PENGEMBANGAN APLIKASI MANAJEMEN REMOTE LABORATORY MENGGUNAKAN METODE RESTFUL WEB SERVICE BERBASIS MOBILE

Febrian Mebiyantara, Ahmad Faisol, F.X. Ariwibisono
Program Studi Teknik Informatika S1, Fakultas Teknologi Industri
Institut Teknologi Nasional Malang, Jalan Raya Karanglo km 2 Malang, Indonesia
briandsmith22@gmail.com

ABSTRAK

Laboratorium jauh adalah lingkungan perangkat lunak yang mendukung kegiatan praktikum jarak jauh, di mana *user* yang berada pada jarak yang jauh dimungkinkan untuk berinteraksi dengan perangkat pengukuran dan peralatan laboratorium yang sesungguhnya. Di masa pandemi saat ini hampir di semua kalangan instansi pendidikan terutama di universitas dan sekolah melakukan sistem pembelajaran *online* guna menekan angka infeksi dari virus corona yang mewabah di seluruh dunia. Dengan di berlakukannya sistem pembelajaran *online* menimbulkan beberapa masalah salah satunya terkait dengan praktikum laboratorium yang sering kali harus dilakukan didalam laboratorium menjadi terhalang akibat adanya sistem pembelajaran dari rumah. Mayoritas laboratorium jauh yang ada saat ini diimplementasikan dengan komputer *desktop*. Sistem berbasis komputer memiliki kekurangan yaitu konsumsi energi listrik yang besar dan biaya investasi alat yang mahal.

Aplikasi *mobile* dapat menjadi *alternative* agar biaya investasi energi dan alat dapat lebih ditekan lagi serta dapat lebih memudahkan dalam manajemen pengaksesan laboratorium jauh karena sifatnya yang dapat digunakan kapan saja dan dimana saja. Aplikasi ini dikembangkan untuk me-*remote* alat *oscilloscope* dan *signal generate* milik *red pitaya*. Dalam manajemen akses alat ini menggunakan teknologi *RESTful Web Service* sehingga untuk transfer data lebih cepat dan memakan ukuran data yang relatif kecil. Untuk me-*remote* alat, aplikasi ini menggunakan *webview plugin* untuk menampilkan simulator yang di akses melalui ip yang telah di sediakan oleh alat tersebut.

Berdasarkan hasil pengembangan pada penelitian ini didapat 6 kebutuhan fungsional utama dan 2 kebutuhan *non* fungsional. Berdasarkan pengujian dapat disimpulkan bahwa pegembangan aplikasi manajemen *remote laboratory* menggunakan metode *RESTful Web service* berbasis *mobile* ini telah memenuhi kriteria sesuai dengan perancangan.

Kata kunci : *Mobile, Representational State Transfer(REST), Web Service, Remote Laboratory, Oscilloscope.*

1. PENDAHULUAN

1.1. Latar Belakang

Laboratorium jauh adalah sebuah arsitektur dari kumpulan perangkat lunak yang dapat mendukung kegiatan praktikum jarak jauh, di mana pengguna yang berada pada jarak yang jauh dapat berinteraksi dengan perangkat pengukuran dan peralatan laboratorium yang sesungguhnya. Keuntungan penggunaan laboratorium jauh yaitu kinerja laboratorium yang lebih baik dan lebih efisien, mendukung kegiatan berbagi pakai sumber daya dan kolaborasi antar laboratorium. Beberapa penelitian laboratorium jauh yang telah dipublikasikan, menyebutkan bahwa mayoritas laboratorium jauh yang ada saat ini diimplementasikan dengan komputer *desktop*. Sistem berbasis komputer memiliki kekurangan yaitu konsumsi energi listrik yang besar dan biaya investasi sistem yang mahal.

Di masa pandemi saat ini hampir di semua kalangan instansi pendidikan terutama di universitas dan sekolah melakukan sistem pembelajaran *online* guna menekan angka infeksi dari virus corona yang mewabah di seluruh dunia. Dengan di berlakukannya sistem pembelajaran *online* menimbulkan beberapa masalah salah satunya terkait dengan praktikum laboratorium yang sering kali harus dilakukan didalam laboratorium menjadi terhalang akibat adanya sistem pembelajaran dari rumah.

Smartphone menjadi sebuah *trend* saat ini karena bersifat mobile dan mudah digunakan. penggunaan smartphone tidak terbatas pada kalangan *IT developer*, bisnis, dan *government* saja, tetapi sudah hampir semua bidang profesional keahlian. Pada era industri 4.0 ini smartphone diadaptasi dalam berbagai bidang seperti bidang entertaiment, *science*, social *networking*, ekonomi, dan Pendidikan dengan mengutamakan fungsi serta keunggulan dalam teknologi komunikasi.

Berbagai alasan yang dapat menjadi faktor utama dalam pengembangan *mobile* learning agar tercipta sebuah paradigma baru dalam pembelajaran yang dapat dilakukan kapan saja dan dimana saja antara lain tingkat perkembangan teknologi perangkat *mobile* yang bergerak sangat cepat, tingkat kemudahan penggunaan, dan harga perangkat yang terjangkau, dibanding dengan perangkat komputer *desktop*, Kemudahan dalam menggunakan perangkat mobile dapat menjadi solusi atas keterbatasan pelajar dalam mengakses materi pelajaran / *resources.*

