**DOKUMENTASI PERANGKAT LUNAK**

**ENTERPRISE RESOURCE PLANNING (ERP) REBORN**



**Versi : 1.00.00.0000**

|  |  |
| --- | --- |
|  | **DAFTAR ISI** |
|  |

**Daftar Isi** Lst-Cnt-1

**Daftar Tabel** Lst-Tbl-1

**Daftar Gambar** Lst-Drw-1

**BAB 1. RENCANA PENGELOLAAN PROYEK PERANGKAT LUNAK** Cnt.1-1

I.1. Pendahuluan Cnt.1-1

I.1.1. Gambaran Proyek Cnt.1-1

I.1.2. Dokumen Dalam Proyek Cnt.1-1

I.1.3. Material Acuan Cnt.1-4

I.1.4. Definisi, Akronim, dan Singkatan Cnt.1-4

I.2. Organisasi Proyek

I.2.1. Model Proses

I.2.2. Struktur Organisasi

I.2.3. Batasan dan Antar Muka Organisasi

I.2.4. Ruang Lingkup dan Tanggung Jawab

I.2.4.1. Penjelasan

I.2.4.1.1. Manager

I.2.4.1.2. Konsultan

I.2.4.1.3. *Front End Developer Coordinator*

I.2.4.1.4. *Front End Developer*

I.2.4.1.5. *Back End Developer Coordinator*

I.2.4.1.6. *Back End Developer*

I.2.4.1.7. *Database Developer Coordinator*

I.2.4.1.8. *Database Developer*

I.2.4.1.9. *Quality Assurance & Documentation Coordinator*

I.2.4.1.10. *Quality Assurance & Documentation*

I.2.4.1.11. *System & Infrastructure Developer*

I.3. Proses Manajerial

I.3.1. Tujuan dan Prioritas Manajemen

I.3.1.1. Prioritas Jadwal

I.3.1.2. Anggaran

I.3.2. Manajemen Risiko

I.3.2.1. Dampak Negatif

I.3.3. Mekanisme Pengawasan dan Pengendalian

I.3.3.1. Pengawasan

I.3.3.2. Pengendalian

I.3.4. Perencanaan Staf

I.4. Proses Teknis

I.4.1. Metode, Alat Bantu, dan Teknik

I.4.2. Dokumentasi Perangkat Lunak

I.5. Paket Pekerjaan, Jadwal, dan Anggaran

I.5.1. Paket Pekerjaan

I.5.2. Ketergantungan dan Keterkaitan

I.5.3. Kebutuhan Sumber Daya

I.5.3.1. Kebutuhan Sumber Daya Manusia

I.5.3.2. Kebutuhan Sumber Daya Perangkat Lunak

I.5.3.3. Kebutuhan Sumber Daya Perangkat Keras

I.5.4. Alokasi Anggaran dan Sumber Daya

I.5.5. Jadwal

**BAB 2. SPESIFIKASI KEBUTUHAN PERANGKAT LUNAK** Cnt.2-1

II.1. Pendahuluan Cnt.2-1

II.1.1. Tujuan

II.1.2. Ruang Lingkup

II.1.3. Definisi, Akronim, dan Singkatan

II.1.4. Referensi

II.2. Gambaran Umum

II.2.1. Perspektif Produk

II.2.1.1. Antarmuka Sistem

II.2.1.2. Antarmuka Pengguna

II.2.1.3. Antarmuka Perangkat Keras

II.2.1.4. Antarmuka Perangkat Lunak

II.2.1.5. Antarmuka Komunikasi

II.2.2. Fungsi-Fungsi Produk

II.2.3. Karakteristik Pengguna

II.2.4. Batasan-Batasan

II.2.5. Asumsi-Asumsi dan Keterkaitan

II.2.6. Kebutuhan Penyeimbang

II.3. Kebutuhan Spesifik

II.3.1. Performa

II.3.2. Keamanan Aplikasi

II.3.3. Keamanan Data

II.4. Informasi Penunjang

**BAB 3. DESKRIPSI PERANCANGAN PERANGKAT LUNAK**  Cnt.3-1

III.1. Pendahuluan Cnt.3-1

III.1.1. Tujuan

III.1.2. Ruang Lingkup

III.1.3. Definisi, Akronim, dan Singkatan

III.2. Referensi

III.3. Deskripsi Dekomposisi

III.3.1. Dekomposisi Modul

III.3.2. Dekomposisi Proses Konkuren

III.3.3. Dekomposisi Data

III.4. Deskripsi Ketergantungan/Keterkaitan

III.4.1. Keterkaitan Inter Modul

III.4.2. Keterkaitan Inter Proses

III.4.2.1. Keterkaitan Proses *API* *Backend* dengan *Frontend*

III.4.3. Keterkaitan Data

III.5. Deskripsi Antarmuka

III.5.1. Deskripsi Antarmuka Pengguna

III.6. Desain Rinci

III.6.1. Desain Rinci Modul

III.6.2. Desain Rinci Data

III.6.3. Diagram Alir

**BAB 4. IMPLEMENTASI PERANGKAT LUNAK** Cnt.4-1

**BAB 5. PENGUJIAN PERANGKAT LUNAK**  Cnt.5.1

V.1. Tujuan

V.2. Strategi dan Pendekatan

V.3. Spesifikasi Pengujian

V.4. Kerangka Rencana Pengujian

V.5. Kriteria Pemenuhan

V.6. Sumber Daya

**BAB 6. PEMELIHARAAN PERANGKAT LUNAK** Cnt.6.1

VI. 1. Pendahuluan

VI. 2. Lingkup Pemeliharaan

VI. 3. Kontrak Perawatan

**BAB 7. MANUAL PERANGKAT LUNAK** Cnt.7.1

VII.1. Pendahuluan

VII.2. Langkah dan Pengenalan Aplikasi

**BAB 8. PENUTUP** Cnt.8.1

**BAB 9. REFERENSI** Ref.9.1

**BAB 10. LAMPIRAN** Atc.10.1

X.1. Kode Sumber Database

X.1.1. Database Front End (dbERPReborn)

X.1.2. Database Back End

X.1.2.1. Database dbERPReborn-SysConfig

X.1.2.1.1. Schema SchSystem

X.1.2.1.1.1. Table

X.1.2.1.1.2. Type

X.1.2.1.1.3. Stored Procedure

X.1.2.1.2. Schema SchSysConfig

X.1.2.1.2.1. Table

X.1.2.1.2.2. Type

X.1.2.1.2.3. Stored Procedure

X.1.2.2. Database dbERPReborn-Data-OLTP

X.1.2.2.1. Schema SchSystem

X.1.2.2.2. Schema SchAccounting

X.1.2.2.3. Schema SchBudgeting

X.1.2.2.4. Schema SchCustomerRelation

X.1.2.2.5. Schema SchFinance

X.1.2.2.6. Schema SchFixedAsset

X.1.2.2.7. Schema SchHumanResource

X.1.2.2.8. Schema SchMaster

X.1.2.2.9. Schema SchProject

X.1.2.2.10. Schema SchSupplyChain

X.1.2.3. Kode Sumber Database dbERPReborn-Data-OLAP

X.1.2.4. Kode Sumber Database dbERPReborn-Data- BinaryObject

X.2. Kode Sumber Pemrograman

X.2.1. Kode Sumber PHP Front End

X.2.2. Kode Sumber PHP Back End

|  |  |
| --- | --- |
|  | **DAFTAR TABEL** |
|  |

Tabel 1.2.3.(1) **Tabel Batasan dan Antarmuka Organisasi**

Tabel 1.2.4.(1) **Tabel Perencanaan Staf**

|  |  |
| --- | --- |
|  | **DAFTAR GAMBAR** |
|  |

Gambar I.1.1.(1) **Logo ERP Reborn**

Gambar I.2.1.(1) **Model Proses Air Terjun**

Gambar I.2.2.(1) **Struktur Organisasi**

|  |  |
| --- | --- |
| **BAB 1.** | **RENCANA PENGELOLAAN PROYEK SOFTWARE** |
|  |

**I.1. Pendahuluan**

**I.1.1. Gambaran Proyek**

Perkembangan persaingan bisnis pada industri bidang jasa konstruksi membutuhkan proses yang semakin kompleks dari hari kehari. Tentunya untuk dapat mewujudkan suatu sinergi, efektivitas, dan efisiensi antar proses dan departemen yang terlibat didalamnya memerlukan suatu sistem digital yang dapat mengakomodir kebutuhan tersebut. Untuk itulah perusahaan membutuhkan memerlukan sistem ERP untuk mengintegrasikan seluruh kebutuhan tersebut.

Saat ini PT QDC Technologies sudah memiliki program ERP dalam mendukung proses bisnisnya. Namun dengan semakin berkembangnya proses bisnis dan meningkatnya kebutuhan dari para pengguna dari waktu ke waktu, maka diperlukan suatu proses *reengineering* ERP dengan perancangan baru yang jauh lebih efisien, reliable, dan stabil. Untuk itulah maka dipandang perlu untuk meluncurkan perangkat lunak baru yang diberikan dengan nama **ERP Reborn**.



*Gambar I.1.1.(1).* ***Logo ERP Reborn***

ERP Reborn menggunakan filosofi dari burung Phoenix yang merupakan simbol dari keabadian, lambang dari siklus kehidupan setelah kematian, dan simbol dari kebangkitan tubuh setelah kematian. Diharapkan dengan dikerjakannya proyek ini kedepannya akan memberikan kontribusi positif yang signifikan untuk meningkatkan *added value* bagi Perusahaan.

**I.1.2. Dokumen Dalam Proyek**

Dalam proses pengembangan proyek agar dapat memenuhi persyaratan teknis yang memadai dalam proses *life cycle*-nya, maka diperlukan dokumentasi yang memenuhi standar internasional. Untuk itulah dokumentasi proyek ini dirancang dengan mengacu pada format **IEEE software life cycle**.

Tujuan dokumentasi ini dibuat agar proses membuat perangkat lunak yang dilakukan menggunakan pendekatan kaidah - kaidah atau prinsip - prinsip rekayasa (*engineering*), sehingga akan dihasilkan perangkat lunak yang berkualitas. Dokumen yang digunakan dalam proyek ini sangatlah berkaitan dengan semua proses aktivitas yang dilakukan. Dokumen-dokumen yang terlibat diantaranya adalah :

1. **Dokumen Spesifikasi Kebutuhan Perangkat Lunak**

Dokumen ini mengacu pada standar **IEEE 830**, yang memuat deskripsi terperinci dari semua aspek perangkat lunak yang akan dibangun terspesifikasi sebelum proyek dimulai. Spesifikasi kebutuhan perangkat lunak menetapkan dasar perjanjian antara pengguna dan pengembang tentang bagaimana produk perangkat lunak seharusnya berfungsi. Spesifikasi kebutuhan perangkat lunak adalah penilaian kebutuhan yang ketat sebelum tahap desain sistem yang lebih spesifik, dan tujuannya adalah untuk mengurangi desain ulang yang mungkin terjadi dikemudian hari. Dokumen ini juga harus memberikan dasar yang realistis dengan memperkirakan faktor biaya, risiko, dan jadwal. Bila digunakan dengan tepat maka spesifikasi kebutuhan perangkat lunak dapat membantu mencegah kegagalan pengembangan perangkat lunak.

1. **Dokumen Deskripsi Perancangan Perangkat Lunak (Software Design Description)**

Dokumen ini mengacu pada standar **IEEE 1016**, yang memberikan penjelasan dan gambaran umum mengenai perancangan arsitektur perangkat lunak yang akan dibangun baik berupa gambaran umum maupun penjelasan terperinci dan menyeluruh. Secara umum dokumen ini terdiri atas :

* 1. **Perancangan Data**, yang menjelaskan struktur data yang digunakan di dalam perangkat lunak. Termasuk didalamnya segala atribut dan relasi yang terjadi antara objek data dengan struktur data;
  2. **Perancangan Arsitektur**, yang menjelaskan karakteristik arus informasi, dan memetakannya ke dalam struktur program. Termasuk transformasi metode pemetaan yang digunakan untuk menghasilkan perbedaan tampilan;
  3. **Perancangan Antarmuka**, yang menjelaskan antarmuka internal dan eksternal dari program yang dirancang berdasarkan model analisis; dan
  4. **Perancangan Prosedur**, yang menjelaskan konsep struktur pemrograman dengan menggunakan pendekatan grafik, tabel, dan notasi tulisan.

1. **Dokumen Implementasi Perangkat Lunak** adalah dokumen yang menjelaskan tahapan implementasi pengkodean *source code* dari seluruh perancangan yang telah didefinisikan pada tahapan sebelumnya.
2. **Dokumen Pengujian Perangkat Lunak**

Dokumen ini mengacu pada standar **IEEE 829**, yang merupakan sekumpulan dokumen pengujian atas perangkat lunak yang terdiri dari :

* 1. **Sub Dokumen Rencana Induk Pengujian / *Master Test Plan (MTP)***, yang memberikan perencanaan pengujian secara menyeluruh dan dokumen manajemen pengujian untuk berbagai tingkat pengujian;
  2. **Sub Dokumen Rencana Tahapan Pengujian / *Level Test Plan (LTP)***, yang menentukan batasan ruang lingkup pengujian, cara pendekatan pengujian, sumber daya pengujian, dan jadwal kegiatan pengujian yang disesuaikan dengan tingkat pengujian yang telah ditentukan dan dijelaskan sebelumnya. Hal-hal yang perlu diidentifikasi diantaranya adalah: Item yang sedang diuji, fitur yang akan diuji, tugas pengujian yang harus dilakukan, personel yang bertanggung jawab untuk setiap tugas, dan risiko yang terkait;
  3. **Sub Dokumen Perancangan Tahapan Pengujian / *Level Test Design (LTD)***, yang menentukan perincian pengujian dan hasil yang diharapkan bila lulus dalam proses pengujian;
  4. **Sub Dokumen Kasus Tahapan Pengujian / *Level Test Case (LTC)***, yang menentukan data pengujian yang digunakan untuk menjalankan pengujian kasus yang diidentifikasi dalam Perancangan Tahapan Pengujian;
  5. **Sub Dokumen Prosedur Pengujian / *Level Test Procedure (LTPr)***, yang menjelaskan bagaimana cara menjalankan setiap pengujian, termasuk segala prasyarat pengaturan dan langkah-langkah yang perlu diikuti;
  6. **Sub Dokumen Catatan Tahapan Pengujian / *Level Test Log (LTL)****,* yang mencatat semua rekaman kronologis detail yang relevan terkait pelaksanaan pengujian, seperti merekam pengujian kasus mana yang dijalankan, siapa yang menjalankannya, bagaimana urutannya, dan apakah hasil pengujian lulus ataukah gagal;
  7. **Sub Dokumen Laporan Anomali / *Anomaly Report (AR)***, yang mendokumentasikan setiap peristiwa yang terjadi selama proses pengujian yang memerlukan investigasi lebih lanjut seperti permasalahan, insiden pengujian, cacat, kesulitan, *issue*, anomali, atau laporan kesalahan. Anomali bahwa perbedaan antara hasil yang diharapkan dan hasil aktual dapat terjadi karena sejumlah alasan selain kesalahan dalam sistem. Ini termasuk kesalahan hasil yang diharapkan, kesalahan perjalanan pengujian, atau inkonsistensi dalam persyaratan yang menghasilkan lebih dari satu interpretasi dapat tercipta. Laporan tersebut terdiri dari semua detail kejadian seperti hasil aktual dan yang diharapkan, kejadian saat gagal, dan bukti pendukung yang sekiranya dapat membantu dalam penyelesaiannya. Laporan ini juga juga mencakup penilaian dampak dari suatu insiden setelah pengujian;
  8. **Sub Dokumen Laporan Status Pengujian Tahapan Interim */ Level Interim Test Status Report (LITSR)***, yang merangkum hasil sementara dari kegiatan pengujian yang telah dilakukan, secara opsional digunakan untuk memberikan evaluasi dan rekomendasi berdasarkan hasil untuk tahapan pengujian khusus;
  9. **Sub Dokumen Laporan Pengujian Tahapan / *Level Test Report (LTR)***, yang merangkum hasil dari kegiatan pengujian yang telah dilakukan dan untuk memberikan evaluasi dan rekomendasi berdasarkan hasil setelah pengujian selesai dilaksanakan untuk tingkat pengujian khusus;
  10. **Sub Dokumen Laporan Induk / *Master Test Report (MTR)***, yang merangkum hasil tahapan kegiatan pengujian yang dilaksanakan dan memberikan evaluasi sesuai hasil yang ada. Laporan manajemen yang menyediakan informasi penting apa pun yang ditemukan oleh pengujian yang dilakukan, dan termasuk penilaian atas kualitas pengujian, kualitas sistem perangkat lunak yang diuji, dan statistik yang berasal dari Laporan Anomali. Laporan ini juga mencatat pengujian apa saja yang telah dilakukan dan durasinya, untuk meningkatkan perencanaan pengujian di masa depan. Dokumen akhir ini digunakan untuk menunjukkan apakah sistem perangkat lunak yang diuji sesuai untuk tujuan sesuai dengan apakah telah memenuhi kriteria penerimaan yang ditentukan oleh para pemangku kepentingan proyek.

1. **Dokumen Pemeliharaan Perangkat Lunak**

Dokumen ini memberikan penjelasan terkait pemeliharaan dan pengelolaan Perangkat Lunak.

1. **Dokumen Manual Perangkat Lunak**

Dokumen ini mengacu pada standar **IEEE 1063**, yang menjelaskan setiap fitur program, dan membantu pengguna dalam menggunakan fitur-fitur ini. Dokumen manual yang baik tentunya dapat memberikan solusi pemecahan masalah yang dihadapi pengguna secara menyeluruh. Penyusunan dokumen bisa menggunakan pendekatan tutorial, tematik, ataupun daftar referensi.

**I.1.3. Material Acuan**

Materi yang menjadi acuan dalam pembuatan proyek ini menggunakan standar IEEE, karena menyediakan kerangka kerja yang menggabungkan seluruh spektrum proses siklus hidup perangkat lunak. Dan juga standar IEEE untuk membentuk model yang diakui secara internasional dari kehidupan perangkat lunak umum, siklus proses yang dapat direferensikan oleh industri perangkat lunak di seluruh dunia.

