Optimasi *Web Service REST* Pada *Backend* Aplikasi *Prospect* Menggunakan Metode *Extreme Programming*

[1]Rahmad Adhi Sasono, [2*]Sepyan Purnama Kristanto, [3]Lutfi Hakim, [4]Dianni Yusuf
[1,2,3,4] Teknologi Rekayasa Perangkat Lunak, Politeknik Negeri Banyuwangi, Banyuwangi
[1]rahmad365@gmail.com@gmail.com, [2]sepyan@poliwangi.ac.id,[3]lutfi@poliwangi.ac.id, [4]dianniyusuf@poliwangi.ac.id

Abstracts *The Prospect application initially used RPC-API to manage task scheduling, team notes, and client meetings. However, issues arose concerning the quality of data integration within the Web Service system. To address this, the research employed the Extreme Programming (XP) methodology. The aim was to achieve superior data integration compared to the previous RPC-API implementation. Testing was conducted using JSON output format and REST architecture for data integration. The study proposes adopting Representational State Transfer (REST) architecture as an alternative to RPC-API. Development followed the Extreme Programming (XP) methodology, with REST architecture utilizing HTTP protocols and JSON format. This approach aims to provide more efficient and flexible data integration across various software systems and operating environments. The results indicate that the use of Extreme Programming (XP) enhances performance and development speed through its iterative approach, including planning, design, coding, and testing. The resulting API endpoints meet user requirements, with appropriate data input, output, and status codes for handling task scheduling and client meetings. REST API, using HTTP protocols, delivers good response speed. The study demonstrates that the implementation of REST and Extreme Programming (XP) provides a strong foundation for the development and maintenance of the Prospect application.*

Abstract Dalam era digital yang terus berkembang, web service menjadi elemen penting dalam pengembangan aplikasi web. PT. Galang Kangin *Software* menghadapi tantangan dalam manajemen penjadwalan tugas, pencatatan dalam tim, dan pengaturan pertemuan dengan klien. Aplikasi *Prospect* sebelumnya menggunakan *RPC-API,* yang meskipun sudah memenuhi kebutuhan manajemen tersebut, menghadapi masalah dalam kualitas layanan integrasi data. Dengan sistem aplikasi sebelumnya sudah menghasilkan solusi dengan aplikasi *Prospect* dengan *RPC-API.* Dimana sudah menghasilkan kebutuhan untuk mengelolah manajemen penjadwalan tugas, catatan dalam tim dan pada bagian tugas tertentu, mengatur pertemuan dengan *client*, namun persoalan dalam kualitas layanan integrasi data pada sistem *Web Service.* Solusi penyelesaian penelitian ini penulis memanfaatkan metode *Extreme Programming (XP)* untuk optimasi pada pengembang pada aplikasi *prospect* dengan menggabungkan arsitektur *representational state tramsfer (Rest)* sebagai alternative untuk *RPC-API.* Arsitektur *Rest* menggunakan *protocol HTTP* dan format json, bertujuan untuk memberikan integerasi data yang lebih efesien dan fleksibel di berbagai sistem perangkat lunak dan lingkungan operasi. Hasil pengujian yang dilakukan menunjukan bahwa metode yang digunakan yaitu *extreme programming (XP),* menunjukan kinerja dan kecepatan dalam pengembangan, berkat pendekatan yang mencangkup perencanaan, perancangan, pengkodean, pengujian. *Endpoint API* yang dihasilkan memenuhi kebutuhan pengguna dengan data *input* dan *output* serta status kode yang sesuai dalam menangani penjadwalan tugas dan pertemuan *client. Penggunaan Rest API* dengan *protocol HTTP* menghasilkan kecepatan *respon* yang lebih baik dan optimasi pada pengembangan pada masa depan menjadi lebih testruktur pada sistem, penelitian ini menunjukan bahwa penerapan Rest dan extreme programming (XP) sebagai fondasi yang kuat untuk bagian pengembang dan pemeliharaan aplikasi *prospect*

I. Pendahuluan

Dalam perkembangan pesat era digital, aplikasi web telah menjadi komponen penting yang tak terpisahkan dari keseharian. Kehadiran layanan web memberikan pengaruh signifikan terhadap proses perancangan dan pengembangan sistem di lingkungan organisasi, di mana layanan ini berperan sebagai perantara yang memfasilitasi interaksi antara basis data dan aplikasi klien[1]. Selain itu, persaingan pasar memicu pembuatan dan pengembangan *web service* semakin inovatif dan terus menurus mengalami perubahan, dari tahun ke tahun, bagaimana semakin efesienya suatau aplikasi yang memiliki *database*, semakin cepat respon yang diberikan maka akan memberikan landasan yan lebih kokoh.

