Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 6 No. 1 Januari 2024 Hal. 1-10
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

Penerapan Web Api Untuk Sentralisasi Data Sistem Pembukuan

Henrikus Karel Dwiputra[*a], Argo Wibowo[b] , Umi Proboyekti[c]
[a]Sistem Informasi, Fakultas Teknologi Informasi, Universitas Kristen Duta Wacana Yogyakarta, henrikus.karel@si.ukdw.ac.id
[b]Sistem Informasi, Fakultas Teknologi Informasi, Universitas Kristen Duta Wacana Yogyakarta, argo@staff.ukdw.ac.id
[c]Sistem Informasi, Fakultas Teknologi Informasi, Universitas Kristen Duta Wacana Yogyakarta, othie@staff.ukdw.ac.id

Submitted: 10-10-2023, Reviewed: 31-10-2023, Accepted 21-11-2023
https://doi.org/10.47233/jteksis.v6i1.1073

Abstract
One type of financial business is a provider of lending services in the form of money or goods. The Remedial Department is a department in a company that operates in the financial sector and has the task of marking, evaluating and recovering loans that have been issued by the company. In carrying out its duties, the Remedial Department already has a system called Harbag, but this system is considered ineffective because it only contains HTML and does not have a database and architecture. So the system can only perform input and output. Therefore, the Remedial Department must input data several times. This research offers a solution in the form of creating an API that is useful for creating a system to help the work of the Remedial Department. This system is integrated with a web service which allows data to be centralized. Because there is centralized data, the Remedial Department does not need to enter data repeatedly because the data can be accessed between branches. This research was developed using the Personal Extreme Programing (PXP) method and utilized PL/SQL in Oracle and SOA in SoftwareAG. At the end of the research, it was obtained that the software was quite suitable for use, namely experiencing 89.8% success in the tests that had been carried out and 10.2% failure.
Keywords: *web service, data centralization, Personal Extreme Programing, PL/SQL, SOA*

Abstrak
Salah satu jenis usaha finansial adalah penyedia jasa peminjaman dana berupa uang atau barang. Departemen Remedial merupakan departemen yang ada pada perusahaan yang bergerak di bidang keuangan dan memiliki tugas untuk menandai, mengevaluasi, dan melakukan pemulihan terhadap pinjaman-pinjaman yang telah dikeluarkan perusahaan. Dalam melakukan tugasnya Departemen Remedial sudah memiliki sistem yang bernama Harbag, namun sistem ini dinilai tidak efektif karena hanya berisi HTML dan tidak memiliki *database* dan arsitektur. Sehingga sistem tersebut hanya dapat melakukan *input* dan *output* saja. Oleh karena itu Departemen Remedial harus melakukan beberapa kali *input* data. Penelitian ini menawarkan solusi berupa pembuatan API yang berguna untuk pembuatan sistem dalam membantu pekerjaan Departemen Remedial. Sistem ini terintegrasi dengan *web service* yang memungkinkan data dapat tersentralisasi. Karena adanya data yang tersentralisasi, Departemen Remedial tidak perlu memasukkan data berulang kali karena data dapat diakses antar cabang. Penelitian ini dikembangkan dengan metode *Personal Extreme Programing (PXP)* dan memanfaatkan PL/SQL dalam Oracle dan SOA dalam SoftwareAG. Di akhir penelitian didapatkan software yang cukup layak digunakan yakni mengalami 89,8% keberhasilan dalam pengujian yang telah dilakukan dan 10,2% kegagalan.
Keywords: *web service, sentralisasi data, Personal Extreme Programing, PL/SQL, SOA*

PENDAHULUAN

Usaha finansial merupakan usaha yang berfokus pada bidang keuangan. Salah satu jenis usaha finansial yaitu penyedia jasa peminjaman dana berupa uang atau barang. Sistem kerja dari penyedia jasa peminjaman tersebut, perusahaan akan meminjamkan dana berupa uang kepada pelanggan yang nantinya pelanggan tersebut harus melakukan pembayaran secara berskala hingga dana pengembalian yang telah diputuskan oleh perusahaan selesai terbayarkan. Ada juga perusahaan penyedia jasa peminjaman uang yang membutuhkan jaminan berupa barang atau surat dari pelanggan. Ketika nominal pengembalian terbayarkan maka barang atau surat tersebut akan dikembalikan kepada pelanggan. Dalam menjalankan bisnis finansial tersebut ada pelanggan-pelanggan yang bermasalah dalam pembayaran seperti jatuh tempo pembayaran tidak ditepati dan tidak ada pembayaran menyebabkan perusahaan penyedia jasa peminjaman keuangan mengalami kerugian dan berpeluang menyebabkan

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 6 No. 1 Januari 2024 Hal. 623-632
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

kebangkrutan. Salah satu kesalah pahaman antar anggota koperasi mengenai pembayaran angsuran pinjaman adalah keterbatasan daya ingat manusia [11].

Perusahaan melakukan pencegahan dengan cara membuat departemen-departemen yang dapat membantu masalah kerugian tersebut. Salah satunya yaitu departemen Remidial. Departemen Remidial bertugas untuk menandai, mengevaluasi, dan melakukan pemulihan terhadap pinjaman-pinjaman yang telah dikeluarkan perusahaan. Transaksi atau pinjaman yang mengalami kendala tersebut ditandai sebagai transaksi yang merugikan. Transaksi yang ditandai dengan kerugian, dievaluasi agar dari kerugian tersebut perusahaan masih mendapatkan keuntungan. Upaya mendapatkan penghasilan disaat transaksi rugi ini disebut pemulihan.

Departemen Remidial membutuhkan sebuah sistem yang dapat membantu kebutuhan pekerjaan tersebut. Departemen Remidial memiliki sistem bernama Harbag yang beberapa tahun belakang sudah tidak digunakan lagi, dikarena keterbatasan fitur dalam sistem. Sistem Harbag belum memiliki arsitektur dan *database*, sehingga sistem hanya dapat membantu pekerjaan pemasukan data transaksi saja. Departemen Remidial harus melakukan perulangan pemasukan data antar cabang, karena pelanggan yang berpindah-pindah kota. Oleh karena itu departemen Remidial mengalami kesulitan karena data antar cabang tidak menjadi satu. Departemen Remidial perlu–sistem baru yang memiliki fitur yang lebih dari pada sistem Harbag. Sistem baru ini diharapkan dapat membantu departemen Remidial untuk kebutuhan kinerja dan mengatasi kendala-kendala yang ada pada sistem Harbag.