Untuk menekan pengeluaran konsumsi daya dan investasi alat yang mahal sekaligus mendukung konsep dari penggunaan sumber daya komputer

secara efisien atau sering disebut dengan konsep *green it.* Perlu dilakukan pengembangan aplikasi *mobile* untuk memanajemen *user* dalam penggunaan laboratorium jauh yang sudah ada. Pengembangan aplikasi ini menggunakan metode *RESTful API* untuk melakukan komunikasi dan penggunaan data baik di *database* maupun pada sistem tertanam laboratorium jauh. salah satu mekanisme pengiriman data oleh *server* dilakukan menggunakan *Web service*. *Web* service merupakan sebuah teknologi yang dapat mengubah kemampuan internet dengan cara menambahkan teknologi *transactional Web,* yaitu teknologi *Web* yang dapat berkomunikasi dengan pola *program-to- program (P2P)*. [2] *Web service* bekerja ketika *user me-request* suatu fungsi pada *Server* dan kemudian *Server* melakukan perintah sesuai dengan layanan yang diinginkan *user*. Arsitektur yang dapat mendukung transmisi data pada teknologi *Web service* antara lain *Simple Object Access Protocol*(*SOAP*) dan *Representational State Transfer*(*REST*). *SOAP* merupakan *Web service* yang menggunakan XML (*Extensible Markup Languange*) format data dan HTTP sebagai protokol transmisi data. *REST* merupakan *architectural design* untuk *distributed hypermedia system* dan untuk mengidentifikasi *resource*, *REST* menggunakan *Uniform Resource Identifier*(URI). [1]

Adapun tujuan yang ingin dicapai dari pelaksanaan penelitian ini adalah dalam rangka mengembangkan aplikasi manajemen *remote laboratory* berbasis *mobile*. Disamping itu tersedianya alat laboratorium jauh. Secara tidak langsung implementasi dari penelitian ini diharapkan dapat menjadi aplikasi yang lebih efektif, efisien dan praktis dalam penggunaan laboratorium jauh.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan diatas, maka permasalahan yang dibahas dalam program ini:

1. Bagaimana menciptakan aplikasi manajemen *remote laboratory* yang efisien dan mendukung konsep green it?
2. Bagaimana mengimplementasikan *RESTful API* untuk melakukan komunikasi dengan sistem alat laboratorium jauh yang sudah ada?
3. Seberapa besar pengaruh aplikasi manajemen *remote laboratory* terhadap sistem alat laboratorium jauh yang sudah ada?

1.3. Tujuan

Berdasarkan masalah yang telah dirumuskan sebelumnya, maka tujuan penyusunan dari laporan penelitian ini adalah sebagai berikut :

1. Menghasilkan perangkat lunak yang dapat memudahkan manajemen *user* untuk menjadwalkan penggunaan laboratorium jauh kapan saja dan dimana saja.
2. Dengan adanya aplikasi ini diharapkan dapat lebih efisien dan menekan angka pengeluaran daya dan biaya investasi alat yang mahal serta mendukung konsep *green it*.

1.3. Batasan Masalah

Agar pembahasan dalam penelitian ini tidak meluas, maka pada penelitian ini memiliki beberapa batasan masalah sebagai berikut.

1. Bahasa pemrograman yang digunakan untuk mengembangkan aplikasi adalah Dart dengan framework Flutter, PHP versi 4, dan MySQLi sebagai *database*.
2. Metode yang digunakan adalah *RESTfull API* yang digunakan untuk mengambil data dari *database* dan melakukan *remote* ke alat laboratorium jauh.
3. Lokasi Penelitian di Prodi Teknik Elektro Institut Teknologi Nasional Malang.
4. Pengembangan aplikasi hanya untuk *user*.
5. Pada pengembangan aplikasi ini hanya berfokus me-*remote* alat untuk melakukan praktikum *oscilloscope* dan *signal generate* dengan *red pitaya*.
6. Sistem hanya berjalan pada perangkat *mobile*.
7. Menggunakan IDE *Visual Studio Code*.
8. Tidak membahas secara rinci tentang perangkat pada arsitektur laboratorium jauh.

2. TINJAUAN PUSTAKA

2.1. Penelitian Terkait *RESTfull API*

Menurut penelitian Faisal Roufa Rohman dkk. [6], dengan judul "Pengembangan Perangkat Lunak Aplikasi Monitoring Klimatologi Menggunakan Metode *RESTful Web service* Berbasis Android (Studi Kasus: Stasiun Klimatologi Karangploso Malang)", Penelitian ini bertujuan untuk menerapkan metode *RESTful Web service* pada studi kasus data klimatologi. Penyimpanan data klimatologi dilakukan pada sisi *Server* dan sisi *Client* akan melakukan akses menggunakan *REST*. Pada sisi *client* sendiri di bangung menggunakan *platform* android. Android merupakan Os dengan jumlah pengguna teerbanyak di dunia, sehingga dapat di gunakan oleh banyak pengguna. Sisi *Client* dikembangkan dengan menggunakan bahasa pemrograman java sehingga disebut *native* android. Pengembangan aplikasi yang lebih efektif, efisien dan praktis dalam hal monitoring klimatologi merupakan hasil yang di harapkan.

Menurut penelitian yang dilakukan oleh Dudhe dan Sherekar.[5], metode *RESTful* memiliki keunggulan pada *requested* data yang lebih kecil dibandingkan metode *SOAP*. Ukuran data *request* yang lebih kecil memungkinkan mempersingkat waktu transmisi, konsumsi daya lebih rendah dan *Web service* yang lebih cepat dibanding dengan *SOAP*.

2.2 . Laboratorium Jauh (*Remote Laboratory*)

Laboratorium untuk melakukan penelitian dan pengujian adalah komponen yang sangat penting dalam pendidikan dan penelitian bidang teknik. Percobaan-

percobaan di laboratorium berguna untuk meningkatkan motivasi belajar siswa dan memperkuat pemahaman mereka tentang konsep-konsep abstrak dan teori-teori yang diajarkan dalam perkuliahan. Aktivitas laboratorium meliputi pengukuran, pengumpulan data, analisa, desain, dan pengalaman menggunakan peralatan secara langsung.