**I.1.4. Definisi dan Akronim**

Dalam dokumentasi perangkat lunak ini ada beberapa istilah dan akronim yang digunakan, diantaranya adalah :

|  |  |  |
| --- | --- | --- |
| **Apache** | : | *Adalah server web yang dapat dijalankan di banyak sistem operasi (Unix, BSD, Linux, Microsoft Windows dan Novell Netware serta platform lainnya) yang berguna untuk melayani dan memfungsikan situs web. Protokol yang digunakan untuk melayani fasilitas web/www ini menggunakan HTTP. Apache memiliki fitur-fitur canggih seperti pesan kesalahan yang dapat dikonfigurasi, autentikasi berbasis basis data dan lain-lain. Apache juga didukung oleh sejumlah antarmuka pengguna berbasis grafik (GUI) yang memungkinkan penanganan server menjadi mudah.* |
| **API** | : | ***Application Programming Interface***  *Penerjemah komunikasi antara klien dengan server untuk menyederhanakan implementasi dan perbaikan software. Diartikan juga sebagai sekumpulan perintah, fungsi, serta protokol yang dapat digunakan oleh programmer saat membangun perangkat lunak untuk sistem operasi tertentu. API memungkinkan programmer untuk menggunakan fungsi standar untuk berinteraksi dengan sistem operasi.* |
| **DBMS** | : | ***Database Management System***  *Sistem atau perangkat lunak yang dirancang untuk mengelola suatu basis data dan menjalankan operasi terhadap data yang diminta banyak pengguna.* |
| **Docker** | : | *Platform perangkat lunak yang memungkinkan Anda membuat, menguji, dan menerapkan aplikasi dengan cepat. Docker mengemas perangkat lunak ke dalam unit standar yang disebut kontainer yang memiliki semua yang diperlukan perangkat lunak agar dapat berfungsi termasuk pustaka, alat sistem, kode, dan waktu proses.* |
| **ERP** | : | ***Enterprise Resource Planning***  *Sistem informasi yang diperuntukkan bagi perusahan manufaktur maupun jasa yang berperan mengintegrasikan dan mengotomasikan proses bisnis yang berhubungan dengan aspek operasi, produksi maupun distribusi di perusahaan bersangkutan.* |
| **FDW** | : | ***Foreign Data Wrapper***  *Salah satu modul milik PostgreSQL yang memungkinkan pengguna untuk dapat mengakses data pada database PostgreSQL yang lain (baik sesama local database maupun remote database). FDW memungkinkan pengguna untuk mendapatkan referensi tabel yang berada pada database lainnya, sehingga seakan-akan tabel tersebut terdapat pada database project yang sedang dikerjakan.* |
| **Git** | : | *Perangkat lunak pengendali versi atau proyek manajemen kode perangkat lunak yang diciptakan oleh Linus Torvalds, yang pada awalnya ditujukan untuk pengembangan kernel Linux.* |
| **GitHub** | : | *Layanan host web bersama untuk proyek pengembangan perangkat lunak yang menggunakan sistem kendali versi Git dan layanan hosting internet.* |
| **IEEE** | : | ***Institute of Electrical and Electronics Engineers***  *Sebuah organisasi profesi nirlaba yang terdiri dari banyak ahli dibidang teknik yang mempromosikan pengembangan standar-standar dan bertindak sebagai pihak yang mempercepat teknologi-teknologi baru dalam semua aspek dalam industri dan rekayasa (engineering), yang mencakup telekomunikasi, jaringan komputer, kelistrikan, antariksa, dan elektronika.* |
| **Laravel** | : | *Kerangka kerja aplikasi web berbasis PHP yang sumber terbuka, menggunakan konsep Model-View-Controller (MVC). Laravel berada dibawah lisensi MIT, dengan menggunakan GitHub sebagai tempat berbagi kode.* |
| ***Mockups*** | : | *Rancangan desain profil statis statis dari suatu desain atau perangkat, yang digunakan untuk mewakili struktur informasi, memvisualisasikan konten dan menunjukkan fungsi dasar dengan cara statis. Mockups memberikan visual seperti warna dan tipografi dengan terperinci, sehingga memberikan kesan yang lebih realistis tentang bagaimana produk akhir akan terlihat.* |
| ***Monolithic Application*** | : | *Aplikasi perangkat lunak satu tingkat di mana antarmuka pengguna dan kode akses data digabungkan menjadi satu program pada satu platform tungga****l*** |
| **MVC** | : | ***Model-View-Controller***  *Sebuah metode untuk membuat sebuah aplikasi dengan memisahkan data (Model) dari tampilan (View) dan cara bagaimana memprosesnya (Controller). Dalam implementasinya kebanyakan kerangka kerja (framework) dalam aplikasi web adalah berbasis arsitektur MVC. MVC memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, antarmuka pengguna, dan bagian yang menjadi kontrol dalam sebuah aplikasi web.* |
| **PGSQL** | : | **PostgreSQL**  *Sebuah RDBMS open-source (didistribusikan secara free) yang menekankan pada pemenuhan standar teknis dan fleksibilitas (keluwesan) data. PostgreSQL didistribusikan dengan lisensi bebas/gratis, sehingga dapat digunakan, dimodifikasi, dan didistribusikan kembali kepada publik secara bebas/gratis untuk tujuan pribadi, komersial, maupun akademik. PostgreSQL dirancang untuk menangani beban kerja terhadap data dari sebuah mesin menuju layanan web yang diakses banyak orang secara bersamaan. PostgreSQL merupakan sebuah sistem manajemen database yang didesain agar bekerja dengan baik pada sistem MacOS Server, namun development-nya juga diberlakukan untuk platform Linux, FreeBSD, OpenBSD, dan Windows 32-bit dan 64 bit sampai pada versi PostgreSQL 10. Selanjutnya, mulai dari versi PostgreSQL 11 dikembangkan untuk platform MacOS 64-bit dan Windows 64-bit saja.* |
| **PHP** | : | ***PHP: Hypertext Preprocessor***  *Bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk memprogram situs web dinamis.* |
| **Postman** | : | *Platform kolaborasi yang digunakan untuk mempercepat pengembangan API dengan memanfaatkan fitur-fitur yang dapat menyederhanakan setiap langkah pembangunan API dan kolaborasi streamline.* |
| **Redis** | : | ***Remote Dictionary Server***  *Penyimpanan data nilai utama di dalam memori yang super cepat dengan sumber terbuka untuk digunakan sebagai database, cache, broker pesan, dan antrian. Redis mampu memberikan respons dalam waktu di bawah satu milidetik yang memungkinkan jutaan permintaan per detik untuk aplikasi real-time pada Permainan, Ad-Tech, Layanan Finansial, Layanan Kesehatan, dan IoT. Redis adalah pilihan populer untuk caching, manajemen sesi, permainan, papan peringkat, analisis real-time, geospasial, tumpangan berkendara, obrolan/perpesanan, streaming media, dan aplikasi pub/sub.* |
| **Sandbox** | : | *Lingkungan pengujian yang mengisolasi perubahan kode yang belum diuji dan eksperimen langsung dari lingkungan produksi atau repositori, dalam konteks pengembangan perangkat lunak termasuk pengembangan web dan kontrol revisi.* |
| **Website Fireframe** | : | *Dikenal juga sebagai* ***skema laman*** *atau* ***cetak biru layar****, adalah panduan visual yang mewakili kerangka-kerangka pada situs web yang menggambarkan tata letak halaman atau pengaturan konten, termasuk elemen antarmuka dan sistem navigasi, dan bagaimana mereka saling bekerjasama.*  *Wireframes berfokus pada : rentang fungsi yang tersedia, prioritas relatif dari informasi dan fungsi, aturan untuk menampilkan jenis informasi tertentu, dan efek dari berbagai skenario pada tampilan* |

**I.2. Organisasi Proyek**

**I.2.1. Model Proses**

Model proses yang digunakan dalam pengembangan perangkat lunak ini menggunakan model air terjun (*watefall model*)



*Gambar I.2.1.(1).* ***Model Proses Air Terjun***

Model ini dipilih atas pertimbangan beberapa keunggulan diantaranya adalah :

* 1. Kualitas dari sistem yang dihasilkan akan baik, disebabkan pelaksanaannya dilakukan secara bertahap sehingga tidak terfokus hanya pada tahapan tertentu saja; dan
  2. Dokumen pengembangan sistem sangat terorganisir, karena setiap tahapan harus terselesaikan dengan lengkap sebelum melangkah ke tahapan berikutnya. Jadi setiap tahapan akan memiliki dokumen tertentu.

**I.2.2. Struktur Organisasi**

Struktur organisasi yang diperlukan untuk pengembangan perangkat lunak ini adalah :

