Rancang Bangun Perangkat Lunak *Trusted Business Listing* dan *Complaint Management System* Berbasiskan Aplikasi Website

Yefta Sutanto, Undergraduate Student, Universitas Ciputra Surabaya, UC Town, CitraLand, Surabaya 60219
Adi Suryaputra Paramita, Information Technology Lecturer, Universitas Ciputra Surabaya, UC Town, CitraLand, Surabaya 60219

ABSTRAK

Perkembangan dan pertumbuhan ekonomi di Indonesia sudah sangat maju pesat jika dibandingkan dengan beberapa tahun silam. Pada tahun 2014, jumlah Usaha Mikro, Kecil, dan Menengah di Indonesia tercatat mencapai angka 56,5 juta unit, dengan hampir 98,9% dari jumlah tersebut adalah usaha mikro. Dengan semakin meningkatnya jumlah UMKM yang tercatat di Indonesia, maka semakin banyak pula dampak positif maupun dampak negatif yang dirasakan oleh rakyat Indonesia. Dampak positif yang ada yaitu berkurangnya angka pengangguran secara drastis serta banyaknya usahawan-usahawan muda yang mulai ber-enterpreneur sejak usia dini. Hal tersebut tentu saja menjadi suatu kebanggaan dan cikal bakal usaha besar di Indonesia. Sedangkan dampak negatif dari perkembangan dan pertumbuhan UMKM yang pesat di Indonesia adalah masalah klasik yang timbul dalam dunia bisnis, yaitu Kepercayaan dan *After Sales Service* dari suatu perusahaan terkait. Banyak sekali kasus penipuan yang dilakukan oleh perusahaan-perusahaan palsu dan tidak bertanggung jawab atas kerugian yang tidak sedikit bagi kliennya. Selain itu bagi orang awam atau non-bisnis yang merasa kurang puas atas produk atau jasa yang mereka beli seringkali tidak mendapatkan perlakuan yang menyenangkan dalam hal komplain atau mengadukan kekurangan produk perusahaan terkait. Dari kasus dan kejadian yang ada di lapangan perekonomian Indonesia inilah kami tercetus sebuah ide untuk mengakomodasi *Trusted Business Listing* dan *Complaint Management System* yang akan sangat membantu dalam hal memajukan dan mengembangkan perekonomian serta persaingan ekonomi yang sehat di Indonesia.

Kata kunci: Ekonomi, Website, Usaha Mikro Kecil Menengah, Trusted Business Listing, Complaint Management System

1. Pendahuluan

1.1. Latar Belakang

Indonesia adalah negara berkembang, dengan banyaknya bisnis-bisnis baru yang bermunculan setiap tahunnya. Mulai dari bisnis level mikro, level menengah atau yang biasa kita sebut dengan UKM / Usaha Kecil Menengah, hingga bisnis berlevel besar.

Namun, berdasarkan hasil survey singkat yang penulis lakukan, didapati bahwa permasalahan yang sering terjadi adalah adanya kesulitan ketika para pelaku usaha tersebut hendak mencari vendor atau partner untuk mendukung kinerja usaha yang mereka lakukan, seperti digambarkan pada Gambar 1.1 berikut.

Gambar 1.1 – Hasil Survey "Sistem Manajemen Komplain dan Daftar Bisnis Terpercaya"

Tidak sedikit dari para pelaku usaha mendapatkan partner yang kurang memuaskan dalam segi kinerja / hasil kerjanya. Permasalahan kedua yang muncul akibat kurang puasnya pelaku usaha adalah masalah komplain. Seringkali ketika pelaku usaha mengirimkan komplain kepada perusahan partner, mereka tidak mendapat balasan dan merasa diacuhkan.

Berdasarkan Feasibility Analysis, maka terdapat dua

Yefta Sutanto. Tel.: (+62) 81 857 4333.
E-mail: yefta@bisnisbagus.id

Adi Suryaputra Paramita. *Tel:* (+62-31) 745 1669
E-mail: adi.suryaputra@ciputra.ac.id

hal yang dapat penulis kembangkan untuk mengatasi permasalahaan tersebut, yaitu :

- Political : Kondisi politik di Indonesia yang berkaitan dengan perekonomian dan perdangangan semakin maju dan berkembang. Ditandai dengan banyaknya dukungan terhadap UMKM / Usaha Mikro, Kecil, dan Menengah dari pihak pemerintahan. Dukungan tersebut berupa peraturan perdagangan yang lebih menunjang dan mengatur tentang keamanan perdangan di Indonesia maupun dukungan finansial seperti adanya kompetisi-kompetisi dan pendanaan bagi para pelaku usaha mikro.
- Economical : Perkembangan perekonomian di Indonesia semakin pesat tiap tahunnya, hal tersebut dapat kita lihat pada data dari Badan Pusak Statistik[1], bahwa pada tahun 2012 terdapat sebanyak 56,5 juta unit UMKM. Angka tersebut terus mengalami peningkatan dari tahun ke tahunnya, sehingga hal tersebut menjadi peluang yang tinggi bagi penulis untuk mengembangkan aplikasi berbasis website ini.
- Social : Kehidupan sosial di masyarakat Indonesia sangatlah tinggi. Kebutuhan akan berkomunikasi, bertukar pendapat, menyampaikan pendapat dan kritik maupun saran sangatlah tinggi di Indonesia. Akan tetapi tidak ada sarana / media yang dapat menampung pendapat dan kritik saran dari para konsumen suatu produk. Seringkali konsumen melayangkan suara kritik dan saran untuk sebuah perusahaan, tetapi tidak mendapat hak sesuai yang mereka harapkan. Pihak perusahaan yang terkait pun terkesan tidak customer friendly dalam hal menangani masalah kritik atau komplain. Hal tersebut sangat menjadi peluang penulis dalam pengembangan market untuk aplikasi berbasis website ini
- Technological : Penggunaan teknologi untuk pengembangan aplikasi berbasis website ini sangatlah mencukupi berdasarkan infrastruktur teknologi di Indonesia. Berbasis pembuatan website pada umumnya dengan koneksi internet standard di Indonesia, pengguna sudah dapat mengakses aplikasi ini di mana saja.

1.2. Rumusan Masalah

- Bagaimana merancang dan membangun daftar bisnis terpercaya *(Trusted Business Listing)* berbasis aplikasi website sehingga memudahkan calon pelanggan untuk mencari perusahaan partner (dalam konteks ingin membeli atau menggunakan barang atau jasa dari perusahaan) ?
- Bagaimana merancang dan membangun sistem manajemen komplain *(Complaint Management System)* berbasis aplikasi website sehingga memudahkan pelanggan untuk menyampaikan komplain kepada perusahaan (dalam konteks setelah membeli atau menggunakan barang atau jasa dari perusahaan) ?

[1] ("Badan Pusat Statistik," n.d.)

2. Landasan Teori

2.1. Business Model dan Profit Model

Business Model adalah suatu teori yang menjelaskan tentang bagaimana suatu perusahaan atau organisasi menciptakan, menyampaikan, dan mengambil suatu nilai secara rasional dan nyata. Business model dapat digambarkan dengan mudah melalui Business Model Canvas. Business Model Canvas sendiri terbagi menjadi 9 kolom yang mewakili keseluruhan proses sebuah bisnis itu dijalankan.