Kegiatan praktikum di laboratorium adalah bentuk kegiatan paling umum dalam lingkungan laboratorium, dimana siswa dapat melakukan percobaan secara nyata yang berhubungan dengan riset atau materi pendidikan. Dari hasil pengamatan bahwa untuk menyelenggarakan praktikum di laboratorium memerlukan biaya yang tinggi, hal ini berkaitan dengan penyediaan peralatan laboratorium, ruangan laboratorium dan staf laboratorium untuk pelayanan dan pemeliharaan. Investasi mahal, seperti biaya perawatan dan keamanan berkaitan dengan laboratorium merupakan hal penting yang perlu dipertimbangkan dalam memilih dan mendisain laboratorium teknik. Menggunakan cara berbagi sumber daya laboratorium dengan menggunakan internet atau memanfaatkan eksperimen versi simulasi adalah pilihan terbaik untuk mengatasi keterbatasan biaya, [7].

Menurut J. Garcia-Zubia, I. Angulo dkk. [8], laboratorium jauh adalah kombinasi antara perangkat keras dan perangkat lunak yang memungkinkan siswa untuk melakukan kegiatan praktikum dari jauh. Siswa mengontrol peralatan yang berada di laboratorium secara remote melalui antarmuka *Web* dan mengamati hasil keluaran melalui *Web*cam.

2.3. Metode *RESTful API*

REST (*Representional State Transfer*) adalah standar *de-facto* untuk arsitektur perangkat lunak untuk aplikasi interaktif yang biasanya menggunakan beberapa layanan *Web*. Untuk digunakan dalam aplikasi berbasis *REST*, Layanan *Web* harus memenuhi batasan tertentu; Layanan *Web* semacam itu disebut *RESTful*. Layanan *Web* yang tenang diperlukan untuk menyediakan akses aplikasi ke sumber daya *Web*nya dalam representasi tekstual dan mendukung pembacaan dan modifikasi mereka dengan protokol stateless dan serangkaian operasi yang telah ditentukan. Dengan *RESTful*, Layanan *Web* menyediakan interoperabilitas antara sistem komputer diinternet yang menyediakan layanan ini. *REST* menawarkan alternatif, misalnya, *SOAP* sebagai metode akses ke Layanan *Web*.

Cara kerja dari *REST API* ini adalah *server* akan dianggap sebagai sebuah *object* yang dapat dibuat, diupdate, dihapus dan juga dibaca oleh *client*. Dengan kata lain *create* sebagai request POST, update sebagai request *PUT* atau *PATCH*, hapus sebagai *request DELETE*, dan baca sebagai *request GET*.

Beberapa *method* menunjukkan tipe dari *request* yang biasanya digunakan untuk melakukan beberapa aksi yaitu *CREATE, READ, UPDATE, DELETE* (CRUD). Membuat, Membaca, Mengubah dan Menghapus.

GET. adalah *method* yang digunakan untuk membaca atau mendapatkan data yang terdapat pada *server*. Pada saat kita melakukan pengambilan data atau *request* dengan *method GET* maka *server* akan mengembalikan data yang dicari melalui *response*. Proses ini sama dengan aksi *READ* (Membaca).

POST. adalah *method* yang digunakan untuk men-*create* atau membuat data baru pada *database server*. Proses pengiriman data dibuat melalui *body* dari *request* yang kita kirimkan. Proses ini sama dengan aksi *CREATE* (Membuat).

PUT / *PATCH*. adalah *method* yang digunakan untuk lakukan perubahan data yang sudah ada pada *database server*. Prosesnya hamper seperti *POST* namun biasanya method ini hanya digunakan untuk mengubah data yang sudah ada. Proses ini sama dengan aksi *UPDATE* (Memperbarui).

DELETE. adalah *method* yang digunakan untuk menghapus data yang sudah ada pada *database server*. Proses ini sama dengan aksi *DELETE* (Menghapus),

2.4. Green IT

Istilah *Green IT* berkaitan dengan teknologi informasi dan lingkungan. *Green IT* meliputi proses perancangan, pembuatan, penggunaan serta pembuangan limbah yang berkaitan dengan komputer beserta subsistem yang terkait, secara efisien dan efektif dengan dampak minimal atau tidak berdampak pada lingkungan. *Green IT* juga berusaha untuk mencapai kelayakan ekonomi, meningkatkan kinerja sistem, menurunkan konsumsi energi, menghemat ruang dan menurunkan biaya sistem termasuk biaya pembuangan dan daur ulang. Dalam sebuah *survey* yang dilakukan oleh *Sun Microsystems* Australia yang melibatkan 1.500 responden dari 758 organisasi besar dan kecil di Australia dan Selandia Baru, responden mengatakan mengurangi konsumsi daya dan menurunkan biaya adalah alasan utama untuk menggunakan ecoresponsible practices, diikuti dengan dampak lingkungan yang lebih rendah dan meningkatkan penggunaan sistem, [7].

3. ANALISIS DAN PERANCANGAN

Pengembangan aplikasi manajemen remote laboratory menggunakan metode *RESTful Web* service dan flutter. *RESTful Web* service digunakan sebagai server untuk melayani data request sedangkan flutter digunakan sebagai client untuk mengambil data dan melakukan request ip *console oscilloscope* terhadap alat laboratorium jauh.

3.1. Studi literatur

Mempelajari ilmu-ilmu yang berkaitan dengan pengembangan aplikasi menggunakan android dan *RESTful Web service* diantaranya:

1. *Representational State Transfer*(*REST*)
2. Flutter
3. Dart
4. Pengujian aplikasi
5. Berbagai *paper*, jurnal dan buku yang terkait dengan *REST AP*

Literatur tersebut dipaparkan dalam bentuk *paper*, jurnal, dan buku.