*Gambar I.2.2.(1).* ***Struktur Organisasi***

**I.2.3. Batasan dan Antar Muka Organisasi**

Untuk menjamin setiap personil yang ditentukan dalam Struktur Organisasi menjalankan fungsi kerja sesuai dengan kewenangannya, maka diperlukan pembatasan ruang lingkup masing-masing unit sebagaimana dijelaskan dalam tabel berikut:

|  |  |
| --- | --- |
| **UNIT KERJA** | **PEMBATASAN KEWENANGAN** |
| ***Manager*** | * Mengawasi dan mengevaluasi *progress* dari seluruh unit yang berada di bawah koordinasinya * Berkoordinasi dan berkomunikasi dengan konsultan |
| ***Konsultan*** | * Berkoordinasi dan berkomunikasi dengan konsultan * Berkoordinasi dan berkomunikasi dengan seluruh anggota tim secara langsung untuk memberikan *guidance* dan *expertise* terkait sistem |
| ***Front End Developer Coordinator*** | * Melaporkan semua pencapaian, pengawasan, dan evaluasi dari tim yang dipimpinnya kepada Manager * Membangun komunikasi dan koordinasi yang efektif dengan anggota tim yang dipimpinnya * Bertukar pikiran dengan konsultan dan anggota tim * Mengimplementasikan rancangan terbaik sistem sesuai *guidance* dan *expertise* dari konsultan |
| ***Front End Programmer*** | * Menjalankan dan mengeksekusi semua instruksi dan arahan yang diberikan oleh koordinator (pimpinan) diatasnya terkait pengembangan *Front End Web* |
| ***Back End Developer Coordinator*** | * Melaporkan semua pencapaian, pengawasan, dan evaluasi dari tim yang dipimpinnya kepada Manager * Membangun komunikasi dan koordinasi yang efektif dengan anggota tim yang dipimpinnya * Bertukar pikiran dengan konsultan dan anggota tim * Mengimplementasikan rancangan terbaik sistem sesuai *guidance* dan *expertise* dari konsultan |
| ***Back End Programmer*** | * Menjalankan dan mengeksekusi semua instruksi dan arahan yang diberikan oleh koordinator (pimpinan) diatasnya terkait pengembangan *Back End Web* |
| ***Database Developer Coordinator*** | * Melaporkan semua pencapaian, pengawasan, dan evaluasi dari tim yang dipimpinnya kepada Manager * Membangun komunikasi dan koordinasi yang efektif dengan anggota tim yang dipimpinnya * Bertukar pikiran dengan konsultan dan anggota tim * Mengimplementasikan rancangan terbaik sistem sesuai *guidance* dan *expertise* dari konsultan |
| ***Database Developer*** | * Menjalankan dan mengeksekusi semua instruksi dan arahan yang diberikan oleh koordinator (pimpinan) diatasnya terkait pengembangan *Database* |
| ***Quality Assurance & Documentation Coordinator*** | * Melaporkan semua pencapaian, pengawasan, dan evaluasi dari tim yang dipimpinnya kepada Manager * Membangun komunikasi dan koordinasi yang efektif dengan anggota tim yang dipimpinnya * Bertukar pikiran dengan konsultan dan anggota tim * Mengimplementasikan rancangan terbaik sistem sesuai *guidance* dan *expertise* dari konsultan |
| ***Quality Assurance & Documentation*** | * Menjalankan dan mengeksekusi semua instruksi dan arahan yang diberikan oleh koordinator (pimpinan) diatasnya terkait proses pengujian dan pendokumentasian |
| ***System & Infrastructure Developer*** | * Melaporkan semua pencapaian, pengawasan, dan evaluasi dari pekerjaan yang dilakukannya kepada Manager |

*Tabel I.2.3.(1).* ***Tabel Batasan dan Antarmuka Organisasi***

**I.2.4. Ruang Lingkup dan Tanggung Jawab**

**I.2.4.1. Penjelasan**

Lingkup dan tanggung jawab ini berisi tugas dari setiap elemen anggota dalam pembuatan proyek ini

**I.2.4.1.1. Manager**

Manager adalah seseorang mempunyai tanggung jawab dan tugas yang besar dalam sebuah tim, tidak hanya terfokus pada hal-hal yang teknis sifatnya. Manager juga harus mampu memanajemen tim dengan baik, agar proyek dapat tercapai sesuai target.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

1. Memberi petunjuk dan pengarahan;
2. Mendelegasikan fungsi pengawasan dan pengendalian kepada para koordinator tim dibawah subordinasinya;
3. Membagi tugas, pekerjaan, tanggung jawab kepada seluruh anggota;
4. Memonitoring kinerja konsultan;
5. Memonitoring kinerja tim;
6. Mengevaluasi pencapaian dari seluruh tim;
7. Membuat dan mengajukan penjadwalan dan pengajuan anggaran untuk pekerjaan;
8. Membuat dan mengajukan rencana kebutuhan seluruh sumber daya yang diperlukan; dan
9. Memberikan laporan kepada atasan yang menjadi superordinasinya.

**I.2.4.1.2. Konsultan**

Konsultan bekerja bermitra dengan tim internal dengan memberi saran dan anjuran melalui pendekatan teknis dan konsultatif untuk memenuhi sasaran bisnis atau menyelesaikan proyek. Konsultan bekerja untuk memperbaiki struktur dan efisiensi dan sistem IT organisasi agar tujuan proyek tercapai dengan lebih baik.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

1. Melakukan pertemuan dengan pengguna jasanya (*client*) untuk menetapkan beberapa keperluan terkait pekerjaan;
2. Bekerja dengan *client* untuk bersama-sama menetapkan tujuan dan jangkauan dari suatu proyek;
3. Merencanakan *timescale* dan kebutuhan sumber daya;
4. Menjelaskan spesifikasi sistem *client*, memahami kebiasaan kerja *client*, dan sifat dasar dari bisnisnyal
5. Berhubungan dengan staff pada semua tingkat dari organisasi *client*;
6. Menetapkan *software*, *hardware*, dan kebutuhan jaringan untuk kebutuhan sistem;
7. Menganalisa kebutuhan IT dalam perusahaan dan memberikan saran dan masukan yang independen dan objektif dalam penggunaan IT;
8. Mengembangkan solusi yang cocok dan mengimplementasikan sistem baru;
9. Memberikan solusi dalam laporan tertulis ataupun lisan;
10. Membantu *client* pada aktivitas perubahan manajemen;
11. Merancang, menguji, memasang, dan memonitoring sistem baru;
12. Menyiapkan dokumentasi dan memberikan laporan proses pada *client*;
13. Mengatur pelatihan untuk *user* dan konsultan lain; dan
14. Mengenali potential *client* dan membangun dan memelihara hubungan dengan mereka

**I.2.4.1.3. *Front End Developer Coordinator***

*Front End Developer Coordinator* selain menjadi *leader* bagi tim dibawahnya, diharapkan bisa menjadi seorang sistem analis yang berkaitan dengan pemecahan permasalahan dan menentukan solusi yang tepat yang terkait dengan pengembangan *Front End Web*. Seorang *Front End Developer Coordinator* harus bisa memberikan arahan kepada timnya agar bisa bekerja secara tepat sasaran, efektif, dan efisien sesuai dengan rencana kerja dan target yang sudah ditentukan.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

1. Mengadaptasi dan mentranslasikan kebutuhan pengguna terkait interaksinya dengan sistem;
2. Merancang *website wireframe* atau *mockups* sebagai dasar acuan pembentukan antarmuka pengguna (*user interface*);
3. Merancang bangun antarmuka pengguna yang mudah diaplikasikan, aman, dan nyaman sesuai kebutuhan pengguna;
4. Merancang bangun model komunikasi data terbaik yang aman, efisien, dan efektif dengan antarmuka pemrograman aplikasi (*API / Application Programming Interface*) *Backend Web* (*Web Service*);
5. Mendelegasikan pekerjaan kepada anggota tim yang menjadi subordinasinya;
6. Mengevaluasi semua pekerjaan anggota timnya;
7. Mengusulkan penambahan, pengurangan, pertukaran, atau pergantian personil tim apabila diperlukan; dan
8. Melaporkan semua pencapaian yang diraih oleh timnya kepada Manajer;

**I.2.4.1.4. *Front End Developer***

*Front End Developer* adalah *programmer* yang bertugas untuk mengimplementasikan dari sistem yang sudah dirancang didesain terkait *Front End Web*. *Front End Developer* dituntut dapat menuliskan kode program dengan baik, efektif, dan efisien. Hal ini dimaksudkan untuk menjamin ketepatan waktu dan menghindari terjadinya banyak *error* dalam proses implementasi.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

Mengimplementasikan instruksi dan rancangan yang sudah ditentukan oleh koordinatornya kedalam bahasa pemrograman;

Mengusulkan modifikasi dari implementasi apabila diperlukan kepada koordinatornya, dengan tujuan untuk meningkatkan modularitas, keamanan, efektivitas, dan efisiensi;

Melakukan koreksi implementasi pemrograman saat ditemukan kesalahan dan anomali; dan

Memastikan terimplementasinya *exception handling* yang terdefinisi dengan baik pada setiap *block* pemrograman, agar bisa menangani semua kesalahan data, proses, dan *event* dengan lebih baik.

**I.2.4.1.5. *Back End Developer Coordinator***

*Back End Developer Coordinator* selain menjadi *leader* bagi tim dibawahnya, diharapkan bisa menjadi seorang sistem analis yang berkaitan dengan pemecahan permasalahan dan menentukan solusi yang tepat yang terkait dengan pengembangan *Back End Web*. Seorang *Back End Developer Coordinator* harus bisa memberikan arahan kepada timnya agar bisa bekerja secara tepat sasaran, efektif, dan efisien sesuai dengan rencana kerja dan target yang sudah ditentukan.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

1. Merancang bangun antarmuka pemrograman aplikasi (*API / Application Programming Interface*) *Backend Web* (*Web Service*) yang akan digunakan oleh *Front End Web Application*;
2. Merancang bangun keamanan *multilayer* untuk menjamin ketersediaan layanan *web service*;
3. Melakukan modularitas API untuk memudahkan pengembangan dan pemeliharaan;
4. Memastikan tidak terjadinya anomali pada seluruh fungsi objek yang telah didefinisikan;
5. Mendelegasikan pekerjaan kepada anggota tim yang menjadi subordinasinya;
6. Mengevaluasi semua pekerjaan anggota timnya;
7. Mengusulkan penambahan, pengurangan, pertukaran, atau pergantian personil tim apabila diperlukan; dan
8. Melaporkan semua pencapaian yang diraih oleh timnya kepada Manajer;

**I.2.4.1.6. *Back End Developer***

*Back End Developer* adalah *programmer* yang bertugas untuk mengimplementasikan dari sistem yang sudah dirancang didesain terkait *Back End Web*. *Back End Developer* dituntut dapat menuliskan kode program dengan baik, efektif, dan efisien. Hal ini dimaksudkan untuk menjamin ketepatan waktu dan menghindari terjadinya banyak *error* dalam proses implementasi.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

Mengimplementasikan instruksi dan rancangan yang sudah ditentukan oleh koordinatornya kedalam bahasa pemrograman;

Mengusulkan modifikasi dari implementasi apabila diperlukan kepada koordinatornya, dengan tujuan untuk meningkatkan modularitas, keamanan, efektivitas, dan efisiensi;

Melakukan koreksi implementasi pemrograman saat ditemukan kesalahan dan anomali; dan

Memastikan terimplementasinya *exception handling* yang terdefinisi dengan baik pada setiap *block* pemrograman, agar bisa menangani semua kesalahan data, proses, dan *event* dengan lebih baik.

