

BAB I

TATA CARA PENGEMBANGAN APLIKASI

3.1 Metodologi Pengembangan Aplikasi

Metodologi Pengembangan Aplikasi Driver Online Transportation Terjadwal untuk Operator Platform dilakukan melalui beberapa tahapan sebagai berikut :

1. Analisa Kebutuhan. Pada tahap ini dilakukan evaluasi permasalahan yang dihadapi operator.
2. Kebutuhan Fungsional. Menentukan *feature* yang dibutuhkan guna mengatasi permasalahan yang dihadapi operator.
3. Mendefinisikan Database yang Dibutuhkan. Pada tahap ini akan ditentukan database yang paling sesuai digunakan.
4. Mendefinisikan Implementasi dan Pengujian. Tahap implementasi merupakan penjelasan dari proses implementasi yang kemudian diikuti oleh tahap pengujian sistem.

3.2 Analisa Kebutuhan

Aplikasi Driver Online Transportation Terjadwal untuk Operator Platform dikembangkan untuk membantu kinerja operator sehingga laju bisnis berjalan dengan lebih terorganisir. Terkait dengan hal tersebut, operator membutuhkan :

1. Operator membutuhkan sebuah jadwal untuk menampilkan data order.
2. Data yang dibutuhkan merupakan data yang terstruktur.
3. Data tersebut harus terstruktur berdasarkan *driver*.
4. Data yang dibutuhkan berasal dari order-order yang sudah masuk.
5. Data yang dibutuhkan merupakan data yang telah tersimpan di dalam database.

6. Operator membutuhkan jadwal tersebut untuk ditampilkan dengan jangka per minggu.

Dalam order yang ditampilkan pada aplikasi terjadwal ini terdapat beberapa data yang dibutuhkan. *Order title* digunakan untuk ditampilkan langsung pada jadwal. Sementara *client*, *driver* serta waktu dan tempat ditampilkan di rincian order.

3.3 Kebutuhan Fungsional

Kebutuhan fungsional dapat ditentukan berdasarkan hasil dari analisa kebutuhan. Setelah menentukan apa saja kebutuhan yang diinginkan maka kebutuhan fungsional dapat ditentukan. Kebutuhan fungsional dari aplikasi penjadwalan yang akan dibutuhkan adalah :

1. Sistem dapat melakukan pengambilan data dari database.
2. Sistem hanya menyediakan data yang diinginkan.
3. Dapat mengambil data dalam rentang waktu 1 minggu dimulai dari hari senin di minggu tersebut.
4. Dapat mengeluarkan data dengan terstruktur.

3.4 Mendefinisikan Database yang Digunakan

Data yang dibutuhkan untuk aplikasi penjadwalan ini ditentukan oleh data order yang sebelumnya telah masuk ke sistem database dalam bentuk *collection orders*. Data yang tersimpan dalam tabel trip ada banyak macam, mulai dari *id trip* yang memiliki tipe data *Object ID* hingga lokasi penjemputan bertipe koordinat. Setiap kali order selesai *rostering* maka secara otomatis data akan mendapat *id trip* dan tersimpan di tabel *trip*.

MongoDB mendukung pengoperasian data dalam bentuk geospasial. Yang artinya ketika sebuah lokasi di peta dipilih, format data ini tidak perlu melalui proses konversi ke bentuk data yang lain agar dapat disimpan langsung ke dalam database. Proses tanpa konversi ini yang membuat MongoDB dipilih sebagai database yang sesuai untuk pengoperasian aplikasi.

3.5 Mendefinisikan Implementasi dan Pengujian

Pada tahap implementasi dilakukan proses pengembangan sistem berdasarkan komponen-komponen yang telah ditentukan. Data dari MongoDB diambil menggunakan *service* yang terbentuk dari framework Express.js. Kemudian agar proses pengiriman data terjadi lebih cepat maka digunakan *path* sebagai perantara yang membawa data langsung ke program. Seluruh proses yang ada di dalam program menggunakan Bahasa Node.js serta semua library yang digunakan adalah library yang berjalan dibawah *Node Package Manager* (NPM). Data yang telah masuk akan disortir oleh Moment.js hingga menghasilkan output berdasarkan waktu satu minggu. Output ini akan diseleksi terus menerus hingga menjadi hasil yang diinginkan. Setelah itu program akan memanggil data-data yang lolos dan melakukan rekonstruksi hingga menjadi output yang lebih terstruktur. Output terstruktur dari program tersebut akan dikirim kepada *frontend* dalam bentuk JSON untuk selanjutnya diolah menjadi tampilan pada halaman web.

Proses pengujian aplikasi dilakukan secara manual pada setiap step. Disini dilakukan *debugging* dan *running program* untuk mengetahui apakah setiap proses berjalan dengan benar, sesuai dengan alur proses yang diharapkan, serta apakah aplikasi berjalan dengan baik, dan memastikan data yang dikirim telah sesuai dan terstruktur dengan baik.