Business Model Canvas dapat dibagi menjadi 3 kelompok berdasarkan tipe nya, yaitu Operational Exellence, Product Leadership, dan Customer Intimacy. Ketiga tipe ini memiliki perbedaan dalam menentukan tujuan dan arah suatu bisnis itu sendiri.

Operational Excellence merupakan tipe yang digunakan bisnis yang mengutamakan penggunaan bisnis model yang tepat sebagai kunci dari revenue stream mereka. Seperti pemanfaatan profit model, koneksi atau rekan kerja yang tepat, struktur kerjasama yang tepat, dan proses bisnis yang tepat. Product Leadership merupakan tipe yang mengutamakan kualitas dan keunggulan produk yang dipasarkan, cocok untuk suatu bisnis dengan tipe manufaktur untuk suatu produk tertentu.

Sedangkan tipe Costumer Intimacy mengutamakan relasi dan hubungan baik dan dekat dengan konsumen mereka, cocok untuk tipe bisnis jasa dan kepercayaan.

Tipe Business Model yang penulis gunakan adalah tipe Operational Excellence dengan mengambil Profit Model sebagai patokan Business Model-nya. Sedangkan untuk tipe Profit Model yang digunakan oleh penulis merupakan tipe Freemium, dimana skema berlangganan akan terbagi menjadi 4 pembagian harga untuk pengguna aplikasi. Model Freemium digunakan oleh penulis dengan alasan agar calon pengguna aplikasi ini dapat mencoba terlebih dahulu fitur-fitur yang terdapat pada aplikasi berbasis website ini, sehingga jika calon pengguna tersebut merasa puas dan bermanfaat maka akun pengguna dapat dengan mudah di upgrade menjadi akun premium yang berbayar.

2.2. Sistem Informasi

"Sistem Informasi" terdiri atas dua kata yaitu "sistem" dan "informasi". Dimana "sistem" dapat diartikan sebagai suatu kumpulan atau himpunan dari berbagai unsur, komponen, atau variabel yang terhubung, terorganisasi, saling berinteraksi, saling tergantung satu sama lain dan terpadu. (Jogiyanto, 2008) Sedangkan "informasi", adalah data yang telah diolah menjadi sebuah bentuk yang berarti bagi penerimanya dan bermanfaat bagi pengambilan keputusan saat ini atau saat mendatang. (Kadir, 2003). Sehingga, "informasi" merupakan kumpulan data yang diolah menjadi bentuk yang lebih berguna dan lebih berarti bagi yang menerima (Kristanto, 2003).

Dengan demikian "Sistem Informasi" dapat diartikan sebagai suatu sistem dengan berbagai komponen yang saling terhubung, contohnya dalam konteks website adalah HTML, CSS, dan PHP, yang mampu memberikan informasi yang memiliki nilai bagi penggunanya.

2.3. Website

Website adalah kumpulan dari halaman yang menampilkan informasi seperti data, teks, gambar, animasi dan lainnya,

dapat bersifat statis maupun yang dinamis, yang memiliki rangkaian atau tautan yang saling terkait dengan jaringan pada halaman. (Dwi, 2005).

2.3.1. HTML

HTML (*HyperText Markup Language)* adalah sebuah bahasa *markup* yang digunakan untuk membangun halaman web. Merupakan unsur dasar dari sebuah website, diciptakan oleh Tim Berners-Lee pada tahun 1999. (Dwi, 2005). Sebuah file HTML terdiri dari kumpulan tag-tag dengan spesifikasi HTML. Kode tag inilah yang diterjemahkan oleh browser sehingga dapat ditampilkan menjadi suatu gambaran yang sesuai dengan keinginan pembuatnya.

Secara umum file HTML dibagi menjadi empat jenis elemen:

- *Structural*, yaitu tag yang menentukan level atau tingkatan sebuah bagian (*heading*, paragraf, kutipan, dan lain-lain).
- *Presentational*, yaitu tag yang menentukan tampilan sebuah bagian text (cetak tebal, cetak miring, garis bawah, dan lain-lain.)
- *Hypertext*, yaitu tag yang merujuk atau merupakan tautan kepada halaman lain atau bagian lain didalam dokumen HTML tersebut.
- *Widget*, yaitu tag yang menghasilkan objek-objek tertentu seperti tombol, garis horizontal, dan lain-lain.

2.3.2. CSS

Cascading Style Sheet (CSS) atau yang biasa disingkat dengan CSS, merupakan suatu dokumen yang digunakan untuk melakukan pengaturan halaman web yang ditulis dengan HTML atau XHTML. (Saputra, 2012) Penggunaan CSS tidak memerlukan perangkat lunak tertentu karena CSS merupakan script yang telah embedded dengan HTML. CSS digunakan oleh Web Designer untuk menentukan warna, jenis huruf, tata letak, dan berbagai aspek tampilan dokumen.

Dokumen CSS dibuat terpisah dari dokumen HTML dan hal ini dilakukan untuk memisahkan antara isi dokumen (yang ditulis dengan HTML atau bahasa *markup* lainnya) dengan dokumen presentasi (yang ditulis dengan CSS). Dengan dilakukan pemisahan ini, Web Designer memiliki lebih banyak keleluasaan dan kontrol terhadap tampilan, dan mengurangi kompleksitas serta repetisi pada strukur isi.

2.3.3. Framework Bootstrap

Bootstrap adalah platform CSS yang dikembangkan secara opensource oleh Twitter pada pertengahan tahun 2010. ("About · Bootstrap", n.d.). Bootstrap digambarkan sebagai CSS sederhana yang dibangun secara khusus bersama dengan JavaScript sehingga dapat menyediakan lebih banyak fitur dan fleksibilitas dibandingkan CSS standar.

Dalam Bootstrap telah ada formatting standar untuk tabel, teks dengan tujuan tertentu, hingga gambar, sehingga seseorang yang masih pemula dalam membangun CSS sendiri, dapat dengan mudah membuat website dengan CSS yang memenuhi standar dan terlihat indah.

2.3.4. JavaScript

JavaScript merupakan bahasa *cross-platform* yang dikembangkan oleh Netscape dan pertama kali digunakan dalam browser Netscape. JavaScript dibuat agar dapat dengan mudah diintegrasikan ke dalam program dan aplikasi lain, dan mudah di eksekusi oleh browser. Seluruh browser saat ini sudah mendukung JavaScript.

JavaScript tidak memerlukan compiler atau penerjemah khusus untuk menjalankannya, dikarenakan compiler JavaScript sudah terintegrasi dengan browser.

2.3.5. PHP

PHP merupakan singkatan dari *Hypertext Preprocessor* yang pada umumnya digunakan sebagai bahasa pemrograman website dikarenakan memudahkan proses pembuatan *website* yang dinamis dengan cepat, skalability yang fleksible, dan kemampuan atau batasan pengembangan yang hampir tidak terbatas. (Saputra, 2012).

Dalam perangkat lunak sistem akreditasi dan rekomendasi bisnis berbasis website, bahasa ini digunakan diatas framework Laravel untuk mempermudah dan mempercepat proses pembuatan aplikasi ini.