3.2. Analisis Kebutuhan Sistem

Analisis kebutuhan merupakan proses untuk menentukan kebutuhan apa saja yang dibutuhkan oleh sistem.

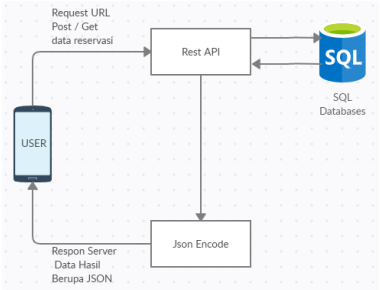
Aplikasi *mobile* dipilih karena dapat memanajemen dan mengakses data dengan cepat. Tidak perlu membuka *browser* untuk mengakses *console oscilloscope* untuk praktikum *oscilloscope*. Tampilan *console* simulasi akan di *generate* menggunakan *Webview plugin* pada aplikasi *mobile*. Penentuan jadwal dan akses dapat di lakukan kapan saja dan dimana saja sehingga lebih fleksibel dan efisien. Kebutuhan tersebut yaitu:

1. Kebutuhan fungsional, yang merupakan tujuan dikembangkannya sistem meliputi *input* data reservasi, melihat jadwal penggunaan *remote* lab, dan dapat akses *console* praktikum sesuai dengan waktu yang telah di jadwalkan.
2. Kebutuhan non fungsional yang merupakan ukuran performa dari sistem.

3.3 Perancangan

Sistem yang akan dibangun memiliki dua sisi, yaitu sisi *Client* dan sisi *Server*. Pada sisi *Server* sebagai sisi yang menyediakan data reservasi untuk dikirimkan pada *Client* sebagai landasan pengaturan waktu akses *console* simulasi *oscilloscope*. Sisi *Server* menggunakan *RESTful Web service* dan akan dibangun menggunakan php *native*.

Web service tidak menggunakan tampilan *User interface* karena berfokus pada layanan. *Client* merupakan aplikasi yang langsung terhubung dengan *user*. *Client* menggunakan *User interface* sebagai perantara untuk berkomunikasi antara *User* dan *Server*.



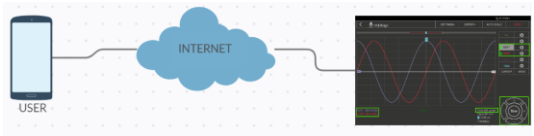
Gambar 1. Hubungan REST Web service dengan client

Alur komunikasi antara *Client* dengan *Server* akan sebagai berikut :

1. User memasukkan data *input* reservasi berupa tanggal, modul, dan waktu awal.
2. Hasil input yang dilakukan *user* akan diubah menjadi *string*.
3. Client mengirimkan *input* ke *server* menggunakan HTTP dengan *method* POST.
4. *Input* diterima dan disimpan pada *database* oleh *server*.
5. *Server* mengambil data dari tabel reservasi pada *database* sesuai dengan data *input* yang telah dilakukan.
6. Hasil data reservasi di-*encode* menjadi JSON.
7. JSON dikembalikan kepada *client* sebagai respon.
8. *Client* menangkap respon dan melakukan *decode*.

Hasil *decode* ditampilkan pada *user interface Client* sebagai *list schedule remote laboratory* dan digunakan untuk mengatur waktu akses *console* simulasi *oscilloscope*.

3.3. Topologi Konektivitas Aplikasi

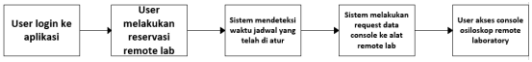


Gambar 2. Topologi Konektivitas Akses Console Praktikum

Pada gambar 2. Merupakan gambaran konektivitas terjadi antara aplikasi *mobile* dengan *remote laboratory*. Sistem tertanam *remote laboratory* akan memberikan alamat ip kemudian alamat ip tersebut akan di akses oleh aplikasi menggunakan *RESTful http*. Kemudian *console* untuk melakukan praktikum akan di tampilkan melalui *Webview plugin* pada aplikasi *mobile*.

3.5 Blok Diagram Sistem

Dalam implementasi aplikasi manajemen *remote laboratory* berbasis *mobile* menggunakan metode *Resfull API* ini digunakan sebagai *system* yang dapat membantu menetapkan jadwal penggunaan *remote* lab dan mengakses *remote* lab sesuai dengan jadwal yang di tentukan.

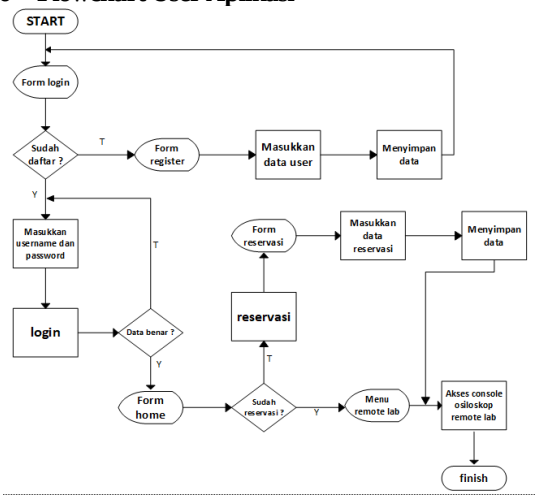


Gambar 3. Blok Diagram Akses Console Oscilloscope Remote Laboratory

Penjelasan Gambar 3. Blok Diagram Sistem yaitu sebagai berikut :

1. *User* login ke aplikasi. Jika login gagal atau *user* belum memiliki akun, maka *user* bisa langsung daftar melalui aplikasi.
2. *User* melakukan reservasi *remote* lab dengan mengisikan tanggal, modul, dan waktu awal.
3. Sistem melakukan pengecekan tanggal dan waktu akses *remote* lab.
4. Sistem akan melakukan *request* ip *console* praktikum ke *remote* lab
5. *User* dapat melakukan akses ke *console oscilloscope*.

