

BAB III

TINJAUAN PUSTAKA

3.1 *Website*

Dalam penelitian ini *Website* adalah sejumlah halaman yang memiliki topik saling terkait, terkadang disertai pula dengan berkas gambar, video, atau jenis-jenis berkas lainnya. Sebuah *Website* biasanya ditempatkan setidaknya pada sebuah *Web Server* yang dapat diakses melalui jaringan seperti internet, ataupun jaringan wilayah lokal (LAN) melalui alamat internet yang dikenali sebagai Uniform Resource Locator (URL). *Website* berdasarkan teknologinya dibagi menjadi dua yaitu *Website* statis dan *Website* dinamis. *Website* statis adalah *Website* yang menampilkan informasi informasi yang bersifat tetap. Disebut statis karena pengguna tidak dapat berinteraksi dengan *Website* tersebut[1].

3.2 *Web Server*

Web Server adalah software yang menjadi tulang belakang dari world wide web (www) yang pertama kali tercipta sekitar tahun 1980 an. *Web Server* menunggu permintaan dari client yang menggunakan browser seperti Netscape Navigator, Internet Explorer, Mozilla Firefox, dan program browser Lainnya. Jika ada permintaan dari browser, maka *Web Server* akan memproses permintaan itu kemudian memberikan hasil prosesnya berupa data yang diinginkan Kembali ke browser[2].

Data ini mempunyai format yang standar, disebut dengan format SGML (Standard General Markup Language). Data yang berupa format ini kemudian akan ditampilkan oleh browser sesuai dengan kemampuan browser tersebut. Contohnya, bila data yang dikirim berupa

gambar, browser yang hanya mampu menampilkan teks (misalnya lynx) tidak akan mampu menampilkan gambar tersebut, dan jika ada akan menampilkan alternatifnya saja[2].

Web Server, untuk berkomunikasi dengan client-nya (web browser) mempunyai protokol sendiri, yaitu HTTP (hypertext transfer protocol). Dengan protokol ini, komunikasi antar *Web Server* dengan client-nya dapat saling dimengerti dan lebih mudah. Seperti telah dijelaskan diatas, format data pada world wide web adalah SGML. Tapi para pengguna internet saat ini lebih banyak menggunakan format *HTML* (hypertext markup language) karena penggunaannya lebih sederhana dan mudah dipelajari. Standarisasi — *Web Server* dalam — penerapan penggunaannya antara lain dikeluarkan oleh W3C (World Wide Web Consortium), IETF Internet Engineering Task Force), dan beberapa organisasi lainnya. Sampai saat ini, sudah lebih dari 110 spesifikasi yang dirilis oleh W3C (W3C Recommendations)[2].

3.3 PHP

PHP (atau resminya *PHP: Hypertext Preprocessor*) adalah skrip bersifat server – side yang ditambahkan ke dalam *HTML*. Pada prinsipnya server akan bekerja apabila ada permintaan dari client. Dalam hal ini client menggunakan kode-kode *PHP* untuk mengirimkan permintaan ke server. Sistem kerja dari *PHP* diawali dengan permintaan yang berasal dari halaman *Website* oleh browser. Berdasarkan URL atau alamat *Website* dalam jaringan internet, browser akan menemukan sebuah alamat dari *Web Server*, mengidentifikasi halaman yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh *Web Server*. Selanjutnya webserver akan mencarikan berkas yang diminta dan menampilkan isinya di browser. Browser yang mendapatkan isinya segera menerjemahkan kode *HTML* dan menampilkannya. Lalu bagaimana apabila yang dipanggil oleh *User* adalah halaman yang mengandung script *PHP*? Pada prinsipnya sama dengan memanggil

kode *HTML*, namun pada saat permintaan dikirim ke web-server, web-server akan memeriksa tipe file yang diminta *User*. Jika tipe file yang diminta adalah *PHP*, maka akan memeriksa isi script dari halaman *PHP* tersebut. Apabila dalam file tersebut tidak mengandung script *PHP*, permintaan *User* akan langsung ditampilkan ke browser, namun jika dalam file tersebut mengandung script *PHP*, maka proses akan dilanjutkan ke modul *PHP* sebagai mesin yang menerjemahkan script-script *PHP* dan mengolah script tersebut, sehingga dapat dikonversikan ke kode-kode *HTML* lalu ditampilkan ke browser *User*[3].

3.4 *Framework Laravel*

Framework adalah komponen pemrograman yang siap *re-use* (bisa digunakan ulang) kapan saja, sehingga programmer tidak harus membuat skrip yang sama untuk tugas yang sama. Misalkan programmer ingin halaman-halaman web menampilkan data dengan paginasi (paging) halaman, framework telah menyediakan fungsi paging tersebut sedangkan programmer cukup menggunakan fungsi tersebut pada saat coding, tetapi tentu dengan kaidah- kaidah yang ditetapkan oleh masing - masing *framework*. *Laravel* adalah sebuah *framework web* berbasis *PHP* yang *open-source* dan tidak berbayar, diciptakan oleh Taylor Otwell dan diperuntukkan untuk pengembangan aplikasi web yang menggunakan struktur pola MVC. Struktur pola MVC pada *laravel* sedikit berbeda pada struktur pola MVC pada umumnya. Di *laravel* terdapat routing yang menjembatani antara request dari *User* dan *Controller*. Jadi *Controller* tidak langsung menerima request tersebut [4]. Struktur pola MVC membagi *Laravel* menjadi tiga bagian yaitu model yang mengurus interaksi antar aplikasi dan database, view yang mengurus interaksi antara *tampilan* dengan *controller*, dan *controller* yang mengurus logika pemrograman. Dalam penggunaannya *Laravel* memiliki beberapa kekurangan salah satunya yaitu ukuran file yang cukup besar. Di dalam *laravel* terdapat file yang sifatnya *default* seperti *vendor*. File tersebut tidak boleh dihapus sembarangan sehingga ukuran *website* yang dibuat berukuran cukup besar. Selain itu,

dibutuhkan koneksi internet untuk *instalasi* dan mengunduh *library Laravel*[5]. *Framework Laravel* juga memiliki beberapa keunggulan sebagai berikut :

1. Menggunakan *Command Line Interface (CLI) Artisan*.
2. Menggunakan *package manager PHP Composer*.
3. Penulisan kode program lebih singkat, mudah dimengerti, dan ekspresif.
4. *Laravel* cocok untuk pemula yang baru ingin belajar menggunakan *framework* dengan Bahasa pemrograman *PHP* karena lebih mudah dimengerti dan memiliki arsitektur yang sederhana.