**I.2.4.1.7. *Database Developer Coordinator***

*Database Developer Coordinator* bertugas untuk menentukan arsitektur database terbaik yang diperlukan oleh pengembangan perangkat lunak. Reliabilitas dan performance harus menjadi pertimbangan utama dalam menentukan arsitektur database yang paling tepat. Seorang *Database Developer Coordinator* harus bisa memberikan arahan kepada timnya agar bisa bekerja secara tepat sasaran, efektif, dan efisien sesuai dengan rencana kerja dan target yang sudah ditentukan.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

1. Menentukan arsitektur *database* yang paling tepat diimplementasikan untuk menjawab kebutuhan dasar dari sistem;
2. Merancang bangun *layering database* untuk memudahkan *maintenance* dan *scale up* sistem;
3. Menentukan objek yang perlu didefinisikan pada sistem *database* berupa *database*, *schema*, *table*, *view*, *stored procedure*, *data type holder*, dan *plugins*;
4. Merancang skenario proses *backup* dan *restore* terkait dengan *recovery* dan reliabilitas sistem;
5. Merancang proses rutin *maintenance* dan sinkronisasi secara terjadwal dan otomatis;
6. Mendelegasikan pekerjaan kepada anggota tim yang menjadi ordinasinya;
7. Mengevaluasi semua pekerjaan anggota timnya;
8. Mengusulkan penambahan, pengurangan, pertukaran, atau penggantian personil tim apabila diperlukan; dan
9. Melaporkan semua pencapaian yang diraih oleh timnya kepada Manajer;

**I.2.4.1.8. *Database Developer***

*Database Developer* bertugas untuk mengeksekusi dan menjalankan semua instruksi yang berkaitan implementasi semua objek yang terkait dengan pengembangan database.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

Mengimplementasikan instruksi dan rancangan yang sudah ditentukan oleh koordinatornya ke dalam *Data Definition Language* (*DDL*), *Data Manipulation Language* (*DML*), dan bahasa pemrograman DBMS;

Mengusulkan modifikasi dari implementasi apabila diperlukan kepada koordinatornya, dengan tujuan untuk meningkatkan modularitas, keamanan, efektivitas, dan efisiensi; dan

Melakukan koreksi implementasi ditemukan kesalahan dan anomali.

**I.2.4.1.9. *Quality Assurance & Documentation Coordinator***

*Quality Assurance & Documentation Coordinator* merupakan koordinator yang bertugas untuk merancang semua proses pengujian sesuai dokumen yang ada. Pengujian yang dilakukan harus bersifat komprehensif dan detail sehingga perangkat lunak yang tengah dikembangkan dapat berfungsi sesuai dengan harapan tanpa adanya anomali ataupun *error*.

*Quality Assurance & Documentation Coordinator* juga bertugas untuk merancang seluruh dokumentasi yang diperlukan dalam pengembangan perangkat lunak, yang nantinya akan dilengkapi oleh tim yang berada di bawah koordinasinya.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

1. Membuat *template* dasar dokumentasi semua tahapan pengembangan sistem;
2. Memastikan terpenuhinya semua proses dokumentasi sesuai dengan standar yang sudah ditetapkan;
3. Merancang skenario proses pengujian secara bertahap dan menyeluruh;
4. Mengevaluasi semua temuan yang terjadi terkait kesalahan sistem saat pengujian;
5. Melaporkan semua temuan kesalahan saat pengujian kepada tim developer agar dapat dilakukan perbaikan dengan segera;
6. Mendelegasikan pekerjaan kepada anggota tim yang menjadi subordinasinya;
7. Mengevaluasi semua pekerjaan anggota timnya;
8. Mengusulkan penambahan, pengurangan, pertukaran, atau pergantian personil tim apabila diperlukan; dan
9. Melaporkan semua pencapaian yang diraih oleh timnya kepada Manajer;

**I.2.4.1.10. *Quality Assurance & Documentation***

*Quality Assurance & Documentation* bertugas untuk mengimplementasikan semua pengujian yang telah dirancang dan mendokumentasikan seluruh tahapan pengerjaan proyek dari awal sampai akhir.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

1. Melengkapi dokumentasi yang sudah didefinisikan;
2. Mengimplementasikan skenario pengujian dan mencatat semua hasil yang didapatkan secara objektif; dan
3. Melaporkan semua temuan terkait kesalahan dan anomali sistem kepada koordinatornya, sehingga akan ditindaklanjuti sebagai masukan bagi tim developer untuk perbaikan sistem.

**I.2.4.1.11. *System & Infrastructure Developer***

*System & Infrastructure Developer* bertugas untuk mempersiapkan segala kebutuhan terkait *environment* dan *resources* yang dibutuhkan dalam proses dan pasca *development*. Khusus untuk proses *development* diperlukan *sandbox* terisolasi agar segala ujicoba dan implementasi dapat dilakukan secara terbatas dan tidak mengganggu sistem yang sedang berjalan. *Repository* perlu disediakan juga untuk proses *versioning* dari tahapan implementasi pengembangan program.

Beberapa tugas utama yang menjadi tanggung jawabnya adalah sebagai berikut :

Menyediakan sumber daya yang dibutuhkan sistem seperti *server*, *network devices*, *software tools*, dan *environment*;

Menjamin ketersediaan dan kestabilan sumber daya yang disediakan;

Melakukan *troubleshooting* dan *maintenance* dengan segera apabila terjadi *malfunction* sumber daya; dan

Membantu tim developer untuk mengimplementasikan sistem dari *sandbox environment* kepada *real environment*.

**I.3. Proses Manajerial**

**I.3.1. Tujuan dan Prioritas Manajemen**

**I.3.1.1. Prioritas Jadwal**

Prioritas jadwal yang dilakukan pada saat ini adalah membuat sistem yang akan dibuat, dokumen projek, jadwal kegiatan, struktur pembuatan proyek, dan organisasi.

**I.3.1.2. Anggaran**

Prioritas anggaran untuk proyek ini lebih ditekankan pada requirement proyek dan kualitas hardware.

**I.3.2. Manajemen Risiko**

**I.3.2.1. Dampak Negatif**

**I.3.3. Mekanisme Pengawasan dan Pengendalian**

**I.3.3.1. Pengawasan**

Proses pengawasan dilakukan secara kerja tim, adapun proses pelaksanaannya dilakukan setiap hari sesuai dengan jadwal yang sudah ditetapkan.

**I.3.3.2. Pengendalian**

Proses pengendalian dilaksanakan setiap hari rabu bersama dengan konsultan, hal ini dilakukan agar hasil kerja proyek dapat dievaluasi pada pertengahan minggunya.

**I.3.4. Perencanaan Staf**

Agar fungsi organisasi berjalan dengan baik maka diperlukan penempatan orang yang tepat dan sesuai dengan kapasitas dan kapabilitas yang dimiliki. Penempatan staf dalam pengembangan perangkat lunak ini adalah:

|  |  |
| --- | --- |
| **UNIT KERJA** | **PENEMPATAN PERSONIL** |
| *Manager* | *Teguh Pratama Januzir S* |
| *Konsultan* | *Bherly Novrandi* |
| *Front End Developer Coordinator* | *Icha Mailinda* |
| *Front End Developer* | *(2 new comer)* |
| *Back End Developer Coordinator* | *Icha Mailinda* |
| *Back End Developer* | *Suyanto, (new comer)* |
| *Database Developer Coordinator* | *Teguh Pratama Januzir S* |
| *Database Developer* | *(2 new comer)* |
| *Quality Assurance & Documentation Coordinator* | *Teguh Pratama Januzir S* |
| *Quality Assurance & Documentation* | *(2 new comer)* |
| *System & Infrastructure Developer* | *Zainudin Anwar* |

*Tabel I.2.4.(1).* ***Tabel Perencanaan Staf***

**I.4. Proses Teknis**

**I.4.1. Metode, Alat Bantu, dan Teknik**

Metode pengembagan yang digunakan menggunakan pendekatan pemrograman berorientasi objek (*Object Oriented Programming*). Setiap objek pemrograman dan objek didalam database harus menggunakan pendekatan ini. Kelebihan yang bisa didapatkan dengan menggunakan model ini adalah :

1. **Peningkatan Produktivitas Pengembangan Perangkat Lunak**, Sistem program dapat dimodifikasi tanpa melibatkan banyak modul dimana hanya objek saja yang terlibat. Selain itu sistem program dapat dikembangkan sampai skala paling kompleks;
2. **Peningkatan Pemeliharaan Perangkat Lunak**, bagian dari perangkat lunak dapat dengan mudah dikelola jika ada perubahan meskipun dalam skala yang cukup besar;
3. **Pengembangan Lebih Cepat,** metode ini didukung oleh banyak *library* objek, sehingga mempercepat penyelesaian program dan juga projek berikutnya;
4. **Biaya Pengembangan yang lebih Rendah**, dengan adanya pengembangan yang lebih cepat (*faster development*) maka akan mengurangi biaya pengembangan pembuatan program;
5. **Perangkat Lunak Berkualitas Tinggi**, dengan adanya pengembangan yang lebih cepat (*faster development*) maka akan memberikan lebih banyak waktu dan sumberdaya untuk dilakukannya proses verifikasi perangkat lunak.