3.6 Flowchart User Aplikasi

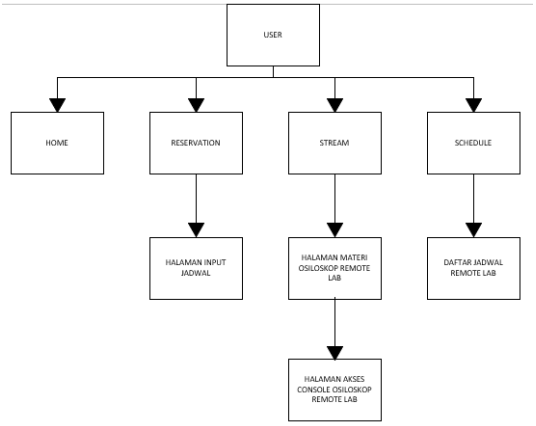


Gambar 4. Flowchart user aplikasi manajemen remote lab.

Pada Gambar 4. Menunjukkan bahwa aplikasi ini dimulai dari tampilan form login yang akan diisikan oleh *user* jika sudah register. Tetapi kalau belum register, maka *user* akan di hadapkan ke halaman register terlebih dahulu kemudian login kembali dengan *user* baru. Setelah itu jika data yang dimasukkan sudah *valid* maka akan dilanjutkan ke *home*. Kemudian *user* akan melakuan reservasi *remote* lab setelah melakukan reservasi dapat melakukan akses ke *console oscilloscope remote* lab melalui halaman menu *stream remote* lab.

3.7 Struktur menu

Struktur menu *user* aplikasi manajemen remote lab merupakan gambaran sistem secara terstruktur untuk menu-menu yang ada di aplikasi *user*. Gambar 5 berikut ini merupakan tampilan dari struktur menu *user* aplikasi manajemen remote lab.



Gambar 5. Struktur menu user aplikasi manajemen remote lab

Struktur menu pada Gambar 5. Merupakan daftar menu yang dapat diakses oleh *user* aplikasi *remote* lab. *User* dapat mengakses menu seperti *home*, *reservation*, *stream*, dan *schedule*.

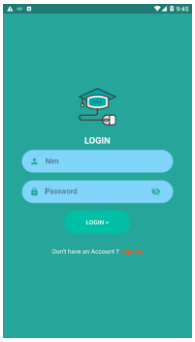
4. HASIL DAN PEMBAHASAN

4.1. Antarmuka Awal

Gambar 6. Antarmuka awal

Tampilan ketika aplikasi pertama kali di jalankan. Terdapat dua *button* yaitu *sign in* untuk pindah ke halaman *login* dan *sign up* untuk pindah ke halaman *register.*

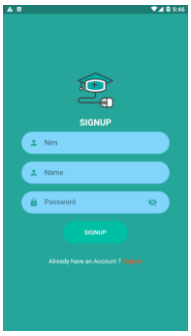
4.2. Implementasi antarmuka login



Gambar 7. Antarmuka login

Gambar 7. Merupakan tampilan dari halaman *login.* Terdapat dua *textfield* yaitu nim dan *password* yang akan digunakan untuk *login.*

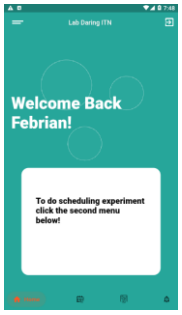
4.3. Implementasi antarmuka register



Gambar 8. Antarmuka register

Tampilan dari halaman *register.* Terdapat tiga *textfield* yaitu nim, *name* dan *password* yang akan digunakan untuk mendaftar *user* baru.

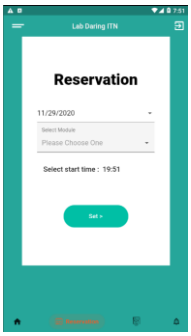
4.4. Implementasi antarmuka home menu



Gambar 9. Antarmuka home menu

Tampilan dari halaman *home* setelah berhasil melakukan *login.* Terdapat empat *menu navigation* yaitu *home, reservation, stream,* dan *schedule.*

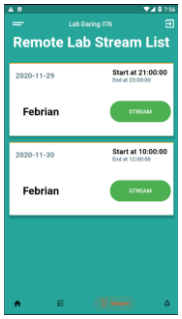
4.5. Implementasi antarmuka reservasi menu



Gambar 10. Antarmuka reservasi menu

Tampilan dari halaman *reservation* yang berfungsi sebagai halaman untuk penjadwalan akses *remote laboratory* pada sistem tertanam*remote laboratory*. Terdapat *datefield* untuk mengisi tanggal, satu *field dropdown* untuk memilih modul dan *timefield* untuk mengisi waktu akses yang dinginkan.

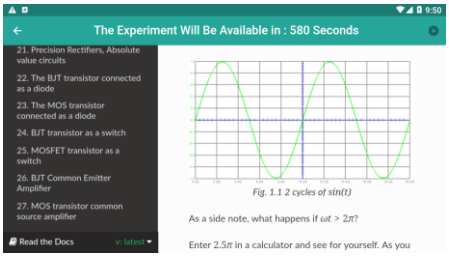
4.6. Implementasi antarmuka stream menu



Gambar 11. Antarmuka stream menu

Pada Gambar 11. Merupakan tampilan *stream menu* dimana pada menu ini akan di lakukan akses dan pengambilan data *console* simulasi *oscilloscope* dari sistem tertanam *remote laboratory* ke *Webview* pada aplikasi *mobile*. Kemudian terdapat *timer* dimana ketika waktu habis maka *user* akan di *logout* paksa dari aplikasi.