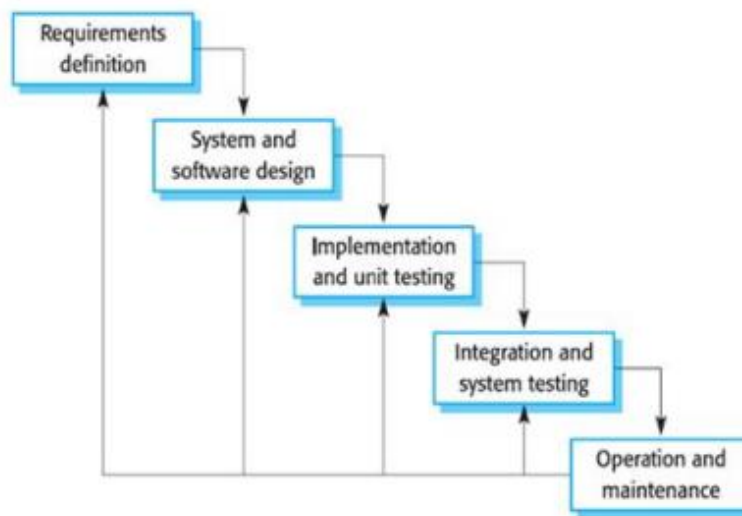
Fitur *framework Laravel* yang ditekankan pada penelitian ini adalah *Blade*, *Migration*, dan *controller*. Berikut adalah penjelasan mengenai ketiga fitur tersebut :

1. *Blade* adalah *template engine*. Pada dasarnya *Blade* adalah *view* namun dengan menggunakan *Blade* akan mempermudah untuk mengatur tampilan *website* dan menampilkan data. Cara untuk membuat file *view* menjadi file *Blade* adalah dengan menambahkan ekstensi *.blade.php* pada file *view*. Dan cara untuk memanggil file *Blade* sama dengan cara untuk memanggil file *view* biasa[6].
2. *Migration* adalah fitur yang menyediakan cara baru untuk membuat *database*. Dengan menggunakan *migration* cara membuat database melalui *Command Line Interface (CLI) database* atau dengan menggunakan aplikasi *database manager* digantikan dengan menggunakan *class*. Tahapan menggunakan *migration* adalah membuat *class* kemudian melakukan perintah *migrate* melalui *Command Line Interface (CLI) artisan*. Keuntungan menggunakan *migration* adalah *class* yang dibuat bisa dipakai untuk membuat database pada berbagai macam *Relation Database Management System (RDBMS)* yang didukung oleh *Laravel*. Keuntungan lain dari menggunakan *migration* adalah semua perubahan yang dilakukan pada *database* akan disimpan pada suatu tabel.

Sehingga bisa dilakukan pembatalan (*rollback*) pada *database* jika melakukan perubahan yang tidak benar[6].

3. *Controller* adalah suatu proses yang bertujuan untuk mengambil permintaan, menginisialisasi, memanggil *model* untuk dikirimkan ke *view*[5].

3.5 Metode *Waterfall*



Gambar 8 Metode *Waterfall*

Metode *Waterfall* memiliki tahapan-tahapan sebagai berikut[7] :

1. Requirements analysis and definition Layanan sistem, kendala, dan tujuan ditetapkan oleh hasil konsultasi dengan pengguna yang kemudian didefinisikan secara rinci dan berfungsi sebagai spesifikasi sistem.
2. System and software design Tahapan perancangan sistem mengalokasikan kebutuhan-kebutuhan sistem baik perangkat keras maupun perangkat lunak dengan membentuk arsitektur sistem secara keseluruhan. Perancangan perangkat lunak melibatkan identifikasi dan penggambaran abstraksi sistem dasar perangkat lunak dan hubungannya.

3. **Implementation and unit testing** Pada tahap ini, perancangan perangkat lunak direalisasikan sebagai serangkaian program atau unit program. Pengujian melibatkan verifikasi bahwa setiap unit memenuhi spesifikasinya.
4. **Integration and system testing** Unit-unit individu program atau program digabung dan diuji sebagai sebuah sistem lengkap untuk memastikan apakah sesuai dengan kebutuhan perangkat lunak atau tidak. Setelah pengujian, perangkat lunak dapat dikirimkan ke customer.
5. **Operation and maintenance** Biasanya (walaupun tidak selalu), tahapan ini merupakan tahapan yang paling panjang. Sistem dipasang dan digunakan secara nyata. Maintenance.

3.6 HTML

HTML adalah suatu bahasa pemrograman yang digunakan untuk pembuatan halaman *Website* agar dapat menampilkan berbagai informasi baik tulisan maupun gambar pada sebuah web browser. Saat ini bahasa *HTML* masih terus dikembangkan. Hal ini dikarenakan pengguna internet semakin hari semakin berkembang pesat. Oleh karena itu bahasa *HTML* harus ditingkatkan lagi agar bisa menciptakan halaman web yang lebih berkualitas. Untuk itulah dibentuk organisasi yang bertanggung jawab mengembangkan bahasa *HTML* organisasi ini bernama W3C[8].

Seiring dengan pesatnya perkembangan zaman *HTML* pun berkembang dari masa ke masa. *HTML* pertama kali diciptakan oleh IBM pada tahun 1980 dengan dibentuknya suatu program untuk melakukan pemformatan dokumen secara otomatis dari susunan elemen elemen tag[8].

3.7 CSS

CSS merupakan singkatan dari Cascading Style Sheet. Kegunaannya adalah untuk mengatur tampilan dokumen *HTML*, contohnya seperti pengaturan jarak antar baris, teks, warna dan

format border bahkan penampilan file gambar. Cascading Style Sheets (CSS) adalah suatu bahasa pemrograman yang digunakan untuk mendukung pembuatan *Website* agar memiliki tampilan yang lebih menarik dan terstruktur. CSS dikembangkan oleh W3C. organisasi yang mengembangkan teknologi internet. Tujuannya tidak lain untuk mempermudah proses penataan halaman web[8].

