



**INSTITUT TEKNOLOGI INDONESIA**

**RANCANG BANGUN APLIKASI TABUNGAN KURBAN  
BERBASIS *MOBILE***

**TUGAS AKHIR**

**FACHRI ALVIANSYAH  
1151500073**

**INFORMATIKA  
TANGERANG SELATAN  
FEBRUARI 2021**



**INSTITUT TEKNOLOGI INDONESIA**

**RANCANG BANGUN APLIKASI TABUNGAN KURBAN  
BERBASIS *MOBILE***

Laporan Tugas Akhir ini di susun sebagai  
salah satu syarat untuk memperoleh gelar sarjana komputer  
Pada Program Studi Informatika  
Institut Teknologi Indonesia

**FACHRI ALVIANSYAH  
1151500073**

**INFORMATIKA  
TANGERANG SELATAN  
FEBRUARI 2021**

**HALAMAN PERNYATAAN ORISINALITAS**

**Skripsi ini adalah hasil karya saya sendiri, dan  
semua sumber baik dikutip maupun dirujuk telah  
saya nyatakan dengan benar.**

<b>Nama</b>	<b>: Fachri Alviansyah</b>
<b>NRP</b>	<b>: 1151500073</b>
<b>Tanda Tangan</b>	
<b>Tanggal</b>	<b>: 18 Februari 2021</b>

### HALAMAN PENGESAHAN

Tugas Akhir ini diajukan oleh:

Nama : Fachri Alviansyah  
 NRP : 1151500073  
 Program Studi : Informatika  
 Judul Skripsi : Rancang Bangun Aplikasi Tabungan Kurban Berbasis *Mobile*

Telah berhasil dipertahankan di depan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar sarjana Teknik pada Program Studi Informatika Institut Teknologi Indonesia.

Pembimbing

: Muhamad Ramli, S.T (.....)

Penguji 1

: Melani Indriasari, M.Kom

Penguji 2

: Suryo Bramasto, M.T

Penguji 3

: Sunarto, M.Kom

Ketua Program Studi Informatika



(Dra. Sulistyowati, M.Kom)

## KATA PENGANTAR

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, saya dapat menyelesaikan Tugas Akhir ini. Penulisan Tugas Akhir ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Teknik pada Program Studi Informatika Institut Teknologi Indonesia. Saya menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak, dari masa perkuliahan sampai pada penyusunan Tugas Akhir ini, sangatlah sulit bagi saya menyelesaikan Tugas Akhir ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Ibu Dra. Sulistyowati, M.Kom sebagai Ketua Program Studi Informatika yang telah mengarahkan saya dalam penyusunan Tugas Akhir ini.
2. Bapak Muhamad Ramli, S.T sebagai Dosen Pembimbing yang telah menyediakan waktu, tenaga dan pikiran untuk mengarahkan saya dalam penyusunan Tugas Akhir ini.
3. Bapak Sunarto, M.Kom sebagai Dosen Penasehat Akademik yang telah membimbing saya dari awal perkuliahan sampai dengan penyusunan Tugas Akhir.
4. Bang Rengga Herlangga, S.Kom sebagai Alumni Informatika 2013 yang telah memberikan semangat, bantuan dan saran.
5. Orang tua dan keluarga saya yang telah memberikan bantuan dukungan material dan moral.
6. Keluarga besar HMIF 2013, HMIF 2015 dan HMIF 2017 yang telah memberikan semangat, bantuan dan saran.

Akhir kata, saya berharap Tuhan Yang Maha Esa berkenan membalaq segala kebaikan semua pihak yang telah membantu. Semoga Tugas Akhir ini membawa manfaat bagi pengembangan ilmu.

Tangerang Selatan, 21 November 2020



Fachri Alviansyah

**HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI  
TUGAS AKHIR / SKRIPSI UNTUK KEPENTINGAN AKADEMIS**

---

Sebagai civitas akademika Institut Teknologi Indonesia, saya yang bertanda tangan di bawah ini:

Nama : Fachri Alviansyah  
NRP : 1151500073  
Program Studi : Informatika  
Jenis karya : Tugas Akhir / Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Institut Teknologi Indonesia Hak Bebas Royalti Non Eksklusif (*Non-exclusive Royalty Free Right*) atas Karya Ilmiah yang berjudul:

**Rancang Bangun Aplikasi Tabungan Kurban Berbasis Mobile**

---

Beserta perangkat yang ada (jika diperlukan). Dengan Hak Bebas Royalti Non Eksklusif ini Institut Teknologi Indonesia berhak menyimpan, mengalih media/pemformatan dan mengelola dalam bentuