RANCANG BANGUN *RESTFUL* API *LIBRARY MANAGEMENT SYSTEM* ELIB UNILA BERINTEGRASI DENGAN APLIKASI ANDROID STATUS PEMINJAMAN BUKU

(Skripsi)

Oleh

ASHA IMALIA ZAHRA

NPM 2015061006

JURUSAN TEKNIK ELEKTRO

FAKULTAS TEKNIK

UNIVERSITAS LAMPUNG

2024

RANCANG BANGUN *RESTFUL* API *LIBRARY MANAGEMENT SYSTEM* ELIB UNILA BERINTEGRASI DENGAN APLIKASI ANDROID STATUS PEMINJAMAN BUKU

Oleh

ASHA IMALIA ZAHRA

Skripsi

Sebagai Salah Satu Syarat untuk Mencapai
Gelar Sarjana Teknik

Pada

Program Studi S1 Teknik Informatika

JURUSAN TEKNIK ELEKTRO
FAKULTAS TEKNIK
UNIVERSITAS LAMPUNG
2024

ABSTRAK

RANCANG BANGUN *RESTFUL* API *LIBRARY MANAGEMENT SYSTEM* ELIB UNILA BERINTEGRASI DENGAN APLIKASI ANDROID STATUS PEMINJAMAN BUKU

Oleh

Asha Imalia Zahra

Perkembangan teknologi informasi memudahkan manusia untuk meningkatkan kinerja, performa bisnis, serta kualitas layanan. Layanan Perpustakaan juga memanfaatkan kemajuan ini untuk bertransformasi sesuai kebutuhan pengguna, meningkatkan efisiensi dan aksesibilitas. Penelitian ini bertujuan merancang dan membangun *RESTful* API untuk aplikasi Android yang terintegrasi dengan ELIB, guna meningkatkan layanan dan akses informasi tanpa mengubah proses peminjaman yang ada. Pada penelitian ini, metode pengembangan perangkat lunak yang digunakan adalah *Rapid Application Development* (RAD). Implementasi metode RAD menggunakan model iteratif (berulang) untuk mengembangkan sistem, dengan iterasi dilakukan sebanyak tiga kali. *RESTful* API yang dibangun menghasilkan sembilan fitur diantaranya fitur *Login, Logout, User, Circulation Account, Circulation History, Circulation Status, Book Title, Book Author*, dan *Book Item*. Aplikasi Status Peminjaman Buku berbasis Android, bernama Perpusta, dikembangkan menggunakan bahasa pemrograman Kotlin. Aplikasi ini memiliki fitur *login*, informasi peminjaman buku, informasi pengembalian buku, notifikasi, dan *logout*. Hasil pengujian aplikasi menggunakan *blackbox testing* dengan enam skenario tes menunjukkan hasil yang sesuai dengan yang diharapkan. Hasil uji *usability* dengan metode *USE Questionnaire* menunjukkan aplikasi memenuhi fungsi utamanya dengan nilai akhir kelayakan 85%. Berdasarkan hasil pengujian performa *RESTful* API menggunakan metode *Stress Test*, tingkat kinerja *RESTful* API optimal saat diakses oleh 9,000 permintaan dalam periode ramp up 10 detik dan loop count 1. Server mengalami penurunan kinerja jika diakses oleh lebih dari 90,000 permintaan.

Kata Kunci : *RESTful API,* Android, RAD, *USE Questionnaire, Stress Testing*

ABSTRACT

RANCANG BANGUN *RESTFUL* API *LIBRARY MANAGEMENT SYSTEM* ELIB UNILA BERINTEGRASI DENGAN APLIKASI ANDROID STATUS PEMINJAMAN BUKU

By

Asha Imalia Zahra

The advancement of information technology facilitates humans in improving performance, business efficiency, and service quality. Library services also leverage these advancements to transform according to user needs, enhancing efficiency and accessibility. This research aims to design and develop a RESTful API for an Android application integrated with ELIB to enhance services and access to information without altering the existing loan process. In this research, the software development method used is Rapid Application Development (RAD). The implementation of the RAD method uses an iterative model to develop the system, with iterations conducted three times. The developed RESTful API includes nine features, including Login, Logout, User, Circulation Account, Circulation History, Circulation Status, Book Title, Book Author, and Book Item. The Android-based Book Loan Status application, named Perpusta, is developed using the Kotlin programming language. This application features login, book loan information, book return information, notifications, and logout. Application testing using blackbox testing with six test scenarios shows results that meet expectations. Usability testing results using the USE Questionnaire method indicate that the application fulfills its main functions with a final feasibility score of 85%. Performance testing results of the RESTful API using the Stress Test method show optimal performance when accessed by 9,000 requests in a ramp-up period of 10 seconds and a loop count of 1. The server experiences a decline in performance when accessed by more than 90,000 requests.

Keywords: RESTful API, Android, RAD, USE Questionnaire, Stress Testing

Judul Skripsi	:	**RANCANG BANGUN *RESTFUL* API *LIBRARY MANAGEMENT SYSTEM* ELIB UNILA BERINTEGRASI DENGAN APLIKASI ANDROID STATUS PEMINJAMAN BUKU**
Nama Mahasiswa	:	Asha Imalia Zahra
Nomor Pokok Mahasiswa	:	2015061006
Program Studi	:	Teknik Informatika
Fakultas	:	Teknik

MENYETUJUI

1.Komisi Pembimbing

Pembimbing Utama	Pembimbing Pendamping
Dr.Eng.Ir.Mardiana, S.T.,M.T.,I.P.M	**Rio Amestia Pradipta, S.Kom, M.T.I**
NIP 197203161999032002	NIP 198603232019031013

2. Mengetahui

Ketua Jurusan Teknik Elektro	Ketua Program Studi Teknik Informatika
Herlinawati, S.T.,M.T.	**Yessy Mulyani, S.T.,M.T**
NIP 197103141999032001	NIP 197312262000122001

MENGESAHKAN

1. Tim Penguji

Ketua : **Dr.Eng.Ir.Mardiana, S.T.,M.T.,IPM.** .....

Sekretaris : **Rio Ariestia Pradipta, S.Kom, M.T.I.** .....

Penguji : **Ir.Meizano Ardhi Muhammad, S.T., M.T.,IPM.** .....

2. Dekan Fakultas Teknik

Dr. Eng. Ir. Helmy Fitriawan, S.T., M.Sc.

NIP. 19750928 200112 1 002

Tanggal Lulus Ujian Skripsi : **11 Juni 2024**

SURAT PERNYATAAN

Saya yang bertandatangan di bawah ini , menyatakan bahwa skripsi saya dengan judul "Rancang Bangun *Restful* Api *Library Management System* Elib Unila Berintegrasi Dengan Aplikasi Android Status Peminjaman Buku" dibuat oleh saya sendiri. Semua hasil yang tertuang dalam skripsi ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti bahwa skripsi ini merupakan salinan atau dibuat oleh orang lain, maka saya bersedia menerima sanksi sesuai dengan ketentuan hukum atau akademik yang berlaku.

Bandar Lampung, 21 Juni 2024

Pembuat pernyataan,

Asha Imalia Zahra

NPM 2015061006

RIWAYAT HIDUP

Asha Imalia Zahra lahir di Kotabumi pada tanggal 1 Januari 2003. Penulis merupakan putri pertama dari tiga bersaudara dari pasangan Bapak Imawan dan Ibu Yuliantina, S.Pd. Penulis menyelesaikan pendidikan dasarnya di SD Negeri 1 Rejosari pada tahun 2014, kemudian melanjutkan pendidikan Sekolah Menengah Pertama di SMP Negeri 7 Kotabumi pada tahun 2017, dan menamatkan pendidikan Sekolah Menengah Atas di SMA Negeri 3 Kotabumi pada tahun 2020. Pada tahun 2020 penulis terdaftar sebagai mahasiswa Program Studi Teknik Informatika, Jurusan Teknik Elektro, Fakultas Teknik Universitas Lampung melalui jalur Seleksi Nasional Masuk Perguruan Tinggi Negeri (SNMPTN). Selama menjadi mahasiswa, penulis melakukan beberapa kegiatan antara lain:

1. Menjadi anggota biasa Himpunan Mahasiswa Teknik Elektro Universitas Lampung, Departemen Komunikasi dan Informasi, Divisi Media Informasi pada tahun 2021.
2. Mengikuti program Studi Independen Kampus Merdeka dari Kementrian Pendidikan dan Budaya dengan mengambil *Android Engineering* di Binar Academy pada tahun 2022.
3. Mengikuti Program Magang dalam Magang *Campus Leaders Program Batch* 6 di Bakrie Center Foundation sebagai divisi IT dan Pengelolaan Database pada tahun 2023.
4. Melaksanakan Kuliah Kerja Nyata di Desa Kerang, Kecamatan Batu Brak, Kabupaten Lampung Barat, Provinsi Lampung pada Januari 2023 sampai dengan Februari 2023.
5. Menjadi anggota Asisten Laboratorium Teknik Komputer Jurusan Teknik Elektro Universitas Lampung pada tahun 2022.

MOTTO

"Hiduplah dengan rasa syukur, beriman, bermartabat dan membahagiakan orang terdekat"

(Penulis)

"Jadilah wanita tangguh yang dapat berdiri di kaki sendiri"

(Ibu Penulis)

"Ingatlah untuk terus berdoa pada apa apa yang sedang diusahakan"

(Ayah Penulis)

"Tanpa cinta kecerdasan itu berbahaya. Dan tanpa kecerdasan, cinta itu tidak cukup"

(BJ. Habibie)

"Maka sesungguhnya bersama kesulitan ada kemudahan. Sesungguhnya bersama kesulitan ada kemudahan"

(Q.S. Al-Insyirah : 5-6)

"Sesungguhnya ketetapan-Nya apabila Dia menghendaki sesuatu Dia hanya berkata kepadanya, "Jadilah!" Maka jadilah sesuatu itu."

(Q.S. Yasin : 82)

PERSEMBAHAN

Puji dan syukur saya ucapkan kepada Allah SWT atas segala nikmat dan karunia-Nya sehingga skripsi ini dapat diselesaikan.

KUPERSEMBAHKAN KARYA KECILKU INI UNTUK:

"Ibu Yuliantina, ibuku tersayang yang telah melahirkanku, merawatku, membesarkan, dan dengan sepenuh hati telah menyayangi serta mendidiku"

"Bapak Imawan, ayahku tercinta yang telah membesarkan, merawatku, dan selalu menyayangiku dan menjagaku"

"Kedua adik lelakiku, M. Fadhil Ali dan M. Hafizh Mulya Ali, yang telah menjadi penyemangatku dan menjadi alasan supaya aku menjadi sosok kakak panutan bagi mereka"

"Sidi dan Nyaik yang kusayangi, (*Alm*). Shabirin Ali dan Rosidana. Ayik dan Mbah yang kusayangi, (*Alm*). A Muluk Hasan dan (*Almh*). Husnah Jaya, yang selalu menyayangiku, mendoakanku dan menjadi motivasiku untuk meraih pendidikan setinggi mungkin"

"Diriku sendiri, terima kasih telah mau berusaha dan bertahan sampai titik ini. Maaf atas hari-hari yang melelahkan dengan fikiran yang ramai dan menguras air mata serta lalai dalam menjaga kesehatan tubuh. Terima kasih telah menjadi kuat disetiap perjalanan waktu."

Serta, Alamamater yang kubanggakan.

"UNIVERSITAS LAMPUNG"

SANWACANA

Puji dan syukur kehadirat Allah SWT. yang telah melimpahkan rahmat dan hidayat-Nya sehingga penulis dapat menyelesaikan penyusunan skripsi / tugas akhir ini dengan judul "Rancang Bangun *Restful* Api *Library Management System* ELIB Unila Berintegrasi Dengan Aplikasi Android Status Peminjaman Buku".

Dalam pelaksaaan dan pembuatan Skripsi / Tugas Akhir ini penulis menerima dukungan baik secara moril maupun materil yang sangat berharga dari berbagai pihak. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada semua pihak yang telah membantu, khususnya kepada:

1. Kedua orangtua tercinta yang menjadi alasan utama penulis dapat bertahan dalam setiap proses yang dijalani selama perkuliahan untuk Bapak Imawan dan Ibu Yuliantina,S.Pd, yang teramat penulis sayangi atas cinta dan kasih sayang, kesabaran yang tulus ikhlas membesarkan, merawat dan yang tidak henti-hentinya selalu memberikan dukungan serta doa-doa yang selalu tercurahkan selama hidup penulis. Semoga Allah senantiasa memuliakan kalian baik didunia maupun diakhirat;
2. Bapak Dr. Eng. Helmy Fitriawan, S.T., M.Sc., selaku Dekan Fakultas Teknik Universitas Lampung;
3. Ibu Herlinawati, S.T., M.T. selaku Ketua Jurusan Teknik Elektro Universitas Lampung;
4. Ibu Yessy Mulyani, S.T., M.T selaku Ketua Program Studi Teknik Informatika Universitas Lampung;
5. Ibu Dr.Eng.Ir. Mardiana, S.T., M.T., I.P.M, selaku pembimbing I dan Bapak

Rio Ariestia Pradipta, S.Kom., M.T.I, selaku pembimbing II, yang dengan tanpa lelah dan bosan telah memberikan bimbingan, semangat, dan mencurahkan waktunya yang demikian banyak dalam menyelesaikan skripsi / tugas akhir ini.

6. Bapak Ir. Meizano Ardhi Muhammad, S.T., M.T., I.P.M, yang telah bersedia menjadi penguji dan telah memberikan saran dan masukkan dalam sidang skripsi / tugas akhir.
7. Seluruh dosen dan staff Program Studi Teknik Informatika Unila yang telah memberikan masukan dan mempermudah proses pembuatan skripsi / tugas akhir ini.
8. Mba Rika Asliana selaku Admin Program Studi Teknik Informatika yang telah banyak membantu penulis dalam segala urusan administrasi selama perkuliahan.
9. Adik-adik penulis untuk M. Fadhil Ali dan M. Hafizh Mulya Ali yang memberikan dukungan dan menjadi orang yang selalu menghibur penulis.
10. Kakek dan Nenek penulis, untuk Alm. Ayik A. Muluk Hasan dan Almh. Mbah Husnah serta Alm. Sidi Shabirin Ali dan Nyaik Rosdiana yang penulis cintai dan sayangi. Terima kasih telah menjadi motivasi dalam hidup penulis, pesan dan nasihat kalian akan selalu penulis ingat sampai kapanpun.
11. Seluruh keluarga besar penulis, terkhusus untuk Paksu Akhmad Amri dan Maksu Inten Cahyani Pratiwi serta adik Hamas dan Hanif terima kasih telah memberikan dukungan selama berada di Bandar Lampung.
12. Sahabat – sahabat penulis grup Kembali Ke Jalannya, untuk Nazmah terima kasih banyak telah menjadi sahabat yang setia untuk saling mendengarkan keluh kesah, saling menyemangati dan mendoakan sejak SMA dan semasa kuliah. Untuk Athip, Tina dan Caca terima kasih selalu memberikan semangat kepada penulis.
13. Sahabat – sahabat grup Threesi untuk Aliya dan Zahra yang penulis sayangi. Terima kasih selalu ada disetiap perjalanan dari sahabat kecil sampai dengan sekarang telah berhasil menyelesaikan pendidikan di perguruan tinggi negeri.
14. Sahabat – sahabat TI A untuk Beltra, Afifah, Elda, Bella, Azzah, Bila, Arista, Nyayu, Adinda, Hesti, Naomi, dan semua rekan-rekan lainnya. Terima kasih

telah membantu dan mendukung penulis selama perkuliahan sampai akhir perkuliahan.

15. Rekan – rekan Asisten Lab Komputer Jurusan Teknik Elektro. Terima kasih telah mendukung dan memberikan pengalaman yang sangat berharga.