Aplikasi berbasis web merupakan salah sagu bentuk perubahan telnologi yg mendukung kebutuhan spesifik. Fungsi dari sistem informasi adalah utk memproses, mengumupulkan, mengirimkan, dan menyimpan informasi sehingga dapat digunakan utk membantu pengambilan keputusan dalam sebuah organisasi [8]. Pengguna dapat menyesuaikan isian data pada sebuah sistem dengab menggunakan web [18].

Permasalahan yang sering dialami oleh pengembang adalah melakukan sinkronisasi sistem yang memiliki perbedaan baik dari segi bahasa pemrograman, gawai, atau *platform* [19]. Kesulitan yang dialami adalah belum tersedianya layanan web atau *web service* yang mampu untuk menyinkronisasi sistem informasi akademik dengan sistem lain [1]. Hal ini menyebabkan sulitnya pengembangan sistem untuk pertukaran, pengolahan, serta sinkronisasi data. Mengambangkan *web service* untuk melakukan sinkronisasi data terkait paisen Covid dikarenakan data Covid yang tersedia hanya mencakup dunia dan Indonesia, tanpa informasi rinci untuk setiap provinsi yang ada di Indonesia [4]. Pengembangan *web service* ini bertujuan untuk mendata perkembangan Covid-19 di provinsi Sulawesi Selatan agar dapat diakses oleh khalayak umum [3]. *Web service* atau layanan *web* adalah mekanisme komunikasi yang ditentukan antara sistem komputer yang berbeda. Layanan *web* memudahkan komunikasi *peer-to-peer* yang dipersonalisasi dan dapat mencakup banyak *platform* [6]. Permasalahan yang dialami adalah terdapat 30 pasar tradisional yang ada di Yogyakarta dimana masing - masing memiliki keunikan dan kekhasan tersendiri. Namun demikian tidak ada upaya dalam penyatuan informasi mengenai karakteristik serta kekhasan dari masing - masing pasar tradisional tersebut [9].

Solusi yang dilakukan adalah pembuatan *web service* dengan teknologi REST [2]. Dalam pembuatan REST, menggunakan *Service Oriented Architecture* (SOA) yang merupakan salah satu konsep arsitektur pada perangkat lunak [2]. *Service Oriented Architecture*(SOA) merupakan sebuah pendekatan arsitektur dengan *service* sebagai konstruksi dasar untuk pengembangan yang cepat dengan biaya yang rendah dan memudahkan pengaturan komposisi aplikasi terdistribusi walaupun dalam lingkungan yang heterogen [10]. Sedangkan *Representational State Transfer (*REST) adalah *web service* yang menerapkan konsep perpindahan antar *state* dimana REST bernavigasi melalui link *Hypertext Trasnfer Protocol (*HTTP) untuk melakukan aktivitas tertentu [13]. Kelebihan yang dimiliki REST adalah interaksi dengan menggunakan HTTP internet. Interasksi tersebut adalah hal yang wajar saat ini, seperti penggunaan kode komunikasi menggunakan kode status HTTP standar. Selain itu REST mengandalkan enskripsi *Secure Sockets Layer* (SSL) dan *Transport Layer Security* (TLS) sehingga detail enskripsi dan integritas transport data tidak diselesaikan dengan penambahan kerangka kerja [16]. Dengan pembuatan *web service* tersebut sangat membantu untuk pengembangan sistem informasi akademik STT Terpadu Nurul Fikri. REST *web service* pada pengembangan sistem informasi akademik STT Terpadu Nurul Fikri dapat digunakan untuk menghubungkan pertukaran data meskipun terdapat perbedaan bahasa pemrograman, gawai, atau *platform.* Sedangkan pada penelitian Baharudin solusi yang dilakukan adalah dengan pembuatan *webiste* yang menggunakan *web service* dengan metode REST API. Metode REST API dimanfaatkan untuk proses pengambilan data [4]. Tujuan dari pengambilan data adalah untuk menyatukan berbagai macam data yang berasal dari

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 6 No. 1 Januari 2024 Hal. 623-632
http://jurnal.unidha.ac.id/index.php/jteksis
E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

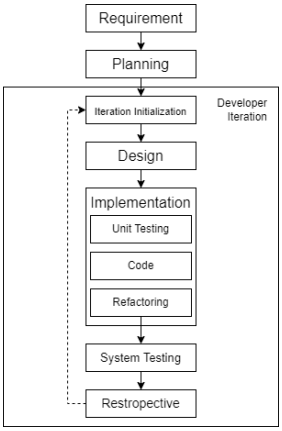
database yang berbeda. Database menggunakan PL/SQL. *Procedural Language/ Structure Query Language* (PL/SQL) merupakan bahasa prosedural yang disajikan dalam bentuk blok atau skrip [12]. Selanjutnya untuk solusi yang dilakukan oleh Filiana adalah pengembangan REST API menggunakan *microframework* PHP yang berhubungan dengan *framework* Laravel [9]. *Application Programming Interface* (API) merupakan salah satu cara yang dapat digunakan untuk mengakses aplikasi dan berkomunikasi antar sistem yang berbeda *platform*. Hal tersebut disebabkan karena API memanfaatkan konsep fungsi antarmuka pemrograman aplikasi [15]. Pengembangan REST API ini menggunakan model *incremental* dengan tiga iterasi. Tujuan dari pembuatan REST API ini adalah agar informasi terkait pasar tradisional di Yogyakarta dapat diakses dari berbagai sisi, seperti kelas, nama alternatif, lokasi, deskripsi, sejarah dan lain sebagainya.

Menurut Filiana dalam pembangunan REST API untuk pasar tradisional memiliki kelebihan yaitu data dapat diakses secara fleksibel dari berbagai *platfrorm* [9]. Selain itu dengan menggunakan REST API akses informasi menjadi lebih mudah dan memberikan akurasi serta konsistensi sebuah data.