Keuntungan dalam menggunakan PHP antara lain:

- Skalability tak terbatas.
- Forked-able bila digabungkan dengan penggunakan apache2-mpm-worker module di server.
- Terdapat lebih dari 25 database yang didukung PHP, beberapa di antaranya adalah MySQL, PostGreSQL, SQLite, dll.
- Merupakan program opensource, sehingga dapat dimodifikasi dan didistribusikan oleh setiap pengguna dan pengembangannya dapat dilakukan secara terbuka dan semua orang dapat berpatisipasi, misal melalui github atau mercurial repository.
- PHP dapat dijalankan di seluruh *platform* server apapun, termasuk Mac OSX dan Windows, serta dapat di compile dan build sesuai kebutuhan.
- Memiliki banyak modul opsional yang dapat dipasang dan dilepas kapanpun sesuai kebutuhan, seperti GD, CURL, dll.
- Dapat melakukan CLI atau berkomunikasi dengan server menggunakan fopen_process sehingga melalui browser, aplikasi dapat melakukan inisialiasi proses dan komputasi langsung terhadap server.

2.3.6. Framework Laravel

Laravel adalah framework PHP yang dikembangkan pertama kali oleh Taylor Otwell pada tahun 2012. (McCool, 2012). Merupakan framework pertama yang mengedepankan ideologi “clean code” dan “expressiveness”. ("Tutorial Laravel Bahasa Indonesia: Kenapa Memilih Laravel?", n.d.). Meskipun masih tergolong baru, komunitas pengguna Laravel telah berkembang cukup pesat dan mampu menjadi alternatif utama dari sejumlah framework umum seperti CodeIgniter & Yii. Laravel oleh para developer disetarakan dengan CodeIgniter dan FuelPHP namun memiliki keunikan tersendiri dari sisi coding yang lebih bersih, ekspresif dan elegan. Gambar 2.1 berikut merupakan gambar logo

Laravel.

Gambar 2.1 - Logo Laravel

Pada awalnya Laravel dikembangkan seorang diri oleh Taylor, namun setelah menginjak versi ke-empat, framework opensource ini dikembangkan bersama oleh komunitas dengan tokoh-tokoh penting seperti Dayle Rees, Shawn McCool, Jeffrey Way, Jason Lewis, Ben Corlett, Franz Liedke, Dries Vints, Mior Muhammad Zaki dan Phil Sturgeon. Dimana mereka adalah kontributor sejumlah framework dan banyak library PHP. ("Tutorial Laravel (1) Apa itu Laravel? | emka.web.id", n.d.).

Beberapa keunggulan Laravel dibandingkan dengan framework sejenis antara lain:

- Tidak terpengaruh dengan "backward compatibility" dari PHP dikarenakan Laravel pertama kali didevelop menggunakan PHP 5.3.
- Tersedia generator yang mutakhir dan memudahkan proses development, seperti Artisan CLI dan Eloquent ORM.
- Fitur Schema Builder dan Migration & Seeding untuk berbagai jenis database, juga tersedia Query Builder.

2.3.6.1. Arsitektur Laravel

Laravel dapat disebut sebagai sebuah framework *"full stack"* karena ia dapat melakukan segalanya, mulai dari menerima *request* dan mengembalikan *response*, manajemen *database*, hingga menghasilkan kode HTML akhir.

Secara umum, interaksi dengan framework ini dilakukan menggunakan sebuah tool berbasis *command-line* untuk mengatur lingkungan kerja project Laravel tersebut. Tool ini dinamakan Artisan yang dapat digunakan untuk menghasilkan mulai dari kerangka kode dasar, skema database dasar, implementasi skema database, hingga manajemen aset (CSS, JavaScript, gambar, dll.) dan file konfigurasi.

2.3.6.2. Model-View-Controller (MVC)

Laravel memiliki salah satu folder utama bernama /app/, yang didalamnya terdapat subfolder /models/, /views/, dan /controllers/. Hal ini menandakan Laravel menerapkan pola arsitektur aplikasi *model-view-controller (MVC)*. Pola MVC menerapkan pemisahan *"business logic"* dari *input* dan *presentation logic* yang terkait dengan *graphical user interface (GUI)*. Pada konteks aplikasi web berbasis Laravel, *business logic* ini pada umumnya berisi *data model* untuk user, artikel, dan GUI tersebut hanyalah sebuah halaman web biasa di browser.

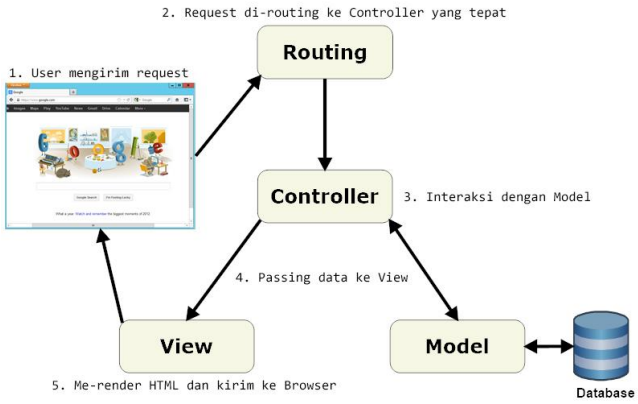
Terdapat tiga komponen pada pola MVC:

Model – Disebut juga sebagai *domain*. Model dibuat berdasarkan benda di dunia nyata, seperti orang, akun bank, atau produk. Dalam kasus membangun sistem ini misalnya, model yang ada bisa berupa *user* dan *komplaint*. Model biasanya bersifat permanen dan disimpan diluar dari aplikasi, pada umumnya didalam database.

View – Merupakan representasi visual dari *model*, yang telah diberikan atau ditambahkan konteks. Hasil akhir daripada *view* adalah tampilan yang direnderer oleh browser, berupa HTML yang memiliki atau didasarkan data dari suatu *model*. Sebagai contohnya, sistem ini akan memiliki kebutuhan untuk menampilkan daftar bisnis. Daftar ini secara data dapat diakses dari *model*, namun sebuah *view*-lah yang akan memproses data ini dan menampilkannya ke user. Apabila kemudian ada input data dari user, bukanlah *view* sendiri yang memproses hal ini, namun *controller*.

Controller – Merupakan koordinator yang menghubungkan antara *view* dan *model*. *Controller* bertanggung jawab untuk memproses input, melakukan proses berdasarkan model, dan menentukan langkah apa yang harus dilakukan selanjutnya, seperti me-*render* sebuah *view* atau mengarahkan user ke halaman lain. Pada sistem ini contohnya, *controller* dapat mencari nama bisnis tertentu secara spesifik kemudian memberikan datanya kepada *view* untuk di-*render*.

2.3.6.3. Alur Kerja Laravel



Gambar 2.2 - Alur Kerja Laravel

Saat berinteraksi dengan sebuah aplikasi Laravel, *browser* mengirimkan *request* yang kemudian diterima oleh *web server* dan diterukan kepada *Laravel Routing Engine*. Router Laravel ini kemudian melanjutkan *request* tersebut kepada *controller* yang tepat dan *class* yang tepat berdasarkan aturan *routing* yang telah ditentukan.

Controller class yang dituju kemudian mengambil alih operasi. Pada halaman statis, *controller* akan langsung me-*render* sebuah *view* yang langsung menjadi HTML dan dikirim balik ke *browser*. Namun pada halaman dinamis, *controller* akan berinteraksi (melakukan transaksi data) dengan *model*, yang merupakan sebuah objek PHP yang merepresentasikan suatu elemen didalam aplikasi (bisa berupa user, komplain, dll.) dan bertugas untuk menjadi mediator dengan database. Setelah memanggil *model* tersebut dan datanya, *controller* memproses data yang diterima kemudian me-*render*-nya bersama dengan *view* (HTML, CSS, dan gambar) untuk dikirimkan kembali ke *browser*.