Akhir kata, semoga skripsi / tugas akhir ini dapat bermanfaat bagi pembacanya

Bandar Lampung, 9 Juni 2024

Penulis

Asha Imalia Zahra

DAFTAR ISI

Halaman

DAFTAR ISI i
DAFTAR GAMBAR iii
DAFTAR TABEL vii
I. PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 3
1.3 Tujuan Penelitian 3
1.4 Manfaat Penelitian 4
1.5 Batasan Masalah 4
1.6 Sistematika Penulisan 4
II. TINJAUAN PUSTAKA 6
2. 1 Perpustakaan 6
2.2 Layanan Sirkulasi 7
2.3 ELIB 8
2.4 Mobile Application 10
2.5 Android 10
2.6 Kotlin 14
2.7 NodeJS 15
2.8 NPM (Node Package Manager) 16
2.9 ExpressJS 17
2.10 Push Notification 17
2.11 Unified Modeling Language (UML) 18
2.12 Basis Data 20
2.13 Microsoft SQL Server 21
2.14 Postman 21
2.15 *RESTful* API 21

2.16 Metode RAD 24
2.17 Stress Testing 25
2.18 Usability Testing 28
2.19 *Blackbox Testing* 29
2.20 USE Questionnaire 30
2.21 Penelitian Terkait 32
III. METODE PENELITIAN 41
3.1 Waktu dan Tempat Penelitian 41
3.2 Jadwal Penelitian 41
3.3 Alat dan Bahan 42
3.3.1 Alat Penelitian 42
3.3.2 Bahan Penelitian 43
3.4 Tahapan Penelitian 43
3.4.1 *Requirement Planning* 44
3.4.2 *User Design* 46
3.4.3 *Construction* 49
3.4.4 *Cutover* 50
IV. HASIL DAN PEMBAHASAN 52
4.1 Hasil 52
4.1.1 *Requirement Planning* 52
4.1.2 User Design 52
4.1.3 Construction 64
4.1.4 Cutover 129
4.2 Pembahasan 135
4.2.1 Testing *Restful API* Dengan Metode Stress Testing 135
4.2.2 *Blackbox Testing* 142
4.2.3 Penerapan Metode RAD 149
4.2.4 Capaian Penelitian 150
V. KESIMPULAN DAN SARAN 151
5.1 Kesimpulan 151
5.2 Saran 152
DAFTAR PUSTAKA 153

DAFTAR GAMBAR

Halaman

Gambar 2. 1 Perpustakaan Universitas Lampung (sumber: *library.unila.ac.id*) 6
Gambar 2. 2 Tampilan Sistem ELIB 8
Gambar 2. 3 Contoh *Mobile Application* (sumber: *pinterest.com*) 10
Gambar 2. 4 Logo Android (sumber: *developer.android.com*) 11
Gambar 2. 5 Struktur Android (sumber: *developer.android.com*)..... 12
Gambar 2. 6 Visualisasi Komponen-Komponen Umum *Push Notification* [18]..... 18
Gambar 2. 7 Arsitektur REST 23
Gambar 2. 8 Tahapan metode RAD. [27]..... 24
Gambar 3. 1 Tahapan Penelitian 44
Gambar 3. 2 Arsitektur Komunikasi Sistem 46
Gambar 3. 3 Relasi Entitas Tabel Bibliografi ELIB..... 48
Gambar 3. 4 Wireframe Aplikasi 49
Gambar 4. 1 *Use Case* Diagram..... 53
Gambar 4. 2 *Activity Diagram Login* Akun Pemustaka..... 55
Gambar 4. 3 *Activity Diagram* Peminjaman Buku..... 56
Gambar 4. 4 *Activity Diagram* Pengembalian Buku..... 57
Gambar 4. 5 *Activity Diagram* Notifikasi 58
Gambar 4. 6 *Activity Diagram Logout*..... 59
Gambar 4. 7 ERD Diagram Database 60
Gambar 4. 8 (a) Halaman *Splash Screen*, (b) Halaman *Login*, (c), Halaman Beranda, (d) Halaman Status peminjaman, (e) Halaman Denda Buku, (f) Halaman *Reminder*, (g) Halaman Buat *Reminder*..... 64
Gambar 4. 9 Koneksi Server Ms SQL Server 65

Gambar 4. 10 *Dependencies* 66
Gambar 4. 11 Koneksi Database 67
Gambar 4. 12 Tabel Entitas *User* Model 67
Gambar 4. 13 Relasi Antar Tabel Entitas Pada *Circulation* Model (Fitur *Circulation* Status) 68
Gambar 4. 14 Relasi Antar Tabel Entitas Pada *Circulation* Model (Fitur *Circulation* History) 69
Gambar 4. 15 Relasi Antar Tabel Entitas Pada *Circulation* Model (Fitur *Circulation* Akun) 69
Gambar 4. 16 Relasi Antar Tabel Entitas Pada *Book* Model 70
Gambar 4. 17 *Source Code* Fitur *Login* 72
Gambar 4. 18 *Source Code Middleware* 73
Gambar 4. 19 *Source Code* Fitur *Logout* 73
Gambar 4. 21 *Source Code* Fitur *Read User* 74
Gambar 4. 22 *Source Code* Fitur *Circulation History* 76
Gambar 4. 23 *Source Code* Fitur *Circulation Status* 77
Gambar 4. 24 *Source Code* Fitur *Circulation Account* 78
Gambar 4. 25 *Source Code* Fitur *Title Book* 80
Gambar 4. 26 *Source Code* Fitur *Item Book* 81
Gambar 4. 27 *Source Code* Fitur *Author Book* 82
Gambar 4. 28 Hasil *Testing* Skenario Normal Fitur *Login* Pada Postman 85
Gambar 4. 29 (a), (b), Hasil *Testing* Skenario Alternatif 86
Gambar 4. 30 Hasil *Testing* Skenario Normal Fitur *Logout* Pada Postman 88
Gambar 4. 31 Hasil *Testing* Skenario Alternatif Fitur *Logout* Pada Postman .. 88
Gambar 4. 32 Hasil *Testing* Skenario Normal Fitur *Read User* Pada Postman 90
Gambar 4. 33 Hasil *Testing* Skenario Alternatif Fitur *Read User* Pada Postman …….. 90
Gambar 4. 34 Hasil *Testing* Skenario Normal Fitur Riwayat Peminjaman Buku Pada Postman 92
Gambar 4. 35 Hasil *Testing* Skenario Alternatif Fitur Riwayat Peminjaman Buku Pada Postman 93

Gambar 4. 36 Hasil *Testing* Skenario Normal Fitur Status Peminjaman Buku Pada Postman 94
Gambar 4. 37 Hasil *Testing* Skenario Alternatif Status Peminjaman Buku Pada Postman……. 95
Gambar 4. 38 Hasil *Testing* Skenario Normal Fitur Akun Denda Buku Pada Postman……. 96
Gambar 4. 39 Hasil *Testing* Skenario Alternatif Fitur Akun Denda Buku Pada Postman……. 97
Gambar 4. 40 Hasil *Testing* Skenario Normal Fitur *Read* Judul Buku Pada Postman……. 98
Gambar 4. 41 Hasil *Testing* Skenario Alternatif Fitur *Read* Judul Buku Pada Postman……. 99
Gambar 4. 42 Hasil *Testing* Skenario Normal Fitur *Read Item Book* Pada Postman……. 100
Gambar 4. 43 Hasil *Testing* Skenario Alternatif Fitur *Read Item Book* Pada Postman……. 101
Gambar 4. 44 Hasil *Testing* Skenario Normal Fitur *Read Author Book* Pada Postman……. 102
Gambar 4. 45 Hasil *Testing* Skenario Alternatif Fitur *Read Author Book* Pada Postman……. 103
Gambar 4. 46 *Permission* Pada Aplikasi..... 104
Gambar 4. 47 File Build.Gradle..... 106
Gambar 4. 48 Halaman *Splash Screen* 108
Gambar 4. 49 *Source Code Splash Screen* 109
Gambar 4. 50 Halaman Fitur *Login* 110
Gambar 4. 51 *Source Code* Fitur *Login* 112
Gambar 4. 52 Halaman Fitur *Home* 113
Gambar 4. 53 *Source Code* Fitur *Home* 115
Gambar 4. 54 Halaman Fitur Status Peminjaman Buku 116
Gambar 4. 55 *Source Code* Fitur Status Peminjaman Buku 118
Gambar 4. 56 Halaman Status Pengembalian Buku 119
Gambar 4. 57 *Source Code* Status Pengembalian Buku 121

Gambar 4. 58 (a) Halaman Membuat Reminder, (b) Halaman Menambahkan Reminder, (c) Tampilan Notifikasi 122
Gambar 4. 59 *Source Code* Fitur Notifikasi 124
Gambar 4. 60 *Source Code* Halaman *Reminder* 126
Gambar 4. 61 Design Halaman Fitur Logout 128
Gambar 4. 62 *Source Code* Fitur Logout 128
Gambar 4. 63 Nilai rata-rata setiap elemen parameter 131
Gambar 4. 64 Skenario *Stress Test* Pada Apache JMeter 136
Gambar 4. 65 Membuat *Thread Group* 136
Gambar 4. 66 Membuat HTTP *Request* 137
Gambar 4. 67 Membuat Listener 137
Gambar 4. 68 Hasil Pengujian Halaman *Login* (a). *Login* berhasil masuk ke aplikasi (b). *Login* gagal 144
Gambar 4. 69 Hasil Pengujian Halaman *Home* 145
Gambar 4. 70 Hasil Pengujian Halaman Status Peminjaman Buku 146
Gambar 4. 71 Hasil Pengujian Halaman Status Pengembalian Buku 147
Gambar 4. 72 Hasil Pengujian Notifikasi (a) notifikasi muncul ketika terdapat tagihan buku yang belum dikembalikan (b) notifikasi muncul ketika user menambahkan reminder 148
Gambar 4. 73 Hasil Pengujian *Logout* 149

DAFTAR TABEL

Halaman

Tabel 2. 1 Spesifikasi Server ELIB 9
Tabel 2. 2 Spesifikasi Server Apielib 9
Tabel 2. 3 Perbandingan Performa Kotlin dengan Java 15
Tabel 2. 4 Simbol Diagram *Use Case* 19
Tabel 2. 5 Simbol *Activity Diagram* 20
Tabel 2. 6 Operasi HTTP 22
Tabel 2. 7 Kriteria Pengukuran *USE Questionnaire.* [34] 30
Tabel 2. 8 Nilai Skala Likert 31
Tabel 2. 9 Kategori Kelayakan [35] 31
Tabel 2. 10 Penelitian Terkait 37
Tabel 3. 1 Jadwal Kegiatan Penelitian 41
Tabel 3. 2 Alat Penelitian 42
Tabel 3. 3 Bahan Penelitian 43
Tabel 3. 4 Definisi *Use Case* 54
Tabel 4. 1 Daftar *Endpoint* 83
Tabel 4. 2 Test Skenario Fitur *Login* 84
Tabel 4. 3 Test Skenario Fitur *Logout* 87
Tabel 4. 4 Test Skenario Fitur *Read User* 89
Tabel 4. 5 Test Skenario Fitur Riwayat Peminjaman Buku 91
Tabel 4. 6 Test Skenario Fitur Status Peminjaman Buku 93
Tabel 4. 7 Test Case Fitur Akun Denda Buku 95
Tabel 4. 8 *Test Case* Fitur *Read* Judul Buku 97
Tabel 4. 9 *Test Case* Fitur *Read Item* Buku 99
Tabel 4. 10 *Test Case* Fitur *Read Author* Buku 101

Tabel 4. 11 *USE Questionnaire* yang digunakan..... 129
Tabel 4. 12 Data Pengujian *USE Questionnaire* 131
Tabel 4. 13 Hasil Pengujian 90 Sample Pada Apache JMeter..... 138
Tabel 4. 14 Hasil Pengujian 900 Sampel Pada Apache JMeter..... 138
Tabel 4. 15 Hasil Pengujian 9000 Sampel Pada Apache JMeter..... 139
Tabel 4. 16 Hasil Pengujian 90000 Sampel Pada Apache JMeter..... 140
Tabel 4. 17 Hasil Pengujian 135000 Sampel Pada Apache JMeter..... 140
Tabel 4. 18 Rangkuman Hasil Pengujian *Stress Test* 141
Tabel 4. 19 Tahapan Pengujian Aplikasi 142
Tabel 4. 20 Pengujian *Login* Aplikasi 143
Tabel 4. 21 Pengujian Halaman *Home* 144
Tabel 4. 22 Pengujian Halaman Status Peminjaman Buku 145
Tabel 4. 23 Pengujian Halaman Status Pengembalian Buku 146
Tabel 4. 24 Pengujian Notifikasi 147
Tabel 4. 25 Pengujian *Logout*..... 148

I. PENDAHULUAN

1.1 Latar Belakang

Keberadaan teknologi informasi dan komunikasi membawa banyak perubahan kehidupan saat ini dengan ditandai oleh era digital dimana pertukaran informasi berlangsung dengan cepat tanpa terhambat batas ruang ataupun waktu. Teknologi informasi membawa kemudahan bagi manusia dan banyak digunakan secara luas untuk meningkatkan kinerja, performa usaha, serta kualitas layanan. Salah satu kegiatan yang memanfaatkan kemajuan teknologi informasi adalah layanan perpustakaan.[1]

Perpustakaan adalah tempat atau institusi yang memfasilitasi akses pada berbagai kegiatan penghimpunan, pengelolaan, dan pelayanan informasi untuk memenuhi kebutuhan intelektualitas penggunanya. Perpustakaan menghimpun berbagai jenis bahan bacaan seperti buku, jurnal, artikel, dan sumber daya lainnya dalam mendukung pendidikan, penelitian, serta akses informasi[2]. Dengan perkembangan teknologi yang pesat, layanan perpustakaan menghadapi transformasi yang disesuaikan dengan kebutuhan penggunanya dalam mengadopsi teknologi guna meningkatkan efisiensi layanan dan aksesibilitas bagi pemustaka.

Universitas Lampung memiliki perpustakaan yang mendukung kegiatan akademik, penelitian, pendidikan, layanan referensi, dan sumber informasi bagi civitas akademika Universitas Lampung. Perpustakaan Universitas Lampung dikelola oleh UPT Perpustakaan Universitas Lampung mempunyai sistem manajemen perpustakaan pada sistem ELIB Universitas Lampung, termasuk layanan sirkulasi atau peminjaman buku menggunakan sistem tersebut. Meskipun sistem ELIB telah

memadai dalam hal manajemen data pada perpustakaan, belum terdapat aplikasi berbasis *mobile* yang dapat memberikan informasi kepada pemustaka tentang layanan sirkulasi atau peminjaman.

Sistem ELIB dirancang khusus untuk penggunaan pustakawan yaitu petugas perpustakaan dengan layanan informasi bersifat internal. Dalam prosedur peminjaman dan pengembalian buku pada perpustakaan Universitas Lampung hak akses pemustaka terhadap informasi peminjaman dan pengembalian buku hanya pada transaksi yang dilakukan pada mesin KIOSK. Berdasarkan hasil wawancara yang dilakukan kepada petugas dan staff ahli perpustakaan keterlambatan pengembalian buku masih menjadi masalah dalam layanan perpustakaan. Hal ini diperlukan untuk mengembangkan sistem *reminder* kepada pemustaka. *Push notification* yang merupakan teknologi pada aplikasi yang memungkinkan dalam pengiriman informasi berupa notifikasi yang bersumber dari penyedia informasi (*server*) ke sebuah perangkat (*client*) secara otomatis tanpa permintaan khusus oleh *client*[3].

Perancangan *Application Programming Interface (API)* memungkinkan sebuah *back-end* dapat dimanfaatkan lebih luas karena *logic* sistem dipisahkan dengan tampilan antarmuka pengguna. API merupakan antarmuka bagi program perangkat lunak ke program lain dengan banyak model bisnis dalam perancangannya. Arsitektur REST dalam pengembangan API menggunakan protokol HTTP untuk berkomunikasi antar data dan memiliki struktur kerja yang mudah dipahami oleh berbagai macam bahasa pemrograman dan platform. API yang mengadopsi gaya arsitektur REST disebut sebagai *RESTful* API[4]. Pengembangan *RESTful* API dipilih untuk komunikasi data antara database sistem ELIB dengan aplikasi android yang akan dirancang.

Menurut DataReportal bersama Hootsuite (*We are Social*) pada januari 2023 terhitung pengguna perangkat *mobile* yang terhubung sebesar 353,8 juta atau sebesar 128% dari total populasi Indonesia yang pada awal tahun 2023 sebesar 276,4 juta jiwa. Kepemilikan perangkat *smartphone* di Indonesia sebesar 99,4%.

Jumlah tersebut lebih besar dibandingkan kepemilikan perangkat lain seperti laptop/komputer, tablet, dan *smartwatch.* Sementara itu, dari penggunaan *smartphone,* sistem operasi yang digunakan yaitu android lebih besar dibandingkan pengguna iOS dari *Apple*[5]. Platform android beradaptasi terhadap versi terbaru dari java dalam pengembangannya karena java adalah bahasa lama yang memiliki banyak tantangan, salah satunya adalah design. Kotlin merupakan bahasa yang aman, ekspresif, ringkas, yang memiliki interoperabilitas besar dengan java dan JavaScript[6]. Pengembangan platform Android dengan bahasa pemrograman kotlin dipilih sebagai basis aplikasi agar dapat diakses dengan mudah oleh pemustaka.

Penelitian ini mengambil gagasan dari penemuan masalah pada layanan sirkulasi atau peminjaman buku pada Perpustakaan Universitas Lampung karena belum terdapat aplikasi yang dapat digunakan oleh pemustaka. Penelitian ini bertujuan untuk merancang dan membangun *RESTful* API untuk mengembangkan aplikasi status peminjaman buku bagi pemustaka berbasis android yang terintegrasi oleh database sistem ELIB perpustakaan Universitas Lampung. Penelitian ini diharapkan dapat memberikan kontribusi terhadap kualitas layanan dengan aksesbilitas informasi yang dapat dijangkau oleh pemustaka tanpa mengubah struktur proses peminjaman yang sudah ada.

1.2 Rumusan Masalah

Berdasarkan latar belakang, rumusan masalah dalam penelitian ini adalah bagaimana mengembangkan *Restful* API pada sistem ELIB Perpustakaan Universitas Lampung dan aplikasi *mobile* status peminjaman buku pemustaka berbasis android.

1.3 Tujuan Penelitian

Tujuan penelitian ini adalah :

1. Mengembangkan *Restful* API pada sistem ELIB Perpustakaan Universitas Lampung untuk pengembangan aplikasi android.

2. Mengembangkan aplikasi *mobile* status peminjaman buku untuk pemustaka berbasis android menggunakan Kotlin.

1.4 Manfaat Penelitian

Manfaat dari penelitian ini yaitu :

1. Tersedianya *Restful* API untuk mempermudah komunikasi integrasi data antara sistem ELIB Unila dengan aplikasi android status peminjam buku agar dapat mengakses sumber daya dari *backend* server serta database tanpa perlu memiliki akses langsung ke sumber daya.
2. Tersedianya aplikasi status peminjaman buku berbasis android yang ditujukan kepada pemustaka untuk meningkatkan akses informasi layanan sirkulasi terkait peminjaman buku, riwayat peminjaman buku, denda buku, serta *reminder* untuk waktu pengembalian buku.