Penelitian ini membangun sistem baru yang memiliki arsitektur dan *database* untuk menggantikan sistem Harbag. Fungsi *database* dan arsitektur ini memungkinkan pengguna untuk memiliki sebuah *database* yang tersentralisasi dengan tujuan untuk mencegah terjadinya pemasukan data ganda antar cabang. Sistem menggunakan arsitektur RESTful Web API yang memungkinkan pertukaran data antar sistem menjadi lebih aman karena pengguna tidak langsung mengakses *database* tetapi mengakses API yang terhubung dengan *database*. *Web* API mendukung fungsi *Create Read Update Delete* (CRUD) yang bekerja melalui HTTP *protocol* dengan menggunakan method *GET, POST, PUT* dan *DELETE*. Selain itu, *web* API memiliki response *Accept Header* dan HTTP status *code. Response* yang dihasilkan oleh API memiliki format yaitu JSON dan XML [19] .Sistem baru yang dibuat merupakan sistem internal departemen Remidial yang bernama *Recovery, Recognition, Registration* (RRR).

METODE PENELITIAN

Metode yang digunakan dalam pembangunan sistem RRR ini adalah *Personal Extreme Programing.* PXP adalah pengembangan perangkat lunak yang disusun dan dilakukan secara individual [17]. Pengembangan dengan metode PXP bersifat iteratif sehingga dalam penerapannya memungkinkan terjadinya perubahan yang fleksibel dan responsive [5]. Extreme programing merupakan metode yang tepat dalam membuat aplikasi yang memerlukan adaptasi pada masa pengembangan [8]. [7] Langkah – langkah pada metode ini dapat dilihat pada Gambar 1.



Gambar 1 Alur metodologi PXP

Requirement: proses requirement merupakan saat dimana eluruh dokumen pendukung yang dibutuhkan dalam pengembangan perangkat lunak akan disiapkan.

Planning: proses planning merupakan proses penyusunan tugas sesuai dengan kebutuhan. Setiap tugas berisi tugas-tugas lebih kecil yang telah dikategorikan dan diberi waktu estimasi pengerjaan.

Iteration Initialization: Proses ini merupakan proses yang menunjukan awal iterasi. Pada proses ini ditentukan pemilihan tugas sebagai fokus utama iterasi. Panjang pengulangan iterasi dapat bervariasi sesuai dengan ruang lingkup projek.

Design: pada proses ini seorang pengembang diharuskan untuk memodelkan sistem yang akan diimplementasikan selama proses iterasi berlangsung. Pada proses ini, pengembang harus memodelkan sesuai dengan kebutuhan yang diminta oleh klien tanpa menambahkan fitur-fitur baru yang mungkin dapat dibutuhkan di masa yang akan datang.

Implemetation. Pada fase ini pengembang mulai membuat code. Fase implementation dibagi menjadi 3 fase kecil yaitu, unit testing, pembuatan kode, dan refactor. Untuk menyelesaikan fase implementation kode yang dibuat harus dapat dikompilasi tanpa eror dan berhasil melewati unit testing.

System testing. Testing merupakan sebuah teknik dalam pengujian dimana untuk menguji inputan yang sudah dibagi dalam beberapa kelompok sesuai dg fungsinya akan dibuat test case yang berbeda [14]. Pada fase ini seluruh fitur yang dibuat diuji selama fase pengujian sistem. Pengujian

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 6 No. 1 Januari 2024 Hal. 623-632
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

fungsionalitas bertujuan untuk memastikan sebuah sistem dapat melakukan fungsionalitas sesuai yang diharapkan. Seperti apakah sebuah sistem dapat mengidentifikasi persyaratan pengguna untuk dapat *login* atau apakah sebuah API yang digunakan untuk *get data* pengguna menghasilkan *response* yang sesuai. Hal ini dapat dilakukan dengan menggunakan teknik desain uji berbasis spesifikasi dan struktur [20]. Apabila ditemukan sebuah fitur yang tidak berjalan sebagaimana mestinya, maka pengembang harus memperbaiki sampai fitur tersebut berjalan sesuai requirement.

Retrospective. Fase ini merupakan fase akhir dari sebuah iterasi. Pengembang harus memverifikasi apakah perkiraan waktu pengerjaan yang telah ditetapkan pada fase planning sesuai atau tidak. Pada fase ini dapat dilanjutkan dengan iterasi yang baru dengan memasuki fase Iteration Initialization untuk menandai akhir pengembangan projek apabil seluruh kebutuhan sistem telah terpenuhi.

HASIL DAN PEMBAHASAN

a. Requirements

Sistem yang dibangun diperlukan untuk memenuhi kebutuhan pengguna. Kebutuhan-kebutuhan sistem tersebut dituliskan dalam bentuk skenario. Skenario yang dituliskan oleh pengguna seperti pada Table 1.

Table 1 Skenario kebutuhan sistem

Skenario:
Departemen remidial melakukan pencatatan transaksi terhadap pinjaman-pinjaman yang diotorisasi perusahaan. Transaksi peminjaman yang mengalami kendala dikumpulkan menjadi satu untuk dilakukan evaluasi dan pemulihan agar perusahaan tidak merugi. Evaluasi transaksi peminjaman yang terkendala pembayarannya dilakukan 3 tahapan yaitu: recovery atau pemulihan, recognition atau pengakuan, dan registration atau pendaftaran. Recovery adalah wewenang Head Office pada Departemen Remidial di setiap cabang. Head Office berwewenang mengevaluasi dan menentukan transaksi tersebut mendapat kesempatan recovery atau tidak. Transaksi yang dinilai mendapat recovery selanjutnya mendapat otorisasi untuk masuk tahap recognition. Pada tahap recognition, Coordinator dan Head Office mengevaluasi transaksi di setiap cabang. Transaksi yang dinilai perlu ditinjau kembali di tahap recovery ditarik mundur ke tahap recovery. Proses tersebut bernama reanalyze atau peninjauan ulang. Sebaliknya, transaksi yang telah dinilai tahap recognition menjalani otorisasi untuk kemudian memasuki tahap registration yang mendaftarkannya sebagai transaksi yang dapat dipulihkan. Pengaturan transaksi pada tahap registration dilakukan oleh Officer yang diawasi Head Office di setiap cabang. Bila ditahap registration terdapat transaksi yang dinilai perlu peninjauan ulang, maka transaksi tersebut kembali ke tahap reconition. Perhitungan jumlah nominal transaksi untuk setiap area atau kota, cabang, dan nasional dilakukan pada tahap registration. Perhitungan untuk nasional artinya penjumlahan semua transaksi pada tahap registration semua cabang ataupun area. Hasil perhitungan tampil pada dashboard yang tersedia bagi seluruh cabang. Head Office berwenang menambahkan transaksi ke dalam tahap recovery apabila terdapat transaksi baru yang mengalami kendala pembayaran. Head Office berwenang menghapus transaksi apabila terjadi kesalahan data pada transaksi baru tersebut dan melakukan perubahan cabang terhadap transaksi apabila transaksi tersebut telah berpindah cabang atau area.