Adapun alat bantu (*tools*) yang digunakan dalam pengembangan perangkat lunak adalah :

|  |  |  |  |
| --- | --- | --- | --- |
| **ALAT BANTU** | | **VERSI** | **KEGUNAAN** |
|  | *Adobe Photoshop* | *CS3 (v.10)* | *Image Processing Tools* |
|  | *Apache* | *2.4.43* | *Web Server* |
|  | *Centos* | *8.3* | *Operating System based on Linux* |
|  | *Docker* | *19.03.12* | *Service Environment* |
|  | *Git* | *2.25.4* | *Local Repository and Versioning* |
|  | *Github* | *-* | *Remote Repository and Versioning* |
|  | *Laravel Framework* | *7.19.1* | *Programming Framework based on PHP* |
|  | *Microsoft Office Excel* | *2013* | *Office Suite for spreadsheet* |
|  | *Microsoft Office Word* | *2013* | *Office Suite for word processing* |
|  | *Microsoft Office Visio* | *2013* | *Office Suite for diagram and chart* |
|  | *Netbeans* | *8.2* | *IDE* |
|  | *PGAdmin4* | *4.2* | *Management Tools for PostgreSQL* |
|  | *PHP* | *7.3* | *Programming Scripting Language* |
|  | *Postman* | *7.0* |  |
|  | *PostgreSQL* | *13* | *SQL Database Management System* |
|  | *Redis* | *6.0.5* | *Non SQL Database based on Memory* |
|  | *Whatsapp* |  | *Chat Messenger* |

*Tabel I.2.4.(1).* ***Tabel Alat Bantu***

**I.4.2. Dokumentasi Perangkat Lunak**

Dokumentasi perangkat lunak yang digunakan berdasarkan standar internasional IEEE, karena telah menyediakan kerangka kerja yang menghubungkan seluruh spektrum siklus hidup perangkat lunak.

**I.5. Paket Pekerjaan, Jadwal, dan Anggaran**

**I.5.1. Paket Pekerjaan**

**I.5.2. Ketergantungan dan Keterkaitan**

**I.5.3. Kebutuhan Sumber Daya Manusia**

**I.5.3.1. Kebutuhan Sumber Daya Manusia**

**I.5.3.2. Kebutuhan Sumber Daya Perangkat Lunak**

**I.5.3.3. Kebutuhan Sumber Daya Perangkat Keras**

**I.5.4. Alokasi Anggaran dan Sumber Daya**

**I.5.5. Jadwal**

|  |  |
| --- | --- |
| **BAB 2.** | **ANALISIS PERANGKAT LUNAK** |
|  |

**II.1. Pendahuluan**

**II.1.1. Tujuan**

Dokumen ini disusun sebagai acuan yang dapat digunakan oleh pengguna maupun pengembang sistem terkait dengan

**II.1.2. Ruang Lingkup**

**II.1.3. Definisi, Akronim, dan Singkatan**

**II.1.4. Referensi**

**II.2. Gambaran Umum**

**II.2.1. Perspektif Produk**

Perspektif produk harus terdefinisikan dengan baik sesuai dengan kebutuhan dan harapan dari sudut pandang pengguna. Dengan terakomodirnya kebutuhan sesuai dengan perspektif pengguna, maka akan semakin memudahkan adaptasi dan keterikatan (*engagement*) pengguna dalam menggunakan sistem. Dengan demikian akan dapat menurunkan peluang kemungkinan terjadinya penolakan dan kegagalan sistem pada saat digunakan oleh pengguna.

**II.2.1.1. Antarmuka Sistem**

Secara umum antarmuka sistem haruslah memenuhi kriteria sebagai berikut :

1. Dapat diakses oleh pengguna kapanpun dan dimanapun;
2. Tersedia secara *realtime* selama 24 jam *non stop*;
3. Mendukung akses *multiuser* dan *multiprivileges*; dan
4. Mendukung akses *multiplatform*.

Untuk memenuhi kebutuhan dengan kriteria sebagaimana dijelaskan diatas, maka antarmuka sistem harus berbasis web dengan koneksi yang terjaga tetap *online* 24 jam sehari semalam secara penuh. Dengan demikian kompleksitas *platform* yang digunakan oleh pengguna untuk mengakses sistem bisa diakomodir dengan sangat baik dan *reliable*.

**II.2.1.2. Antarmuka Pengguna**

**II.2.1.3. Antarmuka Perangkat Keras**

**II.2.1.4. Antarmuka Perangkat Lunak**

**II.2.1.5. Antarmuka Komunikasi**

Secara umum antarmuka sistem haruslah memenuhi kriteria sebagai berikut :

Mampu mengirim dan menerima data *realtime* secara *reliable*;

Mampu menangkap dan menangani kegagalan proses komunikasi apabila terjadi gangguan komunikasi;

Tahan terhadap kemungkinan terjadinya intercept dan gangguan dari pihak ketiga; dan

Mampu memilah potensi serangan yang bertujuan untuk membebani trafik lalu lintas data;

Untuk memenuhi kebutuhan tersebut, maka komunikasi antara *Backend* dan *Frontend* akan menggunakan jalur komunikasi HTTPS (*Hypertext Transfer Protocol Secure*) dengan sistem enkripsi SSL (*Secure Socket Layer*). Sedangkan Format data yang digunakan adalah JSON (*Javascript Object Notation*).

**II.2.2. Fungsi-Fungsi Produk**

**II.2.3. Karakteristik Pengguna**

**II.2.4. Batasan-Batasan**

**II.2.5. Asumsi-Asumsi dan Keterkaitan**

**II.2.6. Kebutuhan Penyeimbang**

**II.3. Kebutuhan Spesifik**

**II.3.1. Performa**

**II.3.2. Keamanan Aplikasi**

Untuk menunjang keamanan dari komunikasi dan transfer data antara *Backend* dan *Frontend*, maka akan digunakan sistem keamanan Token dengan metode JWT (*JSON Web Token*). Dimana *Token* akan berisi kredensial pengguna yang valid dan dienkripsi dengan HMAC SHA256. *Token* akan memiliki masa hidup tertentu dan akan berlaku unik untuk setiap pengguna.

**II.3.3. Keamanan Data**

Diperlukan sistem enkripsi untuk mengamankan kredensial dan informasi sensitif pengguna di dalam database, penggunaan enkripsi Bcrypt untuk data password pengguna dinilai sudah cukup aman.

**II.4. Informasi Penunjang**

|  |  |
| --- | --- |
| **BAB 3.** | **DESKRIPSI PERANCANGAN PERANGKAT LUNAK** |
|  |

**III.1. Pendahuluan**

**III.1.1. Tujuan**

**III.1.2. Ruang Lingkup**

**III.1.3. Definisi, Akronim, dan Singkatan**

**III.2. Referensi**

**III.3. Deskripsi Dekomposisi**

**III.3.1. Dekomposisi Modul**

**III.3.2. Dekomposisi Proses Konkuren**

**III.3.3. Dekomposisi Data**

**III.4. Deskripsi Ketergantungan/Keterkaitan**

**III.4.1. Keterkaitan Inter Modul**

**III.4.2. Keterkaitan Inter Proses**

**III.4.2.1. Keterkaitan Proses *API Backend* dengan *Frontend***



**III.4.3. Keterkaitan Data**

**III.5. Deskripsi Antarmuka**

**III.5.1. Deskripsi Antarmuka Sistem**

Secara umum, sistem dirancang dengan menggunakan 3 lapisan utama yaitu :

1. Front End Web Layer
2. Back End Web Layer
3. Database Layer

**III.5.3. Deskripsi Antarmuka Pengguna**

**III.5.4. Deskripsi Antarmuka Komunikasi**

**III.6. Desain Rinci**

|  |  |
| --- | --- |
| **BAB 4.** | **IMPLEMENTASI PERANGKAT LUNAK** |
|  |

|  |  |
| --- | --- |
| **BAB 5.** | **PENGUJIAN PERANGKAT LUNAK ERP** |
|  |

|  |  |
| --- | --- |
| **BAB 6.** | **PEMELIHARAAN PERANGKAT LUNAK** |
|  |

|  |  |
| --- | --- |
| **BAB 7.** | **MANUAL PERANGKAT LUNAK** |
|  |

|  |  |
| --- | --- |
| **BAB 8.** | **PENUTUP** |
|  |

|  |  |
| --- | --- |
| **BAB 9.** | **REFERENSI** |
|  |

*Adiseshiah, E. G. 2016.* ***Wireframes Vs Mockups: what’s the best option?****. (online). (*[*https://www.justinmind.com/blog/wireframes-and-mockups-whats-the-best-option/*](https://www.justinmind.com/blog/wireframes-and-mockups-whats-the-best-option/)*, diakses 16 Juli 2020 10:07 WIB)*

*Amazon.com. 2020.* ***Apa itu Docker?****. (online). (*[*https://aws.amazon.com/id/docker/*](https://aws.amazon.com/id/docker/)*, diakses 7 Juli 2020 10:04 WIB)*