3.8 BOOTSTRAP

Bootstrap adalah kerangka kerja CSS yang bersifat open source dan digunakan untuk kebutuhan pembuatan tampilan desain visual dari aplikasi web atau situs *Website*. Kerangka kerja yang digunakan berbentuk template desain berbasis *HTML* dan *CSS* untuk kebutuhan pengembangan navigasi, tombol, tipografi, formulir, dan komponen antarmuka yang lainnya. Bootstrap terus mengalami perbaikan dan pembaharuan sistem untuk memberikan layanan atau fitur yang lebih kompleks. Sehingga, dapat membantu mengatasi permasalahan para developer serta mempermudah proses pengembangan produk dari sisi coding[9].

3.9 MySQL

MySQL merupakan software database open source yang paling populer di dunia, dimana saat ini digunakan lebih dari 100 juta pengguna di seluruh dunia. Dengan kehandalan, kecepatan dan kemudahan penggunaannya, MySQL menjadi pilihan utama bagi banyak pengembang software dan aplikasi baik di platform web maupun desktop. Proses menggunakan MySQL pada dasarnya adalah mengelola data dan informasi agar data dan informasi tersimpan dengan tertata dan rapih, proses-proses yang sering terjadi biasanya adalah membuat database, membuat sebuah table, memodifikasi struktur sebuah table, mengisikan data dalam sebuah table, menghapus data dalam sebuah table, memodifikasi (merubah atau mengedit) data dalam sebuah table dan mencari data dalam sebuah table[10]

3.10 *Javascript*

Javascript merupakan bahasa scripting yang populer di internet dan dapat bekerja di sebagian besar browser populer seperti Internet Explorer (IE), Mozilla Firefox, Netscape dan Opera. Kode *Javascript* dapat disisipkan dalam halaman web menggunakan tag SCRIPT. *Javascript* adalah bahasa pemrograman web yang bersifat Client Side Programming Language. Client Side Programming Language adalah tipe bahasa pemrograman yang pemrosesannya dilakukan oleh client. Bahasa pemrograman Client Side berbeda dengan bahasa pemrograman Server Side seperti *PHP*, dimana untuk server side seluruh kode program dijalankan di sisi server[8].

BAB IV

PEMBAHASAN

Hasil dari penelitian ini adalah berupa *chart* yang dapat *memonitoring* ketika *user* melakukan *login* kedalam sistem dan melihat jumlah *user* yang bertambah. Penerapan *Laravel* diterapkan dengan menggunakan konsep *MVC(Model View Controller)* dimana *Model* berfungsi untuk berhubungan dengan *database*, *view* yang berhubungan dengan tampilan website dengan menggunakan *template engine* dari *Laravel* yaitu *blade* dimana dengan menggunakan *blade* kode-kode yang digunakan akan dipisah menjadi kesatuan yang lebih kecil yang kemudian nantinya akan menerima data dari *controller*. Pada *controller* dilakukan pembuatan *function* untuk mengolah data yang akan ditampilkan ke *template engine* dengan pengambilan data yang diambil dari *model* yang telah dibuat melalui *migration*. Kemudian data-data tersebut dapat ditambah, dihapus dan diedit.

4.1 Analisis Kebutuhan

1. Analisis Kebutuhan dilakukan untuk memberikan gambaran mengenai permasalahan dan prosedur. Pada PT. Kazee Digital Indonesia terdapat banyak *User* yang tersebar di seluruh indonesia yang menggunakan produk dari PT. Kazee Digital Indonesia. Oleh karena itu diperlukan suatu sistem yang bisa memonitoring *user*. Dalam perancangan sistem diarahkan langsung oleh pembimbing lapangan yang menginginkan sebuah sistem dimana admin dapat memantau perkembangan *user* atau client perhari dan melihat jumlah dari *user* atau client yang aktif perharinya yang dipresentasikan melalui line chart. Pada sistem yang dibuat hanya membuat Line chart dimana sistem login pada projek ini sebelumnya telah disediakan oleh perusahaan. Pada sistem terdapat admin yang dapat memiliki berbagai hak seperti mencetak data, melihat total *User* perhari dan melihat total aktif *user* perhari. Pembuatan sistem ini diarahkan dengan menggunakan framework *PHP* yaitu *Laravel* dengan *chart* menggunakan *highchart*

yang merupakan *library* dari *Javascript* yang disertai dengan pengelolaan *Database* yang dapat dilakukan oleh admin untuk menambah, menghapus serta mengedit data.

2. Kebutuhan Fungsional :

- a. Melihat Dashboard *User* Aktif Perhari
- b. Melihat Dashboard Pertambahan *User* Per Hari

3. Kebutuhan Non Fungsional :

- a. Antarmuka yang dibuat merupakan antarmuka yang sederhana
- b. Proses dari pengguna dari setelah melakukan login sampai dengan menampilkan linechart berlangsung tidak lebih dari 5 detik.

