Halaman Gallery

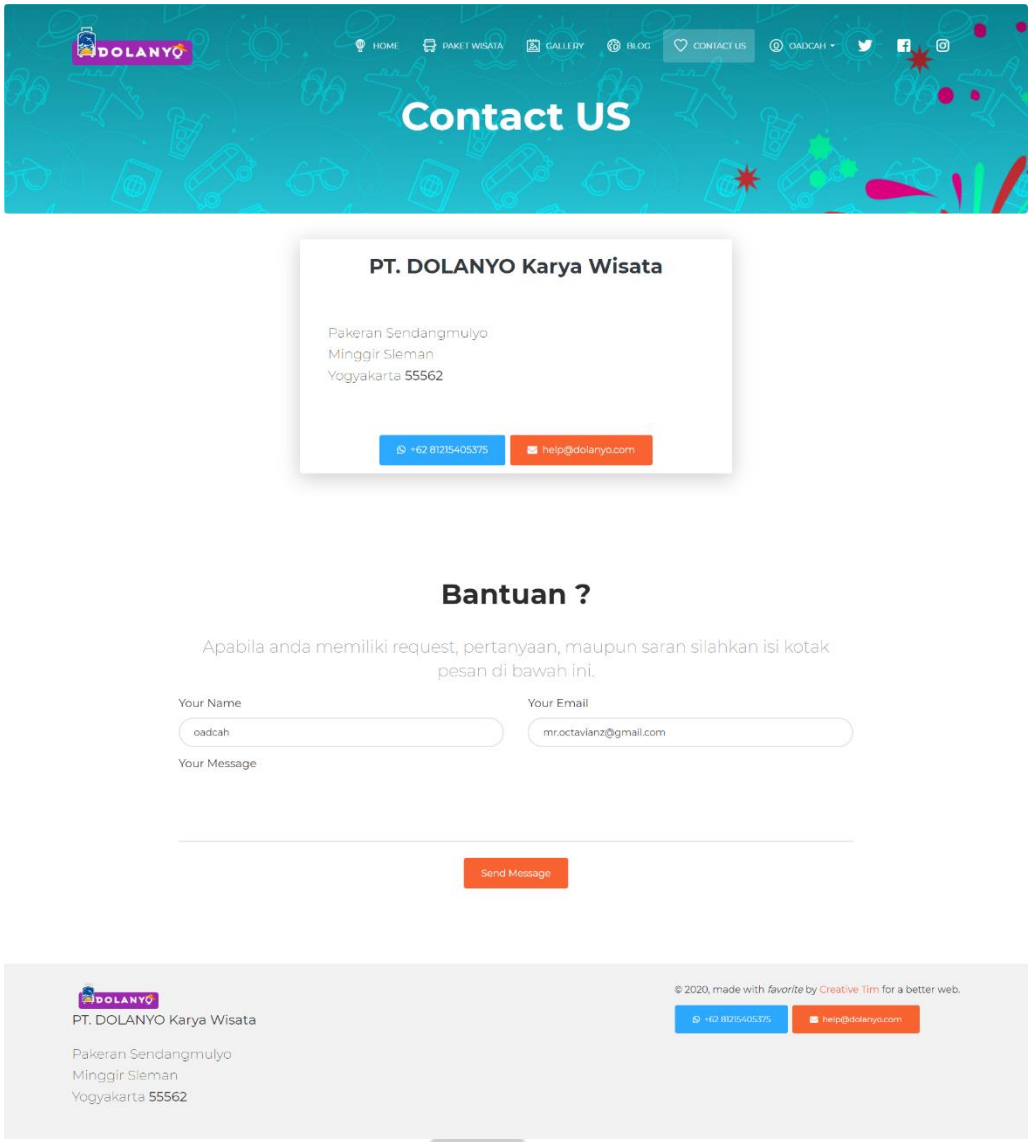
Untuk menampilkan data Gallery seperti gambar 5.22 dapat dilihat di potongan kode dalam gambar 5.23. Fungsi showgallery akan memanggil model *Gallery* dan menyimpan obyek Gallery yang dipanggil dari basis data ke dalam variabel $gallery dan menampilkannya di *view gallery*.

```
public function showgallery()
{
    $gallery = Gallery::orderBy('created_at', 'DESC')->paginate(6);
    return view('gallery')->with('gallery',$gallery);
}
```

Gambar 5.23 Potongan kode untuk menampilkan halaman gallery.

5.2.17. Implementasi Halaman *Contact Us*

Menurut gambar 5.24, halaman *Contact Us* berisikan informasi mengenai Dolanyo serta terdapat *form* bantuan. *Form* bantuan digunakan untuk meminta saran, *request*, serta pertanyaan yang diajukan oleh pengguna. *Form* meminta masukan berupa *Your Name* yang diisikan nama, *Your Email* yang diisikan email pengguna, serta *Message* yang berisikan kritik, saran, atau pertanyaan.



Gambar 5.24 Implementasi Halaman Contact Us.

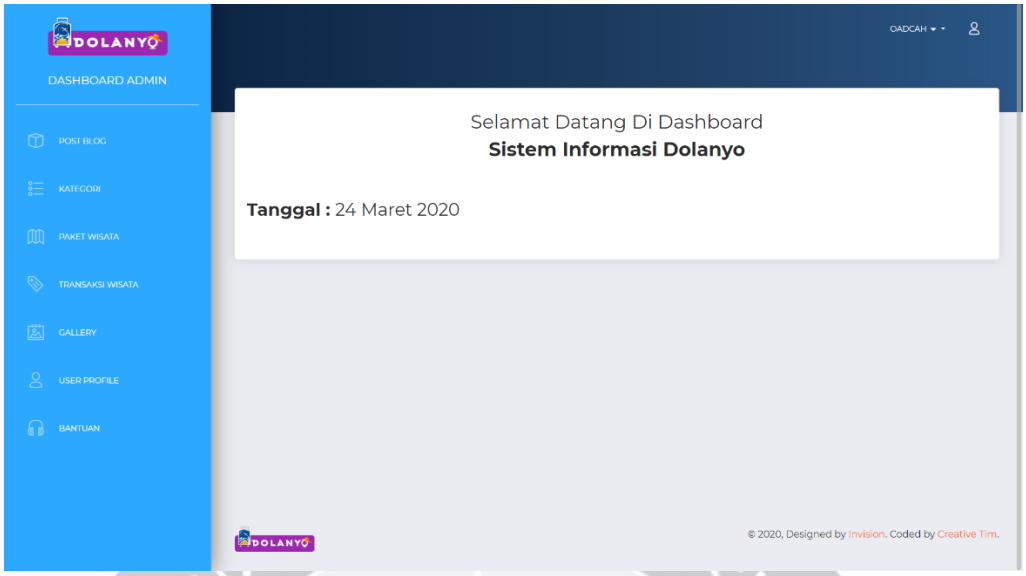
Untuk menyimpan data bantuan ke basis data potongan kode dapat dilihat pada gambar 5.25. Fungsi store akan membuat obyek baru bantuan yang disimpan di variabel $bantuan. Setelah itu parameter request yang berisikan data dari form batuan yang sebelumnya diisi dimasukkan ke dalam obyek bantuan yang dari dibuat dan kemudian obyek bantuan disimpan ke dalam basis data.

```
public function store(Request $request)
{
    $bantuan =new Bantuan();
    $bantuan->nama = $request->nama;
    $bantuan->email = $request->email;
    $bantuan->respons =0;
    $bantuan->pertanyaan = $request->pertanyaan;
    $bantuan->save();

    return redirect('/');
}
```

Gambar 5.25 Potongan kode untuk menyimpan data bantuan ke dalam basis data.

5.2.18. Implementasi Halaman *Dashboard* Admin

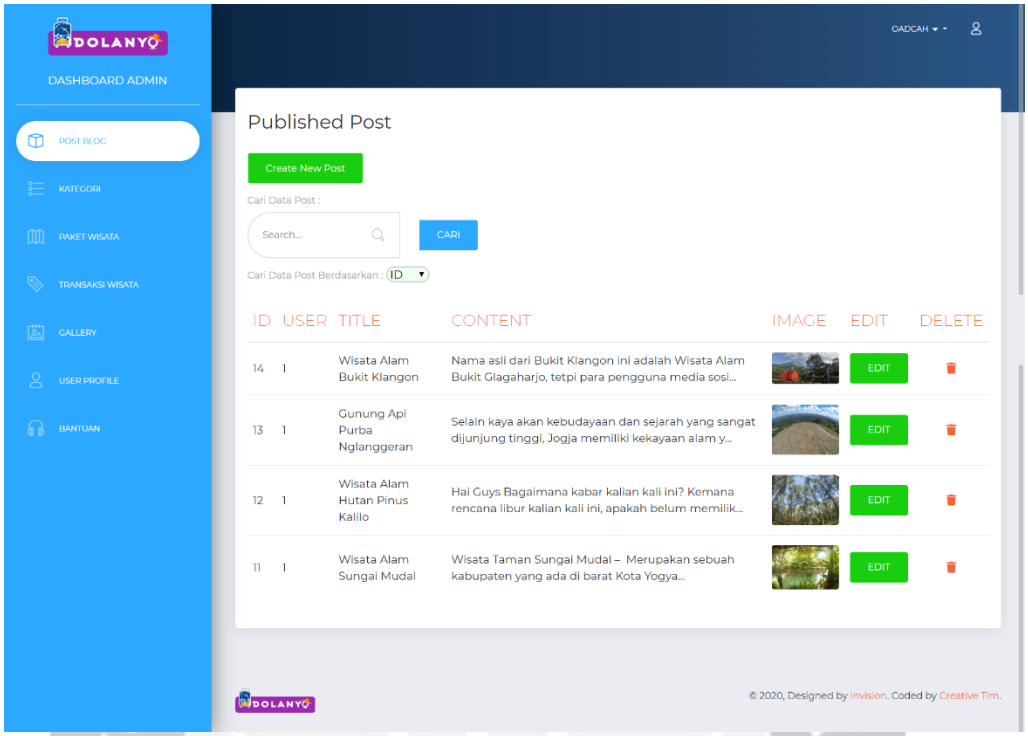
Halaman *Dashboard* Admin digunakan untuk manajemen konten Sistem Informasi Dolanyo dan halaman selamat datang di menu *Dashboard*. Menurut gambar 5.26, halaman ini berisikan menu-menu yang berhubungan dengan *input*, data, pengguna, serta untuk konfirmasi transaksi yang masuk ke sistem. *Dashboard* Admin terdapat menu *Post* Blog yang digunakan untuk melakukan CRUD konten Blog. Menu Kategori yang digunakan untuk melakukan CRUD data Kategori paket wisata. Menu Paket Wisata yang digunakan untuk melakukan CRUD paket-paket wisata yang ditawarkan. Menu Transaksi yang digunakan untuk melakukan pengecekan transaksi dan melakukan konfirmasi transaksi yang masuk ke dalam sistem. Menu *Gallery* digunakan untuk melakukan CRUD data *Gallery* berupa data foto atau *Image.* Menu *User Profile* yang digunakan untuk melakukan CRUD data pengguna yang sudah terdaftar oleh sistem. Menu bantuan yang digunakan untuk mengelola data Bantuan, yaitu memberikan respons dan mengirimkan *Email* balasan kepada pengguna.



Gambar 5.26 Implementasi Halaman Dashboard Admin

5.2.19. Implementasi Halaman *Dashboard Post* Blog

Pada menu *Post* Blog pada halaman *Dashboard* ketika diklik akan menampilkan *Post* atau artikel yang sudah dipublikasikan. Menurut gambar 5.27, terdapat dua tombol utama di bagian atas yaitu Create New Post yang digunakan untuk merujuk ke halaman untuk membuat *Post* baru dan tombol cari yang digunakan untuk mencari data *Post* berdasarkan ID atau *Tittle Post*. Setelah itu rancangan antarmuka halaman *Dashboard Post* Blog memiliki konten tabel dengan atribut kolom ID, *User*, *Tittle*, *Content*, *Image*, Edit, dan *Delete*. Pada atribut kolom ID berisikan ID dari *Post* yang dibuat, *User* berisikan *id User* yang membuat *Post* tersebut. *Tittle* berisikan judul dari *Post*. *Content* berisikan isi artikel *Post*. *Image* berisikan *thumbnail* dari gambar *Post*. Edit berisikan tombol Edit yang digunakan untuk mengedit *Post*, dan kolom *Delete* berisikan tombol *Delete* yang digunakan untuk menghapus *Post*. Halaman ini menampilkan data dengan *Pagination* yang akan menampilkan 5 data dalam satu halaman jika jumlah data lebih dari 5, sehingga lebih efisien untuk *loading* halaman ini.



Gambar 5.27 Implementasi Halaman Dashboard Post Blog

Untuk menampilkan data post blog ke halaman admin dapat dilihat di gambar 5.28. Fungsi posted akan memanggil model Post dan menyimpan setiap obyek post yang dipanggil ke dalam variabel $post yang kemudian akan ditampilkan ke *view* admin.post.

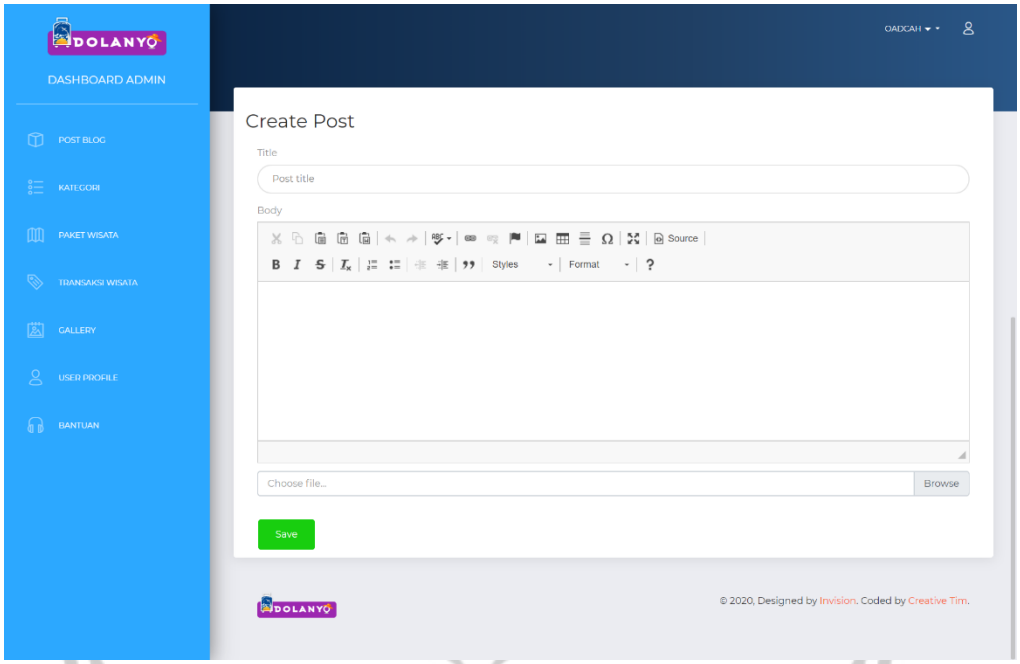
```
public function posted(){

    $post = Post::orderBy('created_at', 'DESC')->paginate(5);
    return view('admin.post')->with('posts',$post);
}
```

Gambar 5.28 potongan kode untuk menampilkan data post blog ke halaman Dashboard post blog.

5.2.20. Implementasi Halaman Dashboard Create New Post

Ketika kita menekan tombol *Create New Post* di halaman *Dashboard Post*, kita akan dirujuk ke halaman *Dashboard Create New Post*. Menurut gambar 5.29, halaman ini terdapat *Form* yang digunakan untuk mengisi atribut dari *Post*. Masukan *Tittle* digunakan untuk memasukkan judul dari *Post*.

Masukan *Body* digunakan untuk memasukkan data *content* artikel yang akan di *Post*. *Image* digunakan untuk mengunggah *file* gambar *Post* ke Sistem. Setelah itu tombol *Save* digunakan untuk menyimpan seluruh data yang sudah ada di *Form* ke dalam basis data.



Gambar 5.29 Implementasi Halaman Dashboard Create New Post.

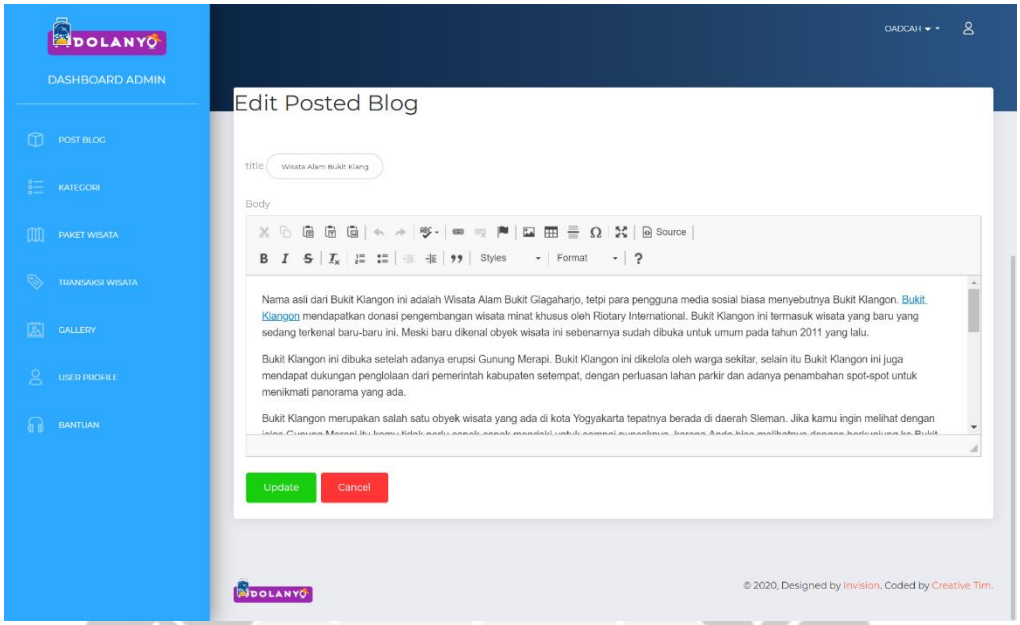
Untuk menyimpan data post blog yang baru disimpan dapat dilihat di potongan kode dalam gambar 5.30. Fungsi store memiliki parameter variabel yang berisikan data post blog yang diisikan di form pada gambar 5.29. Fungsi ini akan membuat obyek Post baru dan disimpan ke dalam variabel $post. Setelah itu akan *generate* nama *file* gambar post blog menjadi dua yaitu *file original* dan *file thumbnail* dan menyimpannya di variabel $file_name karena basis data hanya akan menyimpan nama gambarnya saja, sedangkan *file* gambar akan disimpan di direktori *thumbnail* dan data_file. Setelah itu semua data dari form yang disimpan di parameter *request* disimpan di dalam obyek post dan disimpan ke dalam basis data.

```php
public function store(Request $request)
{
    $post =new Post;
    $file=$request->file('file');
    $nama_file=time()."_".$file->getClientOriginalName();
    $tujuanupload='tumbnail';
    $resize_image=Image::make($request->file('file')->getRealPath());
    $resize_image->resize(200,200,function($constraint){
        $constraint->aspectRatio();
    })->save($tujuanupload .'/'. $nama_file);
    $tujuanupload='data_file';
    $file->move($tujuanupload,$nama_file);
    $post->user_id =$request->user_id;
    $post->title = $request->title;
    $post->body = $request->body;
    $post->image= $nama_file;
    $post->save();

    return redirect('/posted');
}
```

Gambar 5.30 potongan kode untuk menyimpan data post blog baru ke dalam basis data.

5.2.21. Implementasi Halaman *Dashboard Edit Post*

Halaman *Dashboard* Edit *Post* seperti pada gambar 5.31 merupakan halaman yang digunakan untuk mengubah data artikel yang sudah di publikasikan. Data *Post* yang diubah di halaman ini yaitu *Tittle* dan *Body Content*. Di halaman ini terdapat tombol *Update* dan *Cancel*. Tombol Update digunakan untuk mengkonfirmasi perubahan data yang dimasukkan di *Form* dan menyimpan perubahan data ke basis data, sedangkan tombol *Cancel* digunakan untuk membatalkan perubahan data dan mengarahkan pengguna ke halaman *Post.*



Gambar 5.31 Implementasi Halaman Dashboard Edit Post.

Untuk menyimpan hasil perubahan data post metode yang digunakan hampir sama dengan fungsi yang digunakan untuk menyimpan data post blog baru. Menurut gambar 5.32, perbedaannya, pada fungsi *update* akan memanggil obyek post yang dipilih berdasarkan id post yang diedit. Setelah itu menyimpan data form dari parameter ke dalam obyek post yang dipanggil tadi lalu menyimpannya ke basis data.