*Amazon.com. 2020.* ***Apa itu Redis?****. (online). (*[*https://aws.amazon.com/id/redis/*](https://aws.amazon.com/id/redis/)*, diakses 7 Juli 2020 10:28 WIB)*

*Bahiy. 2017.* ***Metode Waterfall (Kelebihan dan Kelemahan)****. (online). (*[*http://bahiy.blogger.mercubuana.ac.id/2017/09/15/metode-waterfall-kelebihan-dan-kelemahan/*](http://bahiy.blogger.mercubuana.ac.id/2017/09/15/metode-waterfall-kelebihan-dan-kelemahan/)*, diakses 16 Juli 2020 10:27 WIB)*

*Belaid, Rachid. 2015.* ***Introduction to PostgreSQL physical storage****. (online). (*[*http://rachbelaid.com/introduction-to-postgres-physical-storage/*](http://rachbelaid.com/introduction-to-postgres-physical-storage/)*, diakses 21 Maret 2018 10:59 WIB)*

*Ghazali, Muhammad. 2009.* ***Konsultan IT****. (online). (*[*https://muhammadghazali.wordpress.com/tag/deskripsi-pekerjaan-konsultan-it/*](https://muhammadghazali.wordpress.com/tag/deskripsi-pekerjaan-konsultan-it/)*, diakses 7 Juli 2020 16:21 WIB)*

*Guntoro. 2020.* ***Mengenal Pemrograman Berorientasi Objek dalam 5 Menit****. (online). (*[*https://badoystudio.com/pemrograman-berorientasi-objek*](https://badoystudio.com/pemrograman-berorientasi-objek)*, diakses 15 Juli 2020 15:58 WIB)*

*Kmkonline.co.id. 2015.* ***Cara Menggunakan PostgreSQL : Foreign Data Wrapper****. (online). (*[*https://blog.kmkonline.co.id/cara-menggunakan-postgresql-foreign-data-wrapper-8dff0ee82ab2*](https://blog.kmkonline.co.id/cara-menggunakan-postgresql-foreign-data-wrapper-8dff0ee82ab2)*, diakses 7 Juli 2020 11:53 WIB)*

*Pratama, Rikzy. 2019.* ***Cara Menulis Daftar Pustaka dari Buku, Jurnal, dan Internet****. (online). (*[*https://bocahkampus.com/cara-menulis-daftar-pustaka*](https://bocahkampus.com/cara-menulis-daftar-pustaka)*, diakses 7 Juli 2020 10:40 WIB)*

*Postman.com. 2020.* ***What is Postman?****. (online). (*[*https://www.postman.com*](https://www.postman.com/)*, diakses 13 Agustus 2020 17:56 WIB)*

*Saputro, Nur. 2019.* ***Pengertian PostgreSQL Beserta Fungsi, Kelebihan dan Kekurangannya****. (online). (*[*https://www.nesabamedia.com/pengertian-postgresql/*](https://www.nesabamedia.com/pengertian-postgresql/)*, diakses 7 Juli 2020 10:45 WIB)*

*Suherman, Edy, Lainatul Jannah, Miftahuda Adam I. N., Siti Ainun Zahroh, dan Nilli Rohmaniyah. 2017.* ***Deskripsi Perancangan Perangkat Lunak (DPPL) Halen Bookstore****. (online).*

*(*[*https://www.academia.edu/32875999/DESKRIPSI\_PERANCANGAN\_PERANGKAT\_LUNAK\_DPPL\_HALEN\_BOOKSTORE*](https://www.academia.edu/32875999/DESKRIPSI_PERANCANGAN_PERANGKAT_LUNAK_DPPL_HALEN_BOOKSTORE)*, diakses 7 Juli 2020 12:36 WIB), 1(1), 7*

*Wikipedia.org. 2019.* ***Antarmuka pemrograman aplikasi****. (online). (*[*https://id.wikipedia.org/wiki/Antarmuka\_pemrograman\_aplikasi*](https://id.wikipedia.org/wiki/Antarmuka_pemrograman_aplikasi)*, diakses 16 Juli 2020 09:22 WIB)*

*Wikipedia.org. 2019.* ***Git****. (online). (*[*https://id.wikipedia.org/wiki/Git*](https://id.wikipedia.org/wiki/Git)*, diakses 6 Juli 2020 18:16 WIB)*

*Wikipedia.org. 2019.* ***Monolithic application****. (online). (*[*https://en.wikipedia.org/wiki/Monolithic\_application*](https://en.wikipedia.org/wiki/Monolithic_application)*, diakses 16 Juli 2020 09:32 WIB)*

*Wikipedia.org. 2019.* ***Sistem manajemen basis data****. (online). (*[*https://id.wikipedia.org/wiki/Sistem\_manajemen\_basis\_data*](https://id.wikipedia.org/wiki/Sistem_manajemen_basis_data)*, diakses 6 Juli 2020 17:56 WIB)*

*Wikipedia.org. 2019.* ***Spesifikasi Kebutuhan Perangkat Lunak****. (online). (*[*https://id.wikipedia.org/wiki/Spesifikasi\_Kebutuhan\_Perangkat\_Lunak*](https://id.wikipedia.org/wiki/Spesifikasi_Kebutuhan_Perangkat_Lunak)*, diakses 7 Juli 2020 12:16 WIB)*

*Wikipedia.org. 2020.* ***Apache HTTP Server****. (online). (*[*https://id.wikipedia.org/wiki/Apache\_HTTP\_Server*](https://id.wikipedia.org/wiki/Apache_HTTP_Server)*, diakses 7 Juli 2020 10:09 WIB)*

*Wikipedia.org. 2020.* ***Exception handling****. (online). (*[*https://en.wikipedia.org/wiki/Exception\_handling*](https://en.wikipedia.org/wiki/Exception_handling)*, diakses 16 Juli 2020 12:21 WIB)*

*Wikipedia.org. 2020.* ***Feniks****. (online). (*[*https://id.wikipedia.org/wiki/Feniks*](https://id.wikipedia.org/wiki/Feniks)*, diakses 6 Juli 2020 17:47 WIB)*

*Wikipedia.org. 2020.* ***GitHub****. (online). (*[*https://id.wikipedia.org/wiki/GitHub*](https://id.wikipedia.org/wiki/GitHub)*, diakses 7 Juli 2020 09:57 WIB)*

*Wikipedia.org. 2020.* ***Laravel****. (online). (*[*https://id.wikipedia.org/wiki/Laravel*](https://id.wikipedia.org/wiki/Laravel)*, diakses 6 Juli 2020 18:07 WIB)*

*Wikipedia.org. 2020.* ***MVC****. (online). (*[*https://id.wikipedia.org/wiki/MVC*](https://id.wikipedia.org/wiki/MVC)*, diakses 6 Juli 2020 18:12 WIB)*

*Wikipedia.org. 2020.* ***Perencanaan sumber daya perusahaan****. (online). (*[*https://id.wikipedia.org/wiki/Perencanaan\_sumber\_daya\_perusahaan*](https://id.wikipedia.org/wiki/Perencanaan_sumber_daya_perusahaan)*, diakses 6 Juli 2020 17:52 WIB)*

*Wikipedia.org. 2020.* ***PHP****. (online). (*[*https://id.wikipedia.org/wiki/PHP*](https://id.wikipedia.org/wiki/PHP)*, diakses 6 Juli 2020 18:02 WIB)*

*Wikipedia.org. 2020.* ***Sandbox (software development)****. (online). (*[*https://en.wikipedia.org/wiki/Sandbox\_(software\_development)*](https://en.wikipedia.org/wiki/Sandbox_(software_development))*, diakses 6 Juli 2020 18:38 WIB)*

*Wikipedia.org.2020 .****Software design description****. (online). (*[*https://en.wikipedia.org/wiki/Software\_design\_description*](https://en.wikipedia.org/wiki/Software_design_description)*, diakses 7 Juli 2020 13:19 WIB)*

*Wikipedia.org. 2020.* ***Software documentation****. (online). (*[*https://en.wikipedia.org/wiki/Software\_documentation*](https://en.wikipedia.org/wiki/Software_documentation)*, diakses 15 Juli 2020 11:23 WIB)*

*Wikipedia.org. 2020.* ***Software project management****. (online). (*[*https://en.wikipedia.org/wiki/Software\_project\_management*](https://en.wikipedia.org/wiki/Software_project_management)*, diakses 15 Juli 2020 11:20 WIB)*

*Wikipedia.org. 2020.* ***Software test documentation****. (online). (*[*https://en.wikipedia.org/wiki/Software\_test\_documentation*](https://en.wikipedia.org/wiki/Software_test_documentation)*, diakses 15 Juli 2020 10:23 WIB)*

*Wikipedia.org. 2020.* ***Website wireframe****. (online). (*[*https://en.wikipedia.org/wiki/Website\_wireframe*](https://en.wikipedia.org/wiki/Website_wireframe)*, diakses 16 Juli 2020 09:53 WIB)*

*Y., Akbar W, Egi Tri F, Hartono, Ibnu Faisal, Ken Dio O, M. Fazlurrahman, Mulya P. H, Rena Syafitri. 2015.* ***Dokumentasi Project RPL "Aplikasi Manajemen Perangkat Lunak Berbasis Dekstop"****. (online).*

*(*[*https://www.academia.edu/14955510/Dokumentasi\_Project\_RPL\_Aplikasi\_Manajemen\_Perangkat\_Lunak\_Berbasis\_Dekstop\_*](https://www.academia.edu/14955510/Dokumentasi_Project_RPL_Aplikasi_Manajemen_Perangkat_Lunak_Berbasis_Dekstop_)*, diakses 7 Juli 2020)*

*Zaiste. 2014.* ***Table inheritance and partitioning with PostgreSQL****. (online). (*[*https://zaiste.net/posts/table\_inheritance\_and\_partitioning\_with\_postgresql/*](https://zaiste.net/posts/table_inheritance_and_partitioning_with_postgresql/)*, diakses 21 Maret 2018 11:40 WIB)*

|  |  |
| --- | --- |
| **BAB 10.** | **LAMPIRAN** |
|  |

BAB I.