Skenario tersebut dapat dibentuk *user story* sebagai acuan dalam pembuatan sistem. Hasil pembentukan user story dapat dilihat pada Table 2.

Table 2 User story

User Story Code	***User Stories***
Story-01	*User* dapat melakukan *login*
Story-02	*Head Office* dapat melakukan pengolahan data transaksi
Story-03	*User* dapat melakukan proses *authorize* & *reanalyze*
Story-04	*User* dapat mengakses data perhitungan/*overview*

Pembentukan *user story* dapat digunakan untuk menentukan *task* yang dibutuhkan dalam pembuatan sistem. *Task* dapat berupa pembuatan *database*, prosedur, dan *service* atau API. *Task* yang ditentukan dari *user story* dapat dilihat pada Table 3.

Table 3 Task dalam pembangunan sistem RRR

User Stories Code	***Task Code***	**Task Description**
Story-01	Task-01	Pembuatan API *Login*
	Task-02	Pembuatan API *Logout*
Story-02	Task-03	Pambuatan API Tambah data Transaksi
	Task-04	Pembuatan API Ambil semua Data Transaksi
	Task-05	Pembuatan API Ubah Cabang
	Task-06	Pembuatan API Hapus data Transaksi
Story-03	Task-07	Pembuatan API Ambil data transaksi berdasarkan status
	Task-08	Pembuatan API Proses *Authorize*
	Task-09	Pembuatan API Proses *Reanalyze*
Story-04	Task-10	Pembuatan API Ambil data *Overview* Cabang
	Task-11	Pembuatan API Ambil data *Overview* Area
	Task-12	Pembuatan API Ambil data *Overview* Nasional

Tabel 2 merupakan tabel hasil penentuan *task* untuk setiap *user story*. Terdapat 12 *task* yang berupa pembuatan API. Tahap berikutnya akan ditentukan *planning* untuk menentukan jangka waktu proses pengerjaan *task* berdasarkan jumlah iterasi yang ditentukan,

b. Planning

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 6 No. 1 Januari 2024 Hal. 623-632
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

Tahap perencanaan dilakukan untuk menentukan waktu proses pengerjaan dan penentuan masing-masing iterasi dari *user story* yang telah ditentukan. Waktu yang direncanakan akan berlangsung selama 1 bulan. Dalam 1 bulan tersebut akan dibagi menjadi 4 iterasi sehingga 1 iterasi akan terhitung 1 minggu. Dari *user story* yang telah ditentukan, dilakukan penentuan *story points* dimana setiap *story points* menunjukkan ukuran atau jangka waktu pengerjaan sebuah produk *backlog* setiap iterasi. *Story points* terbagi menjadi 1, 2, dan 3. Nilai 3 menentukan *story points* untuk *user story* tersebut tertinggi dan berurutan hingga nilai terkecil. Rencana iterasi yang dilakukan dapat dilihat pada Tabel 4.

Table 4 Perencanaan Iterasi

Iteration Code	***Task Code***	***Story Points***
Iterasi-01	Task-01	3
	Task-02	1
Iterasi-02	Task-03	2
	Task-04	1
	Task-05	1
	Task-06	1
Iterasi-03	Task-07	2
	Task-08	3
	Task-09	3
Iterasi-04	Task-10	2
	Task-11	2
	Task-12	1

c. *Iteration Initialization*

Iteration Inialization merupakan tahap awal dari setiap iterasi yang akan di laksanakan. Iterasi tersebut dimulai dengan pemilihan tugas, yang mana nantinya akan menjadi fokus utama dari iterasi tersebut. Tugas-tugas tersebut didapatkan dari hasil perencanaan pada tahap *planning*. *Interation Initialization* dapat digunakan bila pengguna ingin melakukan *requirement* tambahan.

d. *Design*

Dalam tahap *design* membuat rancangan untuk memenuhi semua *task* pada setiap iterasi. Pada tahap *design* setiap iterasi akan dilakukan perancangan *database*, prosedur atau *logic*, dan API.

- Iterasi-01

Untuk memenuhi *task* dalam iterasi-01 diperlukan *design database*, prosedur, dan API. *Database* yang diperlukan dalam iterasi-01 adalah tabel area, tabel jabatan, tabel level, tabel pengguna, dan tabel *login*. Prosedur yang diperlukan dalam iterasi-01 adalah prosedur *login*, prosedur *check token*, dan prosedur *logout.* API yang diperlukan dalam iterasi-01 adalah API *login* dan *logout. Design* pada iterasi-01 dapat dlihat pada Tabel 5.

Table 5 Design iterasi-01

Task Code	***Database***	***Prosedur***	***Web API***
Task-01	Tabel Pengguna	Prosedur *Login*	API *Login*
	Tabel Jabatan		
	Tabel Level		
	Tabel Area		
	Tabel *Login*	Prosedur *Check Token*	
Task-02	Tabel Pengguna	Prosedur *Logout*	API *Logout*
	Tabel *Login*	Prosedur *Check Token*	

Tabel Pengguna merupakan rancangan dari entitas pengguna. Fungsi tabel tersebut sebagai penampung dari data-data pengguna yang terdaftar sebagai pengguna sistem. Terdapat 3 jenis pengguna dalam sistem terbagi dari level 1 sampai dengan 3. Level 1 menandakan bahwa pengguna dapat melakukan akses ke semua fitur yang ada dalam sistem. Level 1 biasanya digunakan oleh pengguna dengan jabatan Officer. Level 1 juga mengatur data pengguna yang ada dalam sistem. Level 2 dan 3 memiliki keterbatasan dalam penggunaan fitur yang ada dalam sistem. Level 2 dapat melakukan proses authorize data transaksi dari tahap recognition ke tahap registration dan proses reanalyze data transaksi dari tahap recognition ke tahap recovery. Level 2 biasanya digunakan oleh pengguna dengan jabatan Coordinator. Level 3 hanya dapat melakukan proses reanalyze data transaksi dari tahap registration ke tahap recognition. Level 3 biasanya digunakan oleh pengguna dengan jabatan Head Office. Level 2 dan 3 tidak dapat mengatur data pengguna.

Tabel Jabatan berisi jabatan yang ada pada Departemen Remidial. Data pada entitas jabatan digunakan sebagai indentitas jabatan untuk pengguna. Terdapat banyak jabatan dalam Departemen Remidial yang tidak dapat disebutkan satu-persatu karena hal tersebut merupakan rahasia dari mitra.