PT. Galang Kangin *Software* merupakan perusahan teknologi informasi yang bergerak di bidang pengembangan teknologi, teknologi yang di kembangkan termasuk *website* dan Aplikasi. Dalam pembuatan teknologi, perusahaan memiliki tim profesional pada masing-masing bidang sesuai dengan pekerjaan, seringkali dalam proses pekerjaan dalam tim terjadi kesalah pahaman antara tim pada bagian bidang tertentu atau pada client. Pembagian penjadwalan tugas, catatan dalam tim pada bagian tugas tertentu, mengatur pertemuan dengan client, merupakan faktor-faktor kendala dalam perusahan, perlu adanya sistem untuk mengatasi kendala tersebut [2].

Pada sistem aplikasi sebelumnya sudah menghasilkan sebuah solusi dengan menggunakan aplikasi *Prospect* dengan sistem *XML-RPC API.* Pada aplikasi yang telah buat sudah menghasilkan kebutuhan untuk memanajemen penjadwalan tugas, catatan dalam tim pada bagian tugas tertentu, mengatur pertemuan dengan *client*. Namun adanya persoalan dalam segi kinerja layanan pada saat integrasi data pada sistem *web service*, proses yang lama karena *overhead* yang disebabkan tidak mengikuti *standart HTTP* sehingga mengurangi fleksibilitas kemampuan untuk mengoptimalkan kinerja saat integrasi data. Menurut penelitian oleh Salih Alas Majapahit, menurut pengujian yang dilakukan, pada *RPC-API* dan Rest dimana hasil menunjukan bahwa dengan arsitektur *REST-API* mengungguli dari hal kinerja dan efektif integrasi data [3].

Solusi yang diusulkan Pada sistem yang baru ini yaitu *Rest*, teknologi ini sangat berguna bagi perusahaan untuk penjadwalan tugas dan juga pada sisi konsumen, untuk

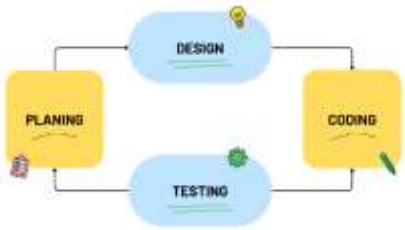
mempermudah operasional perusahaan dan bisnis [4]. Dalam konteks ini, penerapan penggunaan arsitektur *Representational State Transfer (Rest)* untuk merancang layanan *website (web service)* memungkin pengguna untuk dapat menghubungkan berbagai jenis s*oftware* dan sistem operasi yang sama atau berbeda dalam Perusahaan [5]. Dengan *Rest*, dalam hal kecepatan lebih baik dalam bagian kecepatan *response* dan optimasi pada pengembangan pada masa depan menjadi lebih testruktur pada sistem, dengan memanfaatkan protocol *HTTP*, sehingga kinerja penjadwalan perusahaan menjadi mudah, cepat karena efesien waktu dan mampu meningkatkan produktifitas perusahaan [6]. *Rest* memungkinkan *client* pada *server* dimana *client* dapat mengirim r*equest* pada *server* dan *server* memproses *request* dan mengembalikan *response*, dimana menggunakan *protocol HTTP GET, POST, UPDATE, DELETE* [7].

Untuk menghasilkan hasil yang terbaik dari integrasi data dalam hal pertukaran data pada sistem Optimasi *Web Service* Pada *Backend* Aplikasi *Prospect* Menggunakan Metode *Extreme Programming*, oleh karena itu perlunya optimasi arsitektur *Rest* yang dimana pada sistem sebelumnya masih menggunakan *Rpc-Api.*

II. Metode Penelitian

A. Extream Programming (XP)

Metode Pendekatan penelitian yang dapat diterapkan yaitu mengenai pendekatan menggunakan metode Extreme Programming (XP) pada Optimasi *Web Service* Pada *Backend* Aplikasi *Prospect* Menggunakan Metode *Extreme Programming* yang dimana pendekatan ini melibatkan empat tahap yaitu, tahap perencanaan (planning), perancangan (design), pengkodean (coding), pengujian (testing), metode ini menggunakan pendekatan berorientasi objek dan bertujuan untuk tim yang terdiri dari skala kecil sampai menengah [8]. Salah satu alasan *Extreme Programming (XP)* tidak perlu menggunakan tim besar, dimaksudkan untuk mengatasi ketentuan fungsional dan perbahan persyaratan yan cepat [9]. Gambaran proses tahapan metode Extreme Programing (XP).