4.2 Desain Sistem

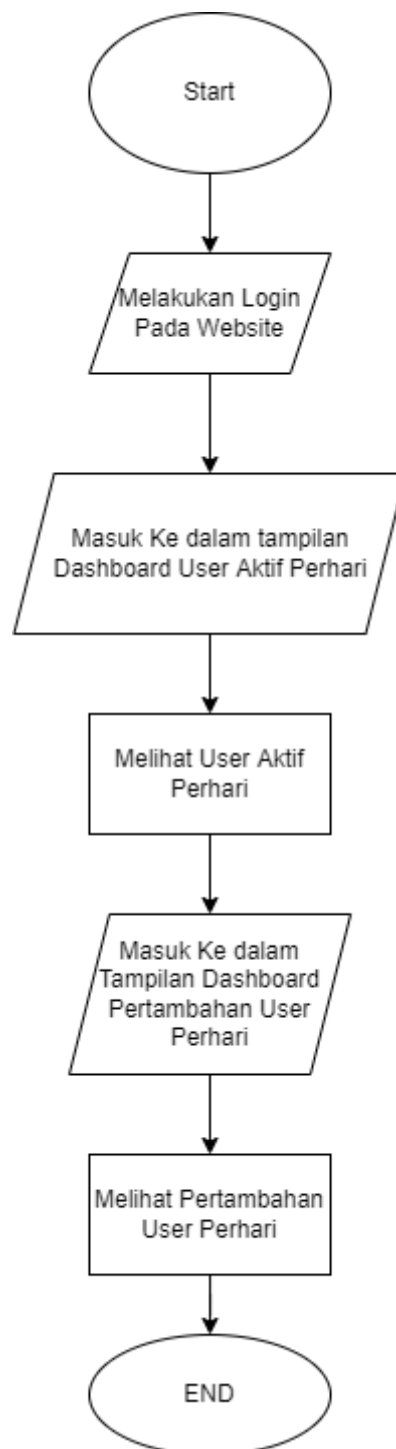
4.2.1. Use Case Diagram



Gambar 9 Use Case Diagram

Pada Gambar 9 merupakan *usecase* dari sistem *Monitoring User*. *Usecase* tersebut menjelaskan *usecase* yang dapat dilakukan oleh admin. Pada *usecase* tersebut admin dapat melakukan hal seperti, melihat data *User* aktif melalui chart, melihat pertumbuhan *User* melalui chart, mengelola *Database* dan mencetak data *User*.

4.2.2. Flowchart



Gambar 10 Flowchart Sistem Monitoring User

Pada Gambar 10 yaitu *Flowchart* sistem *Monitoring User* yang mana dimulai dengan simbol terminator yang digunakan untuk memulai dan mengakhiri proses alur *Flowchart* kemudian pada tahap selanjutnya yaitu login ke *Website* dengan memasukkan email dan

password. Kemudian dilanjutkan dengan dialihkan ke halaman awal dashboard dan memilih dua menu yaitu pertama memilih menu dashboard *User* aktif per hari kemudian memilih menu dashboard penambahan *User* per hari kemudian *Flowchart* berakhir

4.2.3. Definisi Aktor

Table 1 Definisi Aktor

No	Aktor	Deskripsi
1.	Admin	Admin mempunyai wewenang untuk melihat <i>User</i> aktif dan penambahan <i>User</i> serta mencetak data

4.2.4 Definisi Use Case

Table 2 Definisi Use Case

No	Use Case	Deskripsi
1.	Melihat Halaman Dashboard <i>User</i> Aktif Perhari	Admin dapat melihat halaman dashboard <i>User</i> aktif perhari dalam bentuk chart
2.	Melihat Halaman Dashboard Pertambahan <i>User</i> Per Hari	Admin dapat melihat halaman dashboard penambahan per hari dalam bentuk chart
3.	Mencetak Data	Admin dapat mencetak data dari dashboard total aktif <i>User</i> perhari dan penambahan <i>User</i> perhari
4.	Mengelola <i>Database</i>	Admin dapat mengelola <i>Database</i> untuk melakukan penambahan data, pengurangan data, serta pengubahan data

4.2.5 Skenario *Use Case*

Desain berikut ini menggambarkan skenario urutan interaksi antara admin dengan *Use Case* :

Nama *Use Case* : Melihat Halaman Dashboard *User* Aktif Perhari

Skenario :

Table 3 Skenario Use Case Melihat Halaman Dashboard User Aktif Perhari

Work Order	Print Data
Ringkasan	Admin Melihat Halaman Dashboard <i>User</i> Aktif Perhari
Rasional	<i>Usecase</i> agar Admin dapat Melihat Halaman Dashboard <i>User</i> Aktif Perhari
Pengguna	Admin
Pre Kondisi	Admin telah melakukan login dan berada di dashboard
Aliran Dasar	1. Memilih dashboard 2. Masuk ke menu Dashboard <i>User</i> Aktif Perhari
Aliran Alternatif	-
Post Kondisi	Menampilkan <i>chart user</i> aktif perhari

Nama Use Case : Melihat Halaman Dashboard *User* Aktif Perhari

Skenario :

Table 4 Skenario Use Case Melihat Halaman Dashboard Pertambahan *User* Per Hari

Work Order	Print Data
Ringkasan	Admin Melihat Halaman Dashboard Pertambahan <i>User</i> Per Hari.
Rasional	<i>Usecase</i> agar Admin dapat Melihat Halaman Dashboard Pertambahan <i>User</i> Per Hari
Pengguna	Admin
Pre Kondisi	Admin telah melakukan login dan berada di dashboard
Aliran Dasar	1. Memilih dashboard 2. Masuk ke menu Dashboard Pertambahan <i>User</i> Per Hari
Aliran Alternatif	-
Post Kondisi	Menampilkan <i>chart</i> Pertambahan <i>User</i> Per Hari

Nama Use Case : Mencetak Data

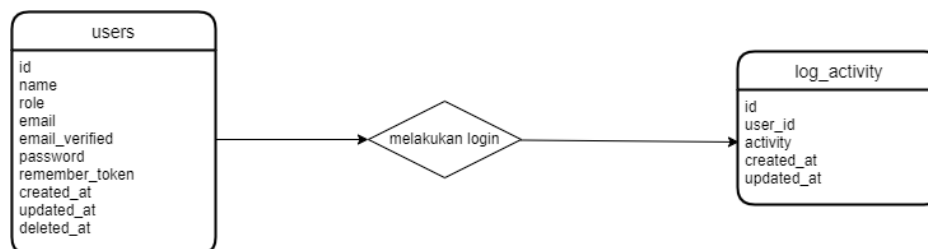
Skenario :

Table 5 Skenario Use Case Mencetak Data

Work Order	Print Data
Ringkasan	Admin Mencetak Data

Rasional	<i>Usecase</i> agar Admin dapat mencetak data Order
Pengguna	Admin
Pre Kondisi	Admin telah melakukan login dan berada di dashboard
Aliran Dasar	3. Memilih dashboard 4. Mencetak data dashboard dalam bentuk chart
Aliran Alternatif	-
Post Kondisi	Menampilkan Unduhan File

4.2.6 Database



Gambar 11 ERD Sistem Monitoring User

Pada Gambar 11 merupakan Entity Relationship Diagram. ERD tersebut memiliki dua entitas yaitu *Users* dan *log_activity*. *Users* merupakan entitas yang akan melakukan login kedalam sistem sedangkan *log_activity* merupakan entitas ketika menyimpan kegiatan *User* ketika melakukan login. Entitas tersebut memiliki satu relasi yaitu melakukan login.