Tabel Level berisi level 1, 2, dan 3 yang digunakan sebagai identitas level untuk pengguna. Seperti yang dijelaskan pada tabel pengguna bahwa setiap level memiliki perannya masing-masing. Apabila kebutuhan Departemen Remidial bertambah dan memerlukan level tambahan, departemen dapat menambahkan level ke dalam tabel tersebut.

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 6 No. 1 Januari 2024 Hal. 623-632
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

Tabel Area berisi area dari setiap cabang kantor mitra yang ada di seluruh Indonesia. Tabel ini diperlukan agar transaksi dapat dipisahkan berdasarkan area atau cabang. Tujuannya agar setiap cabang dengan jelas melakukan pemrosesan data transaksi sesuai cabang yang mereka tempati. Tabel ini juga yang membantu dalam sentralisasi basis data transaksi yang diperlukan Departemen Remidial.

Tabel *Login* merupakan rancangan dari entitas aktivitas *login* pengguna. Fungsi tabel tersebut menyipan data pengguna yang sedang atau telah melakukan *login*. Sistem kerjanya jika pengguna yang sedang *login* belum tercatat di Entitas *login*, maka data pengguna ditambahkan ke tabel tersebut. Namun jika telah tercatat dalam tabel, data diubah berdasarkan tanggal login dan token yang telah dibuat sistem.

Prosedur *Login* dimulai dengan memasukan data input. Data input berupa username menjadi parameter prosedur untuk mencari data pengguna menggunakan username yang telah dimasukan. Jika data tidak ditemukan prosedur mengembalikan pesan error. Jika data ditemukan, dilakukan perbandingan password yang dimasukan dengan yang ada pada tabel. Jika password tidak sesuai, prosedur mengembalikan pesan error. Sementara jika sesuai prosedur mengeluarkan data pengguna. Pada data pengguna tersebut dilakukan pengecekan pada status, Jika status bernilai Y prosedur mencari data pengguna pada tabel login, sesuai dengan username. Jika tidak ditemukan, maka pengguna tersebut didaftarkan pada tabel login. Sementara jika ditemukan, prosedur melakukan pengecekan token yang ada dalam tabel sudah kadaluarsa atau tidak. Jika belum kadaluarsa maka proses login akan berhasil.

Prosedur *Check Token* dimulai dengan input data berupa p_token. Setelah sistem menerima input, sistem akan mencari data login sesuai dengan token yang telah di-input. Apabila data tidak ditemukan sistem akan mengirimkan pesan gagal yang berupa output pesan gagal. Namun, apabila data ditemukan sistem akan meng-update data pada tabel login di kolom date_login yaitu date_login+1. Selanjutnya sistem akan mendefinisikan expired_date dengan melakukan operasi penjumlahan yaitu date_login+1. Setelah expired_date terbentuk, sistem akan membandingkan tanggal pada kolom expired_date dengan tanggal pada saat prosedur dijalankan. Apabila tanggal prosedur dijalankan lebih besar dari tanggal pada expired_date maka sistem akan menampilkan output stat_login = N, data_logout = SYSDATE, dan token = null. Kemudian sistem akan meng-update data login sehingga kolom token akan bernilai sama dengan inputan p_token. Kemudian sistem akan menampilkan pesan gagal. Namun, apabila tanggal prosedur dijalankan lebih kecil dari tanggal pada expired_date maka sistem akan menampilkan output berupa pesan berhasil.

Prosedur *Logout* dimulai dengan sistem menerima input berupa username, kemudian sistem akan mencari data pengguna sesuai dengan username tersebut. Apabila data pengguna sesuai username tidak dapat ditemukan, maka sistem akan mengembalikan pesan gagal. Jika data sesuai username dapat ditemukan, maka sistem akan update data login yaitu pada stat_login menjadi N, date_logout menjadi tanggal saat ini, dan token menjadi null sesuai username yang telah di-input.

Design API yang dibangun pada iterasi-01 dapat dilihat pada Tabel 6.

Table 6 Design API pada Iterasi-01

Task Code	***Task Description***	***Endpoint***	***Method***
Task-01	API *Login*	/rrr2/user/login	POST
Task-02	API *Logout*	/rrr2/user/logout	PUT

- Iterasi-02

Untuk memenuhi *task* dalam iterasi-02 diperlukan *design database*, prosedur, dan API. *Database* yang diperlukan dalam iterasi-02 adalah tabel transaksi dan tabel tipe transaksi. Prosedur yang diperlukan dalam iterasi-02 adalah prosedur *create* transaksi, prosedur *update* cabang transaksi, dan prosedur *delete* transaksi. API yang diperlukan dalam iterasi-02 adalah API *create transaction*, API *read transaksi all*, API *update transaction*, dan API *delete transaction*. *Design* pada iterasi-01 dapat dlihat pada Tabel 7.

Table 7 Design iterasi-02

Task Code	***Database***	***Procedur***	***Web API***
Task-03	Tabel Transaksi Tabel Tipe Transaksi Tabel Area	Prosedur *Create Transaction*	API *Create Transaction*
Task-04	Tabel Transaksi Tabel Tipe Transaksi Tabel Area	-	API *Get All Transaction*
Task-05	Tabel Area Tabel Transaksi	Prosedur *Update Transaction Branch*	API *Update Transaction*

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 6 No. 1 Januari 2024 Hal. 623-632
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

Task-06	Tabel Transaksi	Prosedur *Delete Transaction*	API *Delete Transaction*

Tabel Transaksi merupakan tabel yang paling berperan penting dalam sistem ini. Sistem yang dibangun sebagian besar berfokus pada data transaksi, mulai dari pemindahan status transaksi hingga perhitungan total transaksi yang telah masuk pada tahap terakhir. Atribut-atribut yang ada pada tabel transaksi seluruhnya adalah format yang dimiliki oleh perusahaan.

Tabel Tipe Transaksi diperlukan untuk memisahkan transaksi sesuai dengan kode akun yang tertera pada setiap transaksi. Tujuanya agar mempermudah Departemen Remidial dalam melakukan seleksi data transaksi untuk dilakukan proses authorize atau reanalyze.