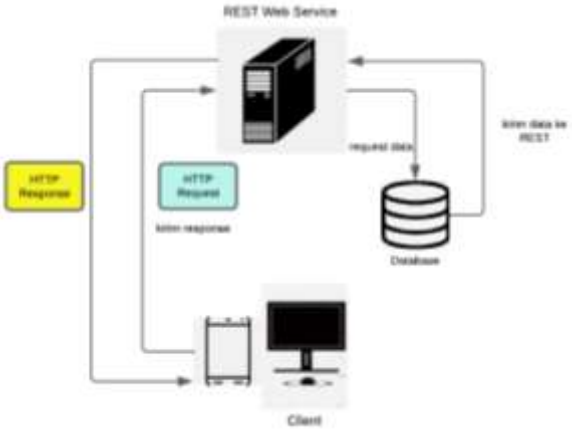
Gambar 1 *Extreme Programming (XP)*

2.1 Perencanaan (*Planning*)

Tahapan pertama dengan merencanakan analisis kebutuhan dan tujuan dari optimasi *web service* dengan menggunakan arsitektur *Rest.* Perbaikan efesiensi untuk mendorong kinerja manajemen tim dan meningkatkan layanan integrasi data. Pelaksaan proses dan pemantauan hasil evaluasi yang di inginkan mamastikan bahwa *web service* dengan menerapkan arsitektur *Rest* berfungsi dengan baik, agar pengguna mendapatkan pengalaman yang terbaik [10].

2.2 Perancangan *(Design)*

Pada tahap berikutnya perancangan dilakukan pembuatan permodelan (design) berdasarkan analisa kebutuhan aktifitas mencangkup, design arsitektur REST, database Entity Relationship Diagram (ERD), Use-Case Diagram. *Card CRC* (*Class Responsibilities Collaborators*) sangat diperlukan dalam penggunaan metode *Extreme Programming (XP)* untuk mengoptimalkan *web service REST* pada *backend* aplikasi *prospect* dalam peningkatan peforma dalam integrasi data, kecepatan pengembangan, peningkatan kualitas *code* [11].

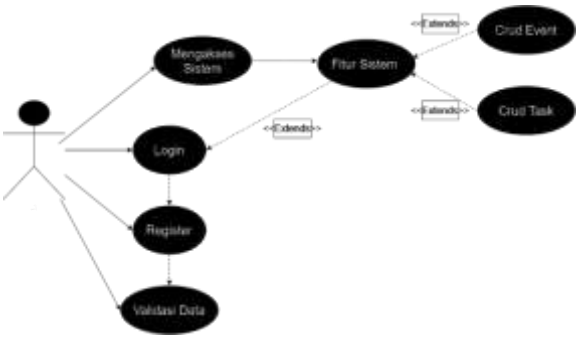


Gambar 2 *Design* Arsitektur *REST*

Pada Gambar 2.2 Diatas menjelaskan cara arsitektur *Rest Web Service* melakukan *request* kepada *server* kemudian *server* membalas *responses* kepada *client* dengan format berbentuk json [12]. Dimana dengan ini dapat membantu perusahaan dalam meningkatkan kinerja, sehingga berbagai penjadwalan dapat disesuaikan dengan mudah.

Gambar 3 *Design ERD (Entity Relationship Diagram)*

Berdasarkan gambar 3 *Entitiy Relationship* Diagram menggambarkan relasi antar entitas atau objek tabel yang di dalamnya terdapat data dan atribut dalam membangun sebuah sistem [13].



Gambar 4 *Design Use Case*

Berdasarkan Gambar 2.4 *use case* menggambarkan proses alur dalam sistem yang memberikan pemahaman aktifitas pengguna sistem dan sistemnya. Berikut penjelasan alur sistemnya.