1.5 Batasan Masalah

Adapun batasan masalah dalam penelitian ini yaitu:

a. Perancangan *RESTful* API pada LMS ELIB Perpustakaan Universitas Lampung difokuskan untuk pengembangan aplikasi terkait sistem sirkulasi perpustakaan.
b. Sistem aplikasi *mobile* yang dikembangkan berbasis android menampilkan informasi terkait status peminjaman buku yang dilakukan pemustaka dan tidak dirancang untuk melakukan peminjaman buku di aplikasi.

1.6 Sistematika Penulisan

Adapun Sistematika penulisan dalam pembahasan laporan penelitian ini yaitu:

BAB I : PENDAHULUAN

Pada bab ini memuat secara umum tentang latar belakang penelitian, rumusan masalah, tujuan penelitian, manfaat penelitian, batasan penelitian, dan sistematika penulisan laporan.

BAB II : TINJAUAN PUSTAKA

Bab II membahas tentang tinjauan pustaka dan dasar teori. Tinjauan pustaka berisi hasil kajian dari berbagai pustaka yang mempunyai relevansi dengan penelitian mengenai *Rancang Bangun Restful API Library Manajement System (LMS) ELIB Unila Berintegrasi Dengan Aplikasi Android Status Peminjaman Buku*. Sedangkan dasar teori membahas definisi dari masing-masing komponen yang digunakan pada penelitian ini. Pada bab ini memuat dasar teori tentang pengertian Perpustakaan, Layanan Sirkulasi, ELIB, *Mobile Aplication,* Android, Kotlin, NodeJS, *Node Package Manajer* (NPM), ExpressJS, *Push Notification, UML,* Basis Data, *Microsoft SQL Server,* Postman, *Restful API,* Metode RAD, *Unit Testing*, *Usability Testing,* dan penelitian terkait.

BAB III : METODE PENELITIAN

Bab III yang merupakan metode penelitian menjelaskan waktu dan tempat penelitian, jadwal penelitian, alat dan bahan penelitian, tahapan penelitian dengan metode *Rapid Application Development* (RAD). Dimana penelitian dimulai dari tahap *Requirement Planning, User Design, Construction,* dan *Cutover.*

BAB IV : HASIL DAN PEMBAHASAN

Bab IV memaparkan hasil dan pembahasan yang didapatkan sesuai dengan tahap penelitian yaitu tahap pada pembuatan sistem dan pengujian sistem.

BAB V : KESIMPULAN DAN SARAN

Bab V berisi kesimpulan dari hasil penelitian dan saran dari hasil penelitian sebagai masukan untuk pengembangan penelitan berikutnya

DAFTAR PUSTAKA

LAMPIRAN

II. TINJAUAN PUSTAKA

2. 1 Perpustakaan

Perpustakaan adalah sebuah institusi yang mendukung masyarakat dalam bidang ilmu pengetahuan dengan menyediakan berbagai bahan pustaka dalam berbagai format untuk keperluan pendidikan. Umumnya Perpustakaan adalah tempat yang mempertemukan pembaca dengan bahan pustaka yang mereka cari. Perpustakaan berperan sebagai penyedia informasi yang sangat penting untuk pendidikan karena berfungsi sebagai media atau jembatan yang menghubungkan pemustaka dengan sumber informasi dan pengetahuan yang tersedia dalam koleksi perpustakaan.

Gambar 2. 1 Perpustakaan Universitas Lampung (sumber: *library.unila.ac.id*)

Terdapat dua tipe layanan perpustakaan yakni layanan akses terbuka (*open access*) dan layanan akses tertutup (*close access*).

1. Sistem Layanan Terbuka (*open access*) memungkinkan pengguna berinteraksi langsung dengan koleksi perpustakaan; contohnya, pengguna

dapat langsung mengakses ruang koleksi dan mengambil buku atau bahan bacaan yang mereka perlukan.

2. Sistem Layanan Tertutup (*close access*), pemustaka tidak dapat menelusuri atau mengambil buku yang mereka butuhkan sendiri; sebaliknya, mereka harus meminta petugas di bagian peminjaman atau sirkulasi perpustakaan untuk mengakses koleksi.[7]

Sedangkan jenis layanan yang terdapat pada perpustakaan diantaranya :

1. Layanan Orientasi Perpustakaan yang menyediakan informasi mengenai koleksi buku serta pencarian informasi umum.
2. Layanan Pendaftaran Kartu Anggota Perpustakaan yaitu layananan administrasi data anggota untuk mempermudah proses layanan bagi pengguna perpustakaan.
3. Layanan Sirkulasi atau Layanan Peminjaman dan Pengembalian Bahan Koleksi Perpustakaan kepada Pemustaka
4. Layanan Referensi merupakan layanan kumpulan referensi atau koleksi bahan rujukan seperti Artikel dan Jurnal.
5. Layanan Penelusuran Pustaka yang dibagi menjadi dua yaitu Layanan Internet dan Layanan CD-ROM
6. Layanan *Reserve Book* atau buku tandon yang merupakan layanan pengumpulan buku atau bahan pustaka dalam satu tempat, dan koleksi tersebut hanya dapat dibaca ditempat,
7. Layanan Koleksi Karya Ilmiah (KKI) adalah layanan dengan koleksi bahan pustaka khusus yang dibuat oleh mahasiswa, dosen dan staff universitas seperti Karya Ilmiah, Skripsi, Tesis, Disertasi, Tugas Akhir Mahasiswa Diploma, dan Laporan Penelitian.

2.2 Layanan Sirkulasi

Layanan sirkulasi perpustakaan adalah layanan operasional perpustakaan yang terkait dengan proses peminjaman dan pengembalian buku. Layanan sirkulasi memungkinkan anggota perpustakaan untuk meminjam buku dan proses pencatatan peminjaman dan pengembalian buku diatur dalam layanan ini. Selain itu, beberapa

fungsi layanan sirkulasi perpustakaan kecuali peminjaman dan pengembalian buku, meliputi : pendaftaran anggota baru, manajemen koleksi buku, manajemen denda, serta menangani permintaan surat bebas pustaka bagi pengguna perpustakaan.[8]

2.3 ELIB

ELIB adalah sistem yang dikembangkan dari perusahaan Bibliotheca yang mendukung Protokol SIP2 yang memungkinkan penggunaan perangkat otomasi perpustakaan. ELIB adalah sistem otomatisasi perpustakaan yang mendukung aktivitas umum pada perpustakaan. ELIB mengotomatiskan transaksi melalui pengumpulan pemindaian dan *Self Service Kiosk.*[9]

Gambar 2. 2 Tampilan Sistem ELIB

Dalam penelitian ini database pada Sistem ELIB yang digunakan sebagai bahan untuk bagian *backend* dalam pengambilan data yang akan terintegrasi dengan aplikasi. Adapun spesifikasi server ELIB dan APIELIB yang digunakan dalam penelitian adalah sebagai berikut :

Tabel 2. 1 Spesifikasi Server ELIB

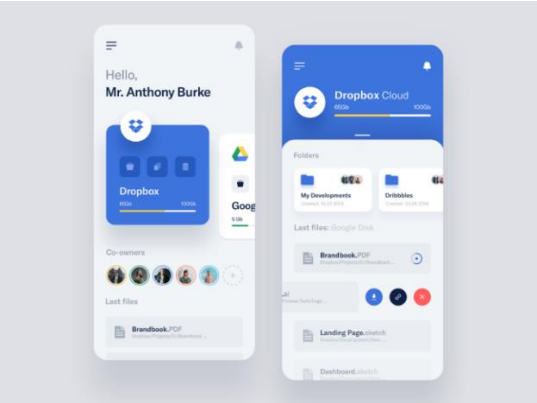
Processor Type	Spesifikasi
Nama Server	ELIB-VER
Sistem Operasi	Windows Server 2012 Standard 64-bit (6.2, build 9200)
System Manufacture	Intel Corporation
System Model	S1200RP_SE
BIOS	BIOS Date: 08/20/09 10:33:39 Ver: 08.00.10
Processor	Intel(R) Xeon(R) CPU E3-1220 v3 @.3.10GHz (4 CPUs), ~3.1GHz
Memory	16384MB RAM
Page File	6609MB *used,* 12015MB *available*
DirectX Version	DirectX 11

Tabel 2. 2 Spesifikasi Server Apielib

Processor Type	Spesifikasi
Host Name	Apielib
IP addresses	1. 10.10.100.48 2. fe80::20c:29ff:fe75:2433
VMware Tools	VMware Tools is not managed by vSphere
Storage	1 disk
CPU Cores	4
Memory	2 GB
Hard disk 1	20GB
USB controller	USB 2.0
CD/DVD drive 1	ISO [datastore1] master ubuntu/ubuntu-20.04.3-live-server-amd64.iso

2.4 Mobile Application

Mobile application (aplikasi seluler) merupakan aplikasi yang berjalan dan dirancang khusus dan dapat diunduh pada platform *mobile* dengan sifat layanan bergerak yang dijalankan pada piranti keras. *Mobile* mempunyai arti dengan mudah berpindah dari satu tempat ke lain tempat tanpa terputus komunikasi, mudah digunakan, dan dapat diakses kapan saja dan dimana saja karena berjalan pada perangkat genggam kecil. *Mobile application* yang berjalan pada platform *mobile* yang umum ditemukan diantaranya *iOS*, *android*, atau *windows mobile*.



Gambar 2. 3 Contoh *Mobile Application* (sumber: *pinterest.com*)

Terdapat perbedaan dasar untuk aplikasi dengan aplikasi *mobile* bahwa umumnya aplikasi berjalan langsung di sistem operasi sedangkan *mobile application* merupakan aplikasi yang dikembangkan di dalam sebuah *framework*. Secara ringkas *mobile application* diartikan sebagai sebuah bahasa pemrograman yang merepresentasikan sesuatu yang harus dilakukan dalam menyelesaikan tugasnya serta terdiri dari perangkat lunak atau sekumpulan program dengan tugas tertentu untuk pengguna.[10][11]

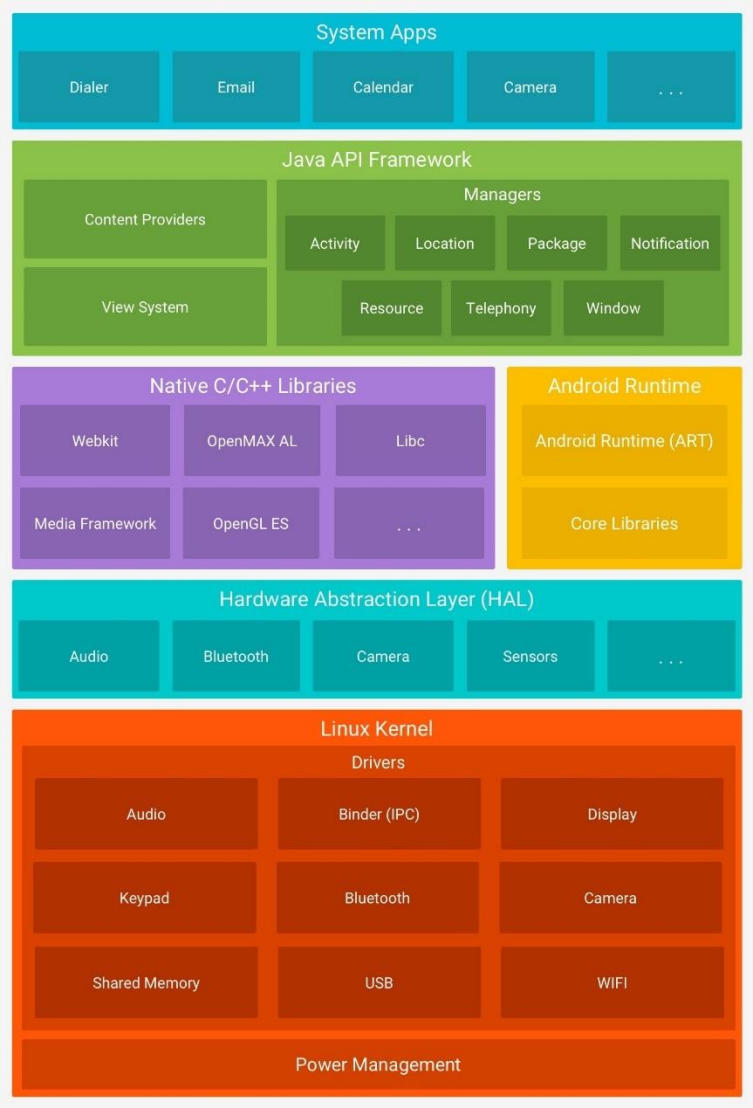
2.5 Android

Android merupakan platform sistem operasi seluler modern yang dibangun dengan kernel linux. Android awalnya dikembangkan pada tahun 2003 oleh Perusahaan Android Inc. Kemudian pada tahun 2005, Android Inc diakusisi oleh Google dan berkembang di tahun 2007 dengan dibentuk Open Handset Alliance (OHA) yang

merupakan akusisi dari 34 perusahaan perangkat keras, lunak, dan telekomunikasi termasuk diantaranya Google, HTC, Intel, LG, Motorola, Nvida, Qualcomm, Samsung Electronics, Sprint Nextel, dan T-Mobile. Android dan Open Handset Alliance mengembangkan standar sumber terbuka pada perangkat seluler. Pada akhir tahun 2008, Google bersama OHA merilis Android 1.0 versi pertama Android yang diluncurkan ke publik dengan kode nama Alpha.[12]

Gambar 2. 4 Logo Android (sumber: *developer.android.com*)

Android sebagai pengembangan aplikasi bersifat *open source* serta ekosistem dari pengguna yang besar dan beragam. Dalam pengembangannya, android menyediakan dokumentasi yang sangat baik dengan komunitas pengembang yang masif serta minim biaya yang diperlukan dalam pengembangan distribusi. Pengembangan aplikasi di platform Android umumnya menggunakan bahasa pemrograman Java, Kotlin, dan Dart yang dieksekusi melalui *Android Run Time* (ART). *Android Software Development Kit* (SDK) mencakup keperluan dalam mengembangkan, menguji, dan men-*debug* aplikasi android. Alat pengembangan dari Android SDK mencakup IDE Android Studio yang dapat mengkompilasi dan men-*debug* aplikasi untuk mengubah sumber kode android menjadi aplikasi yang dapat dieksekusi.



Gambar 2. 5 Struktur Android (sumber: *developer.android.com*)

Struktur operasi Android terdiri enam struktur utama, sebagai berikut:

1. *Linux Kernel*

   Fondasi dari platform Android adalah linux kernel. Pada lapisan linux kernel tidak berinteraksi secara langsung dengan pengguna ataupun *developer*. Lapisan ini diibaratkan sebagai jantung pada seluruh sistem android dikarenakan linux kernel menyediakan berbagai fungsi untuk sistem Android. Fungsi-fungsi tersebut diantaranya:

   a. Abstraksi Hardware
   b. Program Manajemen Memory

c. Pengaturan Sekuritas
d. Manajemen Energi Software (Baterai)
e. Driver
f. *Network Stack*

2. *Hardware Abstraction Layer* (HAL)

   Hardware Abstraction Layer (HAL) terdiri dari beberapa modul pustaka, masing-masing memiliki antarmuka untuk komponen perangkat keras tertentu, seperti modul *bluetooth* atau kamera. Saat API kerangka kerja melakukan panggilan untuk mengakses komponen perangkat keras. HAL memaparkan kemampuan perangkat keras ke kerangka kerja API Java yang lebih tinggi.

3. *Android Runtime*

   Android Runtime adalah kumpulan pustaka Java yang ditunjukan khusus untuk Android. Dalam arsitektur Android, *Android Runtime* ditempatkan pada level yang sama di lapisan pustaka. *Android Runtime* digunakan untuk menjalankan beberapa mesin virtual pada perangkat dengan kapasitas memori rendah dengan cara mengeksekusi file.

4. *Library*

   Library berisi serangkaian instruksi yang mengarahkan perangkat Android dalam memproses berbagai jenis data. Contohnya, perekam dari berbagai format video dam audio yang dikelola oleh *Media Framework Library*.

5. API *Framework*

   API *Framework* berinteraksi langsung dengan aplikasi. Pada lapisan ini sebagai fondasi yang digunakan untuk pengembangan pada aplikasi. API menjadi dasar utama untuk aplikasi Android dengan menyederhanakan penggunaan komponen dan layanan sistem modular, yang meliputi :

   a. *Activity Manager* digunakan untuk mengatur semua fase dari siklus hidup aplikasi dan *Activity Stack*.

b. *Content Providers* akan memperbolehkan aplikasi dalam mempublikasi dan berbagi informasi dengan aplikasi lain.
c. *Resource Manager* akan memberikan akses kepada *resources* yang bukan kode seperti strings dan layout *User Interface*.
d. *View System* berfungsi untuk membuat antarmuka pengguna aplikasi.
e. *Notifications Manager* akan membuat aplikasi menampilkan pemberitahuan kepada pengguna.

6. *System Applications*

   Aplikasi terletak di lapisan teratas dalam arsitektur Android. Pengguna Android akan langsung berinteraksi dengan aplikasi ini. Biasanya aplikasi menjalankan tugas dasar seperti menelepon, mengakses website, dll. *System Appications* berguna sebagai aplikasi untuk pengguna dan diakses oleh kebanyakan *Developer*, Programmer, dan lainnya.