Prosedur Tambah data Transaksi dimulai dengan sistem menerima input berupa data transaksi. Kemudian sistem akan memvalidasi data yang telah di-input. Apabila data yang telah di-input terdapat kesalahan maka sistem akan mengembalikan pesan gagal. Namun, jika data input yang diberikan telah sesuai, maka sistem akan create data transaksi sesuai dengan input yang telah diberikan. Kemudian sistem akan mengembalikan pesan berhasil.

Prosedur Ubah Cabang dimulai dengan sistem menerima input berupa no_aggr, dan cabang. Kemudian sistem akan memvalidasi data tersebut. Apabila data yang di-input tidak sesuai, maka sistem akan mengembalikan pesan gagal. Tetapi apabila sistem berhasil menemukan data sesuai dengan input, maka sistem akan mencari data sesuai dengan no_aggr. Jika data tidak ditemukkan, maka sistem akan mengembalikan pesan gagal. Namun, jika data sesuai no_aggr berhasil ditemukan, maka sistem akan melakukan update data cabang pada tabel transaksi sesuai dengan no_aggr.

Prosedur Hapus data Transaksi dimulai dengan Sistem menerima input berupa ROW ID atau NO AGGR. Kemudian sistem akan mencari data transaksi sesuai dengan ROW ID atau NO AGGR. Apablila data tidak berhasil ditemukan maka sistem akan mengirimkan pesan gagal. Namun jika sistem berhasil menemukan data transaksi sesuai dengan ROW ID atau NO AGGR, maka sistem akan delete data transaksi dan mengembalikkan pesan sukses.

Design API yang dibangun pada iterasi-01 dapat dilihat pada Tabel 8.

Table 8 Design API pada Iterasi-02

Task Code	***Task Description***	***Endpoint***	***Method***
Task-03	API Tambah data Transaksi	/rrr2/trans/create	POST
Task-04	API Ambil semua Data Transaksi	/rrr2/trans/all	GET
Task-05	API Ubah Cabang	/rrr2/trans/update	PUT
Task-06	API Hapus data Transaksi	/rrr2/trans/delete	DELETE

- Iterasi-03

Untuk memenuhi *task* dalam iterasi-03 diperlukan *design* prosedur, dan API. Prosedur yang diperlukan dalam iterasi-03 adalah prosedur *authorize* transaksi, dan prosedur *reanalyze* transaksi. API yang diperlukan dalam iterasi-03 adalah API *Authorize Transaction* dan API *Reanalyze Transaction. Design* pada iterasi-01 dapat dlihat pada Tabel 9.

Table 9 Design Iterasi-03

Task Code	***Database***	***Procedur***	***Web API***
Task-07	Tabel Transaksi Tabel Pengguna Tabel Login Tabel Level Tabel Area Tabel Tipe Transaksi	-	API Ambil data transaksi berdasarkan status
Task-08	Tabel Transaksi Tabel Pengguna Tabel Login Tabel Level Tabel Area Tabel Tipe Transaksi	Prosedur Proses *Authorize*	API Proses *Authorize*
Task-09	Tabel Transaksi Tabel Pengguna Tabel Login Tabel Level Tabel Area Tabel Tipe Transaksi	Prosedur Proses *Reanalyze*	API Proses *Reanalyze*

Prosedur authorize transaksi dimulai dengan Sistem menerima input berupa NO_AGGR dan token. Kemudian sistem akan mencari data transaksi yang sesuai dengan NO AGGR. Apabila data tidak ditemukan, sistem akan mengembalikan pesan gagal. Namun apabila sistem dapat menemukan data transaksi sesuai dengan input NO AGGR, maka sistem akan mengambil semua data yang memiliki NO AGGR sama dengan input. Kemudian sistem akan melakukan pengecekan pada nilai

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 6 No. 1 Januari 2024 Hal. 623-632
http://jurnal.unidha.ac.id/index.php/jteksis
E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

status transaksi sesuai dengan NO AGGR yang telah di-input. Apabila status transaksi bernilai R3 maka sistem akan mengirimkan pesan gagal.
Jika status transaksi tidak bernilai R3, maka sistem akan melakukan pengecekan apakah status transaksi bernilai R2. Jika status transaksi bernilai R2 maka sistem akan mencari data login sesuai dengan token. Jika data tidak ditemukan, maka sistem akan mengirimkan pesan gagal. Jika data login sesuai token ditemukan, maka data akan mengambil data USERNAME dari tabel login yang memiliki token sesuai input. Sistem kemudian akan mencari data pengguna sesuai dengan USERNAME, apabila data tidak ditemukan, sistem akan mengembalikan pesan gagal. Jika data ditemukan, maka sistem akan mengambil data level dari tabel pengguna yang memiliki USERNAME yang sesuai. Kemudian akan dilakukan pengecekkan, apakah pada data tersebut memiliki LEVEL 1 atau LEVEL 2. Jika tidak, sistem akan mengirimkan pesan gagal. Jika data memiliki LEVEL 1 atau LEVEL 2 maka sistem akan mengubah status data transaksi menjadi R3 dan sistem akan mengembalikan pesan berhasil.
Apabila status data transaksi bukan R2, maka sistem akan melakukan pengecekkan apakah status data transaksi adalah R1. Jika betul, maka sistem akan mencari data login sesuai dengan token. Jika data login sesuai token ditemukan, maka data akan mengambil data USERNAME dari tabel login yang memiliki tokensesuai input. Sistem kemudian akan mencari data pengguna sesuai dengan USERNAME, apabila data tidak ditemukan, sistem akan mengembalikan pesan gagal. Jika data ditemukan, maka sistem akan mengambil data level dari tabel pengguna yang memiliki USERNAME yang sesuai. Kemudian akan dilakukan pengecekkan, apakah pada data tersebut memiliki LEVEL 1. Jika data memiliki LEVEL = 1, maka sistem akan mengubah status transaksi menjadi R2 dan sistem akan mengembalikan pesan berhasil.
Flowchart alur prosedur reanalyze transaksi dapat dilihat pada Gambar 3.13. Sistem akan menerima input berupa NO AGGR dan token. Kemudian sistem akan mencari data transaksi yang sesuai dengan NO AGGR. Apabila data tidak ditemukan, sistem akan mengembalikan pesan gagal. Namun apabila sistem dapat menemukan data transaksi sesuai dengan input NO AGGR, maka sistem akan mengambil semua data yang memiliki NO AGGR sama dengan input. Kemudian sistem akan melakukan pengecekan pada nilai status transaksi sesuai dengan NO AGGR yang telah di-input. Apabila status transaksi bernilai R1 maka sistem akan mengirimkan pesan gagal.
Jika status transaksi tidak bernilai R1, maka sistem akan melakukan pengecekan apakah status transaksi bernilai R2. Jika status transaksi bernilai R2 maka sistem akan mencari data login sesuai dengan token. Jika data tidak ditemukan, maka sistem akan mengirimkan pesan gagal. Jika data login sesuai token ditemukan, maka sistem akan mengambil data USERNAME dari tabel login yang memiliki token sesuai input. Sistem kemudian akan mencari data pengguna sesuai dengan USERNAME, apabila data tidak ditemukan, sistem akan mengembalikan pesan gagal. Jika data ditemukan, maka sistem akan mengambil data level dari tabel pengguna yang memiliki USERNAME yang sesuai. Kemudian akan dilakukan pengecekkan, apakah pada data tersebut memiliki LEVEL 1 atau LEVEL 2. Jika tidak, sistem akan mengirimkan pesan gagal. Jika data memiliki LEVEL 1 atau LEVEL 2 maka sistem akan mengubah status data transaksi menjadi R1 dan sistem akan mengembalikan pesan berhasil.
Apabila status data transaksi bukan R2, maka sistem akan melakukan pengecekkan apakah status data transaksi adalah R3. Jika betul, maka sistem akan mencari data login sesuai dengan token. Jika data login sesuai token ditemukan, maka sistem akan mengambil data USERNAME dari tabel login yang memiliki token sesuai input. Sistem kemudian akan mencari data pengguna sesuai dengan USERNAME, apabila data tidak ditemukan, sistem akan mengembalikan pesan gagal. Jika data ditemukan, maka sistem akan mengambil data level dari tabel pengguna yang memiliki USERNAME yang sesuai. Kemudian akan dilakukan pengecekkan, apakah pada data tersebut memiliki LEVEL 1 atau LEVEL 3. Jika data memiliki LEVEL 1 atau LEVEL 3, maka sistem akan mengubah status transaksi menjadi R2 dan sistem akan mengembalikan pesan berhasil.