4.7. Implementasi antarmuka materi praktikum

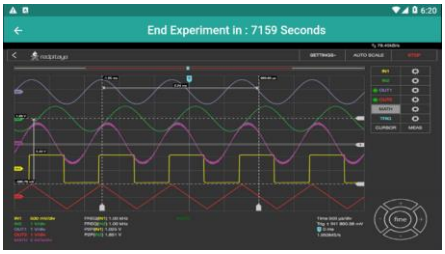


Gambar 12. Antarmuka materi praktikum

Pada Gambar 12 merupakan antarmuka setelah melakukan klik tombol *stream.* Dimana akan diarahkan ke antarmuka materi-materi yang berkaitan dengan *oscilloscope*. Pada antarmuka terdapat button untuk akses *console* simulasi *oscilloscope* dan waktu *countdown.* Jika waktu yang ditentukan untuk praktikum belum masuk maka *button console* akan *disable*.

4.8. Implementasi antarmuka console simulasi *oscilloscope*.

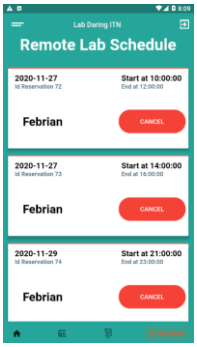
Pada Gambar 13 merupakan tampilan antarmuka setelah waktu *countdown* telah habis. Terdapat beberapa *console* yang digunakan untuk melakukan simulasi praktikum *oscilloscope* dan *signal generate*



Gambar 13. Antarmuka console simulasi oscilloscope

4.9. Implementasi antarmuka schedule menu

Pada Gambar 14. Merupakan antarmuka untuk menampilkan histori reservasi yang pernah dilakukan. Pada halaman ini juga *user* dapat melakukan pembatalan reservasi dengan klik button cancel.



Gambar 14. Antarmuka schedule menu

4.10. Pengujian

Dari hasil implementasi pada sistem aplikasi manajemen remote laboratory menggunakan metode *RESTful Web* service berbasis *mobile* maka dilakukan 4 pengujian yaitu *blackbox*, *response time*, kompatibilitas, dan efisiensi sistem .

4.11. Pengujian Black Box

Pada pengujian *blackbox* terdapat 14 kasus uji untuk kebutuhan fungsional.

Tabel 1. Tabel Pengujian black box

Nama kasus	Hasil yang diharapkan	Hasil yang diperoleh	validitas
Kasus Uji register	Penguji berhasil menyimpan data registrasi	Penguji berhasil menyimpan data registrasi	Valid
Kasus Uji *login*	Penguji dapat masuk ke halaman *home* menggunakan *user* yang telah di register	Penguji dapat masuk ke halaman *home* menggunakan *user* yang telah di register	Valid
Kasus Uji login *username* salah *password* benar	Penguji tidak dapat *login*	Penguji tidak dapat *login*	Valid
Kasus Uji login *username* benar *password* salah	Penguji tidak dapat *login*	Penguji tidak dapat *login*	Valid
Kasus Uji login *username* salah *password* salah	Penguji tidak dapat *login*	Penguji tidak dapat *login*	Valid
Kasus Uji reservasi	Penguji berhasil melakukan *input* data reservasi	Penguji berhasil melakukan *input* data reservasi	Valid
Kasus Uji *stream menu*	Sistem berhasil mengambil dan menampilkan data reservasi sesuai dengan *user* yang *login*	Sistem berhasil mengambil dan menampilkan data reservasi sesuai dengan *user* yang *login*	Valid
Kasus Uji *stream menu*	Berhasil menampilkan *countdown* waktu dan materi praktikum ketika *button stream* di klik	Berhasil menampilkan *countdown* waktu dan materi praktikum ketika *button stream* di klik	Valid
Kasus Uji *stream menu*	Berhasil akses ip *console oscilloscope* dan melakukan simulasi praktikum	Berhasil akses ip *console oscilloscope* dan melakukan simulasi praktikum	Valid
Kasus Uji *stream menu*	Berhasil mencegah dan mengatur akses *console user* sesuai dengan tanggal dan waktu yang telah di tentukan	Berhasil mencegah dan mengatur akses *console user* sesuai dengan tanggal dan waktu yang telah di tentukan	Valid
Kasus Uji *stream menu*	*User* akan di *logout* otomatis saat waktu praktikum habis	*User* akan di *logout* otomatis saat waktu praktikum habis	Valid
Kasus Uji *schedule menu*	Sistem dapat mengambil dan menampilkan data reservasi yang telah di buat sesuai dengan *user* yang *login*	Sistem dapat mengambil dan menampilkan data reservasi yang telah di buat sesuai dengan *user* yang *login*	Valid
Kasus Uji *schedule menu*	Sistem dapat melakukan *cancel* reservasi dengan mengklik *button cancel*	Sistem dapat melakukan *cancel* reservasi dengan mengklik *button cancel*	Valid
Kasus Uji *logout*	Sistem dapat menghapus *session* penguji dan kembali ke halaman awal	Sistem dapat menghapus *session* penguji dan kembali ke halaman awal	Valid

Analisis dari pengujian didapat 14 kasus uji. Setelah dilakukan pengujian pada kasus uji tersebut didapat hasil 14 kasus uji berjalan sesuai dengan harapan dengan tingkat keberhasilan sebesar 100%.

Dengan begitu kebutuhan fungsional dari spesifikasi perancangan telah terpenuhi.