1. Definisi Case
2. lll

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE OBJECT** | **MVC**  **OBJECT** | **CLASS** | **METHOD** |
| **User Login** | **Model** | * **Main Class** * **Namespace :**   **ERP\Model**   * **Class Name :**   **UserLoginTable**   * **File Location :**   **[ROOT]/module/ERP/src/Controller/ UserLoginTable.php** | * **Name :**   **fetchAll()**   * **Description :**   Menampilkan seluruh Record Data User |
| * **Name :**   **getAuthenticationStatus()**   * **Description :**   Menampilkan status dari otentikasi Nama dan Password User |
| * **Additional Class** * **Namespace :**   **(None)**   * **Class Name :**   **(None)**   * **File Location :**   **(None)** |  |
|  |  |  |
| **Controller** | * **Main Class** * **Namespace :**   **ERP\Controller**   * **Class Name :**   **UserLoginController *extends* AbstractActionController**   * **File Location :**   **[ROOT]/module/ERP/src/Controller/ UserLoginController.php**   * **Routing (*module.config.php*) :** * **[/:action]** * **/UserLogin[/:action]** | * **Name :**   **indexAction()**   * **Description :**   Index *default* |
| * **Name :**   **authenticateAction()**   * **Description :** |
| * **Name :**   **cancelAction()**   * **Description :** |
| * **Additional Class** * **Namespace :**   **(None)**   * **Class Name :**   **(None)**   * **File Location :**   **(None)** |  |
|  |  |  |
| **View** | * **Controller Caller Method :**   **UserLoginController::**  **indexAction()**   * **File Location :**   **[ROOT]/module/ERP/view/erp/**  **user-login/index.phtml** |  |
| * **Controller Caller Method :**   **UserLoginController::**  **authenticateAction()**   * **File Location :**   **[ROOT]/module/ERP/view/erp/**  **user-login/authenticate.phtml** |  |
| * **Controller Caller Method :**   **UserLoginController::**  **cancelAction()**   * **File Location :**   **[ROOT]/module/ERP/view/erp/**  **user-login/cancel.phtml** |  |
|  |  |  |  |

**ZEND FRAME MAIN CONFIGURATION**

|  |  |
| --- | --- |
| **FILE NAME** | **PROPERTIES** |
| **global.php** | * **File Location :**   **[ROOT]/config/autoload/global.php**   * **Last Modification :**   **2017-02-28**   * **Description :**   Konfigurasi global sistem |
| *<?*  *use Zend\Db\Adapter;*  *use Zend\ServiceManager\Factory\InvokableFactory;*  *return [*  *'db' => [*  *'driver' => 'Pdo',*  *'dsn' => 'pgsql:host=192.168.1.210;dbname=dbERPReborn',*  *'adapters' => [*  *'DBAdapterWrite' => [*  *'driver' => 'Pdo',*  *'dsn' => 'pgsql:host=192.168.1.210;dbname=dbERPReborn',*  *],*  *'DBAdapterReadOnly' => [*  *'driver' => 'Pdo',*  *'dsn' => 'pgsql:host=192.168.1.210;dbname=dbERPReborn',*  *],*  *],*  *],*    *'service\_manager' => [*  *'factories' => [*  *'Zend\Db\Adapter\Adapter' => 'Zend\Db\Adapter\AdapterServiceFactory',*  *'Application\Db\DBAdapterWrite' => AdapterAbstractServiceFactory::class,*  *'Application\Db\DBAdapterReadOnly' => AdapterAbstractServiceFactory::class,*  *],*  *'abstract\_factories' => [*  *'Zend\Db\Adapter\AdapterAbstractServiceFactory',*  *],*  *],*  *];*  *?>* | |

|  |  |
| --- | --- |
| **FILE NAME** | **PROPERTIES** |
| **erp.local.php** | * **File Location :**   **[ROOT]/config/autoload/erp.local.php**   * **Last Modification :**   **2017-02-28**   * **Description :**   Konfigurasi lokal dari sistem. Didalamnya terdapat *user name* dan *password default* ke dalam *database* |
| *<?*  *return [*  *'db' => [*  *'username' => 'SysAdmin',*  *'password' => '748159263',*  *'adapters' => [*  *'DBAdapterWrite' => [*  *'username' => 'SysAdmin',*  *'password' => '748159263',*  *],*  *'DBAdapterReadOnly' => [*  *'username' => 'SysAdmin',*  *'password' => '748159263',*  *],*  *],*  *],*  *];*  *?>* | |

|  |  |
| --- | --- |
| **URL** | **http://[ HOST ]/WebService/System/getUserAuthentication** |
| **Definition** | Method ini digunakan untuk mengotentifikasi apakah user dapat masuk kedalam sistem atau tidak |
| **Component** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **🢝** | **Controller** | | |  | |  | 🢝 | File Path | : | ***[ ROOT ]/module/ERP/src/Controller/WebService-System-Controller.php*** | |  | 🢝 | Class | : | ***WebService\_System\_Controller*** | |  | 🢝 | Method | : | ***getUserAuthenticationAction()*** | |  | | | | | | **🢝** | **Model** | | |  | |  | 🢝 | File Path | : | ***[ ROOT ]/module/ERP/src/Controller/WebService-System-Model.php*** | |  | 🢝 | Class | : | ***WebService\_System\_Model*** | |  | 🢝 | Method | : | ***GetUserAuthentication*** | |  | | | | | | **🢝** | **View** | | |  | |  | 🢝 | File Path | : | ***[ ROOT ]/module/ERP/view/erp/web-service\_system\_/get-user-authentication.phtml*** | |  | | | | | |
| **Controller Data Parsing** | |  |  |  |  |  | | --- | --- | --- | --- | --- | | **🢝** | **GET** | | : | **-** | |  | | | | | | **🢝** | **POST** | | : | * **UserName** * **UserPassword** * **NavigatorUserAgent** * **NavigatorPlatform** | |  | | | | | | **🢝** | **WebService** | | |  | |  | 🢝 | Request (JSON) | : | **-** | |  | 🢝 | Respond (JSON) | : | * **success** * **message** | |  | | | | | | **🢝** | **SESSION** | | |  | |  | 🢝 | Set | : | **-** | |  | 🢝 | Get | : | **-** | |  | | | | | |
| **Model Database Object** | |  |  |  |  | | --- | --- | --- | --- | | **🢝** | **Stored Procedure** | **:** | **"SchSysConfig"."FuncSys\_General\_GetLoginAuthentication"** | |  | | | | | **🢝** | **Table** | : | *-* | |  | | | | | **🢝** | **View** | : | *-* | |  | | | | |
| **Controller Syntax** | public function getUserAuthenticationAction()  {  $varData = $this->varObjModel->GetUserAuthentication(  $\_SESSION['ZhtFW']['API']['DataParsing']->GetDataParsing('UserName'),  $\_SESSION['ZhtFW']['API']['DataParsing']->GetDataParsing('UserPassword'),  $\_SESSION['ZhtFW']['API']['DataParsing']->GetDataParsing('NavigatorUserAgent'),  $\_SESSION['ZhtFW']['API']['DataParsing']->GetDataParsing('NavigatorPlatform'),  $\_SESSION['ZhtFW']['API']['BasicUtility']->GetHostIPAdress(),  $\_SESSION['ZhtFW']['API']['BasicUtility']->GetHostMACAddress(),  $\_SESSION['ZhtFW']['API']['BasicUtility']->GetHostName()  );    $varJSONDataResponse =  $this->JSONPageRoutine  ([  'success' => boolval($varData['Status']),  'message' => $varData['Notification']  ]);  return $varJSONDataResponse;  } |
| **Model Syntax** | public function GetUserAuthentication($varUserName, $varUserPassword, $varNavigatorUserAgent, $varNavigatorPlatform, $varHostIPAddress, $varHostMACAddress, $varHostName)  {  echo ($this->varVerboseMode==false?'':'[ '.(static::class).' ] '.'Call GetUserAuthentication Method'.'<br>');  $varSQLQuery = "  SELECT  \"Status\"::int AS \"Status\",  \"Notification\"  FROM  \"SchSysConfig\".\"FuncSys\_General\_GetLoginAuthentication\"  (  '".$varUserName."',  '".$varUserPassword."',  '".$varNavigatorUserAgent."',  '".$varNavigatorPlatform."',  '".$varHostIPAddress."',  ".((strcmp($varHostMACAddress, '')==0) ? "NULL" : "'".$\_SESSION['ZhtFW']['API']['BasicUtility']->GetHostMACAddress()."'").",  '".$varHostName."'  )  ";  $varReturn = $\_SESSION['ZhtFW']['API']['Database']->GetQuery(new \Zend\Db\Sql\Sql($this->varDBAdapter), $varSQLQuery);  return [  'Status' => $varReturn['Data'][0]['Status'],  'Notification' => $varReturn['Data'][0]['Notification'],  'UserLoginSession' => $varReturn['Data'][0]['UserLoginSession']  ];  } |