Table 6 Atribut Entitas

No.	Nama	Attribute	Deskripsi
1.	<i>Users</i>	<ol style="list-style-type: none"> 1. <i>id</i> 2. <i>name</i> 3. <i>role</i> 4. <i>email</i> 5. <i>email_verified</i> 6. <i>password</i> 7. <i>remember_token</i> 8. <i>created_at</i> 9. <i>updated_at</i> 10. <i>deleted_at</i> 	Table akun yang akan login ke sistem dan admin yang akan memantau
2.	log_activity	<ol style="list-style-type: none"> 1. <i>id</i> 2. <i>User_id</i> 3. <i>activity</i> 4. <i>created_at</i> 5. <i>deleted_at</i> 	Table untuk mencatat aktivitas <i>User</i>

4.3 Implementasi

4.3.1 Rancangan Website

Dalam perancangan sistem, terdapat tiga Routing yang dirancang sesuai dengan fungsi masing-masing Routing tersebut. Routing tersebut terdiri dari :

1. *ROUTE LogActivity*

```
Route::get('/users', [LogActivityController::class, 'index']);
```

Gambar 12 ROUTE LogActivity

Pada Gambar 12 diatas merupakan Routing LogActivity. Routing tersebut menggunakan METHOD GET karena digunakan untuk get(mengambil data). Routing tersebut berfungsi untuk melihat total data *User* yang aktif dalam satu hari dengan menggunakan fungsi sebagai berikut ini :

```
public function index(){

    $users = DB::select(("SELECT `name` FROM `users`"));
    $totaluser = DB::select(("SELECT DATE_FORMAT(created_at, '%Y-%m-%d') AS date_activity,
                                created_at, user_id AS total_user
                                FROM `log_activity`
                                GROUP BY date_activity
                                ORDER BY date_activity"));

    $categories = [];
    $values = [];
    foreach ($totaluser as $val){
        $categories [] = date( 'd/m/Y', strtotime($val->date_activity));
        $values [] = $val->total_user;
    }
    $series = [
        'name' => $users
    ];
    // dd($values);
    $categories = json_encode($categories);
    $values = json_encode($values);
    $series = json_encode($series);
    return view('chart.usertotal', compact('totaluser','categories','values','series'));
}
```

Gambar 13 Controller Total Online User

Pada Gambar 13 diatas merupakan *Controller Total Online User* yang digunakan untuk melihat total data *User* yang aktif dalam satu hari. Data-data *User* yang aktif lebih dari satu kali dalam sehari akan tetap dihitung menjadi satu dengan menggunakan query `DISTINCT` pada fungsi diatas. Kemudian data tersebut akan dikelompokkan berdasarkan `date_activity` yang merupakan tanggal dari aktivitas tersebut terjadi dan disusun berdasarkan `date_activity` juga sehingga akan berurut dari tanggal lama ke yang terbaru. Kemudian data tersebut akan dikirim dalam bentuk JSON dengan menggunakan syntax `json_encode` dan diterima di Client Side. Kemudian data tersebut diubah dari bentuk JSON kedalam bentuk Object sehingga bisa tampil pada chart.

2. *Route TotalUser*



Gambar 14 Route TotalUser

Pada gambar 14 diatas merupakan gambar *Route TotalUser* yang menggunakan METHOD GET karena digunakan untuk `get`(mengambil data). Route tersebut berfungsi untuk menampilkan penambahan data *User* perhari dengan menggunakan fungsi sebagai berikut :

```

public function index(){
    $totaluser = DB::select("SELECT DATE_FORMAT(created_at, '%Y-%m-%d') AS date_activity,
                           created_at, COUNT(id) AS total_user
                           FROM `users`
                           GROUP BY date_activity");

    $categories = [];
    $values = [];

    foreach ($totaluser as $val){
        $categories [] = date( 'd/m/Y', strtotime($val->date_activity));
        $values [] = $val->total_user;
    }

    $categories = json_encode($categories);
    $values = json_encode($values);

    return view('chart.usertotal', compact('totaluser','categories','values'));
}

```

Gambar 15 Controller Total Data User

Pada gambar 15 diatas merupakan Total Data User. Fungsi dari fungsi tersebut adalah untuk mengelompokan data User yang terdaftar dalam satu hari. Kemudian diurutkan berdasarkan tanggal dibuatnya akun tersebut. Data tersebut kemudian diubah kedalam bentuk JSON yang akan dikirimkan ke Client Side dan dilakukan parsing ke dalam bentuk object sehingga dapat ditampilkan dalam bentuk chart.

4.3.2 Pembuatan Log Login

Pada tahap ini merupakan tahap pembuatan Log Login. Log Login digunakan untuk melihat User yang aktif ketika melakukan login ke sistem. Berikut merupakan Source Code dari Log Login :

```
protected function authenticated(Request $
request, $user)
{
    $activity = new LogActivity();
    $activity->user_id = Auth::id();
    $activity->activity = "login";
    $activity->save();
    $activity->created_at = date("Y-m-d H:i:s"
);
}
```