2.4. Database

Database dapat dibayangkan sebagai sebuah lemari arsip tempat menyimpan folder dan file. Folder dapat merepresentasikan tabel, dan file dapat merepresentasikan

field. (Elmasri & Navathe, 2007).

Database terdiri dari 2 kata, yaitu Data dan Base. Data adalah representasi fakta dunia nyata yang mewakili suatu objek seperti manusia (pegawai, siswa, pembeli, pelanggan), barang, hewan, peristiwa, konsep, keadaan, dan sebagainya, yang direkam dalam bentuk angka, huruf, simbol, teks, gambar, bunyi, atau kombinasinya. Sedangkan Base dapat diartikan sebagai markas atau gudang dimana tempat bersarang atau berkumpul. Sehingga database dapat diartikan sebagai tempat data berkumpul atau dikumpulkan secara sistematis dan terorganisir. (Fathansyah, 2007).

Database digunakan untuk menyimpan seluruh informasi yang besar dan saling berketergantungan satu sama lain dan berdasarkan banyak kategori.

2.4.1. MySQL

MySQL merupakan salah satu jenis DBMS (Database Management System) *open-source* dan data di dalam model MySQL disimpan dalam obyek database yang bernama table atau *table*. Tabel terdiri dari kumpulan data yang saling berhubungan dan tersusun atas kolom dan baris.

Keuntungan dalam menggunakan MySQL:

Merupakan program opensource, sehingga dapat dimodifikasi dan didistribusikan oleh setiap pengguna dan pengembangannya dapat dilakukan secara terbuka dan semua orang dapat berpatisipasi, misal melalui github atau mercurial repository.

Dapat menyimpan berbagai macam tipe data didalam table, termasuk tipe data blob (BL Object).

Skalability tak terbatas. Dapat dijalankan secara independen pada sebuah server sehingga hanya melayani transaksi data.

Multithreading, dapat dimodifikasi untuk berjalan secara multithread.

Memiliki banyak modul opsional dan extention yang dapat dipasang dan dilepas kapanpun sesuai kebutuhan, seperti MySQLi, dll.

2.5. Web Server

Web server merupakan perangkat lunak yang menyediakan layanan akses kepada pengguna melalui protokol komunikasi HTTP atau HTTPS atas berkas-berkas yang terdapat pada suatu situs web dalam layanan ke pengguna.

Fungsi utama sebuah web server adalah untuk mentransfer berkas atas permintaan pengguna melalui protokol komunikasi yang telah ditentukan. Pemanfaatan web server juga berfungsi untuk mentransfer seluruh aspek pemberkasan dalam sebuah halaman web yang terkait, yang di dalamnya terdapat teks, gambar, video, atau lainnya.

2.5.1. Apache

Apache adalah web server yang dapat dijalankan di beragam sistem operasi seperti Windows, Linux, Mac OS X, dan beragam platform lainnya. Apache memiliki fitur-fitur seperti pesan kesalahan yang dapat dikonfigurasi, autentikasi berbasis database dan lain-lain.

Apache merupakan perangkat lunak sumber terbuka dikembangkan oleh komunitas terbuka dibawah naungan Apache Software Foundation (ASF).

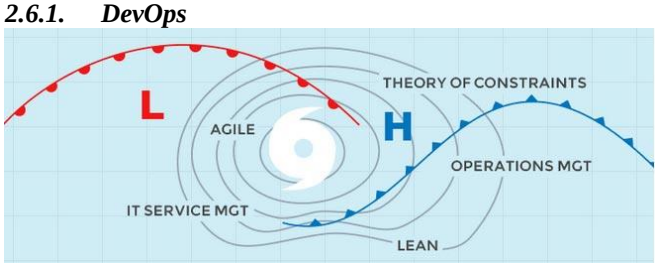
Apache memiliki beberapa kelebihan, antara lain:

- Apache merupakan program opensource.
- Proses implementasi dan deployment yang relatif mudah dibandingkan dengan web server sejenis seperti Internet Information Systems (IIS).
- *Cross-platform*, dapat dijalankan pada berbagai macam sistem operasi.
- Memiliki banyak modul-modul tambahan dan mudah untuk menambahkan modul baru atau menaktifkan atau non-aktifkan modul.

2.6. Systems Development Life Cycle

Systems Development Life Cycle atau siklus hidup pengembangan sistem, dalam rekayasa sistem dan rekayasa perangkat lunak, adalah proses pembuatan dan pengubahan sistem serta model dan metodologi yang digunakan untuk mengembangkan sistem-sistem tersebut. Konsep ini umumnya merujuk pada sistem komputer atau informasi. Dalam rekayasa perangkat lunak, konsep SDLC mendasari berbagai jenis metodologi pengembangan perangkat lunak. Metodologi-metodologi ini membentuk suatu kerangka kerja untuk perencanaan dan pengendalian pembuatan sistem informasi, yaitu proses pengembangan perangkat lunak.

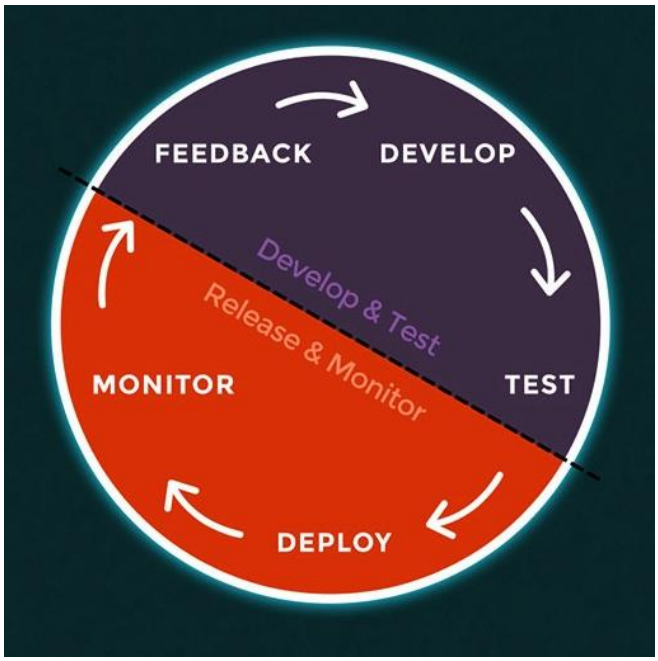
2.6.1. DevOps



Gambar 2.3 - Perfect Storm of 2009

Filosofi DevOps lahir pada sebuah “insiden” yang disebut The Perfect Storm of 2009 yang melahirkan DevOps. Perpaduan metodologi Agile, Operations Management (Systems Thinking & Dynamics), Theory of Constraints, LEAN dan manajemen IT Service pada konferensi-konferensi, debat, dan Twitter (via hashtag #devops). ("DevOps: Collaborative Software Development | New Relic", n.d.).