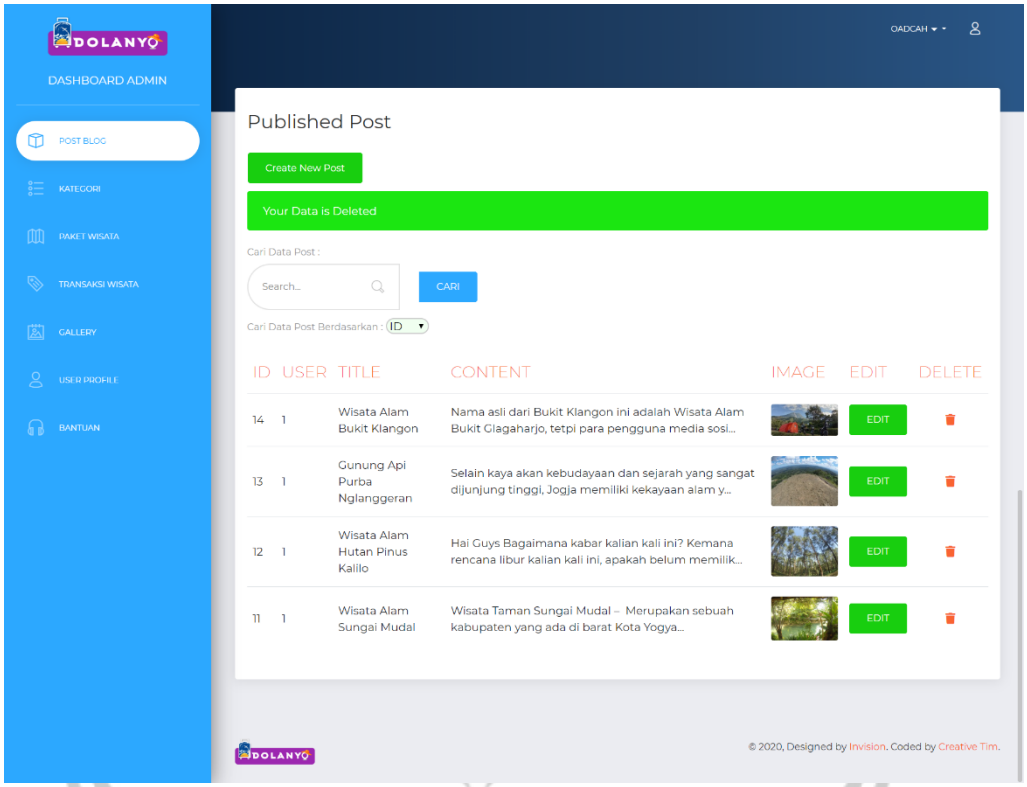
```
public function update(Request $request, $id)
{
    $posts = Post::find($id);
    $posts->title=$request->input('title');
    $posts->body=$request->input('body');
    $posts->update();

    return redirect('/posted')->with('success','Your Data is Updated');
}
```

Gambar 5.32 Potongan kode untuk mengubah data post blog yang ada di basis data.

5.2.22. Implementasi Halaman Dashboard Delete Post

Halaman *Dashboard Delete Post* seperti pada gambar 5.33 adalah halaman yang digunakan untuk menghapus data *Post* yang dipilih berdasarkan letak baris tombol *Delete* di tabel. Tombol *Delete* digunakan untuk

mengkonfirmasi bahwa data di dalam baris tombol tersebut akan dihapus. Setelah dihapus maka akan muncul pemberitahuan *Your data is Deleted* yang berarti data telah berhasil dihapus.



Gambar 5.33 Implementasi Halaman Dashboard Delete Post.

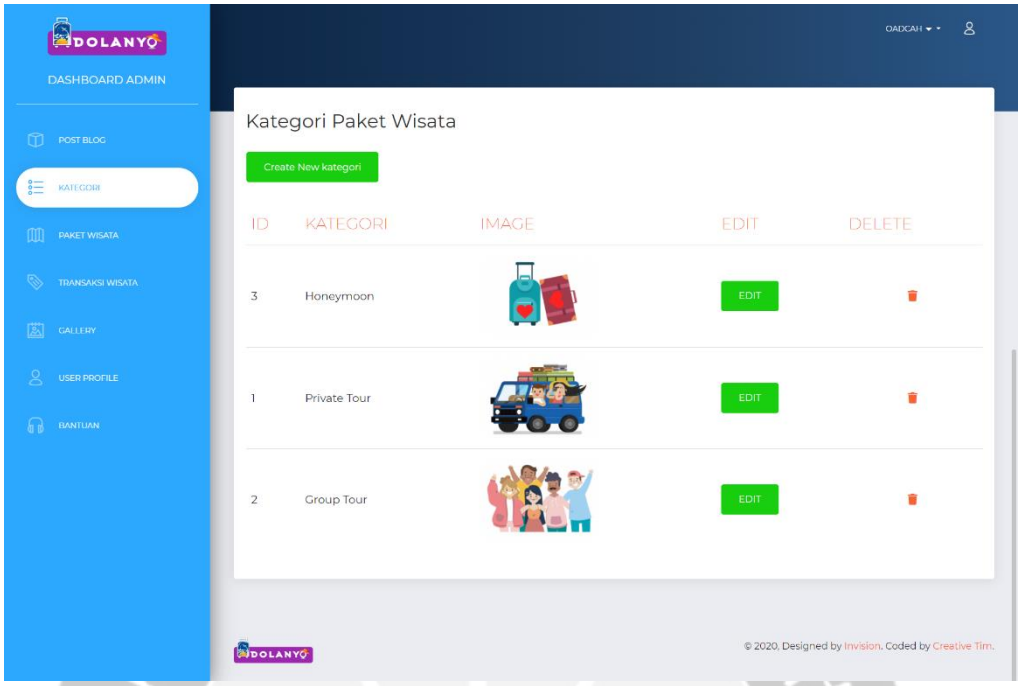
Untuk melakukan hapus data post blog dapat dilihat pada potongan kode di gambar 5.34. fungsi destroy memiliki parameter id post yang digunakan untuk menghapus data berdasarkan parameter id post yang dipilih. Fungsi ini akan memanggil post berdasarkan id post dan melakukan metode *delete.* Setelah itu akan menghapus file gambar di direktori dengan menggunakan bantuan *helper File* yang disediakan oleh *framework Laravel.*

```php
public function destroy($id)
{
    $posts = Post::findOrfail($id);
    $posts ->delete();
    File::delete('data_file/'.$posts->image);
    File::delete('tumbnail/'.$posts->image);

    return redirect('/posted')->with('success','Your Data is Deleted');
}
```

Gambar 5.34 Potongan kode untuk menghapus data post blog dari basis data.

5.2.23. Implementasi Halaman *Dashboard* Kategori Paket Wisata

Halaman *Dashboard* Kategori Paket Wisata adalah halaman yang dipergunakan untuk menampilkan semua data Kategori Paket Wisata yang ada di basis data. Seperti pada gambar 5.35, Data Kategori Paket Wisata yang ditampilkan adalah ID, Kategori, *Image*, Edit, dan *Delete*. Kolom Edit terdapat tombol Edit di setiap baris data tabel, yang digunakan untuk mengarahkan pengguna ke halaman edit kategori paket wisata. Tombol *Delete* di kolom *Delete* digunakan untuk menghapus data berdasarkan baris tombol *Delete* tersebut. Halaman ini juga terdapat tombol *Create New Kategori* yang mengarahkan pengguna ke halaman *Dashboard Create Kategori* Paket yang digunakan untuk membuat kategori baru.



Gambar 5.35 Implementasi Halaman Dashboard Kategori Wisata.

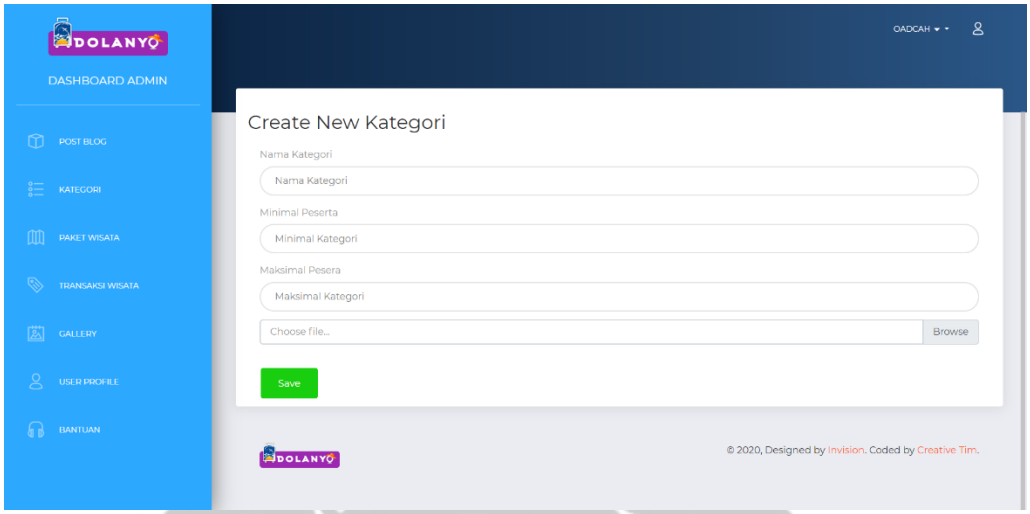
Untuk menampilkan data kategori ke halaman admin dapat dilihat di gambar 5.36. Fungsi show akan memanggil model kategori dan menyimpan setiap obyek kategori yang dipanggil ke dalam variabel $kategori yang kemudian akan ditampilkan ke *view* admin.kategori.

```
public function show(){
    $kategori = Kategori::orderBy('created_at', 'DESC')->paginate(5);
    return view('admin.kategori')->with('posts',$kategori);
}
```

Gambar 5.36 Potongan kode tampil data kategori di halaman admin.

5.2.24. Implementasi Halaman *Dashboard Create* Kategori Paket Wisata

Seperti pada gambar 5.37, halaman *Dashboard Create* Kategori Paket Wisata digunakan untuk membuat data kategori paket wisata yang baru. Halaman ini terdapat *form* yang di dalamnya meminta masukan Nama Kategori, Minimal Peserta, Maksimal Peserta, dan *Image*. Tombol *Save* digunakan untuk mengkonfirmasi data yang dimasukkan di form dan menyimpan data kategori baru ke dalam basis data.



Gambar 5.37 Implementasi Halaman Dashboard Create Kategori Wisata.

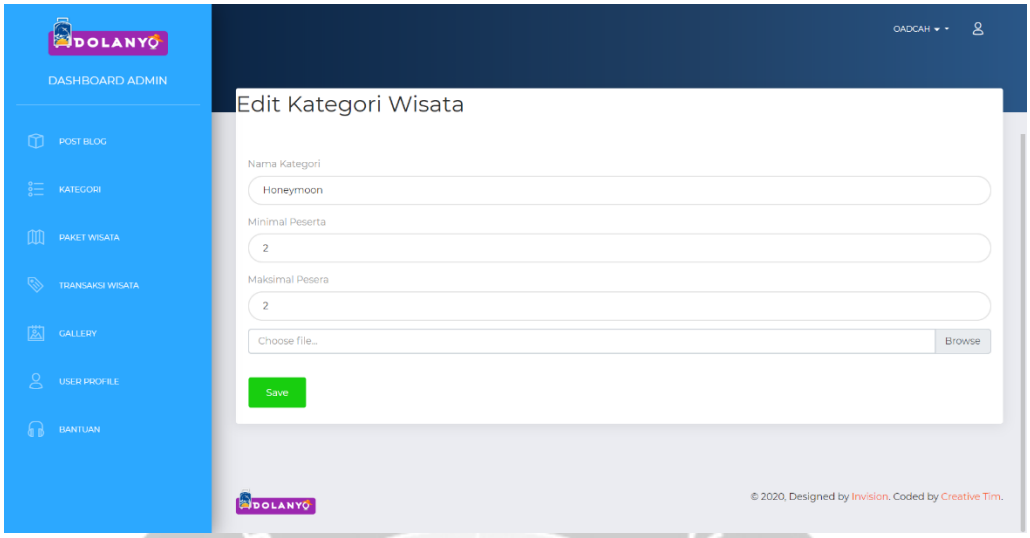
Untuk menyimpan data kategori yang baru disimpan dapat dilihat di potongan kode dalam gambar 5.38. Fungsi store memiliki parameter variabel yang berisikan data kategori yang diisikan di form pada gambar 5.37. Fungsi ini akan membuat obyek Kategori baru dan disimpan ke dalam variabel $post. Setelah itu akan *generate* nama *file* gambar kategori menjadi dua yaitu *file* original dan *file thumbnail* dan menyimpannya di variabel $file_name karena basis data hanya akan menyimpan nama gambarnya saja, sedangkan *file* gambar akan disimpan di direktori *thumbnail* dan data_file. Setelah itu semua data dari form yang disimpan di parameter *request* disimpan di dalam obyek Kategori dan disimpan ke dalam basis data.

```php
public function store(Request $request)
{
    $post =new Kategori();
    $file=$request->file('file');
    $nama_file=time()."_".$file->getClientOriginalName();
    $tujuanupload='tumbnail';
    $resize_image=Image::make($file->getRealPath());
    $resize_image->resize(200,200,function($constraint){
        $constraint->aspectRatio();
    })->save($tujuanupload .'/'. $nama_file);
    $tujuanupload='data_file';
    $file->move($tujuanupload,$nama_file);

    $post->kategori = $request->kategori;
    $post->maxpeserta = $request->maksimal;
    $post->minpeserta = $request->minimal;
    $post->image= $nama_file;
    $post->save();

    return redirect('/admin/kategori');
}
```

Gambar 5.38 Potongan kode input data kategori baru.

5.2.25. Implementasi Halaman *Dashboard Edit* Kategori Paket Wisata

Halaman *Dashboard* Edit Kategori Paket Wisata digunakan untuk mengubah data kategori yang sudah tersimpan di basis data. Seperti pada gambar 5.39, halaman ini terdapat *form* yang berisikan masukan Nama Kategori, Minimal Peserta, Maksimal Peserta, dan *Image*. Halaman ini terdapat tombol *Save* yang digunakan untuk mengkonfirmasi data di form yang akan diubah dan disimpan ke basis data.



Gambar 5.39 implementasi Halaman Dashboard Edit Kategori.

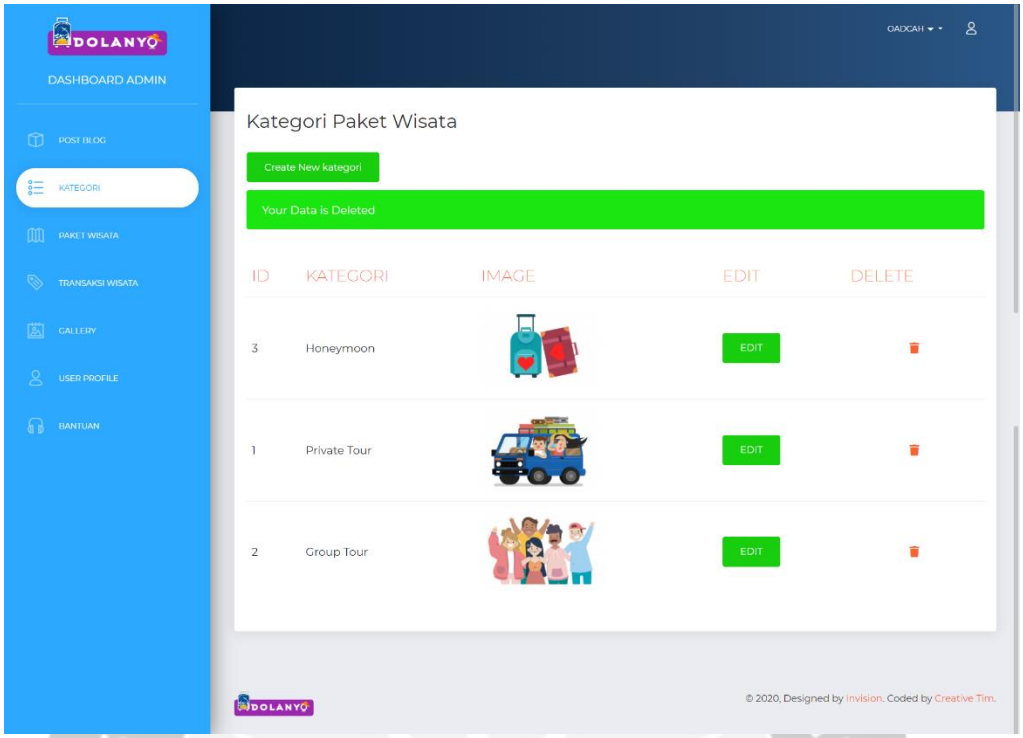
Untuk menyimpan hasil perubahan data kategori metode yang digunakan hampir sama dengan fungsi yang digunakan untuk menyimpan data kategori baru. Menurut gambar 5.40, perbedaannya, pada fungsi *update* akan melakukan pemeriksaan apakah ada file gambar baru atau tidak, kalau ada makan akan menjalankan penyimpanan file seperti dalam menyimpan data kategori baru, jika tidak maka akan lanjut ke step berikutnya. Setelah itu akan memanggil obyek kategori yang dipilih berdasarkan id kategori yang diedit. Setelah itu menyimpan data form dari parameter ke dalam obyek kategori yang dipanggil tadi lalu menyimpannya ke basis data.

```php
public function update(Request $request, $id)
{
   $post = Kategori::find($id);
    if ($request->File('file')) {
        $file=$request->file('file');
        $nama_file=time()."_".$file->getClientOriginalName();

        $tujuanupload='tumbnail';
        $resize_image=Image::make($file->getRealPath());
        $resize_image->resize(200,200,function($constraint){
        $constraint->aspectRatio();
        })->save($tujuanupload .'/'. $nama_file);

        $tujuanupload='data_file';

        $file->move($tujuanupload,$nama_file);
        $post->image= $nama_file;
    }
    $post->kategori = $request->kategori;
    $post->maxpeserta = $request->maksimal;
    $post->minpeserta = $request->minimal;
    $post->save();

    return redirect('/admin/kategori')->with('success','Your Data is Updated');
}
```

Gambar 5.40 Potongan kode ubah data kategori.

5.2.26. Implementasi Halaman *Dashboard Delete* Kategori Paket Wisata

Halaman *Dashboard Delete* Kategori Paket Wisata seperti pada gambar 5.40 adalah halaman yang digunakan untuk menghapus data Kategori Paket Wisata yang dipilih berdasarkan letak baris tombol *Delete* di tabel. Tombol *Delete* digunakan untuk mengkonfirmasi bahwa data di dalam baris tombol tersebut akan dihapus. Setelah dihapus maka akan mengarahkan pengguna ke halaman Kategori Paket Wisata dan muncul pemberitahuan *Your data is Deleted* yang berarti data telah berhasil dihapus.



Gambar 5. 41 Implementasi Halaman Dashboard Delete Kategori Paket Wisata.

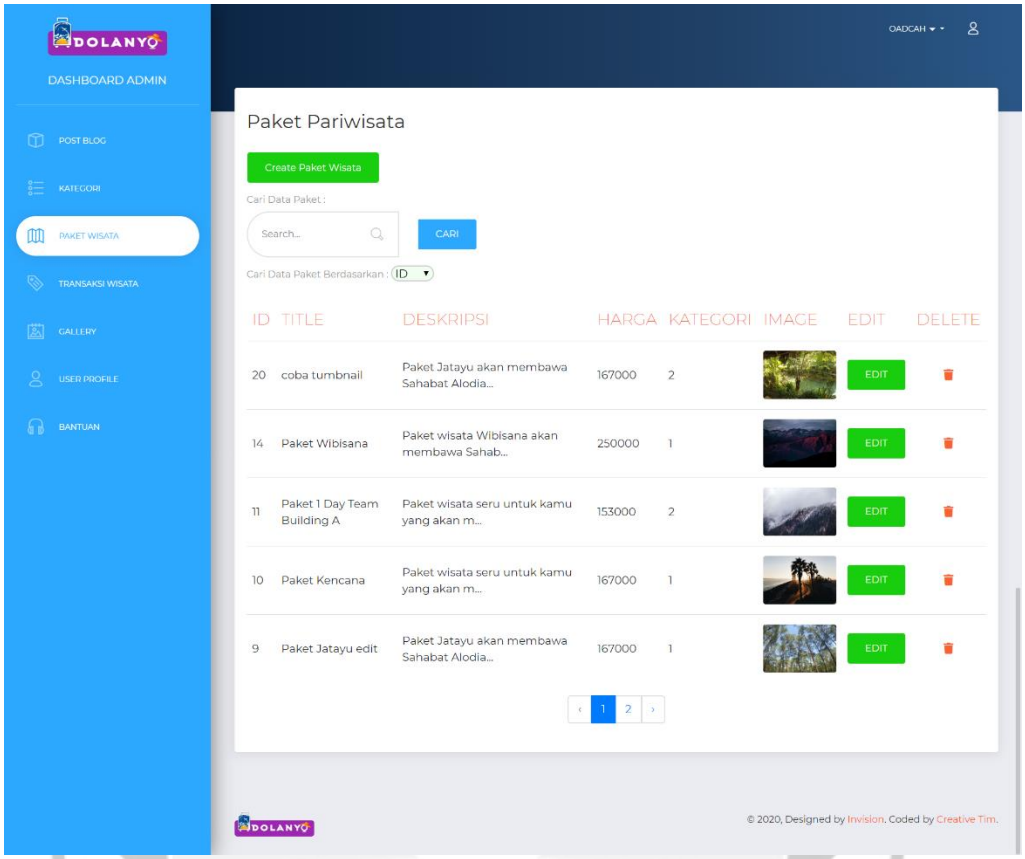
Untuk melakukan hapus data kategori dapat dilihat pada potongan kode di gambar 5.42. fungsi destroy memiliki parameter id kategori yang digunakan untuk menghapus data berdasarkan parameter id kategori yang dipilih. Selain menghapus data kategori fungsi ini juga akan menghapus semua data paket yang memiliki kategori yang sama dengan kategori yang dihapus. Oleh karena itu fungsi ini akan mengambil data obyek paket berdasarkan id paket dan menyimpannya ke dalam variabel $pakets. Karena obyek paket yang diambil bisa lebih dari 1, maka untuk menghapus paket menggunakan perulangan *foreach*. Setiap perulangan akan memanggil *Controller* PaketController, dengan fungsi *destroy* untuk menghapus setiap paket yang ada di obyek $pakets. Setelah itu akan menghapus file gambar di direktori dengan menggunakan bantuan *helper File* yang disediakan oleh *framework Laravel.*

```php
public function destroy($id)
{
    $posts = Kategori::findOrfail($id);
    $pakets = DB::table('pakets')->select('id','image')->where('kategori',$posts->id)->get();
    foreach ($pakets as $paket)
    {
        app()->call('App\Http\Controllers\PaketController@destroy',[$paket->id]);
    }
    File::delete('data_file/'.$posts->image);
    File::delete('tumbnail/'.$posts->image);
    $posts ->delete();

    return redirect('/admin/kategori')->with('success','Your Data is Deleted');
}
```

Gambar 5.42 Potongan kode untuk hapus data kategori.