Gambar 16 Log Login

4.3.3 Pembuatan Tampilan *Dashboard Monitoring User*

Pada tahap ini merupakan tahap design *Dashboard Monitoring User*. Design dibuat sesuai dengan permintaan pembimbing lapangan berikut merupakan *Source Code* design yang dibuat:

```

@section('js')
<script type="text/javascript">
    const categories = JSON.parse('
{!! $categories !!}');
    const data = JSON.parse('{!! $values !!}');
);
Highcharts.chart('container', {
    title: {
        text: 'Pengguna Aktif'
    },
    yAxis: {
        title: {
            text: 'Total Pengguna'
        }
    },
    xAxis: {
        categories
    },
    legend: {
        enabled: false
    },
    credits: {
        enabled: false
    },
    plotOptions: {
        series: {
            label: {
                connectorAllowed: false
            }
        }
    },
    series: [{
        data
    }],
    responsive: {
        rules: [{
            condition: {
                maxWidth: 500
            },
            chartOptions: {
                legend: {
                    layout: 'horizontal',
                    align: 'center',
                    verticalAlign: 'bottom'
                }
            }
        }]
    }
});
</script>
@endsection

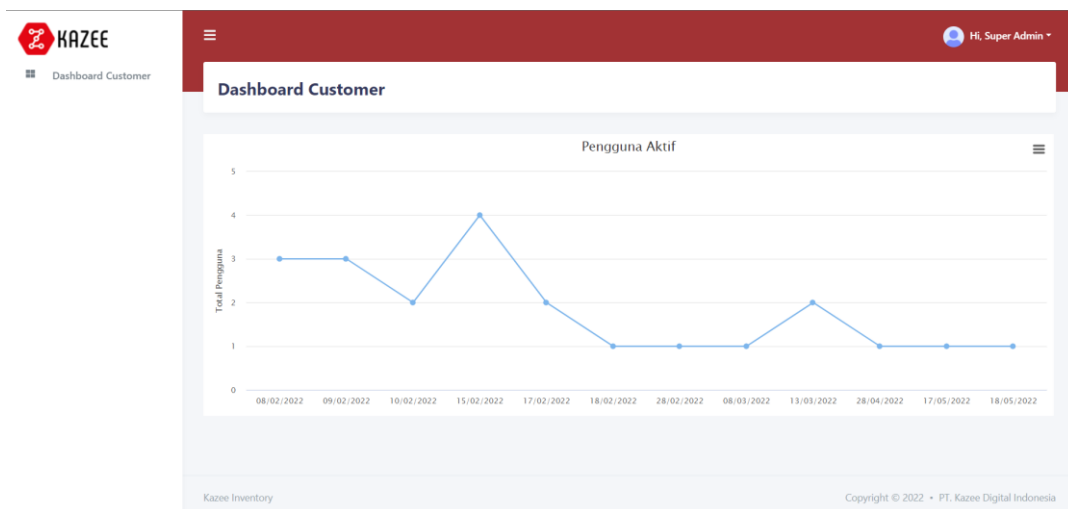
```

Gambar 17 Source Code Tampilan Monitoring User

Pada gambar 17 diatas merupakan *Source Code* tampilan *Monitoring User* menggunakan *HTML 5* dengan plugin *HighChart* yang telah dipisah menjadi lebih sederhana dengan menggunakan `@section` pada blade *Laravel* yang kemudian akan dikirimkan bagian tersebut ke dashboard utama yang berisi code-code *HTML*.

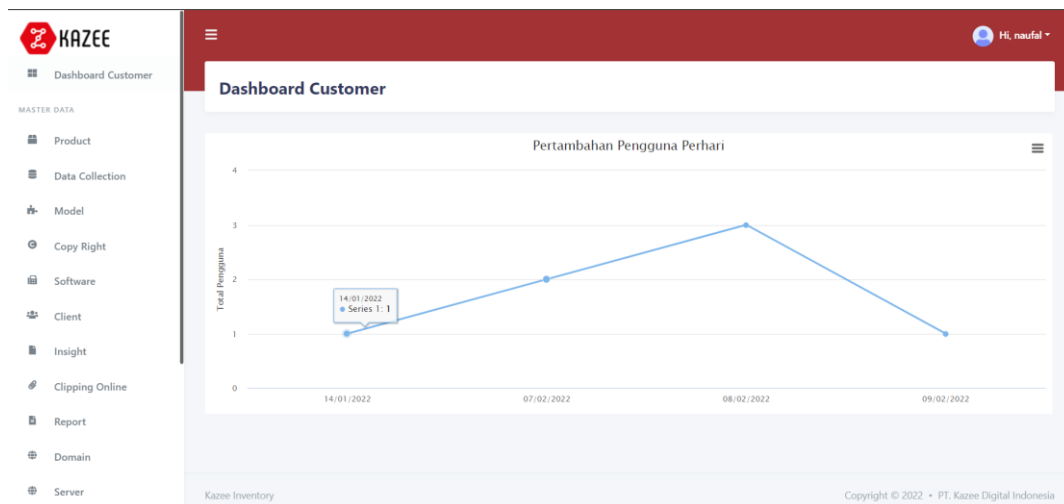
4.4 Hasil

Hasil dari pembuatan dari tahap implementasi diatas menghasilkan satu tampilan *Website* yang terdiri dari dua menu yaitu menu Pengguna Aktif dan Pertambahan pengguna perhari. Hasil ini berasal dari akses web route pada code program Laravel yang dibuat. Sebelum admin dapat melihat dua menu yang diimplementasikan tersebut terdapat login yang dilakukan untuk mengautentikasi bahwa yang login ke sistem merupakan seorang admin atau super admin.



Gambar 18 Client Side Interface Total Online User

Pada gambar 18 merupakan gambar *Client Side Interface Total Online User* yang digunakan oleh admin untuk melihat pengguna yang aktif dengan url <http://127.0.0.1:8000/Users>



Gambar 19 Client Side Interface Total User

Pada gambar 19 merupakan gambar *Client Side Interface Total User* yang digunakan oleh admin untuk melihat pengguna yang aktif dengan url <http://127.0.0.1:8000/Userstotal>

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil kerja praktek yang dilakukan maka dapat diambil kesimpulan sebagai berikut:

1. *Website* yang dibuat telah sesuai dengan permintaan dan kebutuhan dari PT. Kazee Digital Indonesia yang terlihat dari hasil implementasi.
2. *Website* dapat menampilkan data-data dari *User* yang telah melakukan login sehingga dapat dihitung total *User* yang login dalam satu hari tersebut.
3. *Website* telah dapat menampilkan total pengguna yang mengguna aplikasi tersebut yang diambil dari total *User* yang ada.