2.6 Kotlin

Kotlin merupakan bahasa pemrograman yang diperkenalkan oleh JetBrains dan berjalan di atas *Java Virtual Machine*. Kotlin merupakan bahasa pemrograman pengembangan dari bahasa Java. Kotlin direkomendasikan untuk pengembangan di Android dan mendapatkan dukungan dan rekomendasi resmi dari Google, pengembang resmi sistem operasi Android. Kotlin menggabungkan berbagai fitur modern yang memberikan keunggulan khusus dibandingkan bahasa JVM lainnya. Kelebihan kotlin diantaranya :

a. Kotlin *flexible*

   Kode memungkinkan penulisan kode dengan *Object Oriented Programming*, fungsional, imperative, dan reaktif secara terpisah maupun digabungkan. Selain itu, kotlin mendukung fitur modern diantaranya inferensi tipe sehingga *compiler* tidak perlu khawatir terhadap pengetikan yang ketat. Kotlin dapat digunakan di berbagai platform, mulai dari aplikasi perangkat seluler, perangkat IoT, hingga browser
b. Kotlin ekspresif dan ringkas

Penambahan fitur seperti inferensi tipe dan nilai parameter default memungkinkan developer menggunakan jumlah kode yang lebih sedikit. Di sisi lain, fitur seperti kelas data dan kelas objek memungkinkan *developer* untuk mengkomunikasikan kode.

c. Kotlin bersifat *powerful*

Dengan fitur seperti fungsi tingkat tinggi, coroutine, dan pustaka standar yang komprehensif, Kotlin adalah bahasa pemrograman yang lengkap dan kuat yang siap memenuhi tuntutan perangkat lunak modern dan memberi pengembang semua alat yang mereka perlukan untuk mengirimkan perangkat lunak berkualitas tinggi. Meskipun bahasa ini relatif baru, Kotlin memiliki semua fitur yang diperlukan untuk memenuhi tuntutan perangkat lunak modern.[13]

Tabel 2. 3 Perbandingan Performa Kotlin dengan Java

Perbandingan Performa	**Kotlin**	**Java**
CPU Usage (%)	10.19	10.84
Memory Usage (Byte)	25.72	63.29
Execution Time (ms)	1621.00	2268.58

Perbandingan performa aplikasi dilakukan pada aplikasi yang sama dan dibangun dengan bahasa Kotlin dan Java. Pengukuran performa didapatkan hasil dalam penggunaan CPU pada perangkat menghasilkan selisih 0.65% antara keduanya, selanjutnya adalah perbandingan penggunaan memory pada perangkat dengan hasil Kotlin lebih kecil dalam penggunaan memory sebesar 2 kali lipat dibanding Java, dan terakhir adalah waktu eksekusi yang diperlukan antara Kotlin dan Java mendaptkan hasil Kotlin lebih cepat dalam mengeksekusi program.[14].

2.7 NodeJS

NodeJS merupakan perangkat lunak untuk pengembangan aplikasi web. NodeJS diperkenalkan pada tahun 2009 oleh Ryan Dahl. NodeJS dibangun pada lingkungan *runtime* JavaScript lintas platform berdasarkan *engine* JavaScript V8 Chrome dengan beberapa modul bawaan terintegrasi seperti modul HTTP, modul sistem file,

modul keamanan, dan modul penting lainnya. NodeJS memungkinkan pengembang menulis perangkat lunak dari sisi server menggunakan bahasa pemrograman JavaScript pada sisi server (*server-side*).

Secara umum, aplikasi server menggunakan bahasa PHP, Java, Ruby atau Python. Namun, JavaScript lebih banyak digunakan untuk mengembangkan aplikasi sisi browser untuk lebih interaktif. Keunggulan NodeJS salah satunya adalah teknik *non-blocking* dengan sistem kerja secara *asynchronous* pada sistem untuk mengeksekusi operasi secara paralel tanpa menunggu operasi yang belum terselesaikan sehingga memungkinkan permintaan diproses secara paralel.[15]

2.8 NPM (Node Package Manager)

NPM atau *Node Package Manager* adalah proyek sumber terbuka ini bertujuan membantu pengembang JavaScript dengan berbagi modul kode. Kumpulan kode sumber terbuka ini disebut dengan NPM *registry,* yang tidak hanya digunakan sebagai pengembangan web, tetapi juga untuk pengembangan aplikasi seluler, robot, router, dan juga kebutuhan lainnya dalam komunitas pengembang JavaScript NPM terdiri dari 3 komponen yang berbeda, meliputi :

1. Situs web : Situs web digunakan untuk menemukan *package,*menyiapkan profil, dan mengelola aspek lain dari pengalaman NPM oleh user. contohnya yaitu *user* dapat menyiapkan organisasi untuk mengelola akses ke *package* publik atau privat.
2. *Command Line Interface* (CLI): CLI berjalan dari terminal dan cara sebagian besar pengembang berinteraksi dengan NPM.
3. Registri: Registri adalah database publik yang besar dari perangkat lunak JavaScript dan informasi meta yang mengelilinginya.

Saat ini, NPM merupakan perangkat lunak terbesar di dunia. Hal inilah yang menjadikan NodeJS popular dan banyak digunakan perusahaan ternama seperti Slack, Microsoft, Netflix, Adobe, dan banyak perusahaan lainnya. Dengan kehadiran NPM, proses pengembangan perangkat lunak menjadi lebih cepat karena *developer* tidak perlu membuat kode dari awal melainkan hanya perlu menginstall paket yang dibutuhkan dan mengubah kode pada paket tersebut jika diperlukan.[16]

2.9 ExpressJS

ExpressJS merupakan *framework* aplikasi dengan basis web yang menggunakan *core* pemrograman NodeJS dengan menggunakan modul HTTP dan *Connect*. ExpressJS merupakan *framework* aplikasi web yang bersifat fleksible dan minimalis dengan serangkaian fitur tangguh untuk pengembangan aplikasi web dan *mobile.* Sesuai dengan tagline-nya "*Fast, unopinionated, minimalist web framework for NodeJS*". ExpressJS menyediakan 4 mekanisme yaitu:

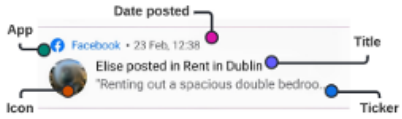
1. Penulisan penangan (*handler*) untuk permintaan menggunakan berbagai kata kerja HTTP di jalur URL atau rute (*routes*) yang berbeda
2. Terintegrasi dengan mesin rendering '*view*' atau bagian *frontend* sehingga dapat menghasilkan respons dengan memasukan data ke dalam template tampilan atau antarmuka pengguna.
3. Mengatur pengaturan umum aplikasi web seperti port yang digunakan untuk menghubungkan server dengan lokasi dalam template tampilan atau antarmuka pengguna yang digunakan untuk merender respons
4. Menambahkan pemrosesan permintaan tambahan "*middleware*" di setiap titik dalam rute (*route*) permintaan (*request*)

ExpressJS menyediakan beberapa library yang dapat membantu pengembang dalam menangani *cookie, session, user login, URL parameters, POST data, security header* dan lain sebagainya. Selain itu, penggunaan ExpressJS sebagai *framework* dalam pengembangan REST API adalah popularitas dari ExpressJS. Popularitas dari sebuah *framework* merupakan indikator apakah kerangka kerja tersebut akan terus berlanjut dan didukung.[17]

2.10 Push Notification

Push notification yang merupakan teknologi pada aplikasi yang memungkinkan dalam pengiriman informasi dalam bentuk notifikasi secara otomatis dari server ke perangkat (*client*) tanpa permintaan khusus oleh *client*. Platform *push notification* memungkinkan pengguna menerima pesan penting dari server ke *client* tanpa perlu klien selalu terhubung ke server[3]. *Push notification* tidak memerlukan perangkat

khusus dalam komputasi *mobile* sehingga pesan dapat diterima. Hal ini memungkinkan *smartphone* untuk menerima dan menampilkan notifikasi walaupun aplikasi sedang tidak berjalan dan layar perangkat terkunci.



Gambar 2. 6 Visualisasi Komponen-Komponen Umum *Push Notification* [18]

Gambar 2.6 diatas merupakan ilustrasi visual dari *push notification*. Komponen-komponen *push notification* memiliki perbedaan tergantung dari platformnya. Umumnya *push notification* terdiri dari judul text, teks penjelasan (panjang teks hingga 240 karakter pada perangkat android), dan ikon. Dalam penerapan *push notification* terdapat metode dan layanan berbeda sesuai dengan perangkat. Perangkat android menggunakan layanan *Google Cloud Messaging* (GMC) sedangkan Pengembangan Apple menggunakan *Apple Push Notification Service's Developers* API untuk mengirimkan *push notification* ke perangkat iOS.[18][19]

2.11 Unified Modeling Language (UML)

Unified Modeling Language (UML) adalah bahasa standar untuk merancang perangkat lunak. UML dapat digunakan untuk memvisualisasikan, menspesifikasikan, membangun, dan mendokumentasikan artefak dari sistem yang kompleks. UML adalah bahasa yang sangat ekspresif, mencakup semua sudut pandang yang diperlukan untuk mengembangkan dan menerapkan sistem-sistem dengan tersebut. UML mencakup sembilan jenis diagram diantaranya : *Class Diagram, Object Diagram, Use Case Diagram, Sequence Diagram, Collaboration Diagram, Statechart Diagram, Activity Diagram, Component Diagram, Deployment Diagram*.[20]

a. *Use Case Diagram*

Sebuah diagram *use case* menunjukkan sekelompok *use case* (kumpulan aktivitas yang sistem harus lakukan) dan aktor-aktor (karakteristik khusus dari kelas yang berinteraksi dengan sistem) serta hubungan antara sistem dan aktor. Diagram *use case* mengatasi pandangan *use case* statis dari

sebuah sistem. Diagram ini memiliki signifikan besar dalam mengatur dan memodelkan perilaku suatu sistem. Berikut merupakan simbol-simbol dalam diagram *Use Case* :

Tabel 2. 4 Simbol Diagram *Use Case*

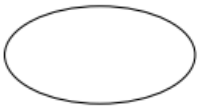
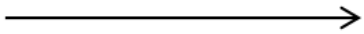
Simbol	Deskripsi
	Aktor, merupakan entitas atau proses yang berinteraksi dengan sistem dan berada diluar konteks lingkungan sistem informasi.
	*Use case*, adalah fungsionalitas dari sistem sebagai unit yang saling bertukar pesan.
	Asosiasi, merupakan simbol komunikasi antar *actor* dan *use case*.
 << extends >>	*Extendes*, merupakan simbol relasi *use case* tambahan ke suatu *use case* dimana *use case* dapat berdiri sendiri.
 << include >>	*Include*, adalah simbol relasi *use case* yang menunjukkan bahwa suatu *use case* memerlukan fungsi tertentu sebagai prasyarat untuk menjalankan *use case* tersebut
	Generalisasi, adalah hubungan antara dua buah *use case* mewakili fungsionalitas yang lebih umum daripada yang lain.

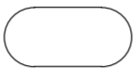
Diagram *use case* mengilustrasikan secara visual bagaimana aktor-aktor berhubungan dengan sistem dan apa yang sistem lakukan sebagai *response*

terhadap aksi-aksi dari aktor-aktor tersebut. Ini berkontribusi pada pemahaman yang lebih baik mengenai perilaku serta fungsi dari sistem yang sedang dikembangkan.

b. *Activity Diagram*

Activity diagram, sebuah varian dari *statechart* diagram mengilustrasikan perpindahan aktivitas dari satu kegiatan ke kegiatan lain dalam suatu sistem. Fungsi *Activity Diagram* terletak pada representasi dinamis sistem, terutama dalam modelasi fungsi sistem dan penekanan alur control antara objek objek. Diagram ini membantu dalam memahami secara detail bagaimana suatu sistem beroperasi dan bagaimana aktivitas-aktivitasnya dijalankan. Adapun simbol-simbol dalam *Activity Diagram* yaitu :

Tabel 2. 5 Simbol *Activity Diagram*

Simbol	Deskripsi
	*Start*, yaitu simbol yang menyatakan awal suatu proses.
	*Stop*, yaitu simbol yang menyatakan akhir suatu proses.
	*Decision*, yaitu simbol yang menyatakan kondisi dari suatu proses
	*Action*, yaitu simbol yang menyatakan aksi yang dilakukan dalam suatu arsitektur sistem

2.12 Basis Data

Basis data merupakan himpunan data yang terkait yang disimpan secara bersama-sama dalam media tertentu yang diorganisasikan secara khusus serta dapat dikelola dengan perangkat lunak untuk tujuan tertentu. Basis data juga dapat didefinisikan sebagai kumpulan data yang tersusun dalam bentuk beberapa tabel yang saling terkait dan mandiri. Operasi dasar basis data meliputi *create database, drop database, create table, drop tabel, insert, update, dan delete*. Basis data merupakan

komponen penting dari sistem informasi karena membantu memberikan informasi dan mengurangi duplikasi data.[21]

2.13 Microsoft SQL Server

SQL server merupakan *Relational Database Management System* (RDBMS) dengan arsitektur *client server*. SQL server menggunakan bahasa kueri utamanya yaitu Transact-SQL adalah implementasi dari standar ANSI/ISO SQL yang digunakan oleh Microsoft dan Sybase. Komunikasi antara Microsoft SQL Server dan Sybase/ASE dapat dilakukan melalui jaringan dengan menggunakan protokol TDS (*Tabular Data Stream*). Selain dari itu, Microsoft SQL Server juga mendukung ODBC (*Open Database Connectivity*), dan memiliki driver JDBC untuk bahasa pemrograman Java. Fitur yang lain dari SQL Server ini mencakup kemampuan untuk membuat basis data *mirroring* dan *clustering*.[22]

2.14 Postman

Postman adalah platform yang dirancang untuk memungkinkan perancangan dan pengujian API yang telah dibuat sebelumnya. Dengan kata lain, itu berfungsi sebagai *client* yang dapat mengakses REST, terotomatisasi pengujian, simulasi *endpoint* secara langsung, menyediakan dokumentasi API, memantau performa API dan waktu *response*, dan menggunakan API secara *real-time*.[23]

2.15 *RESTful* API

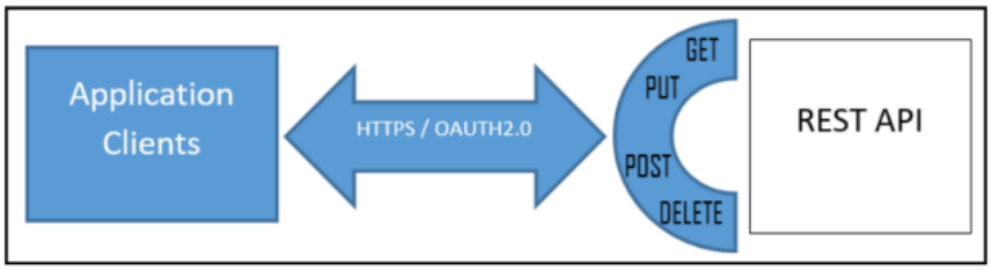
RESTful API adalah jenis layanan web *service* yang beroperasi menggunakan protokol HTTP dan mengikuti prinsip-prinsip arsitektur REST (*Representational State Transfer*). REST merupakan pendekatan arsitektural yang menetapkan aturan-aturan tertentu seperti antarmuka yang konsisten. Implementasi *Restful* API menggunakan perintah HTTP guna memberitahu server hal yang perlu dilakukan terhadap sumber daya. Terdapat empat fungsi metode HTTP yang umum digunakan yaitu :

Tabel 2. 6 Operasi HTTP

Operasi HTTP	Deskripsi
GET	Membaca data dari server
PUT	Memperbarui *resource* yang ada di server
DELETE	Menghapus *resource* dari server
POST	Mengirimkan data ke server untuk membuat *resource* baru

Metode *GET* adalah perintah untuk mengakses sumber daya atau data yang terletak di URI yang ditentukan server. *POST* merupakan perintah untuk mengirimkan data ke server. *PUT*, adalah perintah yang digunakan klien untuk memodifikasi data yang telah ada di server. Berbeda dengan perintah *POST*, permintaan yang dilakukan beberapa kali dengan metode *PUT* akan memberikan hasil yang sama. *DELETE* merupakan perintah untuk menghapus sumber daya atau data pada server.

Web *service* ini digunakan sebagai sarana pertukaran data antara sisi klien dan server yang menyimpan basis data. Data dan fungsi dianggap sebagai sumber daya dalam arsitektur *REST* dan dapat diakses melalui *Uniform Resource Identifier* (URI), yang biasanya merupakan tautan ke web. Dalam prinsip *RESTful* API, method *GET* biasanya digunakan untuk pengambilan data dari sumber karena sifatnya yang *idempotent* dan *cacheable*. Namun, untuk permintaan yang melibatkan data sensitif atau kompleks penggunaan metode *POST* lebih dianjurkan. Meskipun *GET* adalah metode standar untuk mengambil data, *POST* dapat digunakan untuk kueri yang kompleks atau data sensitif yang tidak boleh diekspos di URL[24]. Penggunaan metode *POST* untuk mengirimkan data sensitif seperti mencakup informasi pribadi yang dapat diidentifikasi dapat mengurangi resiko paparan data melalui URL *logging* dan *caching*. Dengan menggunakan metode *POST*, data yang dikirimkan dalam *body request* tidak akan terlihat dan dicatat dalam riwayat browser sehingga memberikan lapisan perlindungan tambahan terhadap akses yang tidak sah[25].