5.2.27. Implementasi Halaman *Dashboard* Paket Wisata

Halaman *Dashboard* Paket Wisata merupakan halaman yang digunakan untuk menampilkan seluruh data Paket Wisata di basis data. Seperti pada gambar 5.43 atribut kolom data yang ditampilkan yaitu ID, *Tittle*, Deskripsi, *Overview*, Fasilitas, *Image*, Edit, dan *Delete*. Di halaman ini terdapat tombol *Create New Paket* Wisata yang digunakan untuk memasukkan data Paket Wisata baru. Tombol Edit digunakan untuk mengedit data Paket Wisata yang ada, sedangkan tombol *Delete* digunakan untuk menghapus data Paket Wisata di basis data. Halaman ini juga terdapat tombol Cari yang digunakan untuk mencari data Paket yang ada di basis data. Pencarian dilakukan dengan mencocokkan masukan Cari dengan data di basis data berdasarkan atribut ID atau *Tittle.* Halaman ini menampilkan data dengan *Pagination* yang akan menampilkan 5 data dalam satu halaman jika jumlah data lebih dari 5, sehingga lebih efisien untuk *loading* halaman ini.



Gambar 5.43 Implementasi Halaman Dashboard Paket Wisata.

Untuk menampilkan data paket wisata ke halaman admin dapat dilihat di gambar 5.44. Fungsi all akan memanggil model Paket dan menyimpan setiap obyek paket yang dipanggil ke dalam variabel $paket yang kemudian akan ditampilkan ke *view* admin.paket.

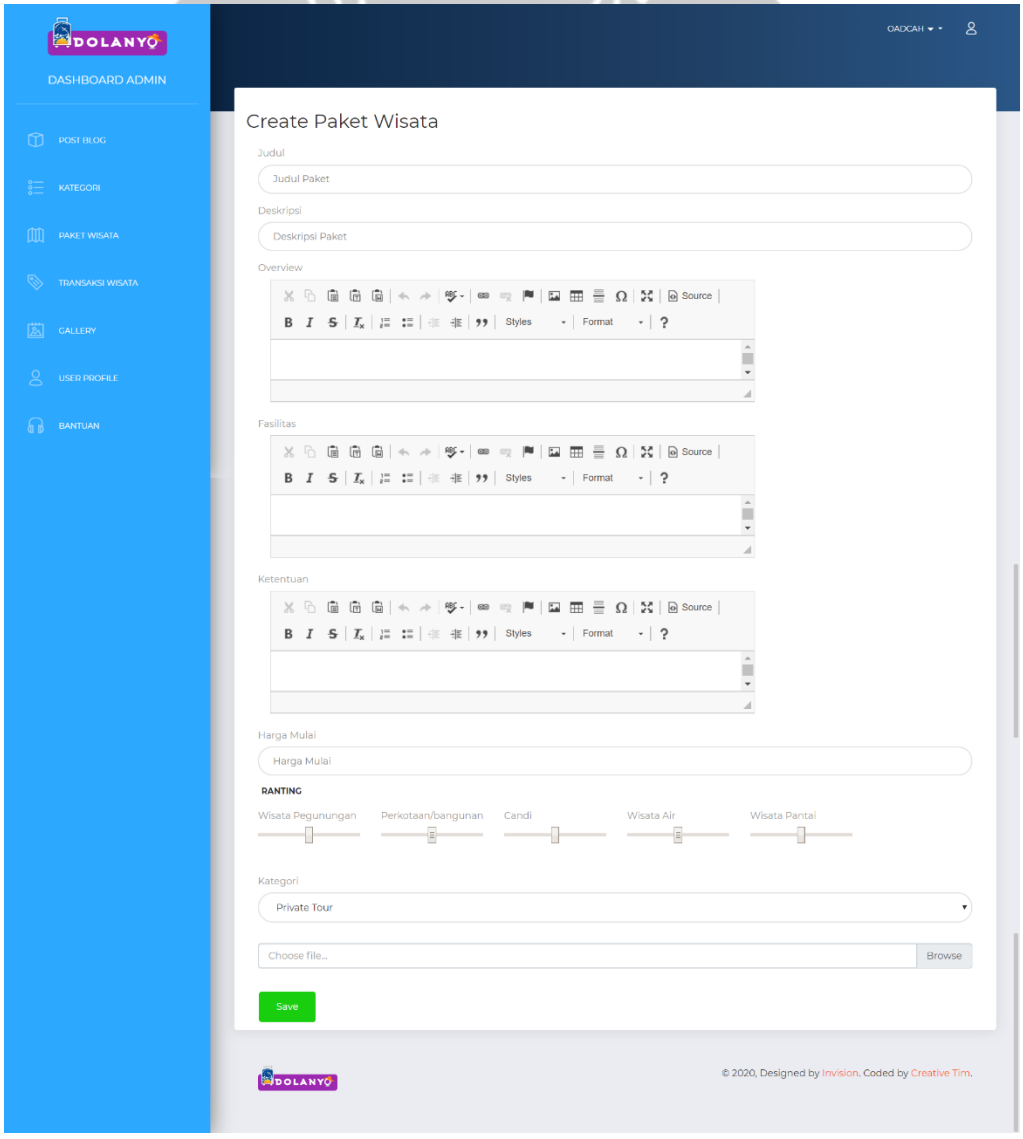
```
public function all(){

    $paket = Paket::orderBy('created_at', 'DESC')->paginate(5);
    return view('admin.paket')->with('paket',$paket);
}
```

Gambar 5.44 Potongan kode untuk tampil paket wisata.

5.2.28. Implementasi Halaman *Dashboard Create* Paket Wisata

Halaman *Dashboard Create New* Paket Wisata digunakan untuk menambahkan data Paket Wisata ke dalam basis data. Seperti gambar 5.45,

halaman ini memerlukan masukan Judul, Deskripsi, *Overview*, Fasilitas, Ketentuan, Harga minimum per pak, Rating, Kategori, dan *Image*. Masukan *Overview*, Fasilitas, dan Ketentuan menggunakan *input Textarea* dan *plugin CKeditor* sehingga bisa memasukkan data dalam bentuk format paragraf dan akan di konversi ke kode HTML. masukan Rating menggunakan *Slider* untuk mempermudah memasukkan data yang sudah di atur jangkauan nilainya. Pada halaman ini terdapat tombol *Save* yang digunakan untuk menyimpan seluruh data yang ada di *Form* ke dalam basis data.



Gambar 5.45 Implementasi Halaman Dashboard Create Paket Wisata.

Untuk menyimpan data paket wisata yang baru disimpan dapat dilihat di potongan kode dalam gambar 5.46. Fungsi store memiliki parameter variabel yang berisikan data kategori yang diisikan di form pada gambar 5.45. Fungsi ini akan membuat obyek paket wisata baru dan disimpan ke dalam variabel $post. Setelah itu akan *generate* nama *file* gambar paket wisata menjadi dua yaitu *file* original dan *file thumbnail* dan menyimpannya di variabel $file_name karena basis data hanya akan menyimpan nama gambarnya saja, sedangkan *file* gambar akan disimpan di direktori *thumbnail* dan data_file. Setelah itu semua data dari form yang ada di parameter *request* disimpan di dalam obyek paket wisata dan disimpan ke dalam basis data.

```
public function store(Request $request)
{
    $post=new Paket;
    $file=$request->file('file');
    $nama_file=time()."_".$file->getClientOriginalName();
    $tujuanupload='tumbnail';
    $resize_image=Image::make($file->getRealPath());
    $resize_image->resize(200,200,function($constraint){
        $constraint->aspectRatio();
    })->save($tujuanupload .'/'. $nama_file);
    $tujuanupload='data_file';
    $file->move($tujuanupload,$nama_file);
    $post->pegunungan = $request->ranting1;
    $post->bangunan = $request->ranting2;
    $post->sungai = $request->ranting3;
    $post->pantai = $request->ranting4;
    $post->title = $request->title;
    $post->deskripsi = $request->deskripsi;
    $post->overview = $request->overview;
    $post->fasilitas = $request->fasilitas;
    $post->ketentuan = $request->ketentuan;
    $post->kategori = $request->kategori;
    $post->harga_mulai=$request->harga_mulai;
    $post->image= $nama_file;
    $post->save();
```

Gambar 5.46 Potongan kode untuk menyimpan data paket wisata baru bagian 1.

Setelah menyimpan data paket wisata ke dalam basis data, fungsi ini akan melakukan pembaharuan pada data tabel rekomendasi. Pembaharuan dilakukan karena ada data paket wisata baru yang masuk sehingga perhitungan rekomendasi setiap pengguna dengan paket baru ini harus dilakukan dan disimpan di tabel rekomendasi. Untuk menyimpan data rekomendasi dapat

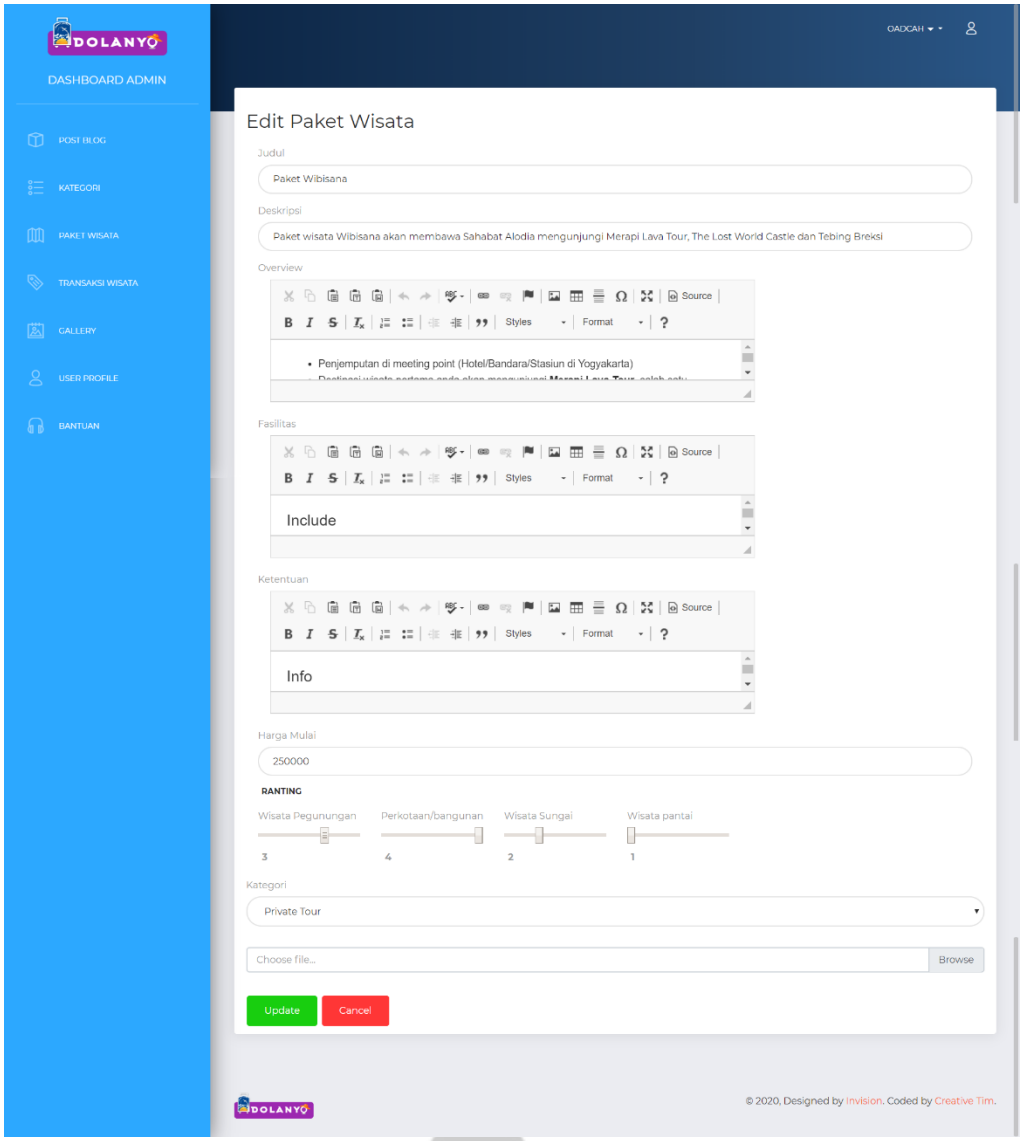
dilihat di gambar 5. 47. Fungsi *store* akan mengambil data seluruh pengguna dari basis data dan menyimpannya dalam variabel $users. Setelah itu akan memanggil paket yang baru saja disimpan di basis data dan disimpan di dalam variabel paket. Untuk melakukan pembaharuan data rekomendasi setiap pengguna yang terdaftar maka menggunakan perulangan *foreach* pada data *user*. Setiap user akan dilakukan perhitungan rekomendasinya dan menyimpan data rekomendasi ke dalam basis data dengan memanggil *Controller* Rekomendsicontroller@create dengan parameter user serta hasil perhitungannya.

```
//UPDATE DATA REKOMENDASI USER SETELAH INPUT PAKET BARU//
$users=DB::table('users')->select('id','pegunungan','bangunan','sungai','pantai')->get();
$paket = DB::table('pakets')->where('title', $request->title)->value('id');
foreach ($users as $user)
{
    $a = [$user->pegunungan, $user->bangunan, $user->sungai, $user->pantai];
    $b = [$request->ranting1, $request->ranting2, $request->ranting3, $request->ranting4];

    $euclidean =$this->eucDistance($a, $b);
    $sim=1/(1+$euclidean);
    app()->call('App\Http\Controllers\RekomendasiController@create',[$user->id,$paket,$request->title,$sim]);
}
return redirect('/admin/paket');
}
```

Gambar 5.47 Potongan kode tambah data paket wisata baru bagian 2.

5.2.29. Implementasi Halaman *Dashboard Edit* Paket Wisata

Halaman *Dashboard* Edit Paket Wisata digunakan untuk mengubah data Paket Wisata yang ada di basis data. Seperti gambar 5.48, halaman ini meminta masukan Judul, Deskripsi, *Overview*, Fasilitas, Ketentuan, Harga minimum per pak, Rating, Kategori, dan *Image*. Pada halaman ini terdapat tombol *Update* yang digunakan untuk menyimpan seluruh perubahan data yang ada di *Form* ke dalam basis data. Selain tombol *Update* terdapat tombol *Cancel* yang digunakan untuk membatalkan perubahan data yang tadinya akan diubah.



Gambar 5.48 Implementasi Halaman Dashboard Edit Paket Wisata

Untuk menyimpan hasil perubahan data paket wisata metode yang digunakan hampir sama dengan fungsi yang digunakan untuk menyimpan data paket wisata baru. Menurut gambar 5.49, perbedaannya, pada fungsi *Update* akan melakukan pemeriksaan apakah ada *file* gambar baru atau tidak, kalau ada makan akan menjalankan penyimpanan *file* seperti dalam menyimpan data kategori baru, jika tidak maka akan lanjut ke prosedur berikutnya. Setelah itu akan memanggil obyek paket wisata yang dipilih berdasarkan id paket wisata yang diedit. Setelah itu menyimpan data form dari parameter ke dalam obyek paket wisata yang dipanggil tadi lalu menyimpannya ke basis data.

```php
public function update(Request $request, $id)
{
    $post = Paket::find($id);
    if ($request->File('image')) {
        File::delete('data_file/'.$post->image);
        File::delete('tumbnail/'.$post->image);
        $file = $request->file('image');
        $nama_file=time()."_".$file->getClientOriginalName();
        $tujuanupload='tumbnail';
        $resize_image=Image::make($file->getRealPath());
        $resize_image->resize(200,200,function($constraint){
            $constraint->aspectRatio();
        })->save($tujuanupload .'/'. $nama_file);
        $tujuanupload='data_file';
        $file->move($tujuanupload,$nama_file);
        $post->image= $nama_file;
    }
    $oldtitle=$post->title;
    $post->pegunungan = $request->ranting1;
    $post->bangunan = $request->ranting2;
    $post->sungai = $request->ranting3;
    $post->pantai = $request->ranting4;
    $post->title = $request->title;
    $post->deskripsi = $request->deskripsi;
    $post->overview = $request->overview;
    $post->fasilitas = $request->fasilitas;
    $post->ketentuan = $request->ketentuan;
```

Gambar 5.49 Potongan kode untuk ubah data paket wisata bagian 1.

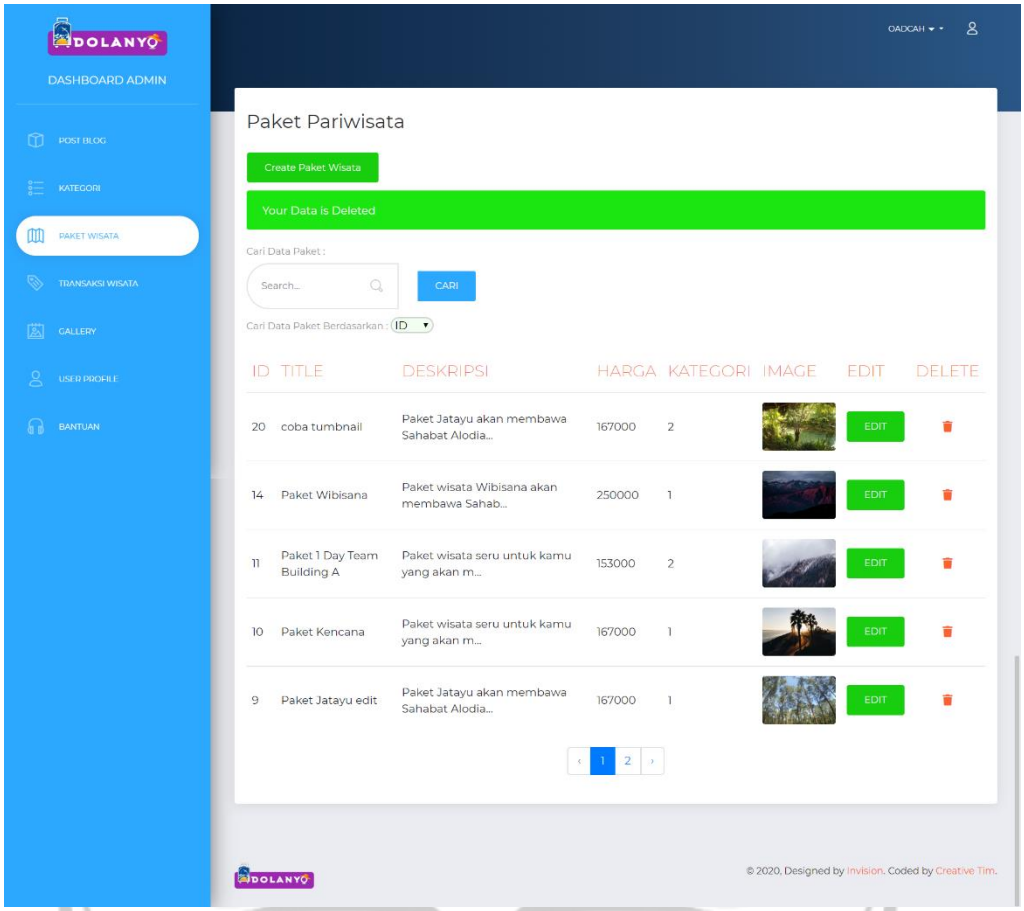
Sama dengan buat paket baru, fungsi *Update* ini juga akan melakukan perubahan pada tabel rekomendasi. Perbedaannya seperti pada gambar 5.50, yaitu sebelum dilakukan perhitungan rekomendasi dan penyimpanan data fungsi ini akan menghapus data rekomendasi berdasarkan id paket yang akan di ubah. Penghapusan data rekomendasi bertujuan supaya data rekomendasi tidak terjadi duplikasi data. Setelah dihapus maka fungsi baru menjalankan prosedur perhitungan dan penyimpanan data rekomendasi ke tabel rekomendasi di basis data.

```
    //UPDATE DATA REKOMENDASI USER SETELAH INPUT PAKET BARU//
    Rekomendasi :: where ('paket',$oldtitle)->delete();

    $users=DB::table('users')->select('id','pegunungan','bangunan','sungai','pantai')->get();
    $paket = DB::table('pakets')->where('title', $request->title)->value('id');
    foreach ($users as $user)
    {
        $a = [$user->pegunungan, $user->bangunan, $user->sungai, $user->pantai];
        $b = [$request->ranting1, $request->ranting2, $request->ranting3, $request->ranting4];
        $euclidean =$this->eucDistance($a, $b);
        $sim=1/(1+$euclidean);
        app()->call('App\Http\Controllers\RekomendasiController@create',[$user->id,$paket,$request->title,$sim]);
    }
    return redirect('/admin/paket')->with('success','Your Data is Updated');
}
```

Gambar 5.50 Potongan kode untuk ubah paket wisata bagian 2.

5.2.30. Implementasi Halaman *Dashboard Delete* Paket Wisata

Halaman *Dashboard Delete* Paket Wisata seperti pada gambar 5.51 adalah halaman yang digunakan untuk menghapus data Paket Wisata yang dipilih berdasarkan letak baris tombol *Delete* di tabel data. Tombol *Delete* digunakan untuk mengkonfirmasi bahwa data di dalam baris tombol tersebut akan dihapus. Setelah dihapus maka akan mengarahkan pengguna ke halaman Paket Wisata dan muncul pemberitahuan *Your data is Deleted* yang berarti data telah berhasil dihapus.



Gambar 5.51 Implementasi Halaman Delete Paket Wisata.