4.12. Pengujian Response Time

Pengujian response time bertujuan untuk mengukur kecepatan response dari server. Pengujian menggunakan *Postman*.

Tabel 2. Tabel Pengujian response time

Request Uji Ke-	Datetime	Status	Hasil Waktu
1	29/11/2020 19:00	200 OK	93,99 ms
2	29/11/2020 20:00	200 OK	87,26 ms
3	29/11/2020 21:00	200 OK	55,55 ms
4	30/11/2020 4:09	200 OK	395,1 ms
5	30/11/2020 5:18	200 OK	49.04 ms
6	30/11/2020 5:39	200 OK	110,57 ms
7	30/11/2020 6:30	200 OK	49,1 ms
8	30/11/2020 7:00	200 OK	76,47 ms
9	30/11/2020 7:19	200 OK	19,68 ms
10	30/11/2020 7:30	200 OK	83,28 ms

Analisis dari pengujian didapat 10 kali uji request dengan melakukan request data reservasi menggunakan *postman*. Setelah dilakukan pengujian pada 10 kasus uji tersebut didapat hasil 10 kasus uji sesuai dengan harapan dengan tingkat keberhasilan uji sebesar 100%. Dengan hasil ini maka kebutuhan non fungsional *response time* telah memenuhi spesifikasi perancangan.

4.13. Pengujian Kompatibilitas

Pengujian kompatibiltas merupakan pengujian perangkat yang digunakan untuk mengetahui mampu atau tidaknya aplikasi manajemen remote laboratory berjalan pada *device*, sistem operasi, ataupun lingkungan instalasi yang berbeda. Berikut hasil percobaan dari *compatibility testing* dalam tabel 3.

Tabel 3. Tabel Pengujian Kompatibilitas

Perangkat Uji	Time out	Ram (Gb)	Os	Orientasi	Masalah
Samsung Galaxy S6 SM-G920F, *API* Level 23	3 min 18 s	3	6.0.1 Marsmellow	Potrait	-
Redmi Note 6 Pro, *API* Level 28	5 min 5s	4	9 Pie	Potrait	-
Nokia 1, *API* Level 27	5 min 4s	1	8.0.1 Oreo	Potrait	-
Vivo 1805, *API* Level 27	1 min 33 s	8	8.0.1 Oreo	Potrait	-
Pixel, *API* Level 26	5 min 3s	4	8.0.0 Oreo	Potrait	-
Sony Xperia Z5 Compact	3 min 10s	2	7.1.1 Nougut	Potrait	-

Pengujian kompabilitas dilakukan dengan menggunakan bantuan *tools Firebase Test Lab*. Dari hasil uji dapat diketahui bahwa pengujian kompatibilitas sistem yang dilakukan di enam perangkat berbeda dengan level *API* yang berbeda juga tercatat tidak terdapat masalah yang ditemukan. Hal ini dapat diartikan bahwa aplikasi manajemen *remote laboratory* memiliki tingkat kompatibiltas yang cukup baik ketika dijalankan di berbagai perangkat android dengan tipe *device* dan *level API* yang berbeda. Aplikasi hanya dapat di *install* dengan *Operating System* minimum 6.0.0 *Marsmellow*. Untuk *performance* yang lebih baik sebaiknya perangkat yang di gunakan memiliki *ram* minimal 2Gb.

4.14. Pengujian Daya, Dimensi, dan Biaya

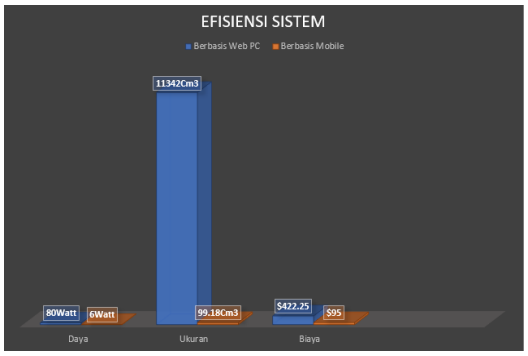
Sesuai dengan konsep dari *Green IT*, maka ada beberapa parameter yang perlu diperhatikan dalam membangun sistem berbasis IT, antara lain meningkatkan kinerja sistem, menurunkan konsumsi energi, menghemat ruang dan menurunkan biaya sistem [7]. Berikut ini akan dibahas perbandingan kebutuhan daya sistem, dimensi atau ukuran ruang yang dibutuhkan dan biaya yang diperlukan untuk membangun sebuah *server Web* berbasis *computer* PC maupun berbasis mobile. Data diambil dari referensi dan analisa perbandingan dinyatakan dalam bentuk grafik.

Dari hasil penelitian menyatakan bahwa kebutuhan daya sebuah PC memerlukan daya antara 60 sampai dengan 100 Watt, atau rata-rata sekitar 80Watt [7]. Sedangkan untuk *mobile* menggunakan *smartphone* Xiaomi Redmi Note 3 memiliki spesifikasi arus maksimum adalah 1500mA dengan tegangan 4V, sehingga kebutuhan daya maksimum adalah sekitar 6Watt. Sedangkan dimensi atau ukuran fisik dari *server Web* berbasis PC dengan *casing type* ATX adalah 9cm x 35,5cm x 35,5cm atau memiliki *volume* 11.342cm3, sedangkan ukuran dimensi *smartphone* Xiaomi Redmi Note 3 adalah 15cm x 7,6cm x 0,87cm atau memiliki *volume* 99,18cm3. Ditinjau dari biaya investasi *server Web* berbasis PC dengan spesifikasi CPU menggunakan prosesor core i5 harga rata-rata adalah sekitar $422.25, sedangkan untuk mobile biaya investasi rata-rata adalah sekitar $95. Tabel 4 menunjukkan rekap perbandingan

kebutuhan daya, dimensi dan biaya serta presentasi efisiensi yang dapat dihemat. Sistem dengan berbasis *mobile* dapat konsumsi daya sekitar 92,5%, penghematan ruang atau dimensi sistem sampai 99,12% dan penghematan biaya sampai 77,51%.

Tabel 4. Tabel Perbandingan Daya, Dimensi, dan Biaya

No	Parameter Efisiensi	Berbasis *Web* PC	Berbasis Mobile	(%)
1	Konsumsi Daya	80Watt	6Watt	92,5%
2	Ukuran Fisik	9cm x35,5cm x 35,5cm (11.342cm3)	15cm x 7,6cm x 0,87cm (99,18cm3)	99,12%
3	Biaya	$422.25	$95	77,51%



Gambar 15. Grafik Efisiensi Daya, Dimensi, dan Harga

5. KESIMPULAN DAN SARAN

5.1. Kesimpulan

Kesimpulan dari penelitian yang telah dilakukan untuk pengembangan perangkat lunak aplikasi Manajemen *Remote Laboratory* menggunakan metode *RESTful Web service* berbasis *Mobile* adalah sebagai berikut :

1. Pada pengembangan perangkat lunak aplikasi manajemen *remote laboratory* terdapat 6 kebutuhan fungsional utama. Aplikasi yang telah berhasil di buat terdiri dari 6 fitur yaitu registrasi yang berfungsi untuk mendaftarkan user baru, login yang berfungsi untuk mengautentikasi *user*, *reservation menu* yang berfungsi untuk melakukan manajemen jadwal akses, *stream menu* yang berfungsi untuk melakukan akses *remote laboratory*, *schedule menu* yang berfungsi untuk melihat *history* reservasi dari *user* dan *logout* yang berfungsi untuk keluar dari aplikasi.
2. Berdasarkan pengujian *black box* yang dilakukan, terdapat total 14 kasus uji terhadap aplikasi Manajemen *Remote Laboratory* dan didapat tingkat keberhasilan sebesar 100%. Sedangkan pada pengujian kebutuhan non fungional, Pengujian *Response Time* sebanyak 10 kali menghasilkan waktu rata-rata *response* sebesar 102,004 ms dengan tingkat keberhasilan sebesar 100%. Pengujian Kompatibilitas dilakukan uji terhadap 6 perangkat berbeda dengan spesifikasi dan level *API* berbeda menghasilkan tingkat keberhasilan sebesar 100%.
3. Analisis efisiensi aplikasi *remote laboratory* dilakukan dengan menganalisa beberapa parameter yang terkait dengan konsep *Green IT*, meliputi penurunan konsumsi energi, penghematan ruang dan penurunan biaya investasi perangkat keras. Dari hasil analisis dapat disimpulkan bahwa penerapan aplikasi manajemen *remote laboratory* berbasis *mobile* dalam sistem laboratorium jauh dapat menghemat konsumsi daya sampai 92,5%, penghematan ruang sebesar 99,12% dan penurunan biaya sistem sampai 77,51%.

5.2. Saran

1. Untuk penelitian kedepan sebaiknya dikembangkan lagi menggunakan database non sql seperti firebase atau aws amazon. Agar keamanan dan manajemen data pada database dapat lebih baik lagi.
2. Untuk penelitian kedepan sebaiknya di kembangkan lagi untuk me-remote instrument alat yang lain.

DAFTAR PUSTAKA

[1] Belqasmi, F., Singh, J., Melhem, S. Y. B. & Glitho, a. R. H., 2012. *SOAP*-Based vs. *RESTful Web* Services A Case Study for Multimedia Conferencing. IEEE Internet Computing, 16(4), pp. 54-63..

[2] Deviana, H., 2011. Penerapan XML *Web* service Pada Sistem Distribusi Barang. Jurnal Generic. , Volume Vol. 6, No. 2, Juli 2011, pp. 61-70.

[3] Luthfillah, I., 2014. Rancang Bangun *RESTful Web* Service Untuk Optimalisasi Kecepatan Akses Studi Kasus Aplikasi Sistem Pakar Berbasis *Web*, Malang: Universitas Brawijaya Malang.

[4] Lee, S., Jo, J., Kim, Y. & Stephen, H., 2014. A Framework for Environmental Monitoring with Arduino-Based Sensors Using *RESTful Web* Service. Services Computing, pp. 275-282.

[5] Dudhe, A. & Sherekar, S., 2014. Performance Analysis of *SOAP* and *RESTful* Mobile *Web* Services in Cloud Environment. IJCA Special Issue on Recent Trends in Information Security, Volume RTINFOSEC, pp. 1-4.

[6] Faisal Roufa Rohman, Arief Andy Soebroto, dan Agi Putra Kharisma., 2018. Pengembangan Perangkat Lunak Aplikasi Monitoring Klimatologi Menggunakan Metode *RESTful Web* service Berbasis Android (Studi Kasus: Stasiun Klimatologi Karangploso Malang)

[7] F. Yudi limpraptono., 2015. Arsitektur baru laboratorium jauh hijau multi-user dan multi-

device berbasis sistem tertanam. Disertasi. Fakultas Teknik, Departemen Teknik Elektro, Universitas Indonesia, Depok.

[8] J. Garcia-Zubia, I. Angulo, J. Irurzun, P. Orduna, J. Ruiz, U. Hernandez, M. Castro, and E. San-Cristobal, "Easily Integrable Platform for the Deployment of a Remote Laboratory for Microcontrollers," International Journal of *Online* Engineering (iJOE), vol. 6, no. 3, pp. pp. 26–31, Jul. 2010.