1. *User* atau admin adalah pengguna yang dapat mengakses sistem, mulai dari *login*, *register*, fitur sistem.
2. *Register* merupakan langkah pertama ketika pengguna belum membuat akun di sistem sebelumnya.
3. *Login* merupakan langkah berikutnya ketika pengguna sudah pernah membuat akun di sistem sebelumnya.
4. Mengaskses sistem merupakan *use case* umum dari berbagai fitur sistem yanag ada di dalamnya.
5. Fitur Sistem merupakan kumpulan fitur atau konten yang akan di akses oleh pengguna dan dapat dirasakan nantinya.
6. *Crud Event* memiliki fungsi menambah, memperbaharui, menghapus dan melihat pada fitur *Event.*
7. *Crud Task* memiliki fungsi menambah, memperbaharui, menghapus dan melihat pada fitur *Task.*

2.3 Pengkodean *(coding)*

Tahap pengkodean ini dilakukan untuk bisa menjalankan perencaan dan perancangan yang sudah di buat untuk pengguna bisa menjalankan Aplikasi *Prospect*. Dalam proses Optimasi *Web Service* Pada *Backend* Aplikasi *Prospect* Menggunakan Metode *Extreme Programming*, pada *code* berfokus pada bagian *backend*, bahasa pemrograman PHP, *Laravel (Frame Work)*, *MySQL* untuk manajemen *database* pada proses pembuatan *code* menggunakan *Model and Controller* [14].

2.4 Pengujian *(testing)*

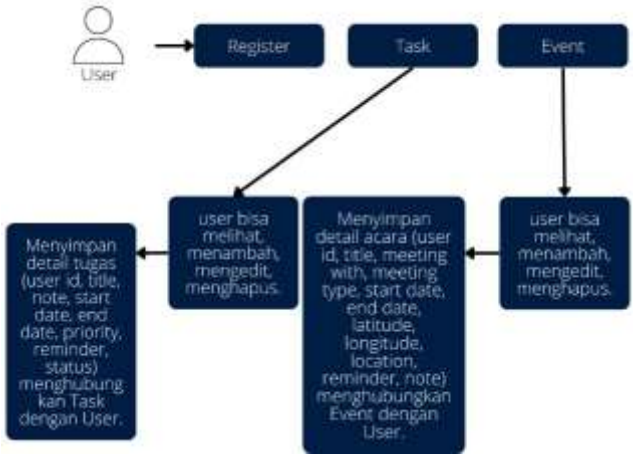
Tahap terakhir ini pengujian untuk memastikan bahwa fungsi berjalan dengan normal, *output* yang di hasilkan yaitu data dalam format json. Pengujian dilakukan dengan menggukan *postman*, dimana *Postman* ini adalah alat untuk menguji *API* dengan *standart HTTP Protokol*, method *POST, PUT, DELETE*, method tersebut dikelompok kedalam Collection yang kemudian nanti akan menjadi output dari penelitian ini. berikut hasil pengujian menggunakan *Postman* [15].

III. Hasil dan Pembahasan

Pada hasil penelitian dan pembahasan dalam penerapan metode *Extreme Programming (XP)* untuk mengoptimalkan *web service REST* pada *backend* aplikasi *prospect* berupa peningkatan peforma dalam integrasi data, kecepatan pengembangan, peningkatan kualitas *code*.

3.1 Perencanaan (*Planing*)

Dalam analisa yang di peroleh bahwa pengguna harus melakukan Registrasi secara otomatis masuk pada fitur *Task* pengguna membuat tugas pada satu karyawan atau lebih dan *Event* pengguna membuat jadwal pertemuan dengan *client* tanpa harus datang langsung ke lokasi.



Gambar 5 *Bussiness Flow*

3.2 Perancangan (*Design*)

Pada c*lass responsibilities collaborators* digunakan dalam memetakan tanggung jawab dan kolaborasi antar *class* dalam sebuah sistem. Bagaimana class tersebut berinteraksi dengan *class* lainya.