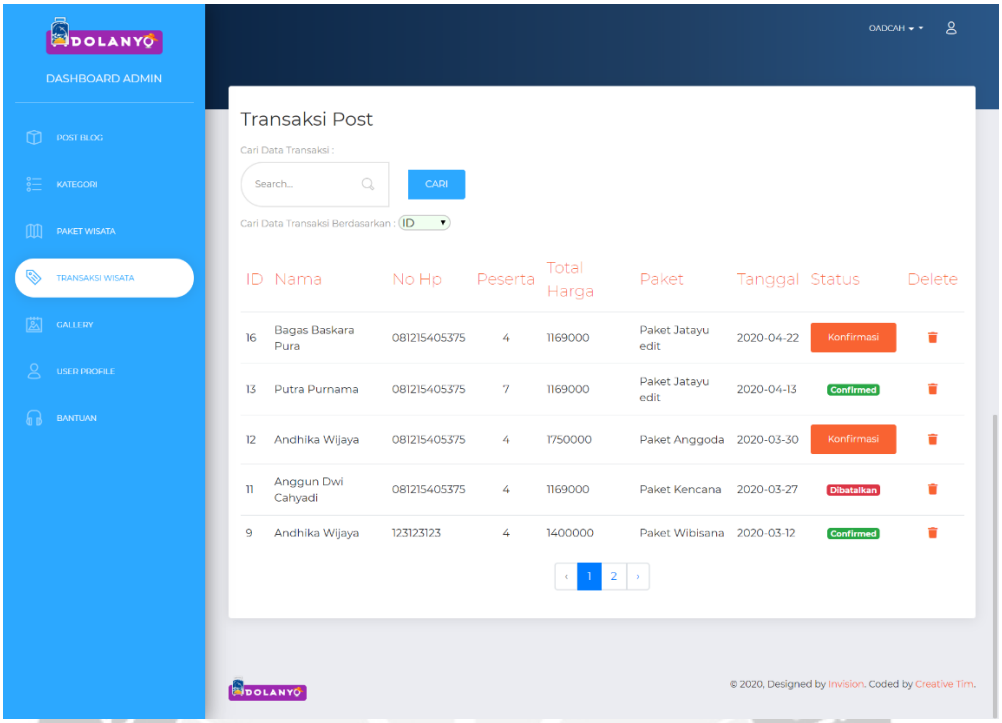
Untuk melakukan hapus data transaksi dapat dilihat pada potongan kode di gambar 5.52. fungsi *destroy* memiliki parameter id paket wisata yang digunakan untuk menghapus data berdasarkan parameter id paket wisata yang dipilih. Selain menghapus data paket wisata fungsi ini juga akan menghapus semua data rekomendasi pengguna yang memiliki id paket yang akan dihapus ini. Fungsi ini akan memanggil data paket wisata berdasarkan id paket dan disimpan di variabel $pakets. Setelah itu akan memanggil *Controller RekomendasiController@destroybypaket* dengan parameter id paket yang digunakan untuk menghapus data rekomendasi berdasarkan id paket yang akan dihapus. Setelah itu akan menghapus file gambar di direktori dengan menggunakan bantuan *helper File* yang disediakan oleh *framework Laravel* dan menghapus data paket menggunakan fungsi $pakets -> *delete()*.

```php
public function destroy($id)
{
    $pakets = Paket::findOrfail($id);
    app()->call('App\Http\Controllers\RekomendasiController@Destroybypaket',[$id]);

    File::delete('data_file/'.$pakets->image);
    File::delete('tumbnail/'.$pakets->image);
    $pakets ->delete();
    return redirect('/admin/paket')->with('success','Your Data is Deleted');
}
```

Gambar 5.52 Potongan kode untuk hapus paket wisata.

5.2.31. Implementasi Halaman *Dashboard* Transaksi

Halaman *Dashboard* Transaksi merupakan halaman yang digunakan untuk menampilkan seluruh data Transaksi yang ada di basis data. Seperti pada gambar 5.53 atribut kolom data yang ditampilkan yaitu *ID*, Nama, *No Hp*, Peserta, Total Harga, Paket, Tanggal, Status dan Delete. Di halaman ini terdapat tombol *Form* Pencarian yang digunakan untuk mencari data Transaksi yang ada di basis data berdasarkan *ID* atau Nama. Kolom Status berisikan data status dari transaksi tersebut. Bila belum dilakukan konfirmasi makan kolom status akan menampilkan tombol Konfirmasi, sedangkan jika statusnya sudah diubah maka kolom Status akan menampilkan status confirmed yang berarti transaksi sudah dikonfirmasi dan status Dibatalkan jika transaksi dibatalkan. Tombol Konfirmasi di kolom Status digunakan untuk melakukan konfirmasi transaksi kepada pelanggan dengan membuka halaman baru yaitu halaman Konfirmasi Transaksi. Pada kolom *Delete* berisikan tombol *Delete* yang digunakan untuk menghapus data transaksi berdasarkan letak barus tombol *Delete* di tabel data.



Gambar 5.53 Implementasi Halaman Dashboard Transaksi.

Untuk menampilkan data transaksi ke halaman admin dapat dilihat di gambar 5.54. Fungsi posted akan memanggil model Transaksi dan menyimpan setiap obyek transaksi yang dipanggil ke dalam variabel $transaksi yang kemudian akan ditampilkan ke *view* admin.transaksi.

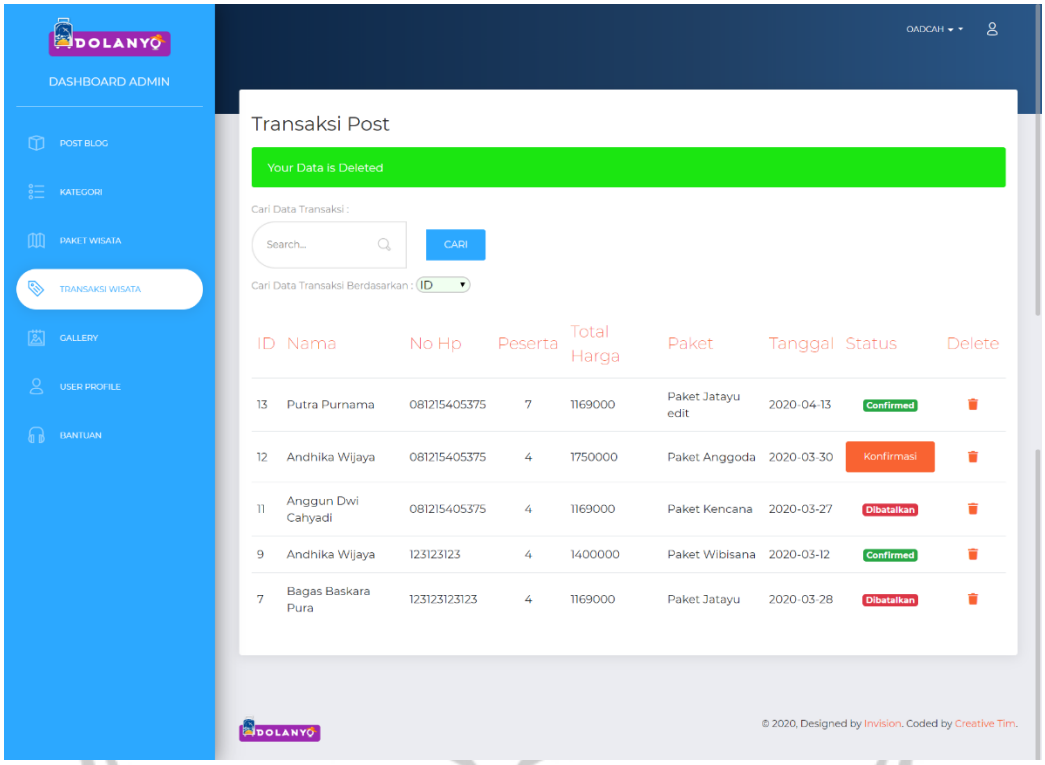
```
public function posted()
{
    $transaksi = Transaksi::orderBy('created_at', 'DESC')->paginate(5);

    return view('admin.transaksi')->with('posts',$transaksi);
}
```

Gambar 5.54 Potongan kode untuk tampil data transaksi.

5.2.32. Implementasi Halaman *Dashboard Delete* Transaksi

Halaman *Dashboard Delete* Transaksi seperti pada gambar 5.55 adalah halaman yang digunakan untuk menghapus data Transaksi yang dipilih berdasarkan letak baris tombol *Delete* di tabel data. Tombol *Delete* digunakan untuk mengkonfirmasi bahwa data di dalam baris tombol tersebut akan dihapus. Setelah dihapus maka akan mengarahkan pengguna ke halaman

Transaksi dan muncul pemberitahuan *Your data is Deleted* yang berarti data telah berhasil dihapus.



Gambar 5.55 Implementasi Halaman Dashboard Delete Transaksi

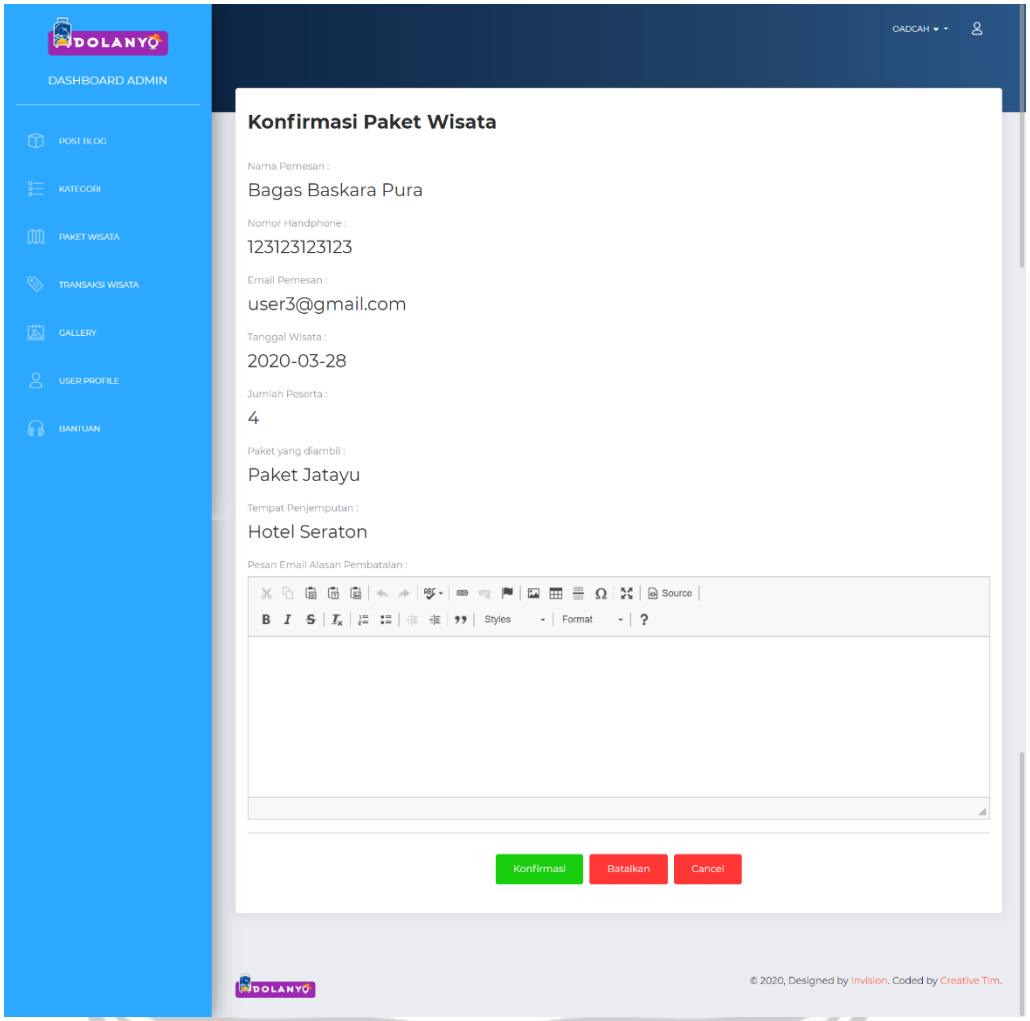
Untuk melakukan hapus data transaksi dapat dilihat pada potongan kode di gambar 5.56. fungsi *destroy* memiliki parameter id transaksi yang digunakan untuk menghapus data berdasarkan parameter id transaksi yang dipilih. Fungsi ini akan memanggil transaksi berdasarkan id transaksi dan melakukan metode *delete.*

```
public function destroy($id)
{
    $posts = Transaksi::findOrfail($id);
    $posts ->delete();

    return redirect('/admin/transaksi')->with('success','Your Data is Deleted');
}
```

Gambar 5.56 Potongan kode untuk hapus data transaksi.

5.2.33. Implementasi Halaman *Dashboard* Konfirmasi Transaksi

Halaman *Dashboard* Konfirmasi Transaksi merupakan halaman yang digunakan untuk menampilkan data transaksi yang akan dikonfirmasi. Seperti pada gambar 4.57, halaman ini menampilkan data transaksi berupa Nama Pemesan, *No Handphone*, *Email* Pemesan, Tanggal Wisata, Jumlah Peserta, Paket yang Diambil, serta Tempat Penjemputan. Selain itu, halaman ini memiliki tiga tombol yaitu tombol Konfirmasi, Batalkan, dan Cancel. Tombol Konfirmasi digunakan untuk mengkonfirmasi transaksi bahwa transaksi sudah diproses dan pengguna akan mendapat *Email* konfirmasi berupa nota transaksi yang berisikan jumlah yang harus dibayarkan dan informasi mengenai Paket Wisata yang diambil. Tombol Batalkan digunakan untuk membatalkan transaksi karena terjadi suatu hal. Tombol Batalkan ini menggunakan masukan Pesan Email Penolakan untuk dikirimkan ke pengguna dalam bentuk *Email* Penolakan transaksi yang di dalamnya berisikan alasan penolakan berdasarkan masukan Pesan *Email* Penolakan yang diisikan di Form tadi. Serta Tombol *Cancel* yang berfungsi untuk membatalkan proses konfirmasi dan kembali ke halaman *Dashboard* Transaksi.



Gambar 5.57 Implementasi Halaman Dashboard Konfirmasi Transaksi.

Ketika tombol konfirmasi pada gambar 5.57 ditekan makan akan menjalankan fungsi *Send* seperti pada potongan kode di gambar 5.58. fungsi Send memiliki parameter $request yang berisikan data yang akan dikirim kepada pengguna. Fungsi akan menyimpan data $request ke dalam data *array* $data. Setelah itu fungsi akan mengubah status transaksi menjadi 1 yang berarti transaksi sudah dikonfirmasi. Setelah itu fungsi akan mengirimkan email kepada pelanggan dengan menggunakan bantuan *helper Mail* yang disediakan *framework Laravel.* Setelah itu admin akan dialihkan ke halaman admin transaksi dengan pesan bahwa email berhasil dikirim kepada pengguna.

```php
public function send(Request $request){
    $data = [
        'nama' => $request->nama,
        'handphone' =>$request->handphone,
        'peserta' =>$request->peserta,
        'paket' =>$request->paket,
        'email' =>$request->email,
        'tanggal' =>$request->tanggal,
        'overview' =>$request->overview,
        'deskripsi' =>$request->deskripsi,
        'idtransaksi' =>$request->idtransaksi,
        'harga' =>$request->harga,
        'idpaket' =>$request->idpaket,
        'tempat' =>$request->tempat,
      ];
      $transaksi=Transaksi::find($request->id);
      $transaksi->konfirmasi=1;
      $transaksi->update();
      Mail::to($request->email)->send(new TransaksiEmail($data));
      return redirect('/admin/transaksi')->with('success','Status sudah dikonfirmasi, Email terkirim ke pelanggan');
}
```

Gambar 5.58 Potongan kode konfirmasi transaksi.

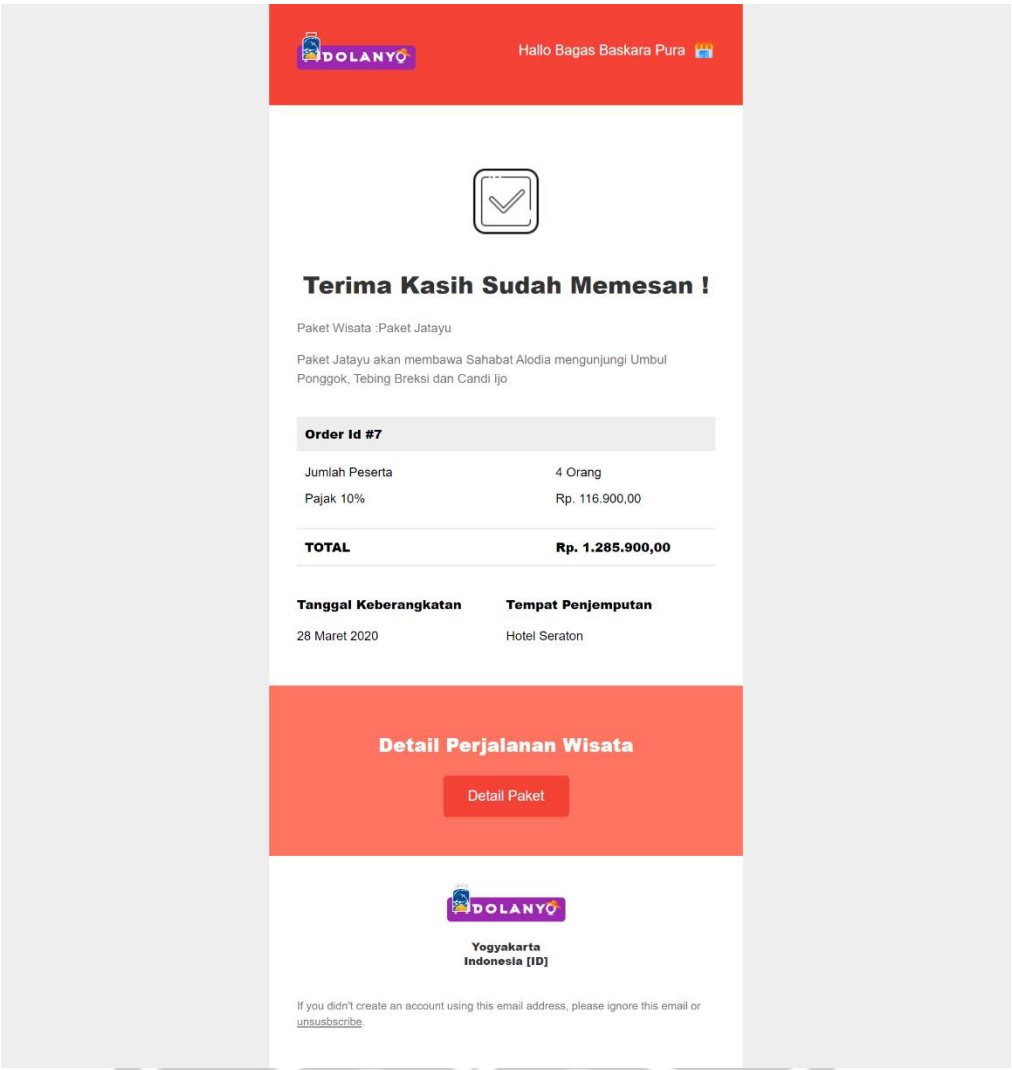
Ketika tombol Batalkan pada gambar 5.57 ditekan makan akan menjalankan fungsi *sendbatal* seperti pada potongan kode di gambar 5.59. sama seperti pada fungsi *send*, fungsi send memiliki parameter $request yang berisikan data yang akan dikirim kepada pengguna. Fungsi akan menyimpan data $request ke dalam data *array* $data. Setelah itu fungsi akan mengubah status transaksi menjadi 1 yang berarti transaksi sudah dikonfirmasi. Setelah itu fungsi akan mengirimkan email kepada pelanggan dengan menggunakan bantuan *helper Mail* yang disediakan *framework Laravel*. Setelah itu admin akan dialihkan ke halaman admin transaksi dengan pesan bahwa email berhasil dikirim kepada pengguna. Perbedaannya fungsi ini akan meminta alasan pembatalan yang akan dikirim kepada pengguna.

```php
public function sendbatal(Request $request){
    $data = [
        'nama' => $request->nama,
        'handphone' =>$request->handphone,
        'peserta' =>$request->peserta,
        'paket' =>$request->paket,
        'email' =>$request->email,
        'tanggal' =>$request->tanggal,
        'overview' =>$request->overview,
        'deskripsi' =>$request->deskripsi,
        'idtransaksi' =>$request->idtransaksi,
        'harga' =>$request->harga,
        'idpaket' =>$request->idpaket,
        'tempat' =>$request->tempat,
      ];
      $transaksi=Transaksi::find($request->id);
      $transaksi->konfirmasi=3;
      $transaksi->update();
      Mail::to($request->email)->send(new BatalTransaksiEmail($data));
      return redirect('/admin/transaksi')->with('success','Status sudah dikonfirmasi, Email Pembatalan terkirim
}
```

Gambar 5.59 Potongan kode pembatalan transaksi

5.2.34. Implementasi Halaman *Email* Konfirmasi Transaksi

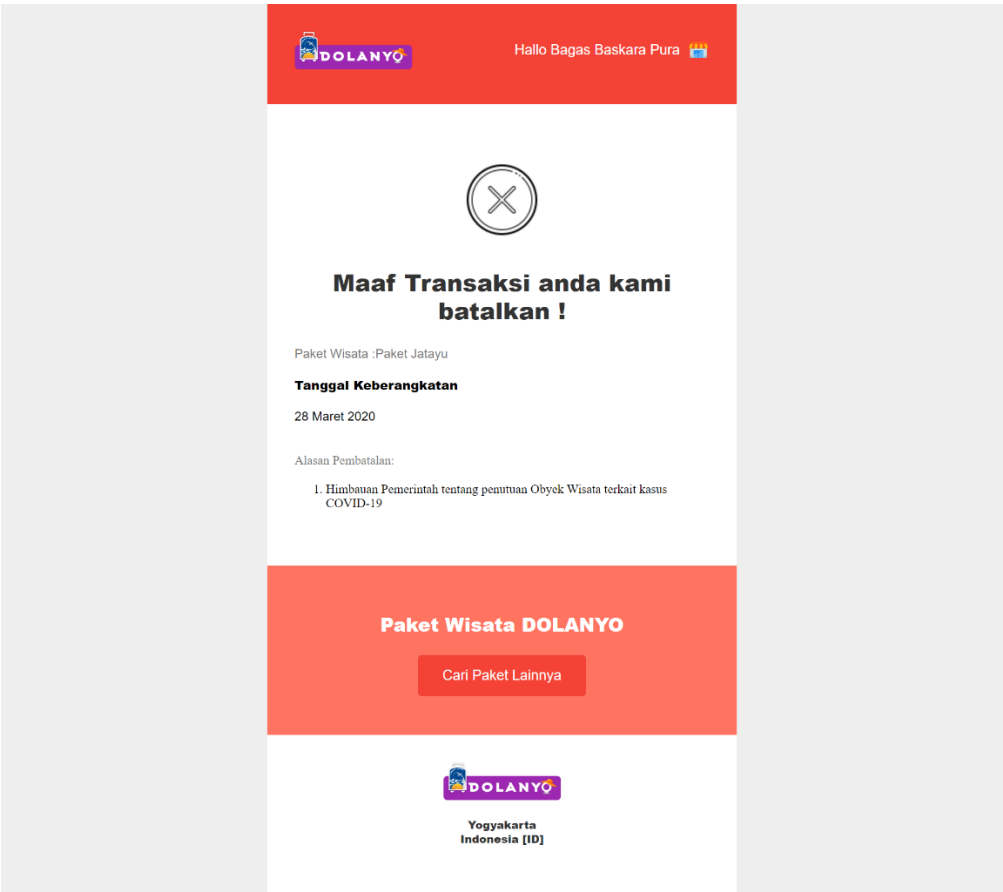
Halaman *Email* Konfirmasi Transaksi seperti pada gambar 5.60, merupakan halaman *Email* untuk menampilkan *invoice* dari transaksi yang sudah dilakukan dan sudah dikonfirmasi oleh Admin. Pada Halaman *invoice* ini data yang ditampilkan yaitu nama pengguna, nama paket wisata, jumlah peserta, harga per pak, total harga untuk seluruh peserta, nilai pajak, tanggal wisata, tempat penjemputan, serta total seluruh uang yang harus dibayarkan. Di halaman ini juga terdapat tombol Detail Paket yang digunakan untuk mengakses data paket yang sudah di ambil dan melihat detail informasi mengenai paket tersebut.