Design API yang dibangun pada iterasi-01 dapat dilihat pada Tabel 10.

Table 10 Design API pada Iterasi-03

Task Code	***Task Description***	***Endpoint***	***Method***
Task-07	API Ambil data transaksi berdasarkan status	/restv2/rrr2/trans/status	GET
Task-08	API Proses *Authorize*	/rrr2/trans/auth	PUT
Task-09	API Proses *Reanalyze*	/rrr2/trans/rean	PUT

- Iterasi-04
Untuk memenuhi *task* dalam iterasi-04 diperlukan *design database* dan API. *Database* yang diperlukan dalam iterasi-04 adalah tabel *overview* cabang, dan tabel *overview area.* API yang diperlukan dalam iterasi-01 adalah API *Read All Overview 1*, API *Read All Overview 2*, dan API

Jurnal Teknologi Dan Sistem Informasi Bisnis
Vol. 6 No. 1 Januari 2024 Hal. 623-632
http://jurnal.unidha.ac.id/index.php/jteksis

E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

Read All Overview 3. Design pada iterasi-01 dapat dlihat pada Tabel 11.

Table 11 Design Iterasi-04

Task Code	***Database***	***Procedur***	***Web API***
Task-10	Tabel Overview Cabang	-	API Ambil data Overview Cabang
Task-11	Tabel Overview Area	-	API Ambil data Overview Area
Task12	-	-	API Ambil data Overview Nasional

Tabel *Overview* Cabang merupakan tabel yang mencatat jumlah transaksi yang telah masuk ke status registration pada setiap cabang yang ada. Pada tabel ini terdapat atribut ID_OVV1, CABANG, dan AMT_TRAN dengan ID_OVV1 sebagai kunci utama tabel ini. Setiap transaksi yang telah di registrasi maka dijumlahkan ke dalam tabel overview sesuai dengan cabang pada transaksi.

Tabel *Overview* Area merupakan tabel yang mencatat jumlah transaksi yang telah masuk ke status registration pada setiap cabang yang ada. Pada tabel ini terdapat atribut ID_OVV2, AREA, dan AMT_TRAN dengan ID_OVV2 sebagai kunci utama tabel ini. Setiap transaksi yang telah di registrasi maka dijumlahkan ke dalam tabel overview sesuai dengan area pada transaksi.

Design API yang dibangun pada iterasi-01 dapat dilihat pada Tabel 12.

Table 12 Design API pada Iterasi-04

Task Code	***Task Description***	***Endpoint***	***Method***
Task-10	API Ambil data Overview Cabang	/restv2/rrr2/trans/status	GET
Task-11	API Ambil data Overview Area	/rrr2/trans/auth	GET
Task-12	API Ambil data Overview Nasional	/rrr2/trans/rean	GET

e. *Implementation*

Pada tahap ini implementasi sistem dilakukan dengan 4 iterasi. Masing-masing iterasi dibagi menjadi 3 tahapan, yaitu *Unit Test*, *Code*, dan *Refactoring*. *Unit Test* dilakukan dengan cara menjalankan 1 fitur pada setiap task untuk memastikan bahwa task tersebut sudah berfungsi atau tidak. *Code* dilakukan dengan cara mencatat *source code* dalam setiap task. *Refactoring* adalah upaya untuk mengoptimalisasi kinerja *source code* dengan melakukan penyempurnaan pada setiap *source code*.

- Unit Test

Hasil pengujian dapat dilihat pada Tabel 5.

Table 13 Hasil *unit test* pada setiap task

Iterasi	***Task Code***	***Method***	***Hasil***
Iterasi-01	Task-01	POST	Pass
	Task-02	PUT	Pass
Iterasi-02	Task-03	POST	Pass
	Task-04	GET	Pass
	Task-05	PUT	Pass
	Task-06	DELETE	Pass
Iterasi-03	Task-07	GET	Pass
	Task-08	PUT	Pass
	Task-09	PUT	Pass
Iterasi-04	Task-10	GET	Pass
	Task-11	GET	Pass
	Task-12	GET	Pass

f. Software Testing

Pada tahap *software testing* dilakukan dengan menggunakan Postman. Setiap API yang telah terbentuk akan dibuatkan masing-masing *test case* kemudian di-*hit* dengan Postman untuk mengecek apakah API tersebut dapat melewati *test case* yang telah dibuat. Berdasarkan hasil *software testing* terdapat 6 buah *test case* yang tidak berhasil dilalui oleh API. Simpulan dari hasil *software testing* dapat dilihat pada Table 6.