Tabel 1 Class Responsibilities Collaborators (*CRC)*

Class	**Responsibilities**	**Collaborators**
User	Menyimpan detail *User (first name, last name, email, password, foto profile).*	Task untuk mendapatkan tugas terkait dengan *User, Event* untuk mendapatkan acara yang terkait dengan *User.*
Task	Menyimpan detail tugas (*user id, title, note, start date, end date, priority, reminder, status*) menghubungkan	User mendapatkan informasi *User* yang membuat atau bertanggung jawab atas *Task.*

Class	Responsibilities	Collaborators
	Task dengan *User.*	
Event	Menyimpan detail acara (*user id, title, meeting with, meeting type, start date, end date, latitude, longitude, location, reminder, note*) menghubungkan *Event* dengan *User.*	User, untuk mendapatkan informasi *User* yang menyelenggarakan atau menghadiri *Event.*

Deskripsi untuk bagian dari *card CRC* (*Class Responsibilities Collaborators*) sebagai berikut :

1. *Class*, menyebutkan nama kelas atau entitas dalam sistem, ini adalah representasi dari sebuah konsep yang memiliki karakteristik dan perilaku tertentu. Contoh pada *class,User, Task, Event.*
2. *Responsibilities* menyediakan tanggung jawab, fungsionalitas utama dari *class* tersebut.
3. *Collaborators* membantu *class* berinteraksi dengan bagian lain dari sistem dengan alasan tertentu sesuai dengan kepentingan bisnis.

3.3 Pengkodeaan (*Coding*)

Tahap pengkodean ini dilakukan untuk bisa menjalankan perencaan dan perancangan yang sudah di buat untuk pengguna bisa menjalankan Aplikasi *Prospect.* Pada proses pembuatan *code* menggunakan *model* dan *controller* kemudian untuk menjalankan data *endpoint* sudah dapat berfungsi dengan normal menggunakan alat *postman*.

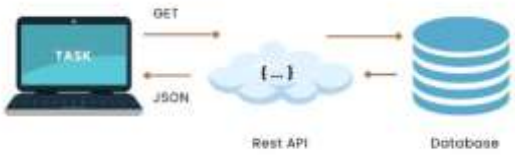
3.4 Pengujian (*Testing*)

Tahap terakhir memastikan bahwa fungsi berjalan dengan normal dan memberikan gambaran arsitektur *Rest* dalam bekerja, *output* yang di hasilkan yaitu data dalam format json. Pengujian dilakukan dengan menggukan postman, dimana *Postman* ini adalah alat untuk menguji API dengan *standart HTTP Protokol, method POST, PUT, DELETE, method* tersebut dikelompok kedalam *collection* yang kemudian nanti akan menjadi *output* dari penelitian ini. berikut hasil pengujian menggunakan *Postman*. Pada proses pengujian terdapat beberapa HTTP Status Code yang muncul setelah melakukan pengujian.

Tabel 2 HTTP Response Code

No	Status Code	Keterangan
1	200	Permintaan berhasil diproses
2	201	Permintaan berhasil dan sumber daya baru telah dibuat (biasanya setelah operasi POST)
3	202	Permintaan diterima untuk diproses, tetapi belum selesai.
4	204	Permintaan berhasil, tetapi tidak ada konten yang dikirim
5	206	Respons berisi sebagian data

A. Task



Gambar 6 Arsitektur *Rest Task*

1. *Get End Point*
   https://prospect.galkasoft.id/api/tasks


```
{
    "id": 3,
    "user_id": 1,
    "user_first_name": "L",
    "user_last_name": "Messi",
    "title": "create restapi untuk client B",
    "note": "dalam project ini client meminta untuk di berikan beberapa fitur ada 7 fitur",
    "start_date": "2023-05-29",
    "end_date": "2024-05-29",
    "priority": "hight",
    "reminder": "2024-05-29 09:30:00",
    "status": "completed",
    "contact_id": 1,
    "first_name": "tegar",
    "last_name": "maharani",
    "full_name": "tegar maharani",
    "phone_number": "082139978890",
    "home_number": "082139978890",
    "work_number": "082139978890",
    "email": "tegarmaharani@gmail.com"
}
```


Gambar 7 *Testing Postman*

Tabel 3 Pengujian *Get Task*

Get Task	
	Status Code Ekspetasi = 200 Hasil = 200

Pada Gambar 7 dan tabel 3 ujicoba pada *method GET* dimana, semua *collection* pada *protocol GET* akan menampilkan semua data yang sudah dibuat dan tampil, output yang di hasilkan method *GET* format json dimana pada uji coba ini mendapatkan keterangan *message "Successfullty Index Data", status "True"*, ekspetasi dan hasil kode sesuai dengan tabel diatas.