Gambar 5.60 Implementasi Halaman Email Konfirmasi Transaksi.

5.2.35. Implementasi Halaman *Email* Pembatalan Transaksi

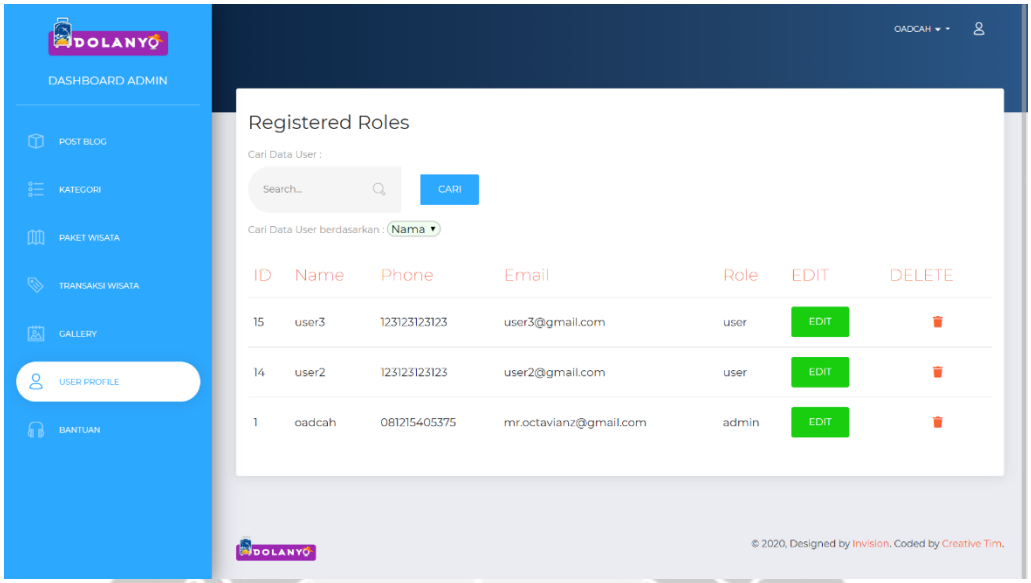
Halaman *Email* Pembatalan Transaksi seperti pada gambar 5.61 digunakan ketika admin memilih untuk membatalkan transaksi karena alasan tertentu. Pada halaman *Email* ini menampilkan data nama pengguna, nama paket, tanggal keberangkatan, serta alasan pembatalan. Di halaman Email ini memiliki tombol Cek Paket Lainnya untuk mengarahkan pengguna untuk menemukan paket lainnya yang tersedia.



Gambar 5.61 Implementasi Halaman Email Pembatalan Transaksi.

5.2.36. Implementasi Halaman Dashboard User Profile

Halaman *Dashboard User Profile* merupakan halaman yang digunakan untuk menampilkan seluruh data *User* atau pengguna yang ada di basis data. Seperti pada gambar 5.62 atribut kolom data yang ditampilkan yaitu *ID*, Nama, *No HP*, *Email*, Edit, dan *Delete*. Di halaman ini terdapat tombol Edit digunakan untuk mengedit data *User* yang ada, sedangkan tombol *Delete* digunakan untuk menghapus data *User* di basis data.



Gambar 5.62 Implementasi Halaman Dashboard User Profile.

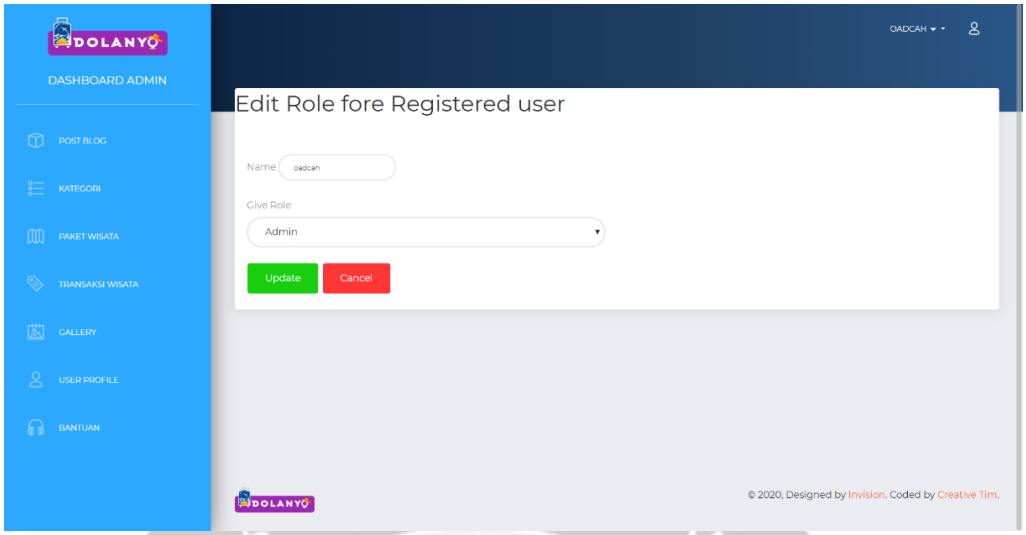
Untuk menampilkan data *user profile* ke halaman admin dapat dilihat di gambar 5.63. Fungsi *registered* akan memanggil model *User* dan menyimpan setiap obyek *user profile* yang dipanggil ke dalam variabel $users yang kemudian akan ditampilkan ke *view* admin.register.

```
public function registered(){

    $users = User::orderBy('created_at', 'DESC')->paginate(5);
    return view('admin.register')->with('users',$users);
}
```

Gambar 5.63 Potongan kode tampil data User profile.

5.2.37. Implementasi Halaman *Dashboard* Edit *User Profile*

Halaman *Dashboard* Edit *User Profile* digunakan untuk mengubah data *User Profile* yang ada di basis data. Seperti gambar 5.64, halaman ini memerlukan masukan Nama dan *Role*. *Role* terdapat dua pilihan yaitu sebagai Admin atau *User*. Pada halaman ini terdapat tombol *Update* yang digunakan untuk menyimpan seluruh perubahan data yang ada di *Form* ke dalam basis data. Selain tombol *Update* terdapat tombol *Cancel* yang digunakan untuk membatalkan perubahan data yang tadinya akan diubah.



Gambar 5.64 Implementasi Halaman Dashboard Edit User Profile.

Ketika tombol *Update* pada gambar 5.64 ditekan maka sistem akan menjalankan fungsi Registeredupdate seperti pada gambar 5.65. fungsi ini memiliki parameter $request yang berisikan data yang diubah, serta parameter $id yang berisikan id *user* yang diubah. Setelah itu fungsi akan memanggil model User dan menyimpan obyek user yang dipanggil berdasarkan id user ke dalam variabel *$users*. Setelah itu memasukkan nilai dari parameter $request ke dalam obyek user dan disimpan ke basis data.

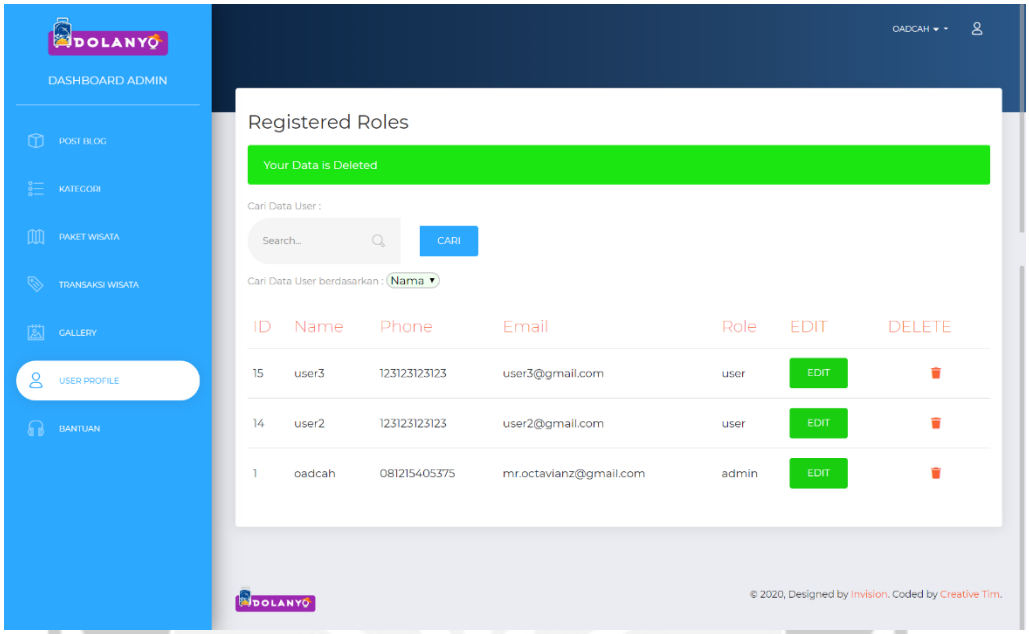
```
public function registerupdate(Request $request, $id)
{
    $users =User::find($id);
    $users->name=$request->input('username');
    $users->usertype=$request->input('usertype');
    $users->update();

    return redirect('/admin/role-register')->with('success','Your Data is Updated');
}
```

Gambar 5.65 Potongan kode ubah role user.

5.2.38. Implementasi Halaman Dashboard Delete User Profile

Halaman *Dashboard Delete User Profile* seperti pada gambar 5.66 adalah halaman yang digunakan untuk menghapus data *User Profile* yang dipilih berdasarkan letak baris tombol *Delete* di tabel. Tombol *Delete* digunakan untuk mengkonfirmasi bahwa data di dalam baris tombol tersebut akan dihapus. Setelah dihapus maka akan mengarahkan pengguna ke halaman

User Profile dan muncul pemberitahuan *Your data is Deleted* yang berarti data telah berhasil dihapus.



Gambar 5.66 Implementasi Halaman Dashboard Delete User Profile.

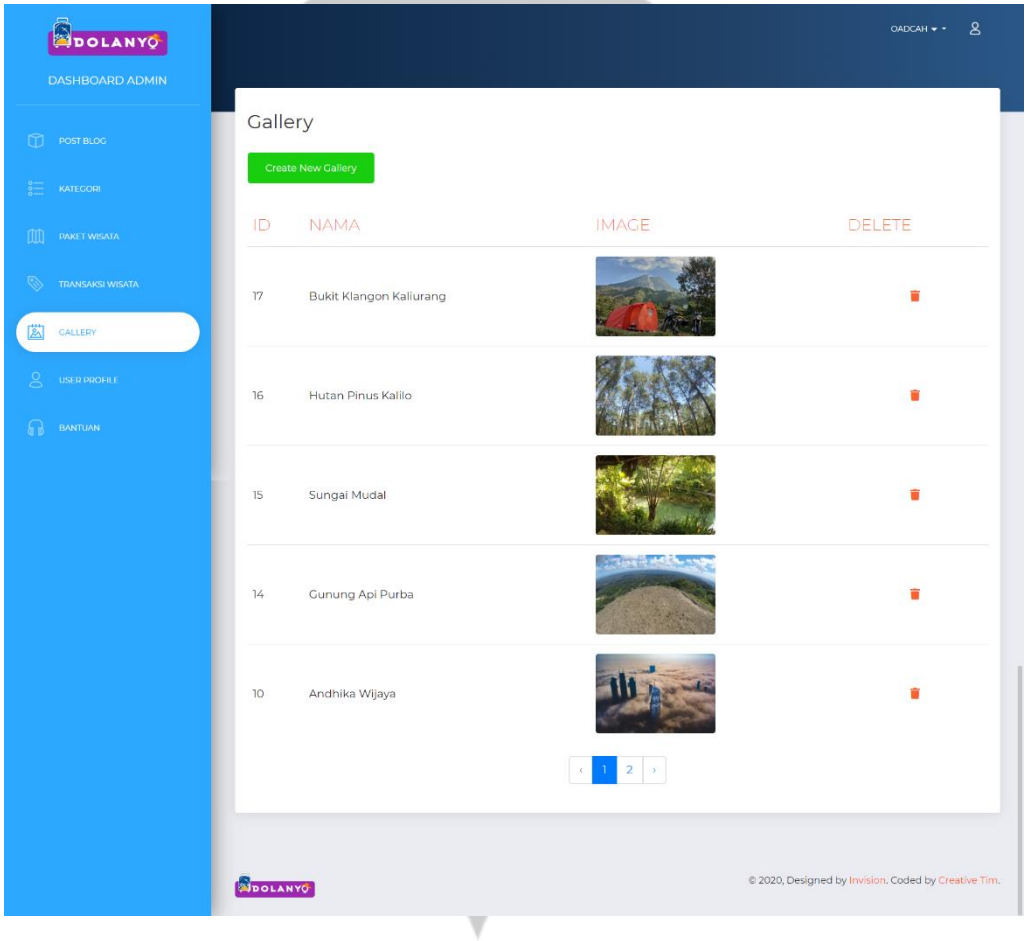
Untuk melakukan hapus data *user* dapat dilihat pada potongan kode di gambar 5.67. fungsi *registerdelete* memiliki parameter id *user* yang digunakan untuk menghapus data berdasarkan parameter id *user* yang dipilih. Fungsi ini akan memanggil *user* berdasarkan id *user* dan melakukan metode *delete*. Setelah itu fungsi akan menghapus data rekomendasi berdasarkan id user yang dihapus. Fungsi akan menghapus data rekomendasi dengan memanggil *Controller* RekomendasiController@Destroybyuser dengan parameter id user yang dihapus.

```
public function registerdelete($id)
{
    $users= User::findOrfail($id);
    $users->delete();
    app()->call('App\Http\Controllers\RekomendasiController@Destroybyuser',[$id]);

    return redirect('/admin/role-register')->with('success','Your Data is Deleted');
}
```

Gambar 5.67 Potongan kode untuk hapus user.

5.2.39. Implementasi Halaman *Dashboard Gallery*

Halaman *Dashboard Gallery* merupakan halaman yang digunakan untuk menampilkan seluruh data *Gallery* atau pengguna yang ada di basis data. Seperti pada gambar 5.68 atribut kolom data yang ditampilkan yaitu *ID*, Nama, *Image*, dan *Delete*. Di halaman ini terdapat tombol *Delete* digunakan untuk menghapus data *Gallery* yang memiliki baris tabel yang sama dengan tombol *Delete*.



Gambar 5.68 Implementasi Halaman Dashboard Gallery.

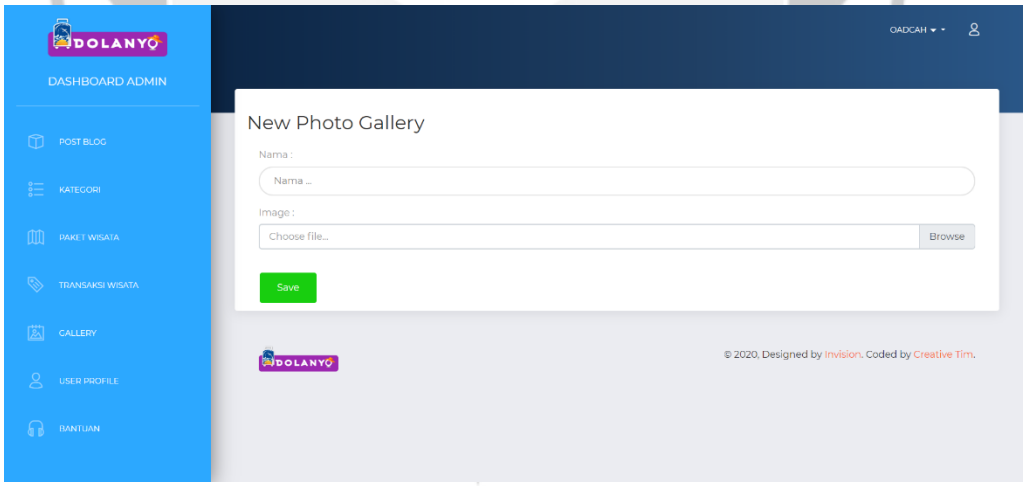
Untuk menampilkan data *gallery* ke halaman admin dapat dilihat di gambar 5.69. Fungsi *all* akan memanggil model *Gallery* dan menyimpan setiap obyek *gallery* yang dipanggil ke dalam variabel $ *gallery* yang kemudian akan ditampilkan ke *view* admin. *gallery*.

```
public function all(){

    $gallery = Gallery::orderBy('created_at', 'DESC')->paginate(5);
    return view('admin.gallery')->with('paket',$gallery);
}
```

Gambar 5.69 Potongan kode tampil gallery.

5.2.40. Implementasi Halaman Dashboard Create New Gallery

Ketika kita menekan tombol *Create New Gallery* di halaman *Dashboard Gallery*, kita akan dirujuk ke halaman *Create New Gallery*. Menurut gambar 5.40, halaman Dashboard *Create New Gallery* terdapat *Form* yang digunakan untuk mengisi atribut dari *Gallery*. Masukan Nama digunakan untuk memasukkan Nama dari *Gallery*. Masukan *Image* digunakan untuk memasukkan data *File* gambar yang akan ditampilkan di *Gallery*. Setelah itu tombol *Save* digunakan untuk menyimpan seluruh data yang sudah ada di *Form* ke dalam basis data.



Gambar 5.70 Implementasi Halaman Dashboard Create New Gallery.

Untuk menyimpan data *gallery* yang baru disimpan dapat dilihat di potongan kode dalam gambar 5.71. Fungsi store memiliki parameter variabel yang berisikan data *gallery* yang diisikan di form pada gambar 5.70. Fungsi ini akan membuat obyek *gallery* baru dan disimpan ke dalam variabel $post. Setelah itu akan *generate* nama *file* gambar kategori menjadi dua yaitu *file* original dan *file thumbnail* dan menyimpannya di variabel $file_name karena

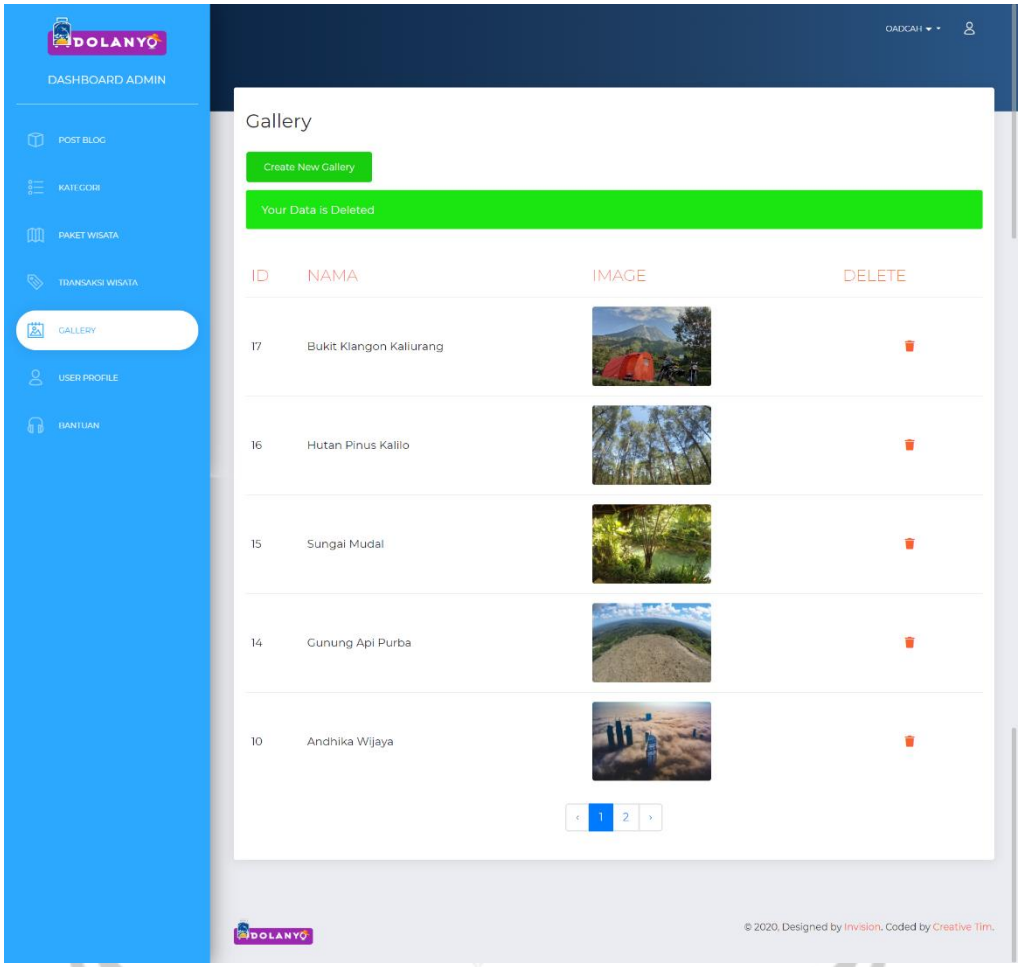
basis data hanya akan menyimpan nama gambarnya saja, sedangkan *file* gambar akan disimpan di direktori *thumbnail* dan data_file. Setelah itu semua data dari form yang ada di parameter $*request* disimpan di dalam obyek *gallery* selanjutnya data disimpan ke dalam basis data.

```
public function store(Request $request)
{
    $post=new Gallery;
    $file=$request->file('file');
    $nama_file=time()."_".$file->getClientOriginalName();
    $tujuanupload='tumbnail';
    $resize_image=Image::make($file->getRealPath());
    $resize_image->resize(400,400,function($constraint){
        $constraint->aspectRatio();
    })->save($tujuanupload .'/'. $nama_file);
    $tujuanupload='gallery';
    $file->move($tujuanupload,$nama_file);
    $post->nama = $request->nama;
    $post->image= $nama_file;
    $post->save();

    return redirect('/admin/gallery');
}
```

Gambar 5.71 Potongan kode untuk tambah gallery.