Model SDLC DevOps lahir dikarenakan adanya kebutuhan untuk meningkatkan agility dari deployment IT Service. Pendekatan DevOps mengedepankan komunikasi, kolaborasi, dan integrasi antara tim developer dengan tim operasional, dimana yang ada pada umumnya saat ini, kedua tim ini seperti berdiri sendiri-sendiri dan cenderung sulit untuk dikolaborasikan.



Gambar 2.4 - Lifecycle DevOps

Metode Agile software development telah berhasil membuat jalan development baru yang berbeda dari model waterfall, sehingga software yang dikembangkan dengan Agile, dapat terus-menerus dikembangkan. Namun, Agile tidak memperhitungkan segi operasional, sehingga meskipun pengembangan dari software dapat terus berjalan, deployment masih berbasiskan waterfall. nted. Sehingga, dalam pengembangan aplikasi menggunakan model DevOps, fungsionalitas lintas fungsi, pembagian tanggungjawab dan kepercayaan menjadi panduan utama. DevOps pada dasarnya melakukan pengembangan pada model Agile, hingga ke deployment yang berlangsung terus-menerus.

Untuk semakin mendukung fungsi ini, DevOps mengedepankan otomasi dari operasi perubahan, konfigurasi dan perilisan aplikasi.

Keuntungan menerapkan DevOps, pada segi teknikal:

- Perilisan aplikasi yang dapat berjalan secara kontinu.
- Problem yang timbul dan perlu diselesaikan dapat dibuat menjadi lebih simpel.
- Waktu yang diperlukan untuk *troubleshooting* masalah semakin singkat.

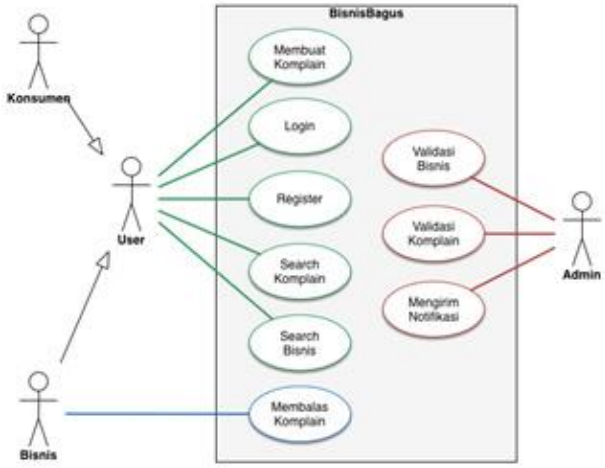
Keuntungan menerapkan DevOps, pada segi bisnis:

- Penambahan fitur baru yang lebih cepat dan kontinu.
- Kondisi operasional yang lebih stabil.
- Lebih banyak waktu yang tersedia untuk menambahkan fitur baru, daripada membenahi problem atau perawatan aplikasi.

3. Desain Sistem

3.1. Use Case Diagram

Use Case Diagram pada dasarnya adalah sebuah bentuk representasi sederhana dari interaksi antara user dengan sistem. Sebuah *use case diagram* sendiri dapat mengandung beberapa *use case* degan cakupan yang lebih kecil dan spesifik terhadap suatu fungsi dalam sistem. User dengan peran berbeda didalam *use case* disebut sebagai aktor.

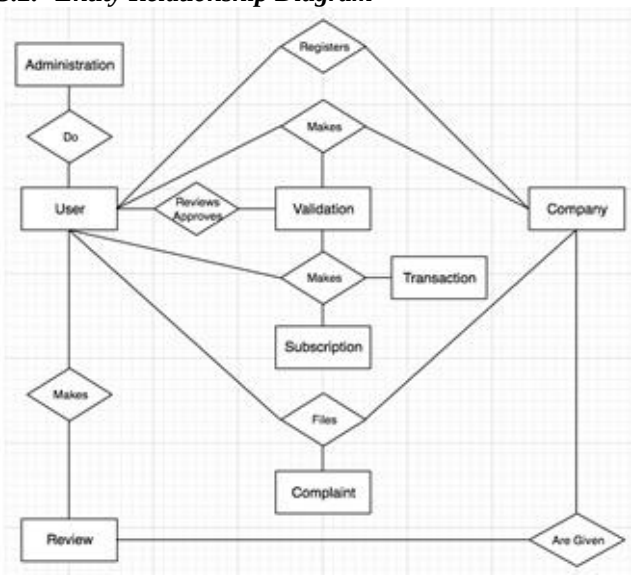


Gambar 3.1 - Use Case BisnisBagus

Terdapat tiga jenis user pada sistem BisnisBagus, yaitu Konsumen, Bisnis, dan Admin.

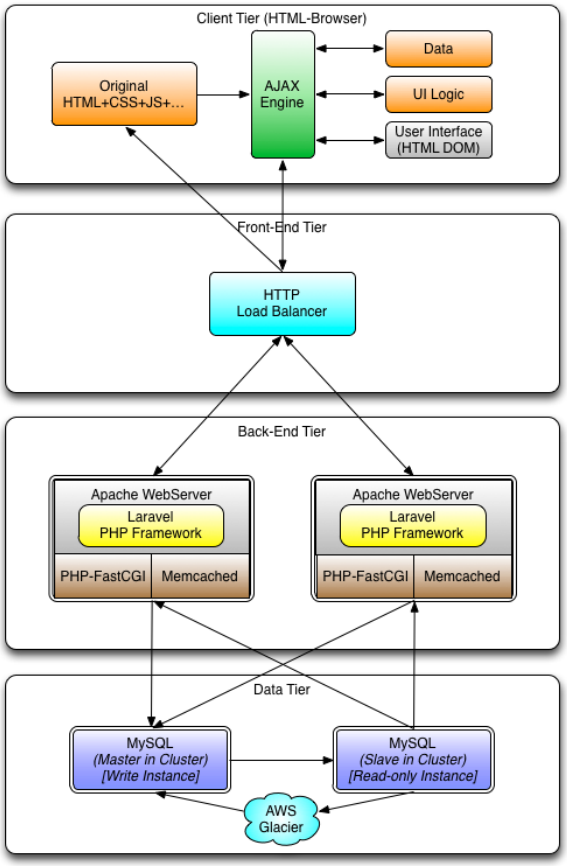
- Konsumen : Merupakan user biasa dengan akses untuk membuat komplain baru dan merespon komplain.
- Bisnis : Merupakan user biasa dengan akses untuk mendaftarkan dan membuat profil bisnis, kemudian dapat merespon komplain yang masuk ke bisnis-nya masing-masing.
- Admin : Merupakan user tingkat atas dengan kemampuan manajemen seluruh situs.

3.2. Entity Relationship Diagram



Gambar 3.2 - Entity Relationship Diagram BisnisBagus

3.3. *Arsitektur Sistem*



Gambar 3.3 - Arsitektur Sistem

Gambar 3.3 menggambarkan Desain Sistem Ideal yang dirancang untuk BisnisBagus. Dimana dibedakan menjadi empat tier, yaitu client, fron-end, back-end, dan data. Dari Desain Sistem Ideal diatas, untuk penelitian ini telah berhasil dicapai pada tahap sebuah WebServer dan sebuah MySQL Database Server.