2. *Post End Point*
   https://prospect.galkasoft.id/api/tasks

Gambar 8 *Testing Postman*

Tabel 4 Pengujian *Post Task*

Post Task	
	Status Code Ekspetasi = 201 Hasil = 201

Pada Gambar 8 dan tabel 4 ujicoba pada *method POST* dimana, *output* yang dihasilkan pada method *POST* tampil data dengan format json dimana pada uji coba ini mendapatkan keterangan *message "Successfullty Create Data", status "True"* , ekspetasi dan hasil kode sesuai dengan tabel diatas.

3. *Update End Point*
   https://prospect.galkasoft.id/api/tasks/{id}

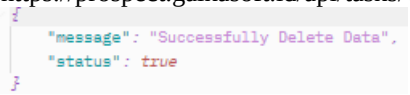
Gambar 9 *Testing Postman*

Tabel 5 Pengujian *Update Task*

Update Task	
	Status Code Ekspetasi = 200 Hasil = 200

Pada Gambar 8 dan tabel 9 ujicoba pada *method PUT* dimana, output yang dihasilkan *method PUT* tampil data dengan format json, dimana pada uji coba ini mendapatkan keterangan *message "Successfullty Update Data", status "True"*, ekspetasi dan hasil kode sesuai dengan tabel diatas.

4. *Delete End Point*
   https://prospect.galkasoft.id/api/tasks/{id}



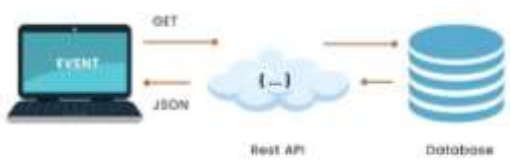
Gambar 10 *Testing Postman*

Tabel 6 Pengujian *Delete Task*

Delete Task	
	Status Code Ekspetasi = 200 Hasil = 200

Pada Gambar 10 dan tabel 6 ujicoba pada *method DELETE* output yang dihasilkan, data dengan format json dimana pada uji coba, mendapatkan keterangan *message "Successfullty Delete Data", status "True"*, ekspetasi dan hasil kode sesuai dengan tabel diatas.

B. Event



Gambar 11 Arsitektur *Rest Event*

1. *Get End Point*
   https://prospect.galkasoft.id/api/events


```
{
    "id": 2,
    "user_id": 1,
    "user_first_name": "L",
    "user_last_name": "Messi",
    "title": "membahas fitur B",
    "meeting_with": "client Z",
    "meeting_type": "create_call_event",
    "start_date": "2023-12-08 00:00:00",
    "end_date": "2024-07-08 00:00:00",
    "latitude": "jalan diponogoro",
    "longitude": "jalan diponogoro",
    "location": "Banyuwangi",
    "reminder": "00:00:15",
    "note": "jangan sampai terlambat bergabung ya"
}
```


Gambar 12 *Testing Postman*

Tabel 7 Pengujian *Get Event*

Get Event	
	Status Code Ekspetasi = 200 Hasil = 200

Pada Gambar 12 dan Tabel 7 Ujicoba pada *method GET* dimana, output yang di hasilkan *method GET* tampil data dengan format json dimana pada uji coba ini mendapatkan keterangan *status message "Successfullty Index Data", status "True"*, ekspetasi dan hasil kode sesuai dengan tabel diatas.

2. *Post End Point*
   https://prospect.galkasoft.id/api/events


```
{
    "data": {
        "id": 2,
        "user_id": 1,
        "user_first_name": "L",
        "user_last_name": "Messi",
        "title": "membahas fitur B",
        "meeting_with": "client Z",
        "meeting_type": 4,
        "start_date": "2023-12-08",
        "end_date": "2024-07-08",
        "latitude": "jalan diponogoro",
        "longitude": "jalan diponogoro",
        "location": "Banyuwangi",
        "reminder": "00:00:15",
        "note": "jangan sampai terlambat bergabung ya"
    },
    "message": "Successfully Create Data",
    "status": true
}
```


Gambar 13 *Testing Postman*

Tabel 8 Pengujian *Post Event*

Post Event	
	Status Code Ekspetasi = 201 Hasil = 201

Pada Gambar 13 dan Tabel 8 ujicoba pada *method POST* dimana, output yang dihasilkan *method POST* tampil data dengan format json dimana pada uji coba ini mendapatkan keterangan *status message "Successfullty Create Data", status "True"*, ekspetasi dan hasil kode sesuai dengan tabel diatas.