5.2.41. Implementasi Halaman Dashboard Delete Gallery

Halaman *Dashboard Delete Gallery* seperti pada gambar 5.72 adalah halaman yang digunakan untuk menghapus data *Gallery* yang dipilih dari basis data. Tombol *Delete* digunakan untuk mengkonfirmasi bahwa data di dalam baris tombol tersebut akan di hapus. Setelah di hapus makan akan muncul pemberitahuan bahwa data berhasil di hapus.



Gambar 5.72 Implementasi Halaman Dashboard Delete Gallery.

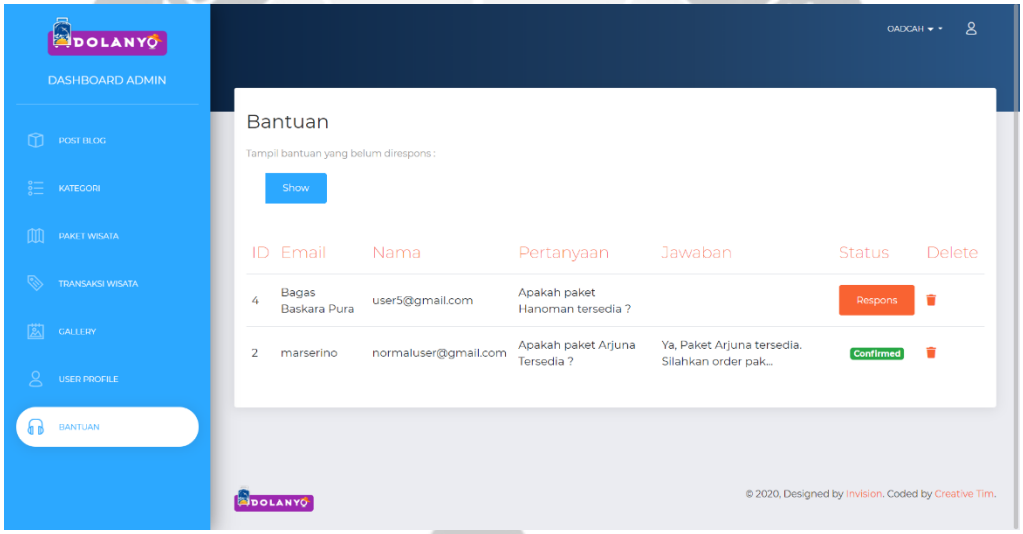
Untuk melakukan hapus data *gallery* dapat dilihat pada potongan kode di gambar 5.73. fungsi *delete* memiliki parameter id *gallery* yang digunakan untuk menghapus data berdasarkan parameter id *gallery* yang dipilih. Fungsi ini akan memanggil *gallery* berdasarkan id *gallery* dan melakukan fungsi *delete*. Setelah itu akan menghapus file gambar *gallery* di direktori dengan menggunakan bantuan *helper File* yang disediakan oleh *framework Laravel.*

```
public function delete($id){
    $gambar = Gallery::where('id',$id)->first();
    File::delete('gallery/'.$gambar->image);
    File::delete('tumbnail/'.$gambar->image);
    Gallery::where('id',$id)->delete();

    return redirect('/admin/gallery')->with('success','Your Data is Deleted');
}
```

Gambar 5.73 Potongan kode untuk hapus gallery.

5.2.42. Implementasi Halaman *Dashboard* Bantuan

Halaman *Dashboard* Bantuan digunakan untuk menampilkan seluruh data bantuan yang dimasukkan oleh pengguna. Seperti pada gambar 5.74, halaman ini terdapat data tabel yang memiliki atribut *ID, Email*, Pertanyaan, Jawaban, dan Respons. Halaman ini memiliki tombol *Show* untuk menampilkan data bantuan yang belum direspons. Kolom Respons terdapat tombol Respons yang digunakan untuk mengarahkan pengguna ke halaman respons bantuan. Ketika sudah direspons tombol tersebut akan berubah menjadi *Confirmed* dan tidak bisa diklik. Kolom *Delete* berisikan tombol *Delete* yang digunakan untuk menghapus data bantuan yang berada di baris yang sama dengan tombol *Delete*.



Gambar 5.74 Implementasi Halaman Dashboard Bantuan.

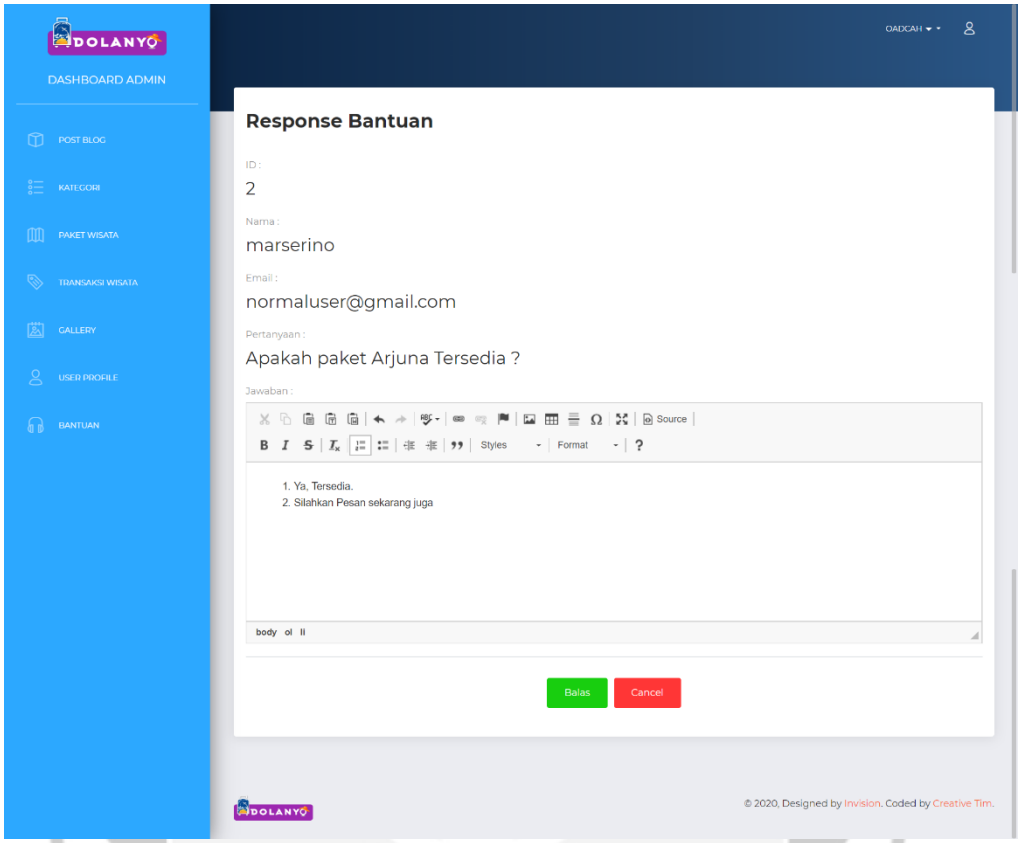
Untuk menampilkan data bantuan ke halaman admin dapat dilihat di gambar 5.75. Fungsi show akan memanggil model Bantuan dan menyimpan setiap obyek bantuan yang dipanggil ke dalam variabel $bantuan yang kemudian akan ditampilkan ke *view* admin.bantuan.

```php
public function show(){
    $bantuan = Bantuan::orderBy('created_at', 'DESC')->paginate(5);

    return view('admin.bantuan')->with('posts',$bantuan);
}
```

Gambar 5.75 Potongan kode untuk tampil bantuan.

5.2.43. Implementasi Halaman *Dashboard* Respons Bantuan

Halaman *Dashboard* Respons Bantuan digunakan untuk memberikan balasan bantuan kepada pengguna. Seperti pada gambar 5.75, halaman ini terdapat *form* yang di dalamnya terdapat masukan Jawaban yang digunakan untuk mengisikan jawaban atas pertanyaan yang ditanyakan oleh pengguna. Halaman ini terdapat dua tombol yaitu tombol Balas yang digunakan untuk mengkonfirmasi *form* yang diisikan dan mengirimkan jawaban kepada pengguna dalam bentuk *Email*, sedangkan tombol *Cancel* digunakan untuk membatalkan aksi dan kembali ke halaman Dashboard Bantuan.



Gambar 5.76 Implementasi Halaman Dashboard Respons Bantuan.

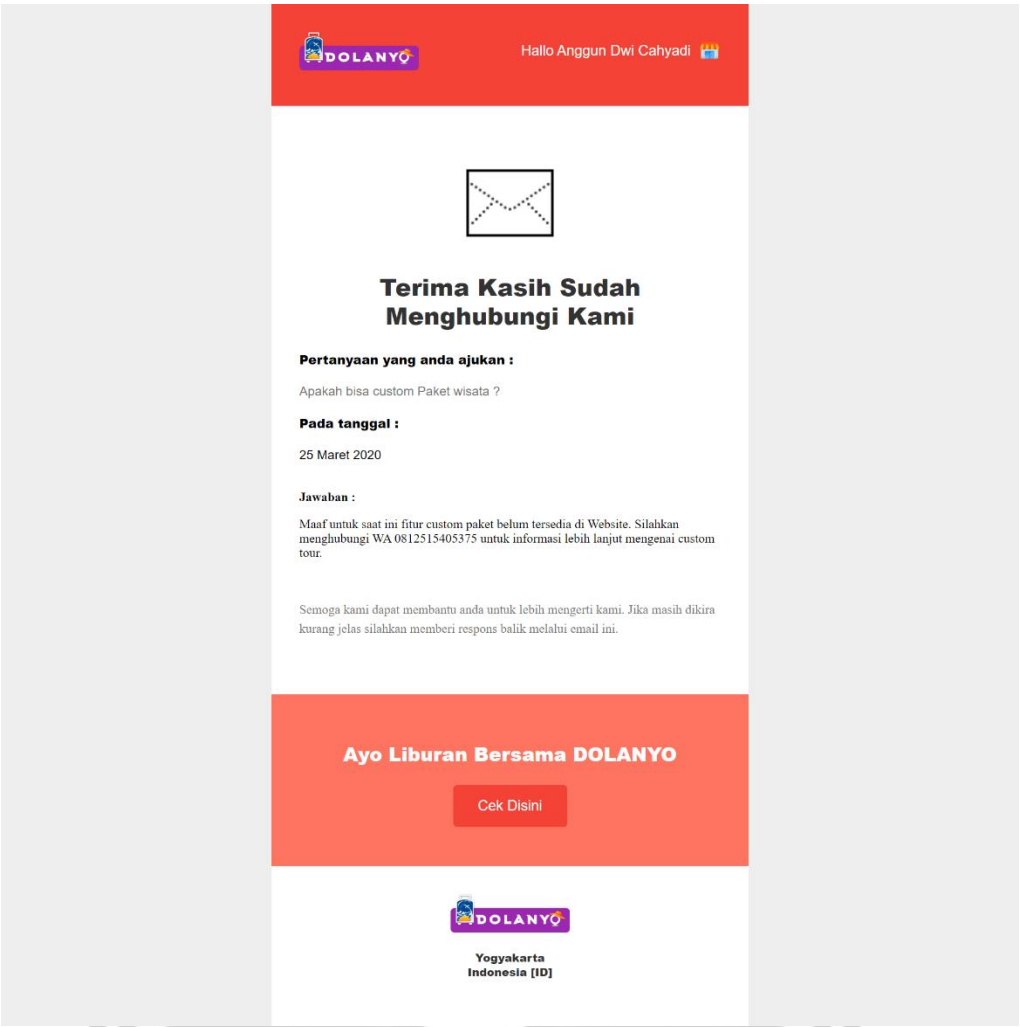
Ketika tombol Balas di gambar 5.76 ditekan maka sistem akan memanggil fungsi *send* seperti pada potongan kode di gambar 5.77. Fungsi ini memiliki parameter variabel $request yang berisikan data bantuan yang akan dikirim kepada pelanggan. Fungsi ini menyimpan data bantuan yang ada di $request ke dalam variabel *array* $data. Setelah itu fungsi akan mengubah status bantuan menjadi 1 yang berarti sudah direspons. Setelah itu sistem akan mengirimkan email balasan kepada pengguna dengan menggunakan bantuan *helper Mail* yang disediakan di *framework Laravel.*

```
public function send(Request $request){
    $data = [
        'id' => $request->id,
        'nama' =>$request->nama,
        'email' =>$request->email,
        'pertanyaan' =>$request->pertanyaan,
        'jawaban' =>$request->jawaban,
        'waktu' =>$request->waktu,
      ];
      $bantuan=Bantuan::find($request->id);
      $bantuan->respons = 1;
      $bantuan->jawaban = $request->jawaban;
      $bantuan->update();
      Mail::to($request->email)->send(new BantuanEmail($data));
      return redirect('/admin/bantuan')->with('success','Bantuan sudah diresponse, Email bantuan terkirim ke pelanggan');
}
```

Gambar 5.77 Potongan kode respons bantuan.

5.2.44. Implementasi Halaman *Email Bantuan*

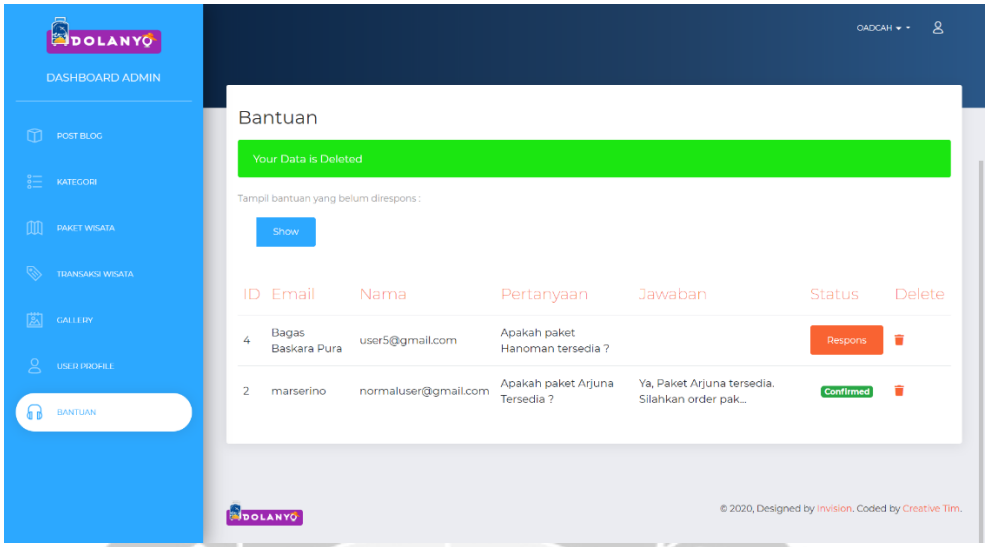
Halaman *Email* Respons Bantuan adalah halaman *Email* yang ditampilkan kepada pengguna ketika bantuan mereka sudah direspons oleh Admin. Seperti pada gambar 5.78, halaman ini terdapat data nama pengguna, pertanyaan yang diajukan, waktu pengajuan bantuan, serta jawaban. Halaman ini juga terdapat tombol Cek Di sini untuk mengarahkan pengguna kembali ke halaman Bantuan.



Gambar 5.78 Implementasi Halaman Email Bantuan.

5.2.45. Implementasi Halaman *Dashboard Delete* Bantuan

Halaman *Dashboard Delete* Bantuan seperti pada gambar 5.79 adalah halaman yang digunakan untuk menghapus data Bantuan yang dipilih dari basis data. Tombol *Delete* digunakan untuk mengkonfirmasi bahwa data di dalam baris tombol tersebut akan di hapus. Setelah di hapus makan akan muncul pemberitahuan bahwa data berhasil di hapus.



Gambar 5.79 Implementasi Halaman Dashboard Delete Bantuan.

Untuk melakukan hapus data bantuan dapat dilihat pada potongan kode di gambar 5.80. fungsi *destroy* memiliki parameter id bantuan yang digunakan untuk menghapus data berdasarkan parameter id bantuan yang dipilih. Fungsi ini akan memanggil bantuan berdasarkan id bantuan dan melakukan metode *delete* untuk menghapus data.

```
public function destroy($id)
{
    $bantuan = Bantuan::findOrfail($id);
    $bantuan ->delete();

    return redirect('/admin/bantuan')->with('success','Your Data is Deleted');
}
```

Gambar 5.80 Potongan kode untuk hapus bantuan.

5.2.46. Implementasi Sistem Rekomendasi

Sistem rekomendasi digunakan untuk menampilkan rekomendasi paket wisata kepada pengguna yang sudah terdaftar dan sudah memberikan penilaian *rating* untuk setiap jenis wisata. Setelah pengguna selesai mendaftar, pengguna akan diarahkan ke halaman *input rating* rekomendasi. Seperti pada gambar 5.7, pengguna memasukkan *rating* untuk setiap jenis wisata dengan menggunakan *Slider*. Setelah selesai memasukkan *rating* dari seluruh jenis wisata yaitu pegunungan, bangunan, sungai, dan pantai pengguna akan mendapatkan hasil rekomendasi seperti pada gambar 5.8.

Sistem rekomendasi yang digunakan oleh Sistem Informasi Dolanyo menggunakan metode *Content-Based Filtering*. Sistem rekomendasi ini merekomendasikan sebuah *item* berdasarkan perbandingan korelasi *content item* dengan preferensi *content* pengguna. Dalam kasus penelitian ini, *item* di sini adalah paket wisata, *content item* adalah *rating* jenis wisata yang dimiliki paket wisata. Preferensi pengguna adalah *rating* jenis wisata yang diberikan diisikan pengguna di *input* rekomendasi. Jadi, jika pengguna memiliki preferensi A, maka sistem akan mencari paket wisata yang memiliki preferensi yang mirip dengan preferensi pengguna yaitu preferensi A. Parameter yang dipilih untuk mempresentasikan preferensi pengguna yaitu jenis wisata pegunungan, bangunan, sungai, dan pantai. Setiap parameter memiliki nilai 0 sampai dengan 4. Nilai 0 berarti paket wisata tersebut tidak terdapat parameter, sedangkan nilai 4 diberikan ketika suatu paket memiliki parameter yang dominan. Dengan demikian, setiap paket wisata direpresentasikan sebagai vektor yang memiliki 4 elemen, di mana setiap elemen merepresentasikan nilai untuk sebuah parameter.

Untuk mencari hasil rekomendasi menggunakan metode *Content-Based Filtering*, sebelumnya dihitung nilai kemiripan atau *Similarity* antara preferensi pengguna dengan preferensi parameter yang dimiliki paket wisata. Untuk menghitung nilai kemiripan dapat menggunakan Algoritma *Nearest Neighbor*. Menurut Algoritma *Nearest Neighbor* nilai kemiripan dapat dicari dengan persamaan sebagai berikut:

$$Sim\ (profile, candidate) = \frac{1}{1 + Dis\ (Profile, candidate)}$$

Di mana:

Dis(Profile, Candidate) =Jarak antara *Profile* dengan *Candidate*

Penerapan persamaan similarity di dalam kode sistem rekomendasi dapat dilihat pada potongan kode di gambar 5.81. Sistem akan memanggil fungsi euDistance($a,$b) untuk mencari nilai jarak atau *Distance*. Parameter $a adalah data preferensi nilai rating untuk setiap paket wisata, sedangkan $b

adalah preferensi nilai rating yang dimiliki oleh *user*. Setelah fungsi eucDistance, hasil akan disimpan di variabel $euclidean yang akan digunakan untuk mencari similarity dan disimpan di variabel *$*sim yang nantinya akan disimpan di basis data di tabel rekomendasi dengan memanggil fungsi *create*.

```
$a = [$paket->pegunungan, $paket->bangunan, $paket->sungai, $paket->pantai];
$b = [$user->pegunungan, $user->bangunan, $user->sungai, $user->pantai];

$euclidean =$this->eucDistance($a, $b);

$sim=1/(1+$euclidean);

$this->create($user->id,$paket->id,$paket->title,$sim);
```

Gambar 5.81 Potongan kode untuk menghitung similarity.

Jarak *(Distance)* disingkat *Dis* dapat dicari dengan menggunakan algoritma *Euclidean* dengan menggunakan persamaan sebagai berikut:

$$Dis(profile, candidate) = \sqrt{\sum_{i=1}^{n}(profile_i - candidate_i)^2}$$

Di mana nilai :

- n = Jumlah elemen dalam vektor paket wisata.
- $Profile_i$ = Nilai parameter ke -*i* dari vektor profil pengguna.
- $Candidate_i$ = Nilai parameter ke -*i* dari vektor paket wisata yang dihitung jaraknya.

Penerapan persamaan di atas dalam sistem rekomendasi di Sistem Informasi DOLANYO dapat dilihat pada potongan kode di gambar 5.82. fungsi euDistance memiliki dua parameter yang di dalam kasus ini $vector1 adalah *profile* $_i$, sedangkan $vector2 adalah *Candidate* $_i$. Variabel $n digunakan untuk menyimpan jumlah data yang akan dibandingkan. Setelah itu menggunakan perulangan *For* untuk memperoleh jumlah hasil dari seluruh data yang dihitung yang disimpan di variabel $sum. Setelah itu fungsi ini akan mengembalikan variabel $sum yang sudah diakar dengan menggunakan fungsi sqrt($sum).

```php
public function eucDistance( $vector1,  $vector2) {
    $n = count($vector1);
    $sum = 0;
    for ($i = 0; $i < $n; $i++) {
        $sum += ($vector1[$i] - $vector2[$i]) * ($vector1[$i] - $vector2[$i]);
    }
    return sqrt($sum);
}
```

Gambar 5.82 Potongan kode untuk mencari jarak atau Distance.