- Client Tier merupakan bagian yang ada pada sisi client atau Browser. Komponen yang terdapat pada bagian ini merupakan hal seperti HTML, JavaScript, dan AJAX.
- Front-End Tier merupakan bagian yang menengahi antara Client dengan Back-End, alias tempat dimana user request pertama kali diterima. Pada bagian ini dapat secara langsung terdapat WebServer atau Load Balancer. Apabila ada Load Balancer, tugas Load Balancer pada bagian ini adalah untuk membagi request kepada sejumlah WebServer dibelakangnya atau di Back-End Tier.
- Back-End Tier merupakan bagian dimana terdapat pemroses utama alias WebServer itu sendiri. Framework aplikasi berjalan di Tier ini dan didukung dengan interface CGI yang ada, contohnya PHP Fast-CGI bisa berada di Tier ini sebagai middleware antara Front-End dengan WebServer misalkan Apache atau Nginx.
- Pada Data Tier terdapat tempat-tempat penyimpanan model, yang disebut data, bernamakan Database. Database Engine yang digunakan disini misalkan MySQL atau PostgreSQL. Dapat terdapat juga lebih dari satu Database Server untuk redundansi.

3.4. *Tampilan Website*

Gambar 3.4 - Tampilan Website

4. Implementasi dan Pengujian

4.1. *Implementasi*

Pada bagian ini akan dijabarkan secara singkat dan padat mengenai basis dari fitur yang ada dan di implementasikan pada BisnisBagus agar tidak repetitif dan penjabaran lebih terfokus, dikarenakan penjelasan secara mendetail termasuk langkah-langkah step-by-step penggunaan modul dan fitur-fitur Laravel telah ada dan tersedia secara lengkap dan bebas di http://laravel.com/docs/4.2 .

4.1.1. *Registration*

```php
<?php

use Illuminate\Auth\UserTrait;
use Illuminate\Auth\UserInterface;
use Illuminate\Auth\Reminders\RemindableTrait;
use Illuminate\Auth\Reminders\RemindableInterface;

use Illuminate\Database\Eloquent\SoftDeletingTrait;

class User extends Eloquent implements UserInterface, RemindableInterface {

    use UserTrait, RemindableTrait, SoftDeletingTrait;

    /**
     * The database table used by the model.
     *
     * @var string
     */
    protected $table = 'users';

    /**
     * The attributes that can be mass-assigned to
     *
     * @var array
     */
    protected $fillable = ['email', 'password', 'privilege', 'is_active'];

    /**
     * The attributes excluded from the model's JSON form.
     *
     * @var array
     */
    protected $hidden = array('password', 'remember_token');

    protected $dates = ['deleted_at'];

    public function userAccount()
    {
        return $this->hasOne('Account')->whereNull('deleted_at');
    }
```

Gambar 4.1 - Potongan Kode Model

Registrasi adalah proses pembuatan (Contruct), pengisian data/atribut (Fill), dan penyimpanan (Store) Model dari suatu representasi data. Dalam BisnisBagus, contoh ada beberapa Model seperti User, Company, dan Komplaint.

Model didalam Laravel, dapat diimplementasikan dengan mudah bila melakukan extend pada interface

Eloquent. Ketersediaan IoC (Inversion of Control) pada Eloquent, membuat semua method Eloquent yang ada, dapat dipanggil dimanapun suatu Model berada.

Gambar 4.1 merupakan contoh pembuatan (Construct) Model CompanyGallery. Tabel yang terdapat di dalam media penyimpanan, atau disebut juga sebagai Database, bernama 'company_gallery'.

Dengan mendeklarasikan '$fillable', menentukan kolom mana sajakah yang boleh di fill secara mass-assign saat inisialisasi suatu Model.

Pembuatan relationship-pun sangat mudah, seperti terlihat pada Gambar 4.2, dengan membuat method 'userAccount()' akan memberitahu Laravel melalui Eloquent bahwa Model User masing-masing memiliki satu Model Account.

```
$user = new User;
$user->email = $input['register_email'];
$user->password = $password;
$user->privilege = $input['register_type'];
$user->is_active = 0;
$user->save();
```

Gambar 4.2 - Potongan Kode Pengisian Model

Proses pengisian data/atribut (Fill), dan penyimpanan (Store) Model User terlihat pada Gambar 4.2. Sebuah Model User baru di inisialisasi, kemudian masing-masing atributnya di isi, lalu di save();

Selain dengan cara diatas, sebuah Model juga dapat di *instantiate*, seperti berikut Gambar 4.3.

```
User::create(array(
    'email' => 'admin@bisnisbagus.id',
    'password' => Hash::make('admin'),
    'privilege' => 'admin',
    'is_active' => 1
    ));
```

Gambar 4.3 - Potongan Kode *Instantiate* Model User

Keduanya melakukan hal yang sama, namun yang pertama menerapkan konsep Object Oriented Programming, dimana sebuah Model diperlakukan sepenuhnya sebagai Object. Sedangkan yang kedua merangkum semua langkah kedalam suatu metode yang bernama *instantiate*.

4.1.2. Mengirim Email

Untuk mengirim email-pun, Laravel telah menyediakan helper dan class yang cocok, yakni Mail. Lihat Gambar 4.4

```
$data = array(
    'email' => Input::get('register_email'),
    'first_name' => Input::get('register_first_name'),
    'activation_code' => $activation_code
    );
Mail::send('emails.auth.activate', $data, function($message) use ($data) {
    $message->from('no-reply@bisnisbagus.id', 'BisnisBagus Indonesia');
    $message->to($data['email'], $data['first_name']);
    $message->subject('Account Activation');
});
```

Gambar 4.4 - Potongan Kode Mengirim Email

Potongan kode tersebut melakukan pengiriman email kepada user yang baru saja mendaftar. Menggunakan view email bernama 'emails.auth.activate' yang artinya adalah '/views/emails/auth/activate.blade.php'. Sedangkan array '$data' adalah data yang dipassing kedalam closure dari function Mail::send untuk di tampilkan isinya didalam email. Email pun mendukung Blade Templating, yang akan dijelaskan berikut-berikut ini.

Function Mail::send memiliki syarat minimal ketersediaan data adalah sebagai berikut:

- $message→from = dari siapakah email tersebut dikirim.
- $message→to = kepada siapakah email tersebut ditujukan.
- $message→subject = judul email.

4.1.3. Routing

Setelah ada Model, View, dan Controller, terakhir dan yang paling penting adalah Routing. Sebuah file routing, mendefinisikan dan mendeklarasikan apa yang harus diperbuat oleh aplikasi setiap kali sebuah request terhadap suatu URL diberikan/diterima oleh webserver dan diteruskan ke aplikasi.