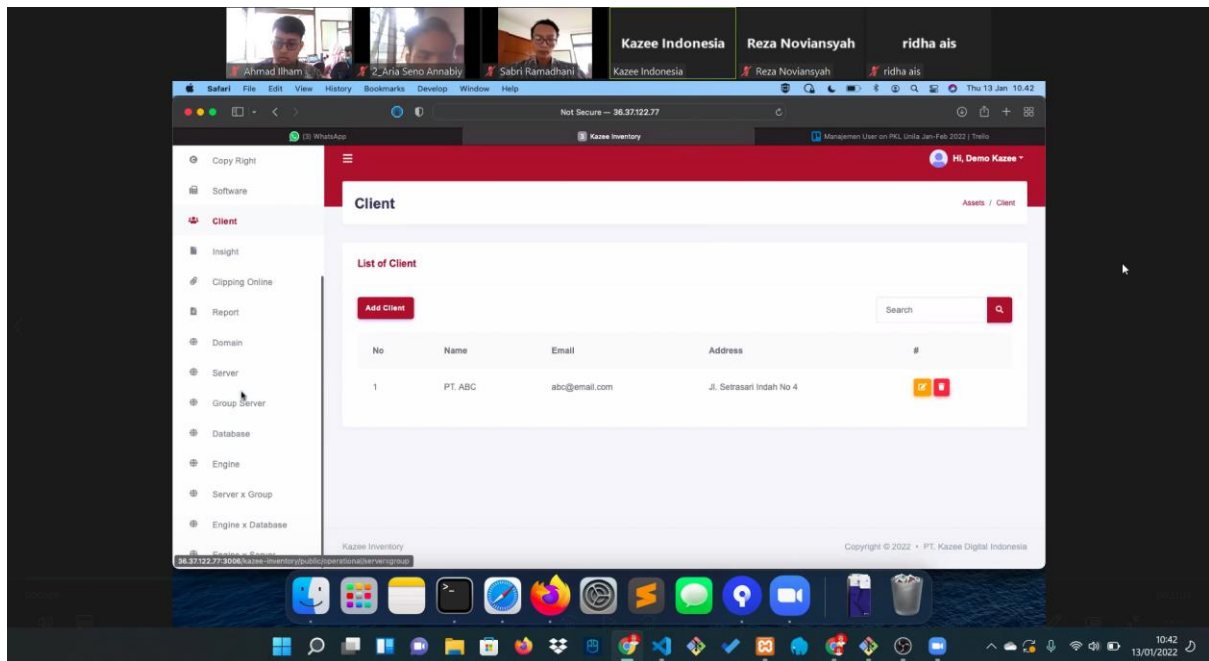
5.2 Saran

Berdasarkan hasil kerja praktek yang telah dilakukan maka didapatkan beberapa saran antara lain sebagai berikut:

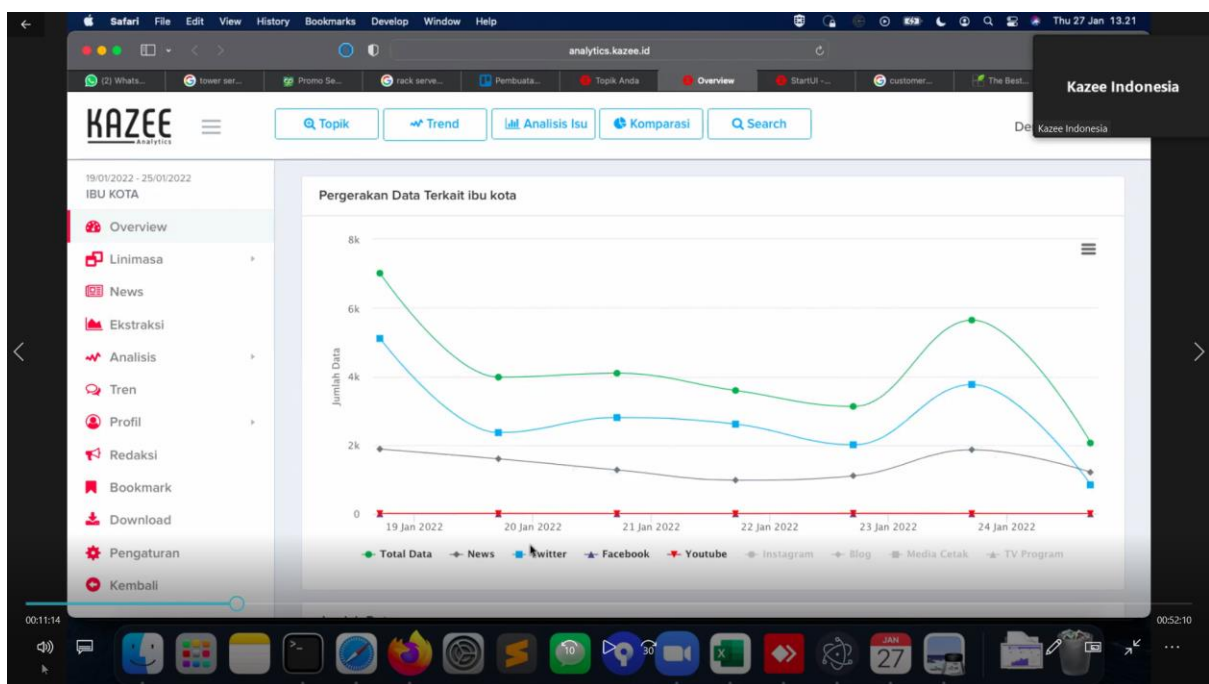
1. Membuat Filter Data Agar Memudahkan Admin Untuk Melihat Dashboard Tersebut.
2. Function pada setiap Task lebih baik digabung ssja menjadi satu *Controller*
3. Untuk penelitian lebih lanjut lebih baik menggunakan Laravel versi terbaru

- [1] Y. Muharam, M. Kom, and A. Agustiasri, “MEMBANGUN WEBSITE SEKOLAH DENGAN MENGGUNAKAN FRAMEWORK LARAVEL 7 UNTUK MEDIA SARANA NFORMASI (STUDI KASUS SMP NURUL HALIM WIDASARI DI KAB.INDRAMAYU),” 2021.
- [2] E. Nurmiati, “Analisis dan Perancangan Web Server Pada Handphone,” *Jurnal Sistem Informasi*, 2012.
- [3] A. Firman, H. F. Wowor, and X. Najosan, “Sistem Informasi Perpustakaan Online Berbasis Web,” 2016.
- [4] D. Purnama Sari, R. Wijanarko, and J. X. Menoreh Tengah, “Implementasi Framework Laravel pada Sistem Informasi Penyewaan Kamera (Studi Kasus Di Rumah Kamera Semarang),” *Jurusan Teknik Informatika, Fakultas Teknik, Universitas Wahid Hasyim*, vol. 2, no. 1, pp. 32–36, 2019.
- [5] D. Mediana and A. I. Nurhidayat, “RANCANG BANGUN APLIKASI HELPDESK (A-DESK) BERBASIS WEB MENGGUNAKAN FRAMEWORK LARAVEL (STUDI KASUS DI PDAM SURYA SEMBADA KOTA SURABAYA),” Surabaya, 2018.
- [6] A. Irawan, “FRAMEWORK LARAVEL UNTUK INFORMASI PENUNJANG PERKULIAHAN,” 2017.
- [7] G. W. Sasmito, J. T. Informatika, H. Bersama, J. Mataram, N. 09, and P. Lor, “Penerapan Metode Waterfall Pada Desain Sistem Informasi Geografis Industri Kabupaten Tegal,” *Jurnal Pengembangan IT(JPIT)*, vol. 2, no. 1, 2017, [Online]. Available: <http://www.tegalkab.go.id>,
- [8] Setiawan Antonius Andy, Lumenta S.M Arie, and Sompie R.U.A Sherwin, “RANCANG BANGUN APLIKASI UNSRAT E-CATALOG,” *Jurnal Teknik Informatika*, 2019.
- [9] A. Zakir, “RANCANG BANGUN RESPONSIVE WEB LAYOUT DENGAN MENGGUNAKAN BOOTSTRAP FRAMEWORK,” 2016. [Online]. Available: www.malasngoding.com
- [10] A. Saputra, P. Bidang, and T. Pengamatan, “Manajemen Basis Data MYSQL pada situs FTP LAPAN BANDUNG,” 2012.