Pada tabel 5.1 merupakan tabel yang berisikan 10 paket wisata dengan nilai parameter pegunungan, bangunan, sungai, dan pantai yang berbeda-beda. Sebagai contoh, pengguna A memberikan *rating* ke preferensi pengguna dengan vektor (4, 1, 3, 0) dan data vektor paket wisata yang ada di dalam sistem diberikan di Tabel 5.1.

Tabel 5.1 Nilai Parameter Paket Wisata.

No	Nama Paket	Pegunungan	Bangunan	Sungai	Pantai
1.	Paket Arjuna	4	1	3	0
2.	Paket Anggoda	1	3	4	0
3.	Paket Bharata	1	0	3	4
4.	Paket Hanoman	2	1	0	3
5.	Paket Jatayu	0	4	1	2
6.	Paket Kencana	3	4	0	0
7.	Paket Wibisana	4	2	1	1
8.	Paket Prahasta	4	3	2	0
9.	Paket Anila	4	4	0	0
10.	Paket Kosalya	2	3	2	0

Dari contoh kasus di atas, sistem akan melakukan perhitungan kemiripan atau (*Similarity)* antara setiap paket wisata dengan preferensi pengguna A. Untuk menghitung kemiripan maka perlu dihitung nilai jarak

(*Distance*) terlebih dahulu. Contoh perhitungan nilai jarak antara pengguna A dengan Paket Wibisana diberikan sebagai berikut:

$$Dis\ (profil, paket) = \sqrt{(4-4)^2 + (1-2)^2 + (3-1)^2 + (0-1)^2}$$
$$= \sqrt{0+1+4+1}$$
$$= \sqrt{6} = 2{,}4495$$

Setelah itu dihitung *Similarity* untuk Paket Wibisana:

$$Sim(profil, paket\ arjuna) = \frac{1}{1+2{,}4495} = 0{,}289898$$

Dengan cara yang sama, makan akan diperoleh nilai kemiripan atau *Similarity* dari pengguna A dengan vektor (4, 2, 1, 0) untuk setiap paket wisata yang ada di sistem dapat dilihat di Tabel 5.2.

Tabel 5.2 Hasil Perhitungan Distance dan Similarity.

No	Nama Paket	*Distance*	*Similarity*
1.	Paket Arjuna	0	1
2.	Paket Anggoda	3,7417	0,210897
3.	Paket Bharata	5,099	0,163961
4.	Paket Hanoman	4,6904	0,175734
5.	Paket Jatayu	5,7446	0,148268
6.	Paket Kencana	4,3589	0,186605
7.	Paket Wibisana	2,4495	0,289898
8.	Paket Prahasta	2,2361	0,309017
9.	Paket Anila	4,2426	0,190744
10.	Paket Kosalya	3	0,25

Sistem rekomendasi di Sistem Informasi Dolanyo akan menampilkan tiga rekomendasi teratas berdasarkan nilai *Similarity* dari pengguna A dan

Paket Wisata. Berdasarkan perhitungan pada Tabel 5.2, dapat disimpulkan bahwa paket wisata yang direkomendasikan kepada pengguna A adalah paket Arjuna dengan nilai *Similarity* 1 sebagai rekomendasi pertama, paket Prahasta dengan nilai *Similarity* 0,30902 sebagai rekomendasi kedua, dan paket Wibisana dengan nilai *Similarity* 0,2899 sebagai rekomendasi ketiga.

Untuk menampilkan hasil rekomendasi di sistem, dapat dilihat pada potongan kode di gambar 5.83. Untuk menampilkan hasil rekomendasi sistem akan memanggil fungsi *show*. Fungsi *show* memiliki parameter id user. Fungsi ini akan mengambil data rekomendasi dari basis data dan disimpan di variabel rekomendasi berdasarkan id *user* yang diurutkan dari nilai sim terbesar ke yang terkecil dan diambil tiga data teratas menggunakan fungsi *take*(3). Setelah itu fungsi akan memanggil seluruh data paket yang ada di basis data dan disimpan di variabel $paket. Fungsi akan mengirimkan data $rekomendasi dan $paket ke halaman rekomendasi.

```
public function show($id)
{
    $rekomendasi= DB::table('hasil_rekomendasi')
    ->select('paket','sim')
    ->where('user_id',$id)
    ->orderBy('sim', 'desc')
    ->take(3)
    ->get();
    $paket = Paket :: all();

    return view('rekomendasi')->with('rekomendasi',$rekomendasi)->with('paket',$paket);
}
```

Gambar 5.83 Potongan kode untuk tampil hasil rekomendasi.

5.3. Pengujian Perangkat Lunak

Pengujian perangkat lunak dilakukan pada Sistem Informasi Dolanyo terhadap semua fungsi yang terdapat pada Sistem Informasi Dolanyo. Pengujian perangkat lunak dilakukan setelah selesai mengimplementasikan semua rancangan antarmuka Sistem Informasi Dolanyo.

Pengujian perangkat lunak bertujuan untuk mengetahui apakah implementasi kode dapat dijalankan dengan baik dan benar sesuai dengan fungsinya. Selain itu, pengujian perangkat lunak digunakan untuk menemukan kekurangan ataupun

kesalahan yang harus diperbaiki. Jika dalam pengujian perangkat lunak ini masih ditemukan kekurangan dan masalah, maka akan dilakukan perbaikan terdapat fungsi kode. Perbaikan dilakukan untuk memperoleh hasil akhir yang sesuai dengan yang diharapkan. Hasil pengujian perangkat lunak dapat dilihat di Tabel 5.3.

Tabel 5.3 Hasil Pengujian Fungsionalitas.

Identifikasi	Deskripsi	Prosedur Pengujian	Masukan	Keluaran yang Diharapkan	Kriteria Evaluasi Hasil	Hasil yang Didapat	Kesimpulan
DOLANYO-01	Pengujian mengisikan *Form* Bantuan	Masuk ke halaman *Landing Page* dan *Contact Us,* masukkan *Your Name, Your Email, Your Message.* Lalu klik tombol *Send Message.*	Masukkan nama, *email,* dan pesan.	Tampil halaman *Home* dan *Dashboard* Bantuan menerima data baru.	Menampilkan halaman *Home*	Tampil halaman *Home* dan *Dashboard* Bantuan menerima data baru.	Handal
DOLANYO-02	Pengujian transaksi Jasa Paket Wisata	Masuk sebagai user, masuk halaman detail paket wisata, klik tombol	Masukan Nama Lengkap, *No Handphone,* Jumlah	Tampil halaman *Home,* muncul notifikasi	Menampilkan halaman *Home* dan muncul pemberitahuan.	Tampil halaman *Home*, muncul notifikasi transaksi berhasil, data	Handal

		Pesan Sekarang Juga! Masukkan data di *Form.* Klik tombol Pesan	Peserta, Tanggal Rencana, Tempat Penjemputan.	transaksi berhasil, data transaksi masuk ke dalam basis data.		transaksi masuk ke dalam basis data.	
DOLANYO-03	Pengujian *Login*	Masuk ke halaman *Login,* masukkan data *form login.* Klik tombol *Login.*	Masukan *Email* dan *Password.*	Tampil halaman *Home* untuk *role user* dan halaman *Dashboard* untuk *role* Admin.	Tampil halaman *Home* untuk *role user* dan halaman *Dashboard* untuk *role* Admin.	Tampil halaman *Home* untuk *role user* dan halaman *Dashboard* untuk *role* Admin.	Handal
DOLANYO-03-1	Pengujian *Logout*	Masuk sebagai pengguna, klik menu *dropdown user,* klik menu *Logout.*	Tidak ada	Tampil halaman *landing page* dan pengguna berhasil	Tampil halaman *landing page* dan pengguna berhasil keluar	Tampil halaman *landing page* dan pengguna berhasil keluar dari sistem.	Handal

				keluar dari sistem.	dari sistem.		
DOLANYO-04	Pengujian reset *Password*	Masuk halaman *Forgot Your Password,* masukkan *Email user* yang akan dihapus. Klok tombol *Send Password Reset Link. User* menerima *Email Reset Password.* Klik tombol *Reset Your Password* di *Email.* Masukkan *email* dan *Password*	Masukan *Email,* masukan *Password* baru, *Confirm Password.*	*User* menerima *Email Reset Password,* tampil halaman *Reset Password. Password* baru disimpan di basis data	*User* menerima *Email Reset Password,* tampil halaman *Reset Password. Password* baru disimpan di basis data	*User* menerima *Email Reset Password,* tampil halaman *Reset Password. Password* baru disimpan di basis data	Handal

		baru. Klik tombol Reset *Password.*					
DOLANYO-05	Pengujian Register	Masuk ke halaman Register, masukkan data di *form* pendaftaran, klik tombol Register.	Masukan *Username, Phone Number, Email Address, Password, Confirm Password.*	Tampil halaman *Home* dan data *user* berhasil dibuat dan disimpan di basis data.	Tampil halaman *Home* dan data *user* berhasil dibuat dan disimpan di basis data.	Tampil halaman *Home* dan data *user* berhasil dibuat dan disimpan di basis data.	Handal
DOLANYO-06	Pengujian melihat informasi pariwisata.	Masuk ke halaman Blog. Klik tombol *Read More*		Berhasil menampilkan halaman Blog dan detail Blog.	Berhasil menampilkan halaman Blog dan detail Blog.	Berhasil menampilkan halaman Blog dan detail Blog.	Handal
DOLANYO-07	Pengujian mengisikan *Form*	Selesai melakukan registrasi,	Masukan nilai parameter dengan *Slider*.	Berhasil tampil halaman	Berhasil tampil halaman *show* Rekomendasi	Berhasil tampil halaman *show* Rekomendasi	Handal

	rekomendasi	dialihkan ke halaman rekomendasi. Masukkan input nilai di setiap rekomendasi menggunakan *Slider*. Klik tombol *Next* sampai semua parameter terisi.		*show* Rekomendasi dan menampilkan tiga rekomendasi teratas. Data rekomendasi berhasil disimpan di basis data.	dan menampilkan tiga rekomendasi teratas. Data rekomendasi berhasil disimpan di basis data.	dan menampilkan tiga rekomendasi teratas. Data rekomendasi berhasil disimpan di basis data.	
DOLANYO-08	Pengujian melihat rekomendasi paket wisata.	Masuk sebagai *user*, klik *Dropdown* menu *username* di menu navigasi, klik Rekomendasi.		Berhasil tampil halaman *show* Rekomendasi dan menampilkan tiga	Berhasil tampil halaman *show* Rekomendasi dan menampilkan tiga rekomendasi teratas.	Berhasil tampil halaman *show* Rekomendasi dan menampilkan tiga rekomendasi teratas.	Handal

				rekomendasi teratas.			
DOLANYO-09	Pengujian melihat data transaksi.	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Transaksi Wisata. Masukkan di *input search* dan pilih Cari data berdasarkan ID atau Nama. Klik Cari	Masukan Cari, *dropdown* Cari data berdasarkan ID atau Nama.	Berhasil menampilkan data Transaksi Wisata.	Berhasil menampilkan data Transaksi Wisata.	Berhasil menampilkan data Transaksi Wisata.	Handal
DOLANYO-10	Pengujian menghapus data transaksi	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Transaksi		Tampil notifikasi data berhasil dihapus, dan data	Tampil notifikasi data berhasil dihapus, dan data Transaksi	Tampil notifikasi data berhasil dihapus, dan data Transaksi	Handal

		Wisata. Klik tombol *Delete* di baris data tabel yang akan dihapus.		Transaksi berhasil dihapus dari basis data	berhasil dihapus dari basis data	berhasil dihapus dari basis data	
DOLANYO-11	Pengujian konfirmasi penjadwalan transaksi jasa wisata.	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik tombol Konfirmasi.		Tampil halaman Konfirmasi yang berisikan data transaksi yang akan dikonfirmasi.	Tampil halaman Konfirmasi yang berisikan data transaksi yang akan dikonfirmasi.	Tampil halaman Konfirmasi yang berisikan data transaksi yang akan dikonfirmasi.	Handal
DOLANYO-12	Pengujian kirim *Email* Konfirmasi Transaksi	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Masuk ke halaman	Masukan Pesan Email Pembatalan Transaksi.	Tampil halaman Transaksi. Pengguna menerima pesan *Email*	Tampil halaman Transaksi. Pengguna menerima pesan *Email*	Tampil halaman Transaksi. Pengguna menerima pesan *Email* penerimaan atau	

		Konfirmasi. Klik tombol Konfirmasi untuk menerima transaksi atau masukkan alasan pembatalan, klik tombol Batalkan.		penerimaan atau pembatalan transaksi. Data Transaksi berhasil di *Update* statusnya.	penerimaan atau pembatalan transaksi. Data Transaksi berhasil di *Update* statusnya.	pembatalan transaksi. Data Transaksi berhasil di *Update* statusnya.	
DOLANYO-13	Pengujian melihat artikel Pariwisata	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu *Post* Blog. Masukkan di *input search* dan pilih Cari data berdasarkan ID atau *Tittle*. Klik	Masukan Cari, *dropdown* Cari data berdasarkan ID atau *Tittle.*	Berhasil menampilkan data artikel Pariwisata.	Berhasil menampilkan data artikel Pariwisata.	Berhasil menampilkan data artikel Pariwisata.	Handal

		Cari					
DOLANYO-14	Pengujian menghapus artikel pariwisata.	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu *Post* Blog. Klik tombol *Delete* di baris data tabel yang akan dihapus.		Tampil notifikasi data berhasil dihapus, dan data artikel pariwisata berhasil dihapus dari basis data	Tampil notifikasi data berhasil dihapus, dan data artikel pariwisata berhasil dihapus dari basis data	Tampil notifikasi data berhasil dihapus, dan data artikel pariwisata berhasil dihapus dari basis data	Handal
DOLANYO-15	Pengujian memasukkan artikel pariwisata.	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu *Post* Blog. Klik tombol *Create New Post*. Masukkan data *Form*	Masukan *Tittle, Body, Image*.	Tampil halaman *Dashboard Post Blog*, muncul notifikasi data berhasil ditambahkan dan data	Tampil halaman *Dashboard Post Blog*, muncul notifikasi data berhasil ditambahkan dan data artikel	Tampil halaman *Dashboard Post Blog*, muncul notifikasi data berhasil ditambahkan dan data artikel pariwisata berhasil	Handal

		artikel pariwisata. Klik tombol *Save.*		artikel pariwisata berhasil disimpan di basis data.	pariwisata berhasil disimpan di basis data.	disimpan di basis data.	
DOLANYO-16	Pengujian mengedit artikel pariwisata.	Masuk sebagai Admin, masuk ke halaman *Dashboard.* Klik menu *Post* Blog. Klik tombol Edit. Masukkan data *Form* artikel pariwisata yang akan di edit. Klik tombol *Update.*	Masukan *Tittle* dan *Body*.	Tampil halaman *Dashboard Post Blog,* muncul notifikasi data berhasil diubah dan data artikel pariwisata yang diubah berhasil disimpan di basis data.	Tampil halaman *Dashboard Post Blog,* muncul notifikasi data berhasil diubah dan data artikel pariwisata yang diubah berhasil disimpan di basis data.	Tampil halaman *Dashboard Post Blog,* muncul notifikasi data berhasil diubah dan data artikel pariwisata yang diubah berhasil disimpan di basis data.	Handal

DOLANYO-17	Pengujian melihat kategori paket wisata	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Kategori.		Berhasil Tampil data Kategori.	Berhasil Tampil data Kategori.	Berhasil Tampil data Kategori.	Handal
DOLANYO-18	Pengujian memasukkan kategori paket wisata	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Kategori. Klik tombol *Create New* Kategori, Masukkan data ke dalam *Form* Kategori. Klik tombol *Save.*	Masukan Nama Kategori, Minimal Peserta, Maksimal Peserta, dan *Image*.	Tampil halaman *Dashboard* Kategori, muncul notifikasi data berhasil ditambahkan dan data Kategori berhasil disimpan di basis data.	Tampil halaman *Dashboard* Kategori, muncul notifikasi data berhasil ditambahkan dan data Kategori berhasil disimpan di basis data.	Tampil halaman *Dashboard* Kategori, muncul notifikasi data berhasil ditambahkan dan data Kategori berhasil disimpan di basis data.	Handal
DOLANYO-19	Pengujian	Masuk sebagai	Masukan	Tampil	Tampil	Tampil halaman	

	mengedit kategori paket wisata	Admin, masuk ke halaman *Dashboard*. Klik menu Kategori. Klik tombol Edit. Masukkan data di *Form* edit kategori Wisata. Klik tombol *Update.*	Nama Kategori, Minimal Peserta, Maksimal Peserta, dan *Image*.	halaman *Dashboard* Kategori, muncul notifikasi data berhasil diubah dan data Kategori yang diubah berhasil disimpan di basis data.	halaman *Dashboard* Kategori, muncul notifikasi data berhasil diubah dan data Kategori yang diubah berhasil disimpan di basis data.	*Dashboard* Kategori, muncul notifikasi data berhasil diubah dan data Kategori yang diubah berhasil disimpan di basis data.	
DOLANYO-20	Pengujian menghapus kategori paket wisata	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Kategori. Pilih dan klik tombol *Delete*		Tampil halaman *Dashboard* Kategori, muncul notifikasi data berhasil	Tampil halaman *Dashboard* Kategori, muncul notifikasi data berhasil	Tampil halaman *Dashboard* Kategori, muncul notifikasi data berhasil dihapus dan data	Handal

		sesuai dengan baris data yang akan dihapus.		dihapus dan data Kategori yang dipilih berhasil dihapus dari basis data.	dihapus dan data Kategori yang dipilih berhasil dihapus dari basis data.	Kategori yang dipilih berhasil dihapus dari basis data.	
DOLANYO-21	Pengujian melihat paket wisata	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Paket Wisata. Masukkan di *input search* dan pilih Cari data berdasarkan ID atau *Tittle*. Klik tombol Cari	Masukan Cari, *dropdown* Cari data berdasarkan ID atau *Tittle.*	Berhasil Tampil data Paket Wisata.	Berhasil Tampil data Paket Wisata.	Berhasil Tampil data Paket Wisata.	Handal
DOLANYO-22	Pengujian	Masuk sebagai	Masukan	Tampil	Tampil	Tampil halaman	Handal

	mengedit paket wisata.	Admin, masuk ke halaman *Dashboard*. Klik menu Paket Wisata. Pilih dan klik tombol Edit sesuai dengan baris data tabel yang akan diedit. Tampil halaman Edit data Paket Wisata, isi data *Form* Edit, klik tombol *Update.*	Judul, Deskripsi, *Overview*, Fasilitas, Ketentuan, Harga Mulai, *Rating* Pegunungan, *Rating* Bangunan, *Rating* Sungai, *Rating* Pantai, Kategori, *Image*.	halaman *Dashboard* Paket Wisata, muncul notifikasi data berhasil diubah dan data Paket Wisata yang diubah berhasil disimpan di basis data.	halaman *Dashboard* Paket Wisata, muncul notifikasi data berhasil diubah dan data Paket Wisata yang diubah berhasil disimpan di basis data.	*Dashboard* Paket Wisata, muncul notifikasi data berhasil diubah dan data Paket Wisata yang diubah berhasil disimpan di basis data.	
DOLANYO-23	Pengujian menghapus paket wisata.	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Paket		Tampil halaman *Dashboard* Paket Wisata, muncul	Tampil halaman *Dashboard* Paket Wisata, muncul	Tampil halaman *Dashboard* Paket Wisata, muncul notifikasi data	

		Wisata. Pilih dan klik tombol *Delete* sesuai dengan baris data tabel yang akan dihapus.		notifikasi data berhasil dihapus dan data Paket Wisata berhasil dihapus dari basis data.	notifikasi data berhasil dihapus dan data Paket Wisata berhasil dihapus dari basis data.	berhasil dihapus dan data Paket Wisata berhasil dihapus dari basis data.	
DOLANYO-24	Pengujian memasukkan paket wisata.	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Paket Wisata. Pilih dan klik tombol *Create New* Paket Wisata. Tampil halaman *Create Paket*	Masukan Judul, Deskripsi, *Overview*, Fasilitas, Ketentuan, Harga Mulai, *Rating* Pegunungan, *Rating* Bangunan,	Tampil halaman *Dashboard* Paket Wisata, muncul notifikasi data berhasil dibuat dan data Paket Wisata yang baru berhasil	Tampil halaman *Dashboard* Paket Wisata, muncul notifikasi data berhasil dibuat dan data Paket Wisata yang baru berhasil disimpan di	Tampil halaman *Dashboard* Paket Wisata, muncul notifikasi data berhasil dibuat dan data Paket Wisata yang baru berhasil disimpan di basis data.	Handal

		Wisata, isi data *Form* Paket Wisata, klik tombol *Save.*	*Rating* Sungai, *Rating* Pantai, Kategori, *Image*.	disimpan di basis data.	basis data.		
DOLANYO-25	Pengujian melihat Profil *User*	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu *User Profile*. Masukkan di *input search* dan pilih Cari data berdasarkan Nama atau *Email*. Klik tombol Cari	Masukan Cari, *dropdown* Cari data berdasarkan Nama atau *Email.*	Berhasil Tampil data *User*.	Berhasil Tampil data *User*.	Berhasil Tampil data *User*.	Handal
DOLANYO-26	Pengujian mengedit	Masuk sebagai Admin, masuk	Masukan Nama dan Give	Tampil halaman	Tampil halaman	Tampil halaman *Dashboard User*	Handal