Gambar 2. 7 Arsitektur REST

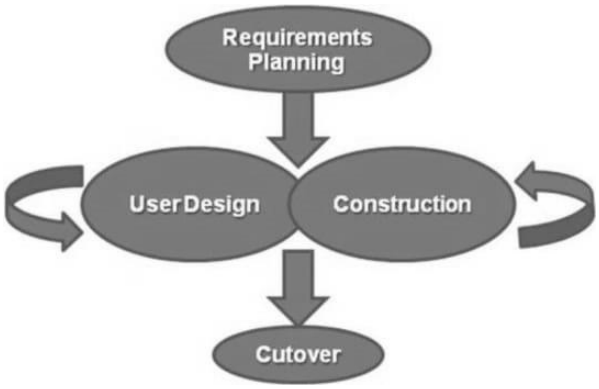
Ketika klien mengirimkan permintaan, server akan memprosesnya, termasuk membuat, menerima, mengubah, dan menghapus sumber daya. Setelah memproses permintaan, server akan memberikan respon kepada klien sebagai bukti selesainya tindakan. Ada dua format data yang dapat dihasilkan: xml atau json. Kedua format ini sangat populer di bidang pertukaran data dan didukung oleh banyak bahasa pemrograman.[26]

Lima karakteristik dasar REST mencakup:

1. *Uniform Interface* yang mengharuskan pemetaan sumber daya dilakukan secara terstruktur, memungkinkan setiap bagian dari *resource* sistem untuk berkembang secara independen.
2. *Stateless* yang memastikan bahwa setiap permintaan menerima *response* tunggal, tanpa adanya interaksi tambahan antara klien dan server, serta tidak menyimpan informasi apapun setelah komunikasi selesai.
3. *Cacheable* yang memungkinkan server untuk memberikan *response* yang sama berdasarkan *request* dari *client* tanpa harus meminta kembali ke *backend* pada pola permintaan tertentu.
4. *Client-Server* yang memisahkan antara proses pada *back-end* dengan antarmuka sistem, memungkinkan pengembangan logika klien dan server secara terpisah.
5. *Layered System* yang melarang permintaan langsung dari klien terhubung ke proses *backend*, meningkatkan skalabilitas dan keamanan sistem.

2.16 Metode RAD

Rapid Application Development (RAD) adalah proses pengembangan aplikasi secara *incremental* (bertingkat) dengan menekankan pada siklus waktu pengerjaan yang singkat dan cepat. Waktu yang singkat pada model pengembangan RAD adalah hal yang penting dimana model kerja sistem (*working model*) dikonstuksikan pada awal pengembangan. Waktu yang diperlukan untuk membuat aplikasi dengan menggunakan RAD dalam waktu 60-90 hari.



Gambar 2. 8 Tahapan metode RAD. [27]

Metode RAD ini sendiri terdapat empat utama, yaitu:

1. *Requirements Planning* atau Rencana Kebutuhan

   Pada tahapan ini, penguna ataupun analis berkumpul untuk menentukan tujuan dari aplikasi atau sistem dan mengidentifikasi informasi yang diperlukan untuk mencapai tujuan tersebut. Peran penting dari tahapan ini adalah keterlibatan kedua belah pihak.
2. *System Design* atau Desain Sistem

   Apabila terdapat ketidaksesuaian desain antara *user* dan analis, proses ini melakukan desain dan perbaikan. Tahapan ini membutuhkan keterlibatan *user* karena masukan dari *user* dapat menentukan apakah tujuan telah dicapai atau tidak. Dengan demikian, *user* dapat langsung memberikan sanggahan apabila terdapat ketidaksesuaian pada desain yang dibuat. Karena perubahan yang terjadi pada sistem aplikasi, tahapan ini juga membutuhkan waktu yang cukup lama karena besarnya aplikasi sistem yang dibuat.
3. *Construction* atau Pengembangan

Setelah desain sistem pada tahap sebelumnya disetujui oleh pengguna dan analis, programmer mulai membangun dan mengembangkan sistem menjadi program. Tahap ini mencakup persiapan kerangka kerja yang cepat, pengembangan program dan aplikasi, serta pembuatan kode pada sistem, unit, integrasi, dan pengujian sistem. Tim pengembang perangkat lunak, penguji, dan pengembang bekerja sama selama tahap ini untuk memastikan semua berjalan lancar dan hasil akhir sesuai dengan harapan pengguna. Pengguna dapat memberikan masukan selama proses ini serta menyarankan perubahan atau ide-ide baru yang dapat menyelesaikan masalah.

4. *Cutover*

   Setelah tahap ini, produk jadi diluncurkan atau dirilis. Selain memberikan pelatihan kepada pengguna, proses ini mencakup pengujian, konversi data, dan penggantian ke sistem baru. Tujuan dari pengujian ini adalah untuk mengidentifikasi kesalahan pada sistem dan untuk mengumpulkan tanggapan dari pengguna mengenai fungsi dan fitur yang baru dibuat. Jika sistem gagal melakukan pengujian, pengembangan atau kembali ke tahap sebelumnya, pembuatan, akan dilakukan untuk memperbaiki oleh analis sebelum kembali dimulai.[28]

2.17 Stress Testing

Stress testing adalah jenis pengujian yang digunakan untuk menilai stabilitas dan keandalan suatu sistem. Aplikasi saat didorong melampaui batas kondisi normalnya atau kondisi puncaknya. Tujuan dari *stress testing* yaitu untuk menemukan *bug* aplikasi yang hanya muncul ketika beban aplikasi sedang dalam kondisi yang tinggi. *Bug* tersebut dapat mencakup hal-hal seperti masalah sinkronasi, *race conditions*, dan kebocoran memori. *Stress test* juga memungkinkan untuk mengidentifikasi titik lemah dari sebuah aplikasi dan juga menunjukkan bagaimana aplikasi kondisi aplikasi di bawah tekanan beban yang ekstrem[29].

Stress testing bisa dilakukan menggunakan alat *load testing* seperti Apache JMeter. Apache JMeter adalah perangkat lunak *open source* yang merupakan aplikasi Java murni, dirancang untuk menguji perilaku fungsional dan mengukur kinerja. Alat ini bisa digunakan untuk menguji kinerja pada sumber daya statis dan dinamis. Apache

JMeter mengirimkan permintaan ke server target dengan mensimulasikan sekelompok pengguna, kemudian mengumpulkan data untuk menghitung statistik dan menampilkan metrik kinerja dalam berbagai format.[30]

Aspek-aspek yang diuji dalam *stress testing* dapat bervariasi tergantung pada alat yang digunakan. Namun umumnya terdapat beberapa aspek umum yang biasanya dievaluasi oleh pengembang saat menjalankan *stress testing* diantaranya :

1. Kinerja Sistem (*Performance*)

   Respon Waktu dan *Throughput* : respon waktu dan throughput adalah metrik utama dalam mengukur kinerja sistem di bawah beban berat. Respon waktu merujuk pada waktu yang dibutuhkan sistem untuk menanggapi permintaan pengguna, sementara *throughput* mengukur jumlah pekerjaan yang dapat diselesaikan dalam periode waktu tertentu.
2. Stabilitas Sistem

   Keandalan dan *Crash Resistance* : stabilitas sistem diukur melalui keandalan dan ketahanan terhadap crash. Penelitian ini menunjukkan pentingnya menguji sistem untuk memastikan tidak ada kegagalan yang signifikan dalam kondisi beban ekstrem .
3. *Scalability*

   Horizontal dan *Vertical Scaling*: skalabilitas horizontal melibatkan penambahan lebih banyak server, sementara skalabilitas vertikal melibatkan peningkatan sumber daya pada server yang ada.
4. *Resource Utilization*

   CPU, *Memory, Disk I/O,* dan *Network Utilization*: pemanfaatan sumber daya seperti CPU, memori, *disk I/O*, dan jaringan sangat penting untuk dinilai dalam *stress testing*. Penggunaan sumber daya ini menunjukkan bagaimana sistem memanfaatkan *hardware* yang tersedia di bawah beban berat .
5. Batas Kapasitas

   Maximum Load dan *Breaking Point*: pengujian batas kapasitas sistem dilakukan untuk menentukan beban maksimum yang dapat ditangani sebelum terjadi penurunan kinerja atau kegagalan. Breaking point adalah titik di mana sistem tidak lagi dapat berfungsi dengan baik .
6. *Recovery*

Error Handling dan *Failover*: pengujian kemampuan pemulihan termasuk penanganan kesalahan dan *failover* untuk memastikan bahwa sistem dapat pulih dari kesalahan dan beralih ke sistem cadangan dengan lancar .

7. *Concurrent Users*

   Simulasi Pengguna konkuren dilakukan untuk menguji bagaimana sistem menangani banyak pengguna yang mengakses secara bersamaan. Hal ini penting untuk mengevaluasi kinerja sistem dalam situasi penggunaan yang tinggi.[31]

Berikut adalah langkah-langkah umum yang perlu diikuti dalam melakukan *stress testing*:

1. Mendefinisikan tujuan dan kriteria keberhasilan

   Tujuan kriteria keberhasilan seperti waktu respons tertentu, tingkat transaksi yang dapat diakomodasi, atau parameter kinerja lainnya.

2. Identifikasi skenario *stress*

   Identifikasi jenis situasi ekstrem yang ingin diuji. Ini bisa mencakup jumlah pengguna yang sangat besar, volume data yang besar, beban transaksi yang tinggi, atau kombinasi dari faktor-faktor ini.

3. Persiapkan lingkungan uji

   lingkungan pengujian bisa mencakup replikasi infrastruktur, jaringan, dan konfigurasi perangkat lunak yang digunakan di produksi.

4. Pilih alat pengujian

   Beberapa alat populer termasuk Apache JMeter, Gatling, dan Siege. Pastikan alat ini mendukung fitur-fitur yang diperlukan untuk mensimulasikan beban yang diinginkan.

5. Konfigurasi skenario *stress*

   Konfigurasikan alat pengujian untuk menjalankan skenario *stress* yang telah identifikasi. Tentukan berapa banyak pengguna yang akan disimulasikan, seberapa sering mereka melakukan tindakan tertentu, dan berapa lama pengujian akan berlangsung.

6. Lakukan pengujian awal

Jalankan pengujian awal dengan skenario stress yang telah dikonfigurasi. Amati perilaku sistem dan catat waktu respons, tingkat kesalahan, penggunaan sumber daya, dan indikator kinerja lainnya.

7. Analisis hasil

   Setelah pengujian awal selesai, analisis hasilnya. Identifikasi apakah ada penurunan kinerja, waktu respons yang tidak dapat diterima, atau masalah lain yang muncul. Tinjau apakah sistem masih berfungsi dengan benar di bawah tekanan ekstrim.

8. Optimasi dan peningkatan

   Jika masalah ditemukan, kerjakan perbaikan dan optimasi yang diperlukan seperti mengubah konfigurasi infrastruktur, mengoptimalkan kode aplikasi, atau memperbarui komponen yang rentan.

9. Ulangi pengujian

   Setelah dilakukan perbaikan, ulangi pengujian dengan skenario stres yang sama. Bandingkan hasilnya dengan pengujian sebelumnya dan perhatikan apakah perbaikan yang dilakukan berhasil mengatasi masalah yang ada.

10. Dokumentasi dan Pelaporan

    Catat semua langkah, hasil pengujian, perbaikan yang dilakukan, dan kesimpulan yang ditarik. Buat laporan resmi tentang hasil pengujian *stress*, termasuk rekomendasi untuk langkah-langkah selanjutnya.[32]

2.18 Usability Testing

Usability Testing adalah evaluasi yang dilakukan terhadap suatu produk, berfokus pada kemampuan pengguna untuk mencapai tujuan yang telah ditetapkan dan merasakan kepuasan. Berdasarkan ISO 9241:11 (1998), *usability* merujuk pada sejauh mana suatu produk dapat digunakan oleh pengguna tertentu untuk mencapai target secara efektif, efisien, dan memuaskan. *Usability* diukur melalui lima komponen, yaitu:

a. Kemudahan (*learnability*) merupakan pengukuran kemudahan dalam menjalankan fungsi dan seberapa cepat pengguna dapat menggunakan
b. Efisiensi (*efficiency*) diartikan sebagai sumber daya yang dikeluarkan untuk mencapai ketepatan dan kelengkapan tujuan

c. Mudah Diingat (*memorability*) adalah kemampuan pengguna untuk mempertahankan pengetahuannya setelah jangka waktu tertentu untuk mengingat dari letak satu menu selalu tetap.
d. Kesalahan dan Keamanan (*errors*) adalah banyak kesalahan yang dibuat pengguna yang termasuk ketidaksesuaian pikiran pengguna terhadap tujuan sistem.
e. Kepuasan (*satisfaction*) merupakan kebebasan dari ketidaknyamanan serta sikap positif terhadap penggunaan produk. [33]

2.19 *Blackbox Testing*

Blackbox Testing merupakan pengujian fungsional sebuah perangkat lunak tanpa menguji desain dan kode program. Pengujian *blackbox* mengabaikan mekanisme internal sistem dan didasarkan pada persyaratan dan spesifikasi perangkat lunak. Pada *blackbox testing* berfokus pada *input* dan *output* dari sistem perangkat lunak. Terdapat banyak jenis pengujian *blackbox*, berikut merupakan pengujian *blackbox* yang sering digunakan :

a) *Functional Testing* adalah pengujian *blackbox* terkait dengan persyaratan fungsional suatu sistem.
b) *Non-Functional Testing* adalah jenis pengujian *blackbox* terkait dengan persyaratan non fungsional seperti kinerja, skalabilitas, dan kegunaan,
c) *Regression Testing* adalah pengujian yang dilakukan setelah perbaikan kode, peningkatan bahan dengan pemeliharaan sistem untuk memeriksa kode baru dan tidak mempengaruhi kode yang ada

Saat ini, ada sepuluh metode yang dapat digunakan untuk menjalankan teknik pengujian *blackbox* diantaranya : *Boundary Value Analysis (BVA), Cause-Effect Graphing (CEG), Decision Tables (DT), Equivalence Partitioning (EP), Orthogonal Array Testing (OAT), Random Testing (RT), Specification-Based Mutation Testing (SBMT), State-Transition Diagram Testing (STT), Syntax Testing (ST), Worst Case Testing (WCT)*.[34]

2.20 USE Questionnaire

USE Questionnaire adalah metode kuisioner dalam pengukuran *usability*. Pengujian ini dilakukan dengan mengevaluasi interaksi antara sistem dan pengguna menggunakan metode *USE Quesionnaire*. Metode *USE Questionnaire* digunakan untuk menilai penerimaan pengguna terhadap aplikasi yang berjalan sesuai dengan standar kebutuhan *user*. Kuisioner USE terdiri dari tiga bagian diantaranya *Usefulness* (kegunaan), *Satisfaction* (Kepuasan), dan *Ease of Use* (kemudahan pengguna). Ketiga dimensi tersebut berfungsi paling efektif dalam pengujian antarmuka dengan dimensi kemudahan pengguna dan kegunaan saling mempengaruhi.[35]

Tabel 2. 7 Kriteria Pengukuran *USE Questionnaire.* [35]

NO	KRITERIA *USABILITY*
A	*Usefulness* (Kegunaan)
1	Aplikasi ini membantu pengguna menjadi lebih efektif
2	Aplikasi ini membantu pengguna menjadi lebih produktif
3	Aplikasi ini bermanfaat bagi pengguna
4	Aplikasi ini membantu pengguna terhadap tugas yang pengguna lakukan
5	Aplikasi ini membuat hal-hal yang ingin dicapai pengguna lebih mudah untuk dilakukan
6	Aplikasi ini menghemat waktu pengguna ketika menggunakannya
7	Aplikasi ini sesuai dengan kebutuhan pengguna
8	Aplikasi ini bekerja sesuai dengan apa yang pengguna harapkan
B	*Easy of Use* (Kemudahan Pengguna)
9	Aplikasi ini mudah digunakan
10	Aplikasi ini praktis digunakan
11	Aplikasi ini mudah dipahami oleh pengguna
12	Aplikasi ini memiliki langkah-langkah pengoperasian yang praktis
13	Aplikasi ini bersifat fleksibel
14	Aplikasi ini tidak sulit ketika digunakan
15	Pengguna dapat menggunakan aplikasi ini tanpa instruksi tertulis
16	Pengguna tidak melihat adanya ketidak konsistenan selama aplikasi ini digunakan
17	Pengguna yang jarang maupun rutin menggunakan aplikasi ini akan menyukainya
18	Pengguna dapat kembali dari kesalahan secara cepat dan mudah
19	Pengguna dapat menggunakan aplikasi ini dengan sukses setiap kali sistem digunakan
C	*Easy of Learning* (Kemudahan Mempelajari)
20	Pengguna belajar menggunakan aplikasi ini dengan cepat
21	Pengguna mudah mengingat bagaimana cara menggunakan aplikasi ini
22	Aplikasi ini mudah untuk dipelajari cara penggunaannya
23	Pengguna cepat menjadi terampil dengan aplikasi ini
D	*Satisfaction* (Kepuasan Pengguna)
24	Pengguna puas dengan aplikasi ini
25	Pengguna akan merekomendasikan aplikasi ini kepada rekan
26	Aplikasi ini menyenangkan untuk digunakan
27	Aplikasi ini bekerja seperti apa yang pengguna inginkan
28	Aplikasi sangat bagus
29	Pengguna merasa harus menggunakan aplikasi ini
30	Aplikasi ini nyaman untuk digunakan

Kuisioner ini menggunakan skala likert untuk perhitungannya. Skala likert digunakan untuk mengukur sikap, pendapat, dan persepsi individu atau kelompok mengenai fenomena sosial yang dijabarkan melalui indikator variabel.[36]

Tabel 2. 8 Nilai Skala Likert

No	Jawaban	Kode	Nilai Skor
1	Sangat Setuju	SS	5
2	Setuju	S	4
3	Ragu-Ragu	RG	3
4	Tidak Setuju	TS	2
5	Sangat Tidak Setuju	STS	1

Pengukuran *usability* dihitung berdasarkan persentase jawaban dari responden menggunakan rumus :

$$Persentase\ Kelayakan\ (\%) = \frac{\text{Skor yang diobservasi}}{\text{Skor yang diharapkan}} x100\%$$

Data yang terkumpul kemudian diubah menjadi kategori kelayakan berdasarkan tabel yang telah ditetapkan sebelumnya. Tabel tersebut menggambarkan nilai-nilai yang masuk ke dalam kategori kelayakan.