Sebuah route dapat di bedakan menggunakan jenis request (GET atau POST, misalnya), nama, dan URL yang dituju.

```
Route::get('/', function()
{
    return View::make('landing');
});

// BISNISBAGUS PROFILE
Route::get('profile', function()
{
    return View::make('profile');
});

// CONTACT US
Route::get('contact', function() {
    return View::make('contact');
});
```

Gambar 4.5 - Potongan Kode Routing

Pada Gambar 4.5, saat ada user yang mengunjungi root ('/'), yang artinya adalah 'https://bisnisbagus.id' itu sendiri, akan ditampilkan view 'landing.blade.php'. Apabila ada yang mengunjungi /profile (https://bisnisbagus.id/profile), maka 'profile.blade.php'-lah yang akan ditampilkan.

```php
// routes available only to guests
Route::group( array('before' => 'guest'), function()
{
    // login
    Route::get('login', array(
        'uses' => 'SessionController@create',
        'as' => 'session.create'
        ));

    Route::post('login', array(
        'uses' => 'SessionController@store',
        'as' => 'session.store'
        ));
```

Gambar 4.6 - Potongan Kode Routing Rules

Routes juga dapat diberikan aturan atau *rules*, dalam contoh pada Gambar 4.6, hanya 'guest' alias user yang belum login yang dapat melihat login page. Pada gambar tersebut juga dibedakan *handler* untuk request GET dan POST. Sebuah *request* tipe GET akan menghasilkan / memberikan / mengembalikan tampilan 'login.blade.php', sedangkan *request* tipe POST akan memulai proses autentikasi.

```php
// business/profile/list - List Business(es) user owns
Route::get('list', function()
{
    $obj = new BusinessController();
    $data['company'] = $obj->showList(Auth::user()->id);
    return View::make('profile/business-list')->with($data);
});
```

Gambar 4.7 - Potongan Kode Passing Data di Routing

Gambar 4.7 menunjukkan bahwa, dari dalam sebuah aturan *routing*, juga dapat me-*request* data dari Controller dan melakukan passing data yang dihasilkan ke *view* yang ditampilkan. Selain itu, aturan routing juga dapat memiliki fungsi *branching* alias *if* seperti pada gambar 4.8 berikut.

```php
if (Auth::check()) {
    $obj = new RegisterController();
    $data['user'] = $obj->show(Auth::user()->id);
}
return View::make('complaint/step4-compose')->with($data);
```

Gambar 4.8 - Potongan Kode Routing dengan If

4.1.4. Controller

Controller adalah middleman daripada View dan Model. Dimana request dari user setelah masuk ke routing, akan diarahkan pada Controller yang tepat dan sesuai dengan routing rules.

Didalam Controller terjadi proses CRUD pada umumnya, seperti mengambil data dari database (yang berupa Model), iterasi dari data tersebut, manipulasi, dan penyimpanan kembali atau passing data ke View untuk ditampilkan kepada user.

```php
/**
 * Display the specified resource.
 *
 * @param  int  $id
 * @return Response
 */
public function show($id)
{
    return User::where('id', $id)->with('userAccount')->first();
}
```

Gambar 4.9 - Potongan Kode Function Show di Controller

Potongan kode pada Gambar 4.9 diatas merupakan function show() yang ada pada RegisterController, sebuah Controller yang menengahi proses registasi dan data tentang user.

Function show() meminta User model dari database yang memiliki 'id' yang sesuai, lengkap dengan Eloquent Relationship-nya yakni userAccount, yang akan mengembalikan sebuah Model User lengkap dengan Model Account.

```php
/**
 * Show the form for editing the specified resource.
 *
 * @return Response
 */
public function edit()
{
    $user = User::where('id', Auth::user()->id)
        ->where('privilege', Auth::user()->privilege)
        ->with('userAccount')
        ->first();
    if ($user->privilege == 'bisnis') {
        $view = 'profile/business';
    } else {
        $view = 'profile/consumer';
    }
    return View::make($view, array('user' => $user));
}
```

Gambar 4.10 - Potongan Kode Function Edit di Controller

Potongan kode pada Gambar 4.10, juga sama mengambil sebuah model User dengan Account dari database, kemudian melakukan pengecekan pada Object User tersebut. Apakah atributnya yang bernama 'privilege' isi datanya adalah 'bisnis'. Jika ya, view yang digunakan adalah 'view/profile/business.blade.php', jika tidak, gunakan 'view/profile/consumer.blade.php'. View tersebut kemudian di render dan ditampilkan kepada user beserta data yang dipassing pada closure View::make tersebut.

Selain basic CRUD seperti diatas, hampir seluruh business logic dari aplikasi juga terjadi, dan sebaiknya terjadi, didalam Controller. Contoh business logic adalah sebagaimana terlihat pada Gambar 4.11 berikut.

```php
$company = Company::where('id', $id)->where('user_id', Auth::user()->id)->first();
if (!$company) {
    return App::abort(404);
}

$input = Input::except('_token', 'company_type', 'company_payment_method');
foreach ($input as $key => $value) {
    if (trim($value) === '') {
        $input[$key] = null;
    }
}
```

Gambar 4.11 - Potongan Kode Business Logic

Setelah meminta sebuah Model Company, dilakukan pengecekan apakah Object yang dikembalikan ada, jika

tidak, hentikan kode dan berikan error 404 (Not Found) kepada user.

Bila ada, kode tidak berhenti, dan lakukan iterasi dari input dari user. Iterasi yang dilakukan diatas adalah untuk mengubah setiap key-value pair dari array di input yang kosong (‘’), menjadi null (null).

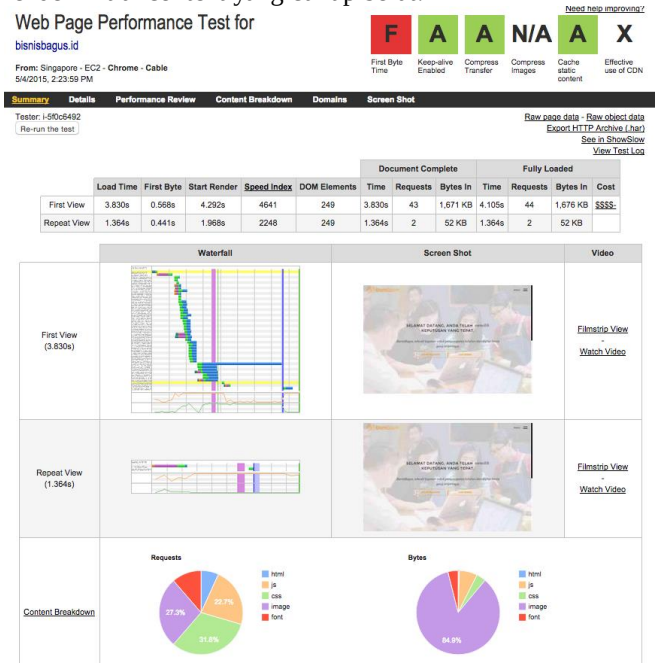
4.2. Pengujian

Untuk melakukan pengujian performa aplikasi akan digunakan online tool dari www.webpagetest.org dan http://nibbler.silktide.com/ .

Hasil pengujian dapat dilihat di URL berikut ini: http://www.webpagetest.org/result/150504_ZQ_BKF/ .

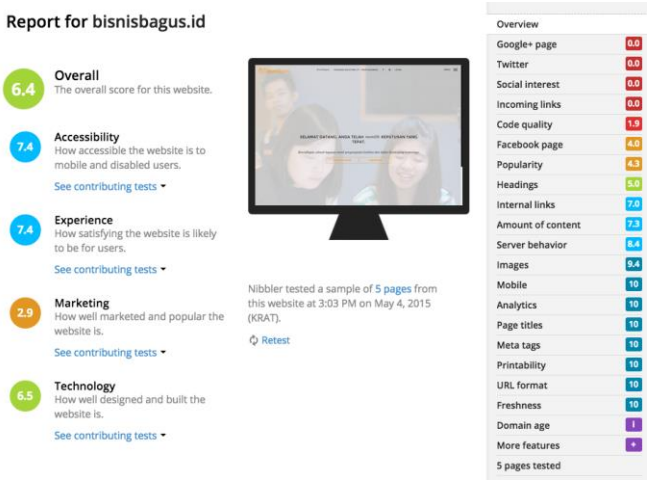
Pengujian pada webpagetest.org lebih berfokus pada performa. First Byte Time yang dimaksud adalah perlu berapa lama waktu setelah user pertama kali mengirim request ke server, sampai server memberikan response balik yang telah berupa content HTML.