3. *Update End Point*
   https://prospect.galkasoft.id/api/events{id}


```
{
    "data": {
        "id": 2,
        "user_id": 1,
        "user_first_name": "L",
        "user_last_name": "Messi",
        "title": "membahas fitur event updated",
        "meeting_with": "client A",
        "meeting_type": "create_call_event",
        "start_date": "2023-11-28",
        "end_date": "2024-04-28",
        "latitude": "jl. diponegoro xx34 xxx534 342343",
        "longitude": "jl. diponegoro xx34 xxx534 342343",
        "location": "Denpasar Bali",
        "reminder": "00:00:00",
        "note": "jangan sampai terlambat bergabung ya"
    },
    "message": "Successfully Update Data",
    "status": true
}
```


Gambar 14 *Testing Postman*

Tabel 8 Pengujian *Update Event*

Update Event	
	Status Code Ekspetasi = 200 Hasil = 200

Pada Gambar 14 dan Tabel 8 ujicoba pada *method PUT* dimana, output yang dihasilkan *method PUT* tampil data dengan format json dimana pada uji coba ini mendapatkan keterangan *status message "Successfullty Update Data", status "True"*, ekpetasi dan hasil kode sesuai dengan tabel diatas.

4. *Delete Controller* dan *End Point*
   https://prospect.galkasoft.id/api/events{id}


```
{
    "message": "Successfully Delete Data",
    "status": true
}
```


Gambar 15 *Testing Postman*

Tabel 8 Pengujian *Delete Event*

Delete Event	
	Status Code Ekspetasi = 200 Hasil = 200

Pada Gambar 15 Dan Tabel 8 ujicoba pada *method DELETE* output yang dihasilkan data dengan format json dimana pada uji coba, mendapatkan keterangan *message "Successfullty Delete Data", status "True"* ekspetasi dan hasil kode sesuai dengan tabel diatas.

IV. Kesimpulan

Optimasi yang dilakuan pada penelitian ini menhasilkan beberapa kriteria utama yaitu

1. Peforma kecepatan integrasi data mengurangi beban pada bagian server *backend*, pengelolahan *log* memungkinkan tim pengembang dengan cepat mendeteksi *error* dan *bug*, kemudian optimasi dalam database dimana memilih tipe data yang tepat untuk mendukung optimasi yang diterapkan. Dalam metode *Extreme Programming (XP)* Optimasi Backend Aplikasi Prospect memalui Implemetasi *Web Service REST* yang dimana pendekatan ini melibatkan empat tahap yaitu, tahap perencanaan *(planning)*, perancangan *(design)*, pengkodean *(coding)*, pengujian *(testing)*, menunjukan peforma dan pengembangan dimana memungkinkan tim pengembang ketika terjadi perubahan kebutuhan, hal ini bisa dilihat dari kecepatan pengembangan
2. Pengembangan pada *endpoint* yang dihasilkan sudah memenuhi kebutuhan pengguna, dimana data *input* dan *output*, kemudian ekspetasi dan hasil pada status code sesuai dalam menangani permasalahan penjadwalan tugas, dan pertemuan dengan *client.*
3. Penggunaan *Rest API* pada aplikasi *prospect* dalam hal kecepatan lebih baik dalam bagian kecepatan *response* dan optimasi pada pengembangan pada masa depan menjadi lebih testruktur pada sistem pada *respons* sistem, dengan memanfaatkan protocol *HTTP*.
4. Penelitian ini menujukan bahwa penerapan optimasi antara arsitektur *Rest* dan *Extreme Programming (XP)* fondasi kuat untuk pengembangan dan pemeliharaan, Saran untuk penelitian selanjunya tetap fokus pada optimasi lebih lanjut menggunakan teknologi *GraphQL* untuk meningkatkan fleksibilitas dan skalabilitas.

V. Daftar Pustaka

[1] P. F. Tanaem, A. F. Wijaya, A. D. Manuputty, and G. N. Huwae, "Penerapan RESTFul Web Service Pada Disain Arsitektur Sistem Informasi Pada Perguruan Tinggi (Studi Kasus: STARS UKSW)," JASIEK (Jurnal Aplikasi Sains, Informasi, Elektronika dan Komputer), vol. 2, no. 1, Jun. 2020, doi: 10.26905/jasiek.v2i1.3483.