Tabel 2. 9 Kategori Kelayakan [37]

Angka (%)	Klasifikasi
< 21	Sangat tidak layak
21 – 40	Tidak layak
41 – 60	Cukup
61 - 80	Layak
81 - 100	Sangat Layak

Usability testing akan dilakukan terhadap 5 responden menggunakan metode *Retrospective Think Aloud* dan pengukuran kinerja. Jumlah responden yang dipilih, yaitu 5 orang, dianggap cukup karena dapat mewakili sebagian besar pengguna untuk mengidentifikasi sekitar 85% masalah kelayakan aplikasi[38]. Penggunaan pengukuran kinerja dalam pengujian *usability* bertujuan untuk menilai tingkat keberhasilan dan kecepatan penyelesaian tugas yang diberikan kepada responden.

Selain itu, penggunaan metode *Retrospective Think Aloud* bertujuan untuk menggali data berdasarkan pengalaman, pemikiran, dan pandangan responden saat berinteraksi langsung dengan aplikasi.[39]

2.21 Penelitian Terkait

Penelitian terkait menjadi referensi pada penelitian ini dengan memuat artikel ilmiah atau jurnal terkait yang mempunyai hubungan dengan penelitian. Beberapa hal yang bisa dijadikan referensi misalnya kecenderungan studi kasus ataupun metode pada pengembangan sistem. Berikut ini merupakan penelitian terkait yang dijadikan sebagai referensi pada penelitian ini.

Penerapan Teknologi REST API Pada Aplikasi Perpustakaan Digital Politeknik Gorontalo yang ditulis oleh Ismail Mohidin dan Saiful Bahri Musa. Penelitian ini mengusulkan sistem rancangan aplikasi *mobile* yang terintegrasi dengan *web service* dengan menerapkan teknologi REST API dan fitur QR (*Quick Response*) *Code* untuk memudahkan management sistem perpustakaan sehingga membantu dalam management dan monitoring peminjaman dan koleksi buku. Penelitian ini dilakukan karena penemuan masalah terkait pelaksanaan peminjaman buku yang ada di perpustakaan ditemukan kendala dalam pengembalian buku, jurnal, yang diakibatkan dosen maupun mahasiswa lupa dalam mengembalikan buku yang menyebabkan kurangnya daftar buku yang akan dipinjam karena status buku belum dikembalikan. Sistem yang dibangun memberikan informasi kepada dosen dan mahasiswa tentang waktu peminjaman buku serta tambahan fitur notifikasi perpanjangan waktu peminjaman untuk mengantisipasi keterlambatan pengembalian buku. [40]

Aplikasi Perpustakaan Digital Pada Perpustakaan Jurusan Ilmu Komputer Universitas Lampung Berbasis Android, yang ditulis oleh Andika Yuda Pratama dan Kurnia Muludi. Studi ini meneliti cara membuat aplikasi untuk perpustakaan digital di Jurusan Ilmu Komputer. Perpustakaan jurusan ilmu komputer masih menggunakan metode manual untuk meminjam dan mengembalikan buku. Sistem aplikasi ini bertujuan untuk menyelesaikan masalah tersebut. Aplikasi perpustakaan

digital ini dirancang berbasis android dan website dengan menggunakan bahasa pemrograman Java dan database PHP dan MySQL. Penelitian ini menggunakan metode waterfall, kesimpulan dari penelitian ini adalah berhasil membangun aplikasi perpustakaan digital berbasis android yang digunakan untuk memudahkan pengelolaan data peminjaman buku untuk laporan perpustakaan. [41]

Pengembangan Sistem Informasi Perpustakaan Pada Dinas Perpustakaan dan Kearsipan Kota Medan Berbasis Android oleh Nur Fadilah, Ali Ikhwan, dan Muhammad Alda. Penelitian ini mengusulkan sistem rancanganan sebagai pengembangan dari sistem informasi perpustakaan yang telah ada yaitu berbasis website menjadi sistem informasi dengan memanfaatkan teknologi berbasis android dengan tujuan mempermudah anggota perpustakaan dalam melakukan pendaftaran, peminjaman, dan melakukan perpanjangan peminjaman buku. Penelitian ini juga mengembangkan fitur notifikasi sebagai pengingat anggota perpustakaan terkait masa peminjaman buku yang akan segera berakhir sebagai solusi terkait keterlambatan pengembalian buku Hasil peneitian ini adalah berhasil dikembangkan sistem informasi perpustakaan yang dapat memfasilitasi anggota perpustakaan berbasis android dengan implementasi firebase sebagai database. [42]

Rancang Bangun Aplikasi Perpustakaan Digital Berbasis Android Menggunakan Framework Flutter oleh Romario Lendo dkk. Penelitian ini membangun layanan E-Library pada Universitas Sam Ratulangi yang menyediakan layanan konten pengunggahan, peminjaman, baca, dan pengembalian buku oleh mahasiswa. Penelitian ini menggunakan metode pengembangan perangkat lunak model *Rapid Application Development* (RAD) yang memuat empat tahapan diantaranya analisa kebutuhan, perancangan, implementasi, dan evaluasi dengan waktu pengembangan 30-90 hari. Penelitian ini juga telah berhasil mengembangkan API menggunakan pola *RESTful* API dengan data-data diambil dari server yang disimpan dalam *Railway* yang terdapat *web server* dengan teknologi ExpressJS dan MongoDB sebagai tempat penyimpanan data. API tersebut disediakan *open* API untuk dapat diakses klien android dan aplikasi PWA. [43]

RESTful Web Service untuk Integrasi Sistem Akademik dan Perpustakaan Universitas Perjuangan oleh Randi Rizal dan Alam Rahmatulloh. Penelitian ini dilatarbelakangi oleh permasalahan duplikasi data pada sistem informasi Universitas Perjuangan. Kemudian dirancang sebuah web service dengan mengintegrasikan sistem informasi yang paling banyak digunakan oleh mahasiswa yaitu sistem informasi perpustakaan dan sistem informasi akademik dengan arsitektur web service menggunakan REST. Hasil dari penelitian ini adalah sistem informasi akademik dan sistem informasi perpustakaan mampu terintegrasi dengan teknologi *web service* dengan format pertukaran data menggunakan format JSON sehingga memudahkan untuk dapat diakses oleh bahasa pemrograman, arsitektur, maupun sistem operasi yang berbeda. [44]

Penerapan REST API Menggunakan Retrofit Untuk Sistem Informasi Film Berbasis Android (Studi Kasus : Sinopsis Film) oleh Robbie Christhover, Sugiyatno, dan Herlawati. Penelitian ini membangun aplikasi android dengan kotlin yang menerapkan REST API bersifat *open source* untuk memberikan informasi Film berbasis Android sehingga dapat melakukan pencarian sinopsis film dengan memanfaatkan API. Penerapan REST API dalam aplikasi android menggunakan Retrofit yang merupakan Pustaka klien HTTP pada android dan java. Retrofit berfungsi sebagai REST Klien untuk mempermudah proses pertukaran data dari format JSON dalam menangani proses transmisi data (permintaan) melalui jaringan (network). Pengambilan data dilakukan dengan Aplikasi akan membuat request dalam bentuk Object POJO dengan format JSON yang selanjutnya REST API memberikan respons dalam format JSON yang akan dijadikan Object POJO ke dalam aplikasi. [45]

Library Automation System Integration (Case: ELIB and SLiMS) oleh Meizano Ardhi Muhammad dan Mardiana. Penelitian ini membahas tentang migrasi data pada sistem otomasi perpustakaan yang mana sistem perpustakaan yang ada telah menerapkan teknologi seperti *barcode* dan RFID. Perpustakaan Unila telah menggunakan sistem otomasi perpustakaan SLiMS untuk mengelola koleksi. Namun, kekurangan dari sistem otomasi ini adalah SLiMS tidak

mendukung protocol SIP2. Adanya sistem otomasi ELIB yang mendukung perangkat perpustakaan RFID dan terhubung dari protokol SIP2 dapat melakukan otomasi transaksi melalui Self Service Kiosk, pengumpulan pemindaian, dll. Oleh karena itu, penelitian ini dilakukan untuk migrasi data dari sistem otomasi SLiMS ke sistem otomasi ELIB yang mendukung pennggunaan teknologi RFID. Penelitian ini menyoroti tantangan dalam sinkronisasi data atara database berbeda antara SLiMS (MySQL 5.5.27) dan ELIB (MS SQL Server 2014) yang memerlukan penyesuaian setiap kali pembaruan terjadi. Hasil dari penelitian ini ialah sistem integrasi yang dibangun sebagai *middleware* untuk menyingkonkan basis data dan mendukung pengembangan sistem lebih lanjut di masa mendatang.[9]

API Features Individualizing of Web Services: REST and SOAP oleh Anshu Soni dan Virender Ranga. Penelitian ini membahas tentang layanan web service yaitu Representasional State Transfer (REST) dan Simple Object Access Protocol (SOAP) dalam memutuskan layanan terbaik pengembangan web sesuai dengan kebutuhan. Penelitian ini merancang REST API dan SOAP API berdasarkan parameter diantaranya waktu respons, arsitektur, dan penggunaan memori. Hasil penelitian ini didapatkan data pertama adalah waktu respon yang digunakan REST dan SOAP untuk *Create message. Delete message, dan Query Message*, REST menghasilkan waktu respons lebih sedikit dibandingkan SOAP karena penggunaan bandwidth yang tinggi dan mengonsumsi lebih banyak sumber daya. Perbandingan kedua yaitu berdasarkan arsitektur dalam pengembangannya dimana REST yang menggunakan gaya arsitektur dengan metode HTTP lebih mudah dimengerti daripada SOAP. Terakhir yaitu penggunaan memori yang mana kebutuhan memori SOAP lebih tinggi dibandingkan REST karena REST menggunakan format JSON yang mudah diurai dibandingkan XML yang merupakan format SOAP. [46]

IDOL : Retrofit-Kotlin Service-Based Online Digital Library Application and College Open Data Repository oleh Taufiq Iqbal dan Muhammad Wali. Penelitian ini melakukan pengembangan IDOL (Integrated Digital Online Library) pada sistem perpustakaan digital dengan mengimplementasikan Retrofit-Kotlin sebagai klien REST yang aman untuk Java dan Android. Metode pengembangan pada

penelitian ini menggunakan RAD (*Rapid Application Development*). Hasil dari penelitian ini adalah berhasil mengimplementasikan penggunaan Retrofit-Kotlin berdasarkan *User Testing* (UT) mencapai tingkat keberhasilan 80%. Layana Retrofit-Kotlin sebagai perpustakaan REST Klien sangat aman untuk aplikasi android dengan penanganan dalam format JSON serta dukungan format JSON/XML dengan sereliasisasi / desentralisasi data dan mampu melayani koneksi data dari android ke internet pada aplikasi IDOL. [47]

Usability Evaluation of Personalized Adaptive E-Learning System Using USE Questionnaire oleh Didik, Moch. Bruri, dan Thomas Kohler. Penelitian ini menggunakan *USE Questionnaire* untuk mengukur kegunaan dari sistem *e-learning.* Kuisioner terdiri dari 30 pertanyaan terdiri dari tiga varibel independen (kegunaan, kemudahan penggunaan, dan kemudahan pembelajaran) yang diberikan kepada 62 responden. Responden dalam penyebaran kuisioner ini adalah siswa dari sekolah menengah kejuruan Provinsi Yogyakarta. Hasil analisis pada *USE Questionnaire* memiliki validitas dan reliabilitas yang baik dalam mengukur kegunaan sistem *e-learning.* [48]

The Evaluation of Final Assignment System Using the USE Questionnaire Approach oleh Aji Purwinarko, Mona Subagja, dan Alfath Yanuarto. Penelitian ini menggunakan metode evaluasi *USE Questionnaire* terhadap sistem informasi penilaian tugas akhir mahasiswa pada Universitas Negeri Semarang. Tujuan kuisioner guna memberikan informasi mengenai kelebihan dan kelemahan suatu produk atau jasa. Pengujian dilakukan dengan metode simple random sampling dari populasi dan menghasilkan 75 orang sebagai responden. Hasil dari pengukuran usability menghasilkan nilai 85,2% yang menunjukkan sistem tugas akhir sangat layak untuk digunakan. Kesimpulan pada penelitian ini adalah pengaruh signifikan antara variabel independen (Kegunaan, Kemudahan Penggunaan, dan Kemudahan Belajar) terhadap variabel dependen (Kepuasan). [37]

Tabel 2. 10 Penelitian Terkait

Judul	Metode Penelitian	Hasil Penelitian
Penerapan Teknologi REST API Pada Aplikasi Perpustakaan Digital Politeknik Gorontalo (2022, Jurnal Technopreneur, Vol. 10 N0. 1)	Waterfall Model	Aplikasi perpustakaan digital Politeknik Gorontalo dapat mengirimkan informasi terkait batas pengembalian buku melalui aplikasi android sehingga keterlambatan dapat diatasi. Aplikasi android menampilkan data real dari web ke aplikasi android menerapkan teknologi REST API.
Aplikasi Perpustakaan Digital Pada Perpustakaan Jurusan Ilmu Komputer Universitas Lampung Berbasis Android (2021, Jurnal Pepadun, Vol. 2 No. 1)	Waterfall Model	Aplikasi Perpustakaan Digital untuk Perpustakaan Jurusan Ilmu Komputer dirancang menggunakan pendekatan *user oriented* dan menggunakan metode *waterfall.* Aplikasi ini berhasil menampilkan informasi tentang data buku pada perpustakaan yang tersedia atau masih dalam peminjaman. Aplikasi Perpustakaan digital memudahkan admin untuk manajemen data buku dan memudahkan pembuatan laporan perpustakaan.
Pengembangan Sistem Informasi Perpustakaan Pada Dinas Perpustakaan dan Kearsipan Kota Medan Berbasis Android (2023,	*Research and Development Model*	Penelitian ini bertujuan mempermudah anggota perpustakaan dalam melakukan pendaftaran, peminjaman, dan melakukan perpanjangan peminjaman buku. Penelitian ini juga mengembangkan

Jurnal Teknologi Sistem Informasi dan Sistem Komputer TGD, Vol. 6 No. 2)		fitur notifikasi sebagai pengingat anggota perpustakaan terkait masa peminjaman buku yang akan segera berakhir sebagai solusi terkait keterlambatan pengembalian buku
Rancang Bangun Aplikasi Perpustakaan Digital Berbasis Android Menggunakan Framework Flutter (2023, Jurnal Teknik Informatika, Vol. 18 No. 1)	RAD Model	Aplikasi dikembangkan dengan tujuan membangun layanan *e-library* pada Universitas Sam Ratulangi yang menyediakan layanan pengunggahan, peminjaman, baca buku, dan pengembalian buku oleh mahasiswa. Penelitian ini berhasil mengembangkan API menggunakan pola Restful API dengan data-data yang diambil dari server kemudian disimpan dalam Railway yang terdapat dalam web server.
RESTful Web Service untuk Integrasi Sistem Akademik dan Perpustakaan Universitas Perjuangan (2019, Jurnal Ilmiah Informatika, Vol.7 No. 1)	Waterfall Model	Penelitian ini merancang sebuah web service dengan mengintegrasikan sistem informasi yang paling banyak digunakan oleh mahasiswa yaitu sistem informasi perpustakaan dan sistem informasi akademik dengan arsitektur web service menggunakan REST. Hasil dari penelitian ini adalah sistem informasi akademik dan sistem informasi perpustakaan mampu terintegrasi dengan teknologi *web service* dengan format pertukaran data menggunakan format JSON sehingga memudahkan untuk dapat diakses oleh bahasa pemrograman,

		arsitektur, maupun sistem operasi yang berbeda
Penerapan REST API Menggunakan Retrofit Untuk Sistem Informasi Film Berbasis Android (Studi Kasus : Sinopsis Film) (2022, Journal of Students Research in Computer Science, Vol. 3 No. 2)	Waterfall Model	Penelitian ini membangun aplikasi android dengan kotlin yang menerapkan REST API bersifat *open source* untuk memberikan informasi Film berbasis Android sehingga dapat melakukan pencarian sinopsis film dengan memanfaatkan API.
Library Automation System Integration (Case: ELIB and SLiMS)	Model Spiral (SDLC)	Penelitian ini membahas tentang migrasi data pada sistem otomasi perpustakaan yang mana sistem perpustakaan yang ada telah menerapkan teknologi seperti barcode dan RFID. Adanya sistem otomasi ELIB yang mendukung perangkat perpustakaan RFID dan terhubung dari protokol SIP2 dapat melakukan otomasi transaksi melalui Self Service Kiosk, pengumpulan pemindaian, dll. Penelitian ini menyoroti tantangan dalam sinkronisasi data atara database berbeda antara SLiMS (MySQL 5.5.27) dan ELIB (MS SQL Server 2014) yang memerlukan penyesuaian setiap kali pembaruan terjadi. Hasil dari penelitian ini ialah sistem integrasi yang dibangun sebagai

		middleware untuk menyingkonkan basis data
API Features Individualizing of Web Services: REST and SOAP (2019, International Journal of Innovative Technology and Exploring Engineering, Vol. 8)	Metode Deskriptif dengan Pendekatan Kuantitatif	Penelitian ini merancang REST API dan SOAP API dengan membandingkan waktu respons,penggunaan memori, dan kecepatan eksekusi. hasil dari penelitian ini adalah waktu respons SOAP lebih lama dibandingkan REST.
IDOL : Retrofit-Kotlin Service-Based Online Digital Library Application and College Open Data Repository(2022, International Journal Software Engineering and Computer Science, Vol. 2 N0. 1)	RAD Model	Penelitian ini melakukan pengembangan IDOL (Integrated Digital Online Library) pada sistem perpustakaan digital dengan mengimplementasikan Retrofit-Kotlin sebagai klien REST yang aman untuk Java dan Android.
The Evaluation of Final Assignment System The Evaluation of Final Assignment System Using the USE Questionnaire Approach (2020, Scientific Journal of Informatics, Vol. 7 No. 2)	Metode Deskriptif dengan Pendekatan Kuantitatif	Penelitian ini menggunakan metode evaluasi USE Questionnaire guna memberikan informasi mengenai kelebihan dan kelemahan suatu produk atau jasa. Hasil penelitian ini adalah pengaruh signifikan antara variabel independent (Kegunaan, Kemudahan Penggunaan, dan Kemudahan Belajar) terhadap variabel dependen (Kepuasan)

III. METODE PENELITIAN

3.1 Waktu dan Tempat Penelitian

Penelitian tugas akhir ini dilakukan pada November 2023 sampai dengan April 2024 yang bertempat di UPT Perpustakaan Universitas Lampung

3.2 Jadwal Penelitian

Jadwal kegiatan penelitian yang akan dilakukan adalah sebagai berikut:

Tabel 3. 1 Jadwal Kegiatan Penelitian

No	Kegiatan	November 2023				Desember 2023				Januari 2024				Februari 2024				Maret 2024				April 2024			
		1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4	1	2	3	4
1	*Require-ment Planning*																								
2	*User Design*																								
3	*Construc-tion*																								
4	*Cutover*																								
5	Pengerj-aan Laporan																								

3.3 Alat dan Bahan

3.3.1 Alat Penelitian

Alat dan bahan yang digunakan dalam penelitian ini adalah sebagai berikut.