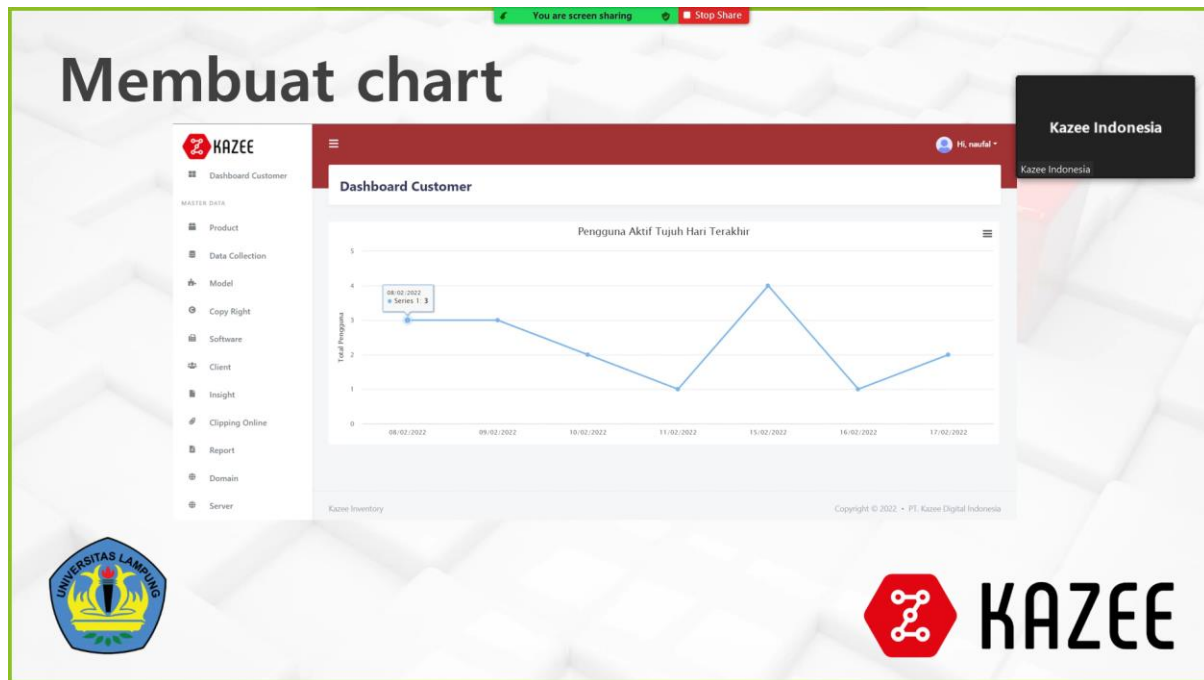
LAMPIRAN



Gambar 20 Pembagian Project



Gambar 21 Penyampaian Task



Gambar 22 Presentasi Akhir



Gambar 23 Foto bersama pembimbing lapangan

NO	KEGIATAN	DESEMBER				JANUARI				FEBRUARI			
		W1	W2	W3	W4	W1	W2	W3	W4	W1	W2	W3	W4
1	1.1 Mempelajari Framework Laravel												
	1.2 Membuat Dashboard Monitoring												
	1.3 Membuat Dashboard Monitoring dengan data dummy												
2	2.1 Pengarahan pergantian project												
	2.2 Membuat mockup monitoring user												
	2.3 Pengimplementasian mockup												
3	3.1 Membuat log login												
	3.2 Membuat query total aktif user												
	3.3 Membuat query perkembangan user												
4	4.1 Penyesuaian chart dengan data												

Gambar 24 Timeline Project

KEMENTERIAN RISET, TEKNOLOGI, DAN PENDIDIKAN TINGGI
UNIVERSITAS LAMPUNG
FAKULTAS TEKNIK-UNIVERSITAS LAMPUNG
PROGRAM STUDI TEKNIK INFORMATIKA
Jln. Soemantri Brojonegoro No. 1 Bandarlampung 35145
Telp.(0721)704947 Fax.(0721)704947
Laman : <https://eng.unila.ac.id>

Bandung, 21 Februari 2022

Nomor :
Perihal : Daftar Nilai Pembimbing Lapangan


Nama : Ahmad Ilham
NPM : 1915061059
Tempat Kerja Praktik : PT Kasee Digital Indonesia
Judul Kerja Praktik : Pengembangan Dashboard Monitoring User dengan Chart Menggunakan Framework Laravel 7

No.	Faktor Yang Dinilai	Bobot	Nilai
1.	Kerajinan Dilapangan	20%	85
2.	Kerja Sama	20%	90
3.	Kemampuan Melaksanakan Tugas	20%	88
4.	Kreatifitas	20%	85
5.	Kemampuan Berkomunikasi	20%	85

Keterangan Konversi Nilai :

Nilai	Huruf Mutu
> 76	A
71 - < 76	B+
66 - < 71	B
61 - < 66	C+
56 - < 61	C
50 - < 56	D
< 50	E

Bandung, 21 Februari 2022
Pembimbing Lapangan Kerja Praktik,

 **KAZEE**
KAZEE DIGITAL INDONESIA
M. Rizki Samsul Ariefin, S.Kom.
Chief Technology Officer

Gambar 25 Nilai KP