	Profil *User*	ke halaman *Dashboard*. Klik menu *User Profile*. Pilih dan klik tombol Edit sesuai dengan baris data tabel yang akan diedit. Tampil halaman Edit data *User Profile*, isi data *Form* Edit, klik tombol *Update.*	Role menggunakan *Dropdown item list.*	*Dashboard User Profile,* muncul notifikasi data berhasil diubah dan data *User Profile* yang diubah berhasil disimpan di basis data.	*Dashboard User Profile,* muncul notifikasi data berhasil diubah dan data *User Profile* yang diubah berhasil disimpan di basis data.	*Profile,* muncul notifikasi data berhasil diubah dan data *User Profile* yang diubah berhasil disimpan di basis data.	
DOLANYO-27	Pengujian menghapus Profil *User*	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu *User Profile*. Pilih dan		Tampil halaman *Dashboard User Profile,* muncul notifikasi	Tampil halaman *Dashboard User Profile,* muncul notifikasi data	Tampil halaman *Dashboard User Profile,* muncul notifikasi data berhasil dihapus dan data *User*	Handal

		klik tombol *Delete* sesuai dengan baris data tabel yang akan dihapus.		data berhasil dihapus dan data *User Profile* berhasil dihapus dari basis data.	berhasil dihapus dan data *User Profile* berhasil dihapus dari basis data.	*Profile* berhasil dihapus dari basis data.	
DOLANYO-28	Pengujian melihat foto galeri	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu *Gallery*.		Berhasil Tampil data *Gallery*.	Berhasil Tampil data *Gallery*.	Berhasil Tampil data *Gallery*.	Handal
DOLANYO-29	Pengujian memasukkan foto galeri	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu *Gallery*. Pilih dan klik tombol *Create*	Masukan Nama dan *Image*.	Tampil halaman *Dashboard Gallery*, muncul notifikasi data berhasil	Tampil halaman *Dashboard Gallery*, muncul notifikasi data berhasil dibuat	Tampil halaman *Dashboard Gallery*, muncul notifikasi data berhasil dibuat dan data *Gallery* yang baru	Handal

		New Gallery. Tampil halaman *Create New Photo Gallery*, isi data *Form Gallery*, klik tombol *Save*.		dibuat dan data *Gallery* yang baru berhasil disimpan di basis data.	dan data *Gallery* yang baru berhasil disimpan di basis data.	berhasil disimpan di basis data.	
DOLANYO-30	Pengujian menghapus foto galeri	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu *Gallery*. Pilih dan klik tombol *Delete* sesuai dengan baris data tabel yang akan dihapus.		Tampil halaman *Dashboard Gallery*, muncul notifikasi data berhasil dihapus dan data *Gallery* berhasil dihapus dari basis data.	Tampil halaman *Dashboard Gallery*, muncul notifikasi data berhasil dihapus dan data *Gallery* berhasil dihapus dari basis data.	Tampil halaman *Dashboard Gallery*, muncul notifikasi data berhasil dihapus dan data *Gallery* berhasil dihapus dari basis data.	Handal

DOLANYO-31	Pengujian melihat Bantuan	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Bantuan. Klik tombol *Show* untuk menampilkan bantuan yang belum direspons.		Berhasil Tampil data Bantuan.	Berhasil Tampil data Bantuan.	Berhasil Tampil data Bantuan.	Handal
DOLANYO-32	Pengujian merespons Bantuan	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Bantuan. Klik tombol Respons,		Berhasil tampil halaman respons bantuan dan menampilkan data Bantuan yang akan direspons	Berhasil tampil halaman respons bantuan dan menampilkan data Bantuan yang akan direspons	Berhasil tampil halaman respons bantuan dan menampilkan data Bantuan yang akan direspons	Handal

DOLANYO-33	Pengujian mengirim *Email* Respons Bantuan	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Bantuan. Klik tombol Respons, masuk ke halaman Respons Bantuan, Masukkan *form* Jawaban, klik tombol Balas.	Masukan Jawaban.	*User* menerima *email* respons Bantuan dari Admin. Tampil halaman *Dashboard* Bantuan, muncul notifikasi *email* bantuan berhasil dikirim ke *User*. Status data Bantuan berhasil diubah dan disimpan di	*User* menerima *email* respons Bantuan dari Admin. Tampil halaman *Dashboard* Bantuan, muncul notifikasi *email* bantuan berhasil dikirim ke *User*. Status data Bantuan berhasil diubah dan disimpan di basis data.	*User* menerima *email* respons Bantuan dari Admin. Tampil halaman *Dashboard* Bantuan, muncul notifikasi *email* bantuan berhasil dikirim ke *User*. Status data Bantuan berhasil diubah dan disimpan di basis data.	Handal

				basis data.			
DOLANYO-34	Pengujian menghapus data Bantuan	Masuk sebagai Admin, masuk ke halaman *Dashboard*. Klik menu Bantuan. Pilih dan klik tombol *Delete* sesuai dengan baris data tabel yang akan dihapus.		Tampil halaman *Dashboard* Bantuan, muncul notifikasi data berhasil dihapus dan data Bantuan berhasil dihapus dari basis data.	Tampil halaman *Dashboard* Bantuan, muncul notifikasi data berhasil dihapus dan data Bantuan berhasil dihapus dari basis data.	Tampil halaman *Dashboard* Bantuan, muncul notifikasi data berhasil dihapus dan data Bantuan berhasil dihapus dari basis data.	Handal

5.4. Hasil Pengujian Terhadap Pengguna

Pengujian Sistem Informasi DOLANYO kepada pengguna menggunakan kuesioner yang dibagikan menjadi dua bagian penelitian, yaitu fungsional Sistem Informasi DOLANYO dan antarmuka aplikasi. Penelitian dilakukan dengan metode pilihan ganda dengan pilihan:

1. Sangat Setuju (SS)
2. Setuju (S)
3. Kurang Setuju (KS)
4. Sangat Tidak Setuju (STS)

Tabel 5.4 Tabel Hasil Pengujian Terhadap Pengguna.

No	Pertanyaan	Jawaban			
		SS	S	KS	STS
1.	Tampilan keseluruhan dari Sistem informasi DOLANYO nyaman dilihat	6	16	8	0
2.	Sistem Informasi DOLANYO mudah digunakan dan mudah dipahami	6	21	3	0
3.	Isi informasi yang diberikan Sistem Informasi DOLANYO tepat dan sesuai antara isi dan judul.	6	23	1	0
4.	Anda memahami fitur rekomendasi paket wisata dengan baik dan memberikan hasil yang sesuai dengan harapan.	4	19	7	0
5.	Anda memahami dengan baik cara memesan paket wisata.	7	21	2	0

Rekapitulasi daya yang diperoleh dari setiap pernyataan pada tabel 5.4 adalah sebagai berikut :

1. Pertanyaan 1

   Dari 30 responden, enam responden sangat setuju, enam belas responden setuju, dan delapan responden kurang setuju.

2. Pertanyaan 2

   Dari 30 responden, enam responden sangat setuju, dua puluh satu responden setuju, dan tiga responden kurang setuju.

3. Pertanyaan 3

   Dari 30 responden, enam responden sangat setuju, dua puluh tiga responden setuju, dan satu kurang setuju.

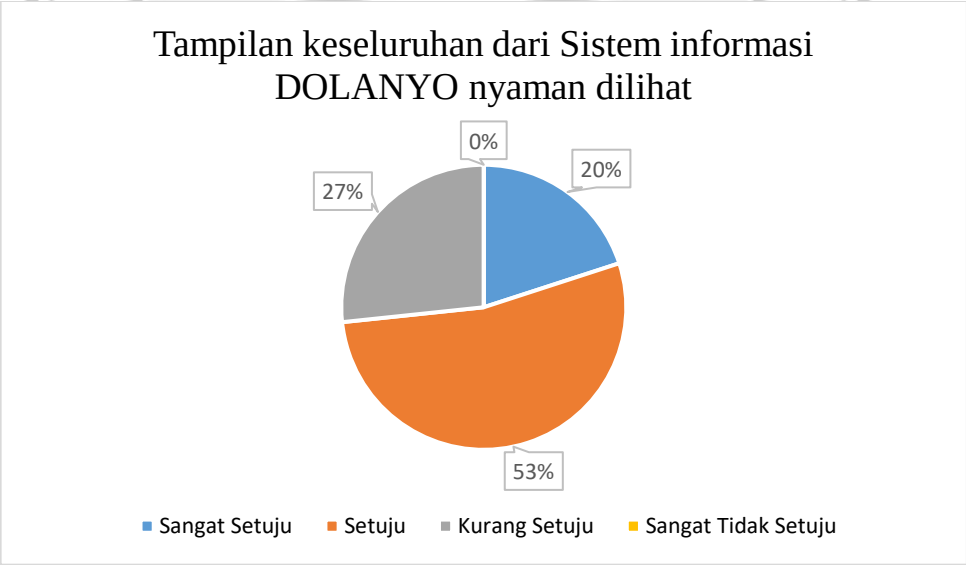
4. Pertanyaan 4

   Dari 30 responden, Empat responden sangat setuju, sembilan belas responden setuju, dan tujuh responden kurang setuju.

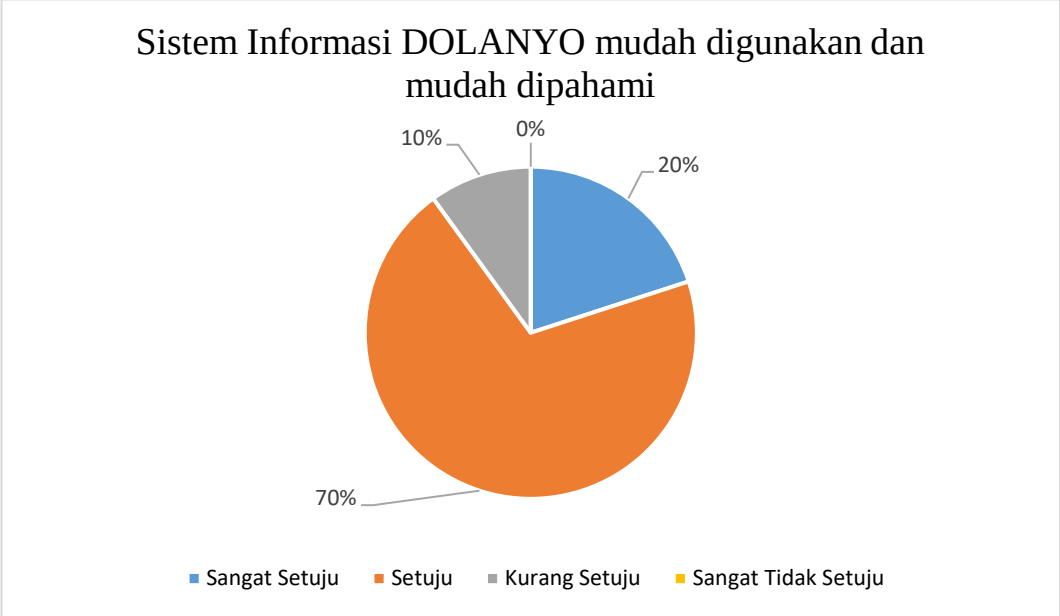
5. Pertanyaan 5

   Dari 30 responden, tujuh responden sangat setuju, dua puluh satu responden setuju, dan dua responden kurang setuju.

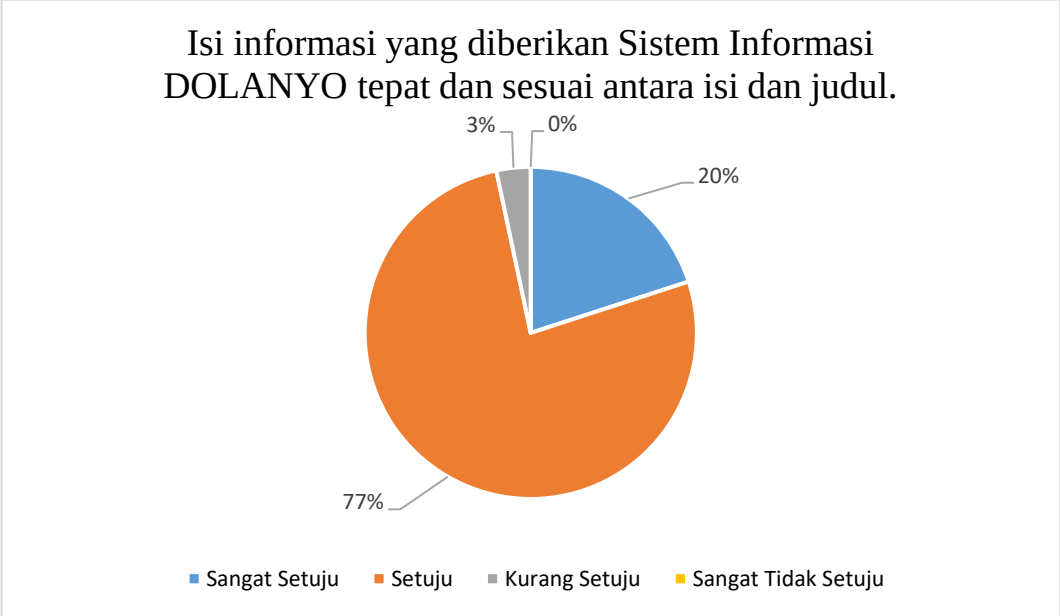
Dari rekapitulasi data kuesioner yang telah didapatkan, maka diperoleh persentase jawaban pada grafik model *pie* seperti yang terlihat pada gambar berikut ini :



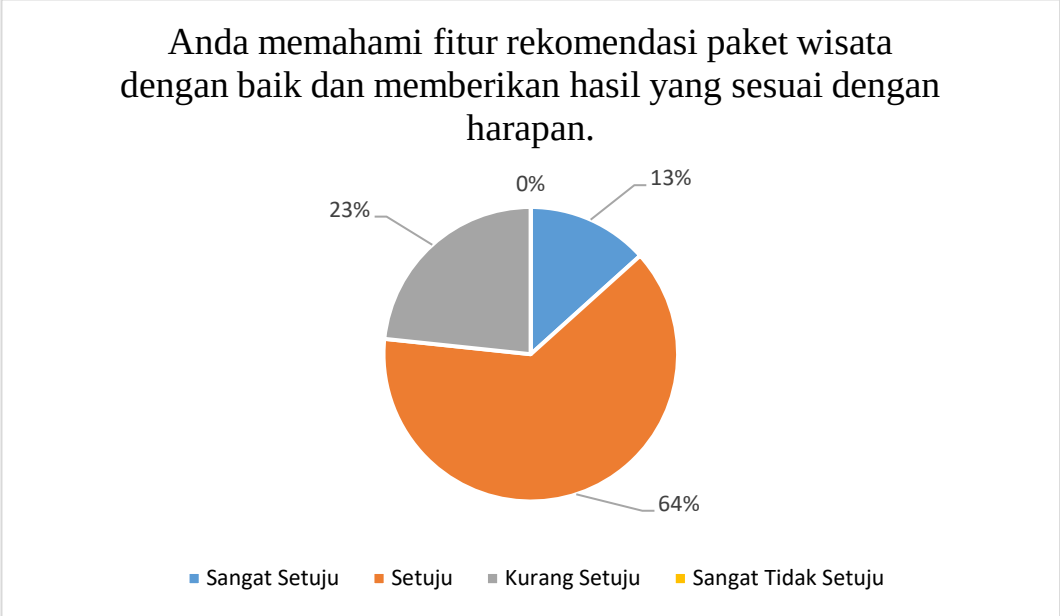
Gambar 5.84 Presentasi pertanyaan 1.



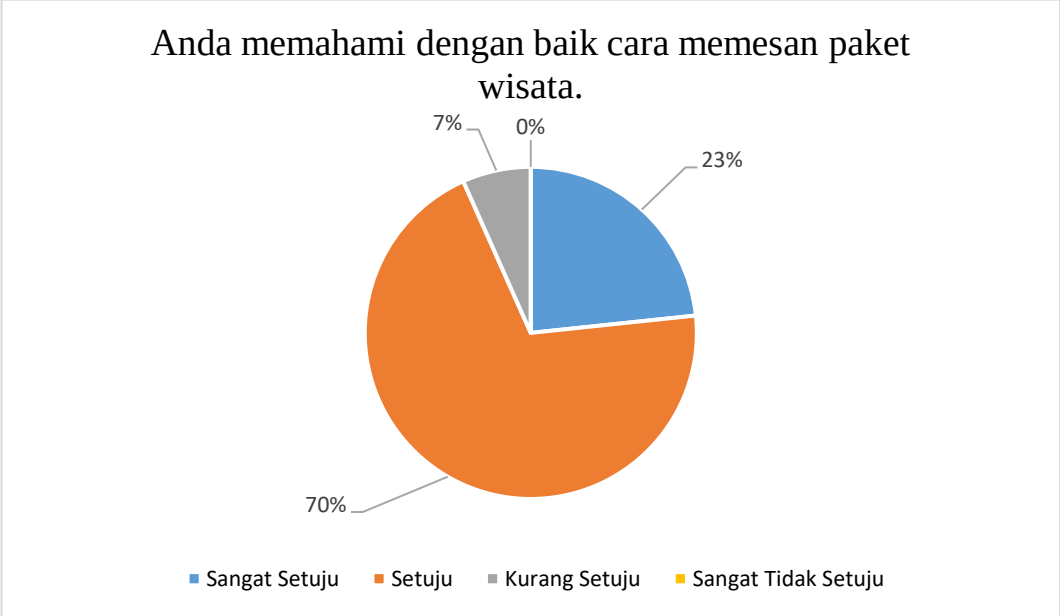
Gambar 5.85 Presentasi pertanyaan 2.



Gambar 5.86 Presentasi pertanyaan 3.



Gambar 5.87 Presentasi pertanyaan 4.



Gambar 5.88 Presentasi pertanyaan 5.

BAB VI
PENUTUP

Bagian ini berisi kesimpulan dari hasil penelitian yang telah dilakukan serta saran-saran yang terkait dengan bagaimana penelitian lebih lanjut.

6.1. Kesimpulan

Berdasarkan hasil perancangan, implementasi, dan hasil pengujian Sistem Informasi DOLANYO yang telah dibuat, maka dapat ditarik kesimpulan sebagai berikut:

1. Sistem Informasi DOLANYO dapat mempermudah para wisatawan dalam mendapatkan informasi pariwisata di Yogyakarta.
2. Wisatawan dapat menemukan paket wisata di Yogyakarta yang sesuai dengan keinginan dari wisatawan dengan adanya fitur rekomendasi paket wisata yang ada di Sistem Informasi DOLANYO.

6.2. Saran

Saran maupun masukan yang dapat penulis sampaikan terhadap pengembangan perangkat lunak Sistem Informasi DOLANYO adalah sebagai berikut :

1. Sistem Informasi DOLANYO diharapkan dapat melakukan pembayaran transaksi jasa pariwisata secara *Online* dari sistem.
2. Sistem Informasi DOLANYO diharapkan dapat dikembangkan lagi ke dalam *platform mobile.*

DAFTAR PUSTAKA

[1] APJII, "Penetrasi & Profil Perilaku Pengguna Internet Indonesia," *Apjii*, p. 51, 2018.

[2] B. Orenzi, "Statistik Pengguna Digital Dan Internet Indonesia 2019," *BOC Indonesia*, 2019. [Online]. Available: https://www.boc.web.id/statistik-pengguna-digital-dan-internet-indonesia-2019/. [Accessed: 08-Jan-2019].

[3] R. T. Watson, P. Berthon, L. F. Pitt, and G. M. Zinkhan, *Electronic Commerce: The Strategic Perspective*. A Global Text, 2008.

[4] L. Sebastia, I. Garcia, E. Onaindia, and C. Guzman, "E-Tourism: A tourist recommendation and planning application," *Int. J. Artif. Intell. Tools*, vol. 18, no. 5, pp. 717–738, 2009, doi: 10.1142/S0218213009000378.

[5] F. Ricci, L. Rokach, B. Shapira, P. B. Kantor, and F. Ricci, *Recommender Systems Handbook*. New York: Springer Science+Business Media, LLC, 2011.

[6] G. G. Maulana, "Sistem Informasi Pelayanan Jasa Tour dan Travel Berbasis Website Electronic Commerce (Studi Kasus Ninetours Indonesia) *," *Sist. Inf.*, vol. 03, no. 01, pp. 49–60, 2015.

[7] J. S. Santosa, "Pembangunan Sistem Rekomendasi Dengan Metode Collaborative Filtering Pada Website Online-Shop," Universitas Atma Jaya Yogyakarta, 2019.

[8] A. Arief, Widyawan, and B. Sunafri Hantono, "Rancang Bangun Sistem Rekomendasi Pariwisata Mobile dengan Menggunakan Metode Collaborative Filtering dan Location Based Filtering," *Jnteti*, vol. 1, no. 3, 2012.

[9] B. T. W. Utomo and A. W. Anggriawan, "Sistem Rekomendasi Paket Wisata Se-Malang Raya Menggunakan Metode Hybrid Content Based Dan Collaborative," *J. Ilm. Teknol. Inf. Asia*, vol. 9, no. 1, pp. 6–13, 2015.

[10] R. Indonesia, *UNDANG-UNDANG REPUBLIK INDONESIA NOMOR 10.TAHUN 2009 TENTANG KEPARIWISATAAN*, vol. 45, no. 1. jakarta,

Indonesia, 2009.

[11] B. Soeherman and M. Pinontoan, *Designing Information System*, Whindy Yoe. Jakarta: Elex Media Komputindo, 2008.

[12] A. Lena and K. Ratna, "Pengertian PHP dan MySQL," *Ilmu Teknol. Inf.*, p. 6, 2008.

[13] F. MUTHRIF, "Tutorial PHP Framework Laravel *Part* 1," *BINUS University*, 2017. [Online]. Available: https://socs.binus.ac.id/2017/09/15/laravel51/. [Accessed: 06-Mar-2020].

[14] H. Maharani and F. A. Gunawan, "Sistem Rekomendasi Mobil Berdasarkan Demographic dan Content-Based Filtering," *J. Telemat.*, vol. 9, no. 2, pp. 64–68, 2014.

[15] pbarrett.net, "Euclidean Distance," *Tech. Whitepaper Ser. 6*, p. 26, 2005.

[16] M. Christianti and C. Hadiguna, "Aplikasi E-Commerce dengan Sistem Rekomendasi Berbasis Collaborative Filtering pada Toko Komputer Ekaria Meliana," *Pros. Semin. Nas. Teknol. Inf. dan Apl. 2015*, vol. 7, pp. 157–175, 2011, doi: 10.1016/j.jcis.2013.03.011.

[17] M. Dabbs, "Service Packages Expertise Work About Blog Contact The Fundamentals of Web Application Architecture," 2009. [Online]. Available: https://reinvently.com/blog/fundamentals-web-application-architecture/. [Accessed: 06-Mar-2020].