Tabel 3. 2 Alat Penelitian

No.	Nama Alat	Spesifikasi	Deskripsi
1.	Laptop	Intel(R) Core(TM) i5-10210U CPU @1.60GHz 2.11 GHz, 8GB RAM, SSD 512GB, Sistem Operasi Windows 10	Perangkat keras yang digunakan dalam proses pembuatan aplikasi.
2.	Android Studio	Versi Dolphin 2021.3.1 Patch 1	Perangkat lunak sebagai penulisan kode untuk aplikasi mulai dari *design* UI sampai *debugging*.
3.	Android Virtual Device	API 33	Perangkat lunak untuk menguji aplikasi pada tingkat daya komputasi dan memori dan dapat berjalan di perangkat dengan spesifikasi berbeda
4.	Visual Studio Code	Versi 1.83.1	Perangkat lunak yang digunakan untuk mengembangkan *RESTful* API
5.	*Design Requirement*	Umlet	Perangkat lunak untuk merancang *design requirement*
6.	Antarmuka dan Prototype	Figma	Perangkat lunak untuk merancang antarmuka aplikasi
7.	SQL Server Management Studio	Versi 2014	Perangkat lunak untuk mengelola database SQL Server pada database *library* sistem ELIB

8.	NodeJS	Versi 18.14.0	Platform untuk pengembangan aplikasi web dan mobile menggunakan JavaScript di sisi server.
9.	ExpressJS	Versi 4.16.1	Framework web app untuk NodeJS yang ditulis dengan bahasa pemrograman JavaScript.
10.	Kotlin	Versi 1.9.24	Bahasa pemrograman untuk pengembangan aplikasi android
11.	Pengujian API	Postman	Perangkat lunak untuk menguji dan mengelola API

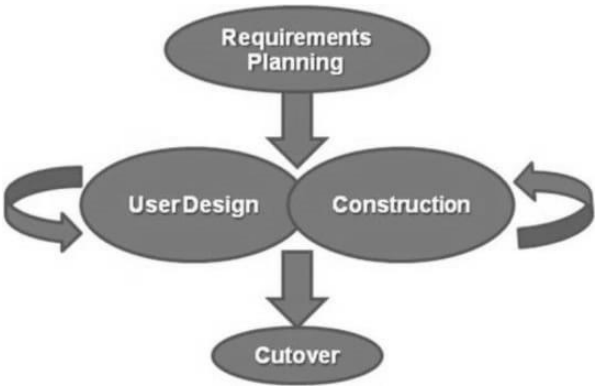
3.3.2 Bahan Penelitian

Tabel 3. 3 Bahan Penelitian

No.	Nama	Deskripsi
1.	Database ELIB Unila	Digunakan sebagai informasi data yang akan dipakai dalam aplikasi untuk informasi *user*, peminjaman buku, pengembalian buku, dan denda yang harus dibayarkan.

3.4 Tahapan Penelitian

Langkah-langkah dalam penelitian ini mengadopsi metodologi pengembangan perangkat lunak Rapid Application Development (RAD). Selama proses penelitian, fase Pelaporan juga dilakukan. Gambaran tahapan penelitian dengan pendekatan RAD terlihat dalam diagram berikut:



Gambar 3. 1 Tahapan Penelitian

Gambar 3.1 menjelaskan urutan pada tahapan metode RAD yang meliputi 4 tahapan, yaitu: *Requirement Planning, User Design, Construction,* dan *Cutover.* Pada tahapan *Requirement Planning* (Perencanaan Kebutuhan) dilakukan untuk mendapatkan informasi dan mengumpulkan data pada layanan sirkulasi / peminjaman buku di perpustakaan Universitas Lampung. Tahapan kedua setelah *requirement planning* adalah *user design* yaitu mendefinisikan sistem dengan melakukan proses *design* sistem (*prototype*). Tahapan setelah rancangan desain sistem, berikutnya adalah tahap *construction*. Tahapan ini merupakan tahap membuat sistem dengan memulai menyusun suatu kode program (*coding*) guna mengubah desain sistem yang telah dibuat untuk selanjutnya diimplementasikan menjadi sebuah aplikasi yang dapat digunakan. Tahap terakhir adalah *Cutover* yaitu implementasi aplikasi diluncurkan kepada pengguna. Pada tahap ini, pengguna memiliki kemampuan untuk memberikan umpan balik terhadap sistem yang telah dibuat dengan dilakukan pengujian pada pengguna untuk memastikan sistem berjalan dengan lancar dan sesuai dengan harapan pengguna.

3.4.1 *Requirement Planning*

Pada tahap awal perancangan sistem dilakukan perencanaan kebutuhan (*requirement planning*). Tujuan dari tahap ini yaitu menghasilkan dokumen *user requirement* yang akan dijadikan sebagai acuan pengembangan sistem pada tahap selanjutnya. Tahap *requirement planning* dilakukan dengan pengumpulan informasi dan pengumpulan data yang diperoleh dengan melakukan metode sebagai berikut:

a. Observasi

Pada tahap ini dilakukan dengan pengamatan secara langsung di Perpustakaan Universitas Lampung untuk mengamati secara langsung terkait alur layanan sirkulasi perpustakaan, menemukan masalah, serta data yang dibutuhkan dalam penelitian. Observasi dilakukan di Perpustakaan Universitas Lampung untuk mengetahui dan memperoleh database sistem ELIB yang akan digunakan dalam pengembangan aplikasi android. Hasil dari observasi yang dilakukan pada database sistem ELIB, belum terdapat API sebagai antarmuka komunikasi pertukaran data antara server database dan klien yang pada penelitian ini adalah aplikasi berbasis android.

b. Interview / Wawancara

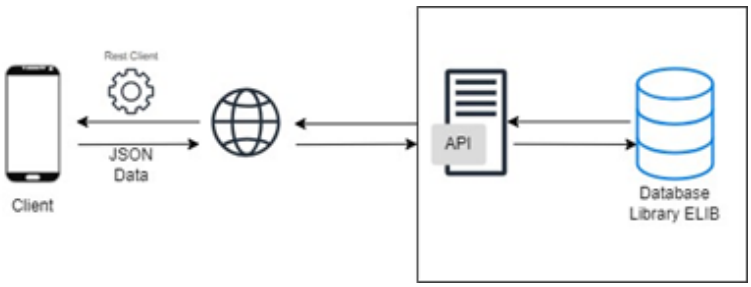
Wawancara adalah sebuah teknik bertatap muka secara langsung dimana dilakukan penulis sebagai pewawancara melalui pertanyaan-pertanyaan yang diajukan kepada narasumber. Narasumber yang dimaksud adalah pihak yang memiliki kepentingan pada layanan sirkulasi / peminjaman buku. Adapun narasumber yang dilakukan wawancara yaitu Pemustaka dan staff Perpustakaan Universitas Lampung, penulis mendapatkan informasi, kritik, dan juga saran terhadap layanan sirkulasi untuk selanjutnya dilakukan usulan sistem yang terdiri dari penemuan masalah yang terjadi dalam layanan sirkulasi / peminjaman terkait akses informasi peminjaman buku oleh pemustaka.

c. Studi Literatur

Setelah mengetahui kebutuhan pemustaka terhadap aplikasi yang akan dikembangkan, Langkah berikutnya adalah studi literatur yang dilakukan untuk memahami dan menambah sumber referensi berdasarkan literatur-literatur sebelumnya. Ini termasuk membaca dan mempelajari buku, jurnal, dan artikel yang berkaitan dengan penelitian ini.

d. Arsitektur Sistem

Arsitektur sistem menggambarkan model sistem status peminjaman buku mulai dari alur transaksi data pada database ELIB, *RESTfu*l Api yang akan dibangun, serta implementasi pada aplikasi berbasis android. Arsitektur sistem diperlukan untuk mengetahui alur sistem yang akan dikembangkan dan hubungan antar objek nya.



Gambar 3. 2 Arsitektur Komunikasi Sistem

Gambar 3.2 menjelaskan arsitektur komunikasi sistem yang akan dibangun pada penelitian ini. Proses dimulai dengan perancangan REST API untuk aplikasi. Database ELIB Unila yang menggunakan DBMS SQL Server menjadi basis data utama yang akan digunakan dalam aplikasi. Aplikasi client yang akan dibangun pada penelitian ini berbasis android akan teritegrasi oleh pemrograman aplikasi antarmuka dengan metode REST. Data akan dikirimkan dalam bentuk JSON kemudian akan diterima sebagai *request* yang akan dikirim ke server API untuk selanjutnya API server memproses permintaan ke database dan dikembalikan dalam bentuk *response* data.

3.4.2 *User Design*

Tahap *User Design* mendefinisikan sistem dengan melakukan proses design sistem (*prototype*) kemudian melakukan uji coba (*test*). Jika terdapat ketidaksesuaian terhadap kebutuhan *user* maka dapat di *rifine* atau diperbaiki. *User Design* akan memudahkan *user* dalam memahami proses bisnis aplikasi yang akan dibangun. Sehingga terdapat beberapa hal yang perlu ditentukan yaitu sebagai berikut :

a. ***Use Case***

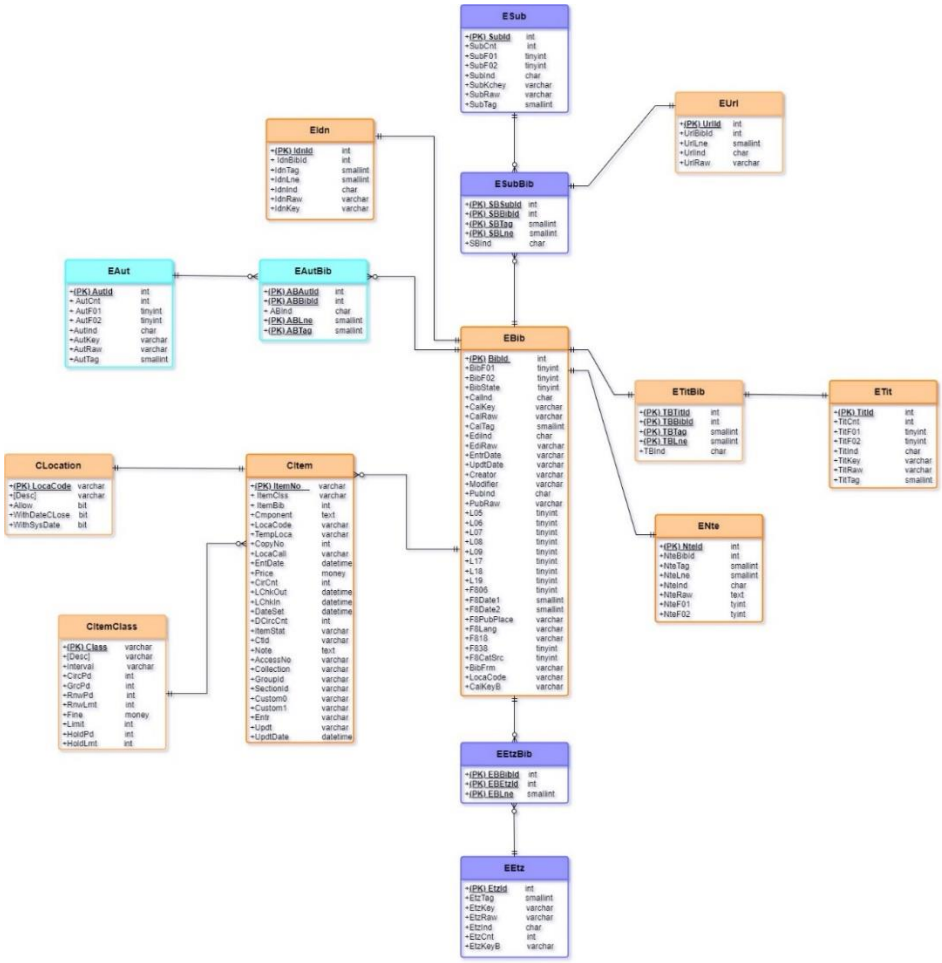
Pada penelitian ini *use case* digunakan untuk menjelaskan interaksi atau skenario *user* dengan sistem Aplikasi Informasi Peminjaman buku *Use case* sistem aplikasi informasi peminjaman buku dibuat berdasarkan dokumen *user* requirement pada tahap perencanaan kebutuhan. *Use case* yang dibuat merupakan rincian yang dilakukan oleh pemustaka.

b. Activity Diagram

Activity Diagram menunjukkan gambaran terkait aliran aktivitas dalam perancangan sistem. Aliran aktivitas dijelaskan mulai dari bagaimana sistem dimulai sampai proses akhir dari aktivitas sistem berakhir. *Activity diagram* menjelaskan proses lebih lanjut dari *use case* diagram.

c. Database Diagram

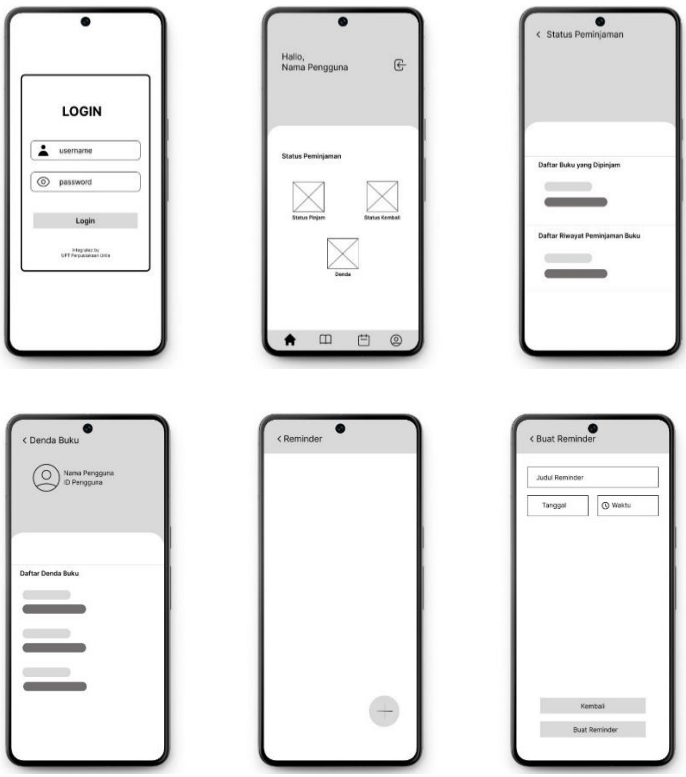
Dalam penelitian, database diagram digunakan untuk mengidentifikasi susunan data yang akan digunakan selama proses pengembangan serta hubungan antar entitas. Gambar 3.3 adalah relasi entitas pada database ELIB untuk bibliografi. Tabel ini berhubungan dengan data terkait layanan sirkulasi terkait data buku baik item buku yang memuat informasi pengarang, biblio, penerbit, tempat penerbit, item, serta lokasi buku disimpan pada perpustakaan. Selanjutnya, pada tabel ini akan dimuat dengan data pemustaka. Beberapa tabel pada tabel entitas bibliografi akan digunakan untuk pengembangan *RESTful* API sesuai dengan kebutuhan aplikasi.



Gambar 3. 3 Relasi Entitas Tabel Bibliografi ELIB

d. Wireframe

Perancangan *wireframe* aplikasi pada penelitian ini bertujuan agar merepresentasikan rencana struktur desain antarmuka dengan tata letak masing-masing fitur pada sistem yang telah ditentukan. *Wireframe* memprioritaskan konsep dan elemen yang penting dalam desain. Desain ini digunakan sebagai panduan awal pengembangan desain dalam membantu memahami konsep , tata letak, dan navigasi



Gambar 3. 4 Wireframe Aplikasi

Pada gambar 3.4 merupakan *wireframe* aplikasi status peminjaman buku yang akan dikembangkan terdiri dari *login*, menu beranda, menu status peminjaman, menu pengembalian buku, dan menu *reminder*. Desain antarmuka serta tata letak yang telah dibuat selanjutnya akan diimplementasikan menjadi UI dari aplikasi pada tahap pengembangan sistem.