Table 14 Hasil Software Testing

Iterasi	**Test Skenar**	**Jumlah test**	**Jumlah Pass**	**Jumlah Gagal**
Iterasi-01	User Login	5	5	0
Iterasi-02	Task-03	4	4	0
	Task-04	3	3	0
	Task-05	5	5	0
	Task-06	5	5	0
Iterasi-03	Task-07	15	9	6
	Task-08	3	3	0
	Task-09	3	3	0
Iterasi-04	Task-10	4	4	0
	Task-11	8	8	0
	Task-12	4	4	0

SIMPULAN

Hasil dari penelitian ini memiliki beberapa kesimpulan sebagai berikut:

1. Web API dan basis data dapat mendukung kinerja Departemen Remidial untuk menandai, mengevaluasi, dan melakukan pemulihan terhadap pinjaman-pinjaman yang terhambat atau mengalami kerugian. Basis data yang dibuat dapat menyelesaikan permasalahan

E-ISSN : **2655-8238**
P-ISSN : **2964-2132**

terkait perulangan input data pada saat terjadi kesalahan.

2. Didapatkan API dengan kualitas cukup bagus yakni mengalami 89,8% keberhasilan dalam pengujian yang telah dilakukan dan hanya 10,2% kegagalan berdasarkan data pengujian tabel 14.

UCAPAN TERIMAKASIH

Terimakasih kepada mitra yaitu PT Cipta Sedaya Digital Indonesia yang telah membantu dari segi pengadaan data serta *environment* yang digunakan seperti server, koneksi VPN, dan *database*.

DAFTAR PUSTAKA

[1] Abbas, C. J. (2016). Rekayasa Arsitektur Sistem Informasi Pelayanan Perijinan Satu Atap Di Kabupaten Kuningan Dengan Menggunakan Metoda Soa (Service Oriented Architecture). Jejaring: Jurnal Teknologi dan Manajemen Informatika, 15-21.

[2] Arianto, M. A., Munir, S., & Khotimah, K. (2016). Analisis Dan Perancangan Representational State Transfer (Rest) Web Service Sistem Informasi Akademik Stt Terpadu Nurul Fikri Menggunakan Yii Framework. Jurnal Teknologi Terpadu.

[3] Asiz, N. (2022). Basis Data Lanjutan dengan PLSQL. Bandung: WIDINA BHAKTI PERSADA BANDUNG.

[4] Baharuddin, Wakkang, H., & Irianto, B. (2022). Implementasi Web Service Dengan Metode Rest Api Untuk Integrasi Data Covid 19 Di Sulawesi Selatan. Jurnal Sintaks Logika, 236-241.

[5] Carolina, I. &. (2019). Penerapan Metode Extreme Programming Dalam Perancangan Aplikasi Perhitungan Kuota Sks Mengajar Dosen. *Jurnal IKRA-ITH Informatika*.

[6] Choirudin, R., & Adil, A. (2019). Implementasi Rest Api Web Service Dalam Membangun Aplikasi Multi Platform Untuk Usaha Jasa. Jurnal Matrik, 284-293.

[7] Eriana, E. S. (2021). Penerapan Metode Personal Extreme Programming Dalam Perancangan Aplikasi Pemilihan Ketua Hmsi Dengan Weighted Product. *Jurnal Ilmu Komputer JIK*.

[8] Fenardi, O. &. (2023). Aplikasi Akademik Berbasis WebsiteMenggunakan Metode Extreme Programming Pada SMAN1 Belinyu. *Jurnal Teknologi Dan Sistem Informasi Bisnis (JTeksis)*, 440-447.

[9] Filiana, A., Rini, M. N., Prabawati, A. G., & Samat, R. A. (2022). Pengembangan Rest Api Untuk Informasi Pasar Tradisional Di Kota Yogyakarta Dengan Metode Incremental. SINTECH JOURNAL, 10-23.

[10] Hantana, J. S. (2013). Pendekatan Service Oriented Architecture (SOA) Pada Pelaksanaan E-Government di Kementerian Hukum dan HAM RI. Jurnal Nasional Pendidikan Teknik Informatika (JANAPATI), 254-260.

[11] Hernando, L. (2020). Perancangan Sistem Informasi Keuangan Pada Unit Koperasi Simpan Pinjam. *Jurnal Teknologi Dan Sistem Informasi Bisnis (Jteksis)*.

[12] Junaedi, I. (2020). Pengembangan Aplikasi Analisis Vendor Performance Di Pt. Suzuki Indomobil. JISAMAR (Journal of Information System, Applied, Management, Accounting and Research), 37-55.

[13] Kurniawan, Y. K., Oslan, Y., & Kristanto, H. (2013). Implementasi Rest - Api Untuk Portal Akademik Ukdw berbasis Android. Jurnal EKSIS, 29-40.

[14] Mintarsih. (2023). Pengujian Black Box Dengan Teknik Transition Pada Sistem Informasi Perpustakaan Berbasis Web Dengan Metode WaterfallPada SMC Foundation. *Jurnal Teknologi Dan Sistem Informasi Bisnis (Jteksis)*, 33-35.

[15] Muri, M. F., Utomo, H. S., & Sayyidati, R. (2019). Search Engine Get Application Programming Interface. Jurnal Sains dan Informatika, 87-97.

[16] Rizal, R., & Rahmatulloh, A. (2019). RESTful Web Service untuk Integrasi Sistem Akademik dan Perpustakaan Universitas Perjuangan. Jurnal Ilmiah Informatika, 54-59.

[17] Suprapto, F. R., Marthasari, G. I., & Nuryasin, I. (2020). Sistem Informasi Penjualan dan Pelelangan Berbasis Web pada Ricardo Corner MLG Menggunakan Metode Personal eXtreme Programming (PXP). *Jurnal Rpositor*.

[18] Widya, S. A. (2023). enerapan Metode Scrum Pada Perancangan Sistem Informasi Manajamen Arsip Surat Berbasis Web. *Jurnal Teknologi Dan Sistem Informasi Bisnis (JTeksis)*, 484-491.

[19] Yanti, S. N., & Rihyanti, E. (2021). Penerapan Rest API untuk Sistem Informasi Film Secara Daring. Jurnal Informatika Universitas Pamulang, 195-201.

[20] Yutia, S. N. (2021). Automated Functional Testing pada API menggunakan Keyword Driven Framework. *Journal of Informatics and Communications Technology (JICT)*, 065-078.