[2] S. Butsianto and A. Iskianto, "SIGMA-Jurnal Teknologi Pelita Bangsa Implementasi Web Service Dengan Metode Representational State Transfer (Rest) Untuk Sistem In Formasi Sales Order Berbasis Mobile," 2019.

[3] S. A. Majapahit and A. Somantri, "Pengukuran Kinerja Web Service Untuk Integrasi Data Sikapeta Dengan SITU2," 2024. [Online]. Available: https://journal-computing.org/index.php/journal-cisa/index

[4] R. Choirudin and A. Adil, "Implementasi Rest Api Web Service dalam Membangun Aplikasi Multiplatform untuk Usaha Jasa," MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer, vol. 18, no. 2, pp. 284–293, May 2019, doi: 10.30812/matrik.v18i2.407.

[5] H. Wakkang and B. Irianto, "Implementasi Web Service Dengan Metode Rest Api Untuk Integrasi Data Covid 19 Di Sulawesi Selatan," vol. 2, no. 1, pp. 12–22, doi: 10.31850/jsilog.v2i1.

[6] F. Pradana, F. A. Bachtiar, M. Dearifaldi, and A. Ikhsan, "Implementasi Weighted Product Untuk Memberikan Rekomendasi Prospek Pelanggan Bagi Sales Marketing Berdasarkan Web Analytics," vol. 7, no. 2, pp. 367–372, 2020, doi: 10.25126/jtiik.202072586.

[7] W. Galindra Wardhana, I. Arwani, and B. Rahayudi, "Implementasi Teknologi Restful Web Service Dalam Pengembangan Sistem Informasi Perekaman Prestasi Mahasiswa Berbasis Website (Studi Kasus: Fakultas Teknologi Pertanian Universitas Brawijaya)," 2020. [Online]. Available: http://j-ptiik.ub.ac.id

[8] D. Rozzi Pratama, E. Budi, I. S. Kom, R. Robbani, and A. Md, "Pengembangan Rest Api Siabang (Sistem Administrasi Pembangunan) Menggunakan Java (SiAbang REST API (Development Administrative System) Using Java Development." [Online]. Available: http://begawe.unram.ac.id/index.php/JBTI/

[9] R. I. Borman, A. T. Priandika, and A. R. Edison, "Implementasi Metode Pengembangan Sistem Extreme Programming (XP) pada Aplikasi Investasi Peternakan," Jurnal Sistem dan Teknologi Informasi (Justin), vol. 8, no. 3, p. 272, Jul. 2020, doi: 10.26418/justin.v8i3.40273.

[10] A. Suwondo, A. Wafa Mansur, J. Akuntansi, and P. Negeri Semarang, "Penerapan Extreme Programming Dalam Pengembangan Sistem Informasi Penjadwalan Kuliah Jurusan Akuntansi Polines," vol. 14, no. 1, pp. 1–7, 2023, [Online]. Available: http://ejurnal.provisi.ac.id/index.php/JTIKP

[11] P. Arta Mandala, J. A. Jenderal Yani No, S. Selatan, and A. Kesehatan, "Mandala et al, Penerapan Metode Extreme
Programming…….… 71
Penerapan Metode Extreme Programming (XP) pada Pengembangan Sistem Informasi Penyewaan Lapangan Olahraga (SIPELA)."

[12] M. Würsch, R. Ingold, and M. Liwicki, "DivaServices-A RESTful web service for Document Image Analysis methods," Digital Scholarship in the Humanities, vol. 32, pp. i150–i156, Apr. 2017, doi: 10.1093/llc/fqw051.

[13] Y. Irawan Chandra, D. Ruri Irawati, M. Riastuti, and S. Jakarta STI, "Penerapan Model Agile-Extreme Programming (XP) Dalam Membuat Aplikasi Pengenalan Daerah Wisata di Wonogiri Berbasis Web", doi: 10.37817/ikraith-informatika.v8i1.

[14] S. Dhina Pohan and I. Firdaus, "Implementation Of Extreme Programming Method In The Development Of Pekanbaru Community Training Information System."

[15] M. K. Naufal, F. Affrianto, and A. B. Cahyono, "Implementasi REST API Untuk Fitur Rencana Strategis Program Pada SIMPEDA."