BisnisBagus.id mendapat hasil yang cukup tinggi yaitu 0.568 detik (normalnya dibawah 0.2 detik) dikarenakan render initial content yang cukup berat.

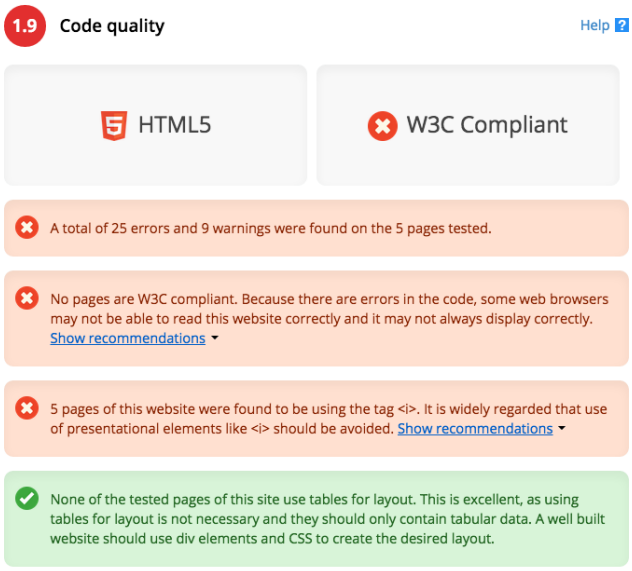


Gambar 4.12 - Hasil Pengujian dari webpagetest.org

Sedangkah untuk hasil dari Nibbler Silktide bisa dilihat hasil laporannya secara online pada URL berikut ini: http://nibbler.silktide.com/en_US/reports/bisnisbagus.id



Gambar 4.13 - Hasil Pengujian Pada nibbler.silktide.com



Gambar 4.14 – Hasil Pengujian Bagian Code Quality

Pengujian pada nibbler.silktide.com lebih berfokus pada tampilan content halman website, kesesuaian atas ketentuan penulisan atau pembuatan website dari WC3, dan user engagement.

Code Quality mendapatkan predikat 1.9 dari nilai total 10 dikarenakan ada beberapa kode tampilan yang belum sesuai dengan standar WC3. Poin ini apabila semakin besar nilainya, artinya semakin baik predikatnya.

5. Kesimpulan

- Aplikasi Sistem Manajemen Komplain dan Daftar Bisnis Terpercaya dapat dibangun menggunakan bahasa pemrograman PHP dengan framework Laravel, dan telah dapat diakses secara online di https://bisnisbagus.id .
- Tampilan pada website BisnisBagus dapat dibangun menggunakan CSS framework BootStrap sehingga menjadi responsive dan tetap dapat diakses dengan mudah meskipun diakses dari berbagai macam device seperti mobile device.

6. Saran

Saran untuk pengembangan kedepan daripada penelitian ini adalah:

- Gunakan *permalink* sebagai pengganti ID bisnis pada URL menuju halaman profil suatu bisnis di BisnisBagus. Sebagai contohnya, dari URL https://bisnisbagus.id/business/info/354 diubah menjadi http://bisnisbagus.id/business/info/universitas-cihuy-surabaya .
- Penambahan fitur untuk mengirim *Review* atau apresiasi kepada bisnis.
- Penambahan fitur menyimpan profil bisnis menjadi PDF dan/atau layout khusus cetak, sehingga apabila dicetak secara langsung oleh user menggunakan *browser* hasilnya rapi.

DAFTAR PUSTAKA

About · Bootstrap. (n.d.). Retrieved May 20, 2014, from http://getbootstrap.com/about/

Architecture of Laravel Applications - Laravel Book. (n.d.). Retrieved March 21, 2015, from http://laravelbook.com/laravel-architecture/

Badan Pusat Statistik. (n.d.). *Tabel Perkembangan UMKM pada Periode 1997-2012*. Retrieved Feb 3, 2015, from http://www.bps.go.id/tab_sub/view.php?kat=2&tabel=1&daftar=1&id_subyek=13%20¬ab=45

Blanchard, B. S., & Fabrycky, W. J. (1981). *Systems engineering and analysis*. Englewood Cliffs, N.J: Prentice-Hall.

Business - Definition and More from the Free Merriam-Webster Dictionary. (n.d.). Retrieved May 22, 2014, from http://www.merriam-webster.com/dictionary/business

Davis, G. B., & Olson, M. H. (1985). *Management information systems: Conceptual foundations, structure, and development*. New York: McGraw-Hill.

DevOps Benefits: Why Do DevOps | New Relic. (n.d.). Retrieved May 15, 2014, from http://newrelic.com/devops/benefits-of-devops

DevOps Lifecycle for Continuous Integration | New Relic. (n.d.). Retrieved May 15, 2014, from http://newrelic.com/devops/lifecycle

DevOps: Collaborative Software Development | New Relic. (n.d.). Retrieved May 15, 2014, from http://newrelic.com/devops/what-is-devops

Dwi, P. D. (2005). *Solusi menjadi Web Master melalui manajemen Web dengan PHP*. Jakarta: Elex Media Komputindo.

Elmasri, R., & Navathe, S. (2007). *Fundamentals of database systems*. Boston: Pearson/Addison Wesley.

Fathansyah (2007). *Basis data*. Bandung: Penerbit Informatika.

Jogiyanto, H. (2008). *Analisis dan desain sistem informasi: Pendekatan terstruktur teori dan praktek aplikasi bisnis*. Yogyakarta: ANDI.

Kadir, A. (2003). *Pengenalan sistem informasi*. Yogyakarta: ANDI.

Kristanto, A. (2003). *Jaringan komputer*. Yogyakarta: Graha Ilmu.

Keeley, L. (2013). *Ten types of innovation: The discipline of building breakthroughs*. Hoboken, N.J: Wiley.

McCool, S. (2012). *Laravel starter: The definitive introduction to the Laravel PHP web development framework*. Birminghan, UK: Packt Pub.

MySQL :: About MySQL. (n.d.). Retrieved May 16, 2014, from http://www.mysql.com/about/

Osterwalder, A., Pigneur, Y., & Clark, T. (2010). *Business model generation: A handbook for visionaries, game changers, and challengers*. Hoboken, NJ: Wiley.

Saputra, A. (2012). *WebTips: PHP, HTML5 dan CSS3*. Jakarta: Jasakom.

Tutorial Laravel (1) Apa itu Laravel? | emka.web.id. (n.d.). Retrieved May 21, 2014, from http://emka.web.id/tutorial/tutorial-laravel/2013/tutorial-laravel-1-apa-itu-laravel/

Tutorial Laravel Bahasa Indonesia: Kenapa Memilih Laravel? (n.d.). Retrieved May 11, 2014, from http://id-laravel.com/post/kenapa-memilih-laravel

What is web server? (n.d.). Retrieved May 12, 2014, from http://www.webdevelopersnotes.com/basics/what_is_web_server.php