3.4.3 *Construction*

Tahapan selanjutnya adalah melakukan pengembangan sistem. perancangan awal adalah membangun *RESTful* API yang bertujuan untuk memungkinkan aplikasi terhubung dengan database ELIB Unila. Database ini kemudian diubah menjadi sebuah *web service* dalam bentuk *RESTful* API. Tujuan utama dari transformasi ini adalah agar aplikasi dapat berinteraksi dengan database tanpa perlu terhubung langsung ke sumber data eksternal. Pengembangan ini menggunakan NodeJS dengan *framework* ExpressJS untuk *RESTful* API didukung dengan *software* Visual Studio Code dan Postman.

Sebelum mengimplementasikan *RESTful* API, dilakukan analisis tabel pada database ELIB Unila. Fokus analisis ini terutama pada tabel yang berkaitan dengan sirkulasi perpustakaan Unila dan tabel yang memiliki relasi dengan fungsi ini. Hal ini penting untuk memastikan bahwa data yang dibutuhkan dalam aplikasi dapat diakses dan dimanipulasi dengan efektif. Adapun hasil dari analisis tabel pada database ELIB yang akan digunakan adalah Tabel CCirculation (data terkait peminjaman buku), CMcirculation (data terkait history peminjaman buku), CPatron (data *user*), dan CAccount (denda), yang kemudian pada tabel tersebut dikembangkan API untuk mendapatkan data sesuai dengan kebutuhan aplikasi android.

Setelah *RESTful* API dibuat, selanjutnya proses pengembangan sistem berlanjut dengan perancangan antarmuka pengguna. Ini melibatkan pembuatan desain sistem berdasarkan *use case* dan desain *wireframe* yang telah disiapkan. Aplikasi ini dikembangkan menggunakan bahasa pemrograman Android, yaitu Kotlin. Android Studio dipilih sebagai editor kode untuk mengembangkan aplikasi Android. Tahap selanjutnya dilakukan implementasi penjadwalan notifikasi. Ini bertujuan untuk memberikan pengingat kepada pengguna mengenai tanggal pengembalian buku.

Setelah tahap-tahap di atas selesai, pengembangan sistem akan terus berlanjut dengan implementasi kode program dan pengujian pada tahap akhir pengembangan aplikasi. Semua langkah ini bertujuan untuk memastikan bahwa aplikasi yang dihasilkan dapat berinteraksi dengan database dengan baik, memiliki antarmuka yang intuitif, dan memberikan manfaat bagi pemustaka terkait informasi layanan sirkulasi status peminajaman buku di perpustakaan di Unila.

3.4.4 *Cutover*

Tahapan implementasi merupakan tahap terakhir aplikasi diluncurkan kepada pengguna. Pada tahap ini, pengguna memiliki kemampuan untuk memberikan umpan balik terhadap sistem yang telah dibuat. *Usability testing* digunakan untuk melakukan pengujian kegunaan dari sistem yang dikembangkan. Ini diperlukan untuk mengetahui pengalaman pengguna dari desain dan sistem yang sudah dibuat. Adapun pengujian dilakukan terhadap responden yaitu pemustaka pada

Perpustakaan Universitas Lampung. Dalam pengujian *usability* terhadap responden, pertama responden akan diberikan beberapa skenario dalam menjalankan aplikasi, kemudian responden akan menilai sesuai dengan *USE Questionnare* yang diberikan.

Pengujian ini akan menghasilkan evaluasi efektivitas dengan mengukur sejauh mana produk dapat mencapai tujuan pengguna, meningkatkan desain dengan memperbaiki desain antarmuka pengguna dan fungsionalitas produk, mengukur kepuasan pengguna melalui kuisioner yang menunjukkan bagaimana pengguna merasa terkait pengalaman mereka dan apakah mereka puas atau tidak, dan memastikan bahwa perubahan yang diusulkan. Jika sistem gagal melakukan pengujian, pengembangan akan dilakukan kembali atau konstruksi akan diperbaiki.

V. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan hasil dan pembahasan yang telah disampaikan, maka kesimpulan yang diperoleh adalah sebagai berikut :

1. Berhasil dibangun *RESTful* API dari Sistem LMS Elib Universitas Lampung pada layanan sirkulasi perpustakaan untuk kebutuhan aplikasi berbasis android. *RESTful* API yang dibangun menghasilkan Sembilan API yaitu: *Authentication (login* dan *logout), User, Circulation History, Circulation Status, Circulation Account, Book Title, Book Author,* dan *Book Item* untuk diintegrasikan dengan Aplikasi Status Peminjaman Buku.
2. Berhasil dikembangkannya sebuah aplikasi Status Peminjaman Buku berbasis Android dengan nama Perpusta menggunakan bahasa pemrograman Kotlin yang memiliki fitur *login,* fitur informasi peminjaman buku, fitur informasi pengembalian buku, fitur notifikasi, dan fitur *logout*.
3. Hasil Pengujian Aplikasi Status Peminjaman buku berbasis Android menggunakan *blackbox testing* dengan 6 skenario tes mendapatkan hasil pengujian yang sesuai dengan yang diharapkan. Aplikasi dibangun dengan target Android 13 (API level 33) sehingga dapat berjalan dan tersedia bagi pengguna pada perangkat yang menjalankan Android OS yang sama atau lebih tinggi dari level API target.
4. Aplikasi dikembangkan menggunakan metode *Rapid Application Development* (RAD) dengan dilakukan iterasi sebanyak 3 kali pada masa waktu pengembangan selama 77 hari. Metode ini cocok digunakan untuk pengembangan pada penelitian.

5. Hasil dari *USE Questionnaire* yang dilakukan pada aplikasi sudah memenuhi fungsi utamanya berdasarkan pengalaman *user* mengetahui penerimaan *user* terhadap aplikasi sesuai dengan standar kebutuhan user. Nilai akhir kelayakan sebesar 85%. nilai ini berada pada rentang kategori 81-100 pada kategori kelayakan yang berarti aplikasi sangat layak digunakan. USE menekankan pada 3 dimensi pengujian yaitu *usefulness* (kegunaan), *satisfaction* (kepuasan), dan *ease of use* (kemudahan pengguna). Hasil yang diperoleh pada masing masing dimensi yaitu sebesar 86% pada aspek *usefulness*, 84% pada aspek *ease of use*, 84% pada aspek *ease of learning*, dan 86% pada aspek *satisfaction*.
6. Berdasarkan hasil pengujian performance *RESTful* API menggunakan metode *Stress Test* dengan menguji 9 bidang *endpoint* API secara paralel, Tingkat kinerja *RESTful* API akan optimal apabila diakses oleh 9000 *request* dalam periode ramp up 10 detik dan *loop count* adalah 1 dalam satu waktu yang sama. Server apielib mengalami penurunan kinerja apabila diakses oleh lebih dari 90000 *request*.

5.2 Saran

Beradasarkan penelitian yang dilakukan, terdapat beberapa saran yang dapat dilakukan pada pengembangan selanjutnya adalah sebagai berikut :

1. Melakukan pengembangan fitur update data user pada aplikasi seperti dapat memperbaharui nama, alamat, nomor hp, dan foto profile dan fitur *search book* untuk menampilkan informasi koleksi buku di perpustakaan.
2. Menyesuaikan fitur authentication untuk login dengan menggunakan username dan password SSO mahasiswa sehingga memungkinkan akses yang lebih aman dan efisien bagi pengguna.

DAFTAR PUSTAKA

[1] V. Sahfitri, "Prototype E-Katalog Dan Peminjaman Buku Perpustakaan Berbasis Mobile," *J. Sisfokom (Sistem Inf. dan Komputer)*, vol. 8, no. 2, pp. 165–171, Aug. 2019, doi: 10.32736/sisfokom.v8i2.665.

[2] A. Asari *et al.*, *Manajemen Perpustakaan*. Get Press, 2022.

[3] J. Hansen, T. M. Grønli, and G. Ghinea, "Cloud to device push messaging on android: A case study," *Proc. - 26th IEEE Int. Conf. Adv. Inf. Netw. Appl. Work. WAINA 2012*, pp. 1298–1303, 2012, doi: 10.1109/WAINA.2012.96.

[4] I. Kurniawan, Humaira, and F. Rozi, "REST API Menggunakan NodeJS pada Aplikasi Transaksi Jasa Elektronik Berbasis Android," *JITSI J. Ilm. Teknol. Sist. Inf.*, vol. 1, no. 4, pp. 127–132, 2020, doi: 10.30630/jitsi.1.4.18.

[5] We Are Social & Hootsuite, "Digital Data Indonesia 2023." [Online]. Available: https://datareportal.com/reports/digital-2023-indonesia

[6] M. Moskala and I. Wojda, *Android Development with Kotlin*. Packt Publishing, 2017.

[7] E. Rahma, *Akses dan Layanan Perpustakaan: Teori dan Aplikasi*. Kencana, 2018.

[8] M. R. Yusuf and H. Hayatuddiniyah, "ANALISIS PERUBAHAN LAYANAN SIRKULASI PERPUSTAKAAN PERGURUAN TINGGI DI MASA PANDEMI CORONAVIRUS DISEASES 2019 (COVID-19) (Studi Kasus di Perpustakaan UIN Sunan Kalijaga Yogyakarta)," *Publ. Libr. Inf. Sci.*, vol. 4, no. 2, pp. 16–28, 2021, doi: 10.24269/pls.v4i2.3121.

[9] M. A. Muhammad and M. Mardiana, "Library Automation Systems Integration (Case: ELIB and SLiMS)," *Insist*, vol. 1, no. 1, p. 60, 2016, doi: 10.23960/ins.v1i1.21.

[10] R. Islam, R. Islam, and T. Mazumder, "Mobile Application and Its Global Impact," *Int. J. Eng. Technol. IJET-IJENS*, vol. 10, no. December, p. 14, 2020.

[11] A. Sasongko, A. Mustopa, and D. Risdiansyah, "Perancangan Prototipe Aplikasi Mobile Ikatan Alumni (Studi Kasus Universitas Bina Sarana Informatika)," *J. Sist. dan Teknol. Inf.*, vol. 9, no. 3, p. 307, 2021, doi: 10.26418/justin.v9i3.47096.

[12] I. Y. Supardi, *Koleksi Program Tugas Akhir dan Skripsi dengan Android*. Elex Media Komputindo, 2017.

[13] N. Ebel, *Mastering Kotlin: Learn advanced Kotlin programming techniques to build apps for Android, iOS, and the web*. Packt Publishing, 2019.

[14] N. S. Sibarani, G. Munawar, and B. Wisnuadhi, "Analisis Performa Aplikasi Android Pada Bahasa Pemrograman Java dan Kotlin. In Prosiding Industrial Research Workshop and National Seminar," *Ind. Res. Work. Natl. Semin.*, no. July, 2018.

[15] S. Gautam, "Deno-A new Node.js?," *Haaga-Helia Univ. Appl. Sci.*, pp. 1–48, 2021,

[16] npmjs.com, "About npm." Accessed: Dec. 16, 2023. [Online]. Available: https://www.npmjs.com/

[17] Nasution, "Implementasi Mongo Db, Express Js, React Js Dan Node Js (Mern) Pada Pengembangan Aplikasi Formulir, Kuis, Dan Survei Online," *Informatics Eng.*, pp. 1–160, 2021,

[18] P. Rosso, V. Basile, R. Martínez, E. Métais, and F. Meziane, *Natural Language Processing and Information Systems: 27th International Conference on Applications of Natural Language to Information Systems, NLDB 2022, Valencia, Spain, June 15–17, 2022, Proceedings*. in Lecture Notes in Computer Science. Springer International Publishing, 2022.

[19] D. Lee, "Designing the Multimedia Push Framework for Mobile Applications," *Int. J. Adv. Sci. Technol.*, vol. 32, pp. 117–124, 2011,

[20] P. Stevens, J. Whittle, and G. Booch, *UML 2003 -- The Unified Modeling Language, Modeling Languages and Applications: 6th International Conference San Francisco, CA, USA, October 20-24, 2003, Proceedings*. in Lecture Notes in Computer Science. Springer Berlin Heidelberg, 2003.

[21] C. A. Pamungkas, *Pengantar dan Implementasi Basis Data*. Deepublish, 2017.

[22] R. Mistry and S. Misner, *Introducing Microsoft SQL Server 2014*. in Introducing. Pearson Education, 2014.

[23] 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)," *J. Pengemb. Teknol. Inf. dan Ilmu Komput.*, vol. 4, no. 2, pp. 680–689, 2020,

[24] M. Masse, *REST API Design Rulebook*. in Oreilly and Associate Series. O'Reilly Media, 2011.

[25] Open Web Application Security Project, "REST Security." [Online]. Available: https://cheatsheetseries.owasp.org/cheatsheets/REST_Security_Cheat_Sheet.html

[26] H. Subramanian and P. Raj, *Hands-On RESTful API Design Patterns and Best Practices: Design, develop, and deploy highly adaptable, scalable, and secure RESTful web APIs*. Packt Publishing, 2019.

[27] J. Martin, *Rapid Application Development*. in The James Martin productivity series. Macmillan Publishing Company, 1991.

[28] R. T. Futrell, D. F. Shafer, and L. Shafer, *Quality Software Project Management*, no. v. 1. in Quality Software Project Management. Prentice Hall, 2002.

[29] N. L. A. S. Ginasari, K. S. Wibawa, and N. K. A. Wirdiani, "Pengujian Stress Testing API Sistem Pelayanan dengan Apache JMeter," *J. Ilm. Teknol. dan Komput.*, vol. 2, no. 3, 2021.

[30] T. A. S. Foundation, "'Apache JMeter' The Apache Software Foundation." Accessed: Nov. 01, 2023. [Online]. Available: http://jmeter.apache.org/

[31] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for Experimental Design, Measurement, Simulation, and Modeling*. Wiley, 1991.

[32] K. A. Prasetia, S. Rizqika Akbar, and R. Primananda, "Implementasi Lingkungan Test pada Moodle dengan Apache JMeter," vol. 6, no. 12, pp. 5719–5725, 2022.

[33] A. Abran, A. Khelifi, W. Suryn, and A. Seffah, "Consolidating the ISO usability models," *Proc. 11th Int. Softw. Qual. Manag. Conf.*, no. January, pp. 23–25, 2003.

[34] A. Nordeen, *Learn Software Testing in 24 Hours: Definitive Guide to Learn Software Testing for Beginners*. Guru99, 2020.

[35] A. M. Lund, "Measuring usability with the USE questionnaire," *Usability interface*, vol. 8, no. 2, pp. 3–6, 2001.

[36] D. P. Sugiyono, *Metode Penelitian Kuantitatif, Kualitatif, dan R&D*. Bandung: CV Alfabeta, 2012.

[37] A. Purwinarko, M. Subagja, and A. Yanuarto, "The Evaluation of Final Assignment System Using the USE Questionnaire Approach," *Sci. J. Informatics*, vol. 7, no. 2, pp. 2407–7658, 2020.

[38] J. Nielsen, "Why You Only Need to Test With 5 Users," *Nielsen Nourman Gr.*, 2000.

[39] L. Nielsen and S. Madsen, "The usability expert's fear of agility - An empirical study of global trends and emerging practices," *Nord. 2012 Mak. Sense Through Des. - Proc. 7th Nord. Conf. Human-Computer Interact.*, no. May, pp. 261–264, 2012, doi: 10.1145/2399016.2399057.

[40] I. K. Mohidin, "Penerapan Teknologi Rest Api Pada Aplikasi Perpustakaan Digital Politeknik Gorontalo," *J. Technopreneur*, vol. 10, no. 1, pp. 34–39, 2022, doi: 10.30869/jtech.v10i1.922.

[41] K. Muludi, "APLIKASI PERPUSTAKAAN DIGITAL PADA PERPUSTAKAN JURUSAN Jurnal Pepadun," *J. Pepadun*, vol. 2, no. 11, pp. 101–106, 2021.

[42] N. Fadilah, A. Ikhwan, and M. Alda, "Pengembangan Sistem Informasi Perpustakaan Pada Dinas Perpustakaan dan Kearsipan Kota Medan Berbasis Android," *Juli*, vol. 6, pp. 298–306, 2023.

[43] R. Bangun, A. Perpustakaan, D. Berbasis, and M. Menggunakan, "Design and Development of Mobile-Based Digital Library Application Using Flutter," vol. 18, no. 1, pp. 353–362, 2023.

[44] R. Rizal and A. Rahmatulloh, "Restful Web Service Untuk Integrasi Sistem Akademik Dan Perpustakaan Universitas Perjuangan," *J. Ilm. Inform.*, vol. 7, no. 01, p. 54, 2019, doi: 10.33884/jif.v7i01.1004.

[45] R. Christhover, Sugiyatno, and Herlawati, "Penerapan Rest Api Menggunakan Retrofit Untuk Sistem Informasi Film Berbasis Android (Studi Kasus: Sinopsis Film)," *J. Students' Res. Comput. Sci.*, vol. 3, no. 2, pp. 159–170, 2022, doi: 10.31599/jsrcs.v3i2.1393.

[46] A. Soni and V. Ranga, "API features individualizing of web services: REST and SOAP," *Int. J. Innov. Technol. Explor. Eng.*, vol. 8, no. 9 Special Issue, pp. 664–671, 2019, doi: 10.35940/ijitee.I1107.0789S19.

[47] T. Iqbal and M. Wali, "IDOL: Retrofit-Kotlin Service-Based Online Digital Library Application and College Open Data Repository," *Int. J. Softw. Eng. Comput. Sci.*, vol. 2, no. 1, pp. 1–8, 2022, doi: 10.35870/ijsecs.v2i1.760.

[48] D. Hariyanto, M. B. Triyono, and T. Köhler, "Usability evaluation of personalized adaptive e-learning system using USE questionnaire," *Knowl. Manag. E-Learning*, vol. 12, no. 1, pp. 85–105, 2020, doi: 10.34105/j.kmel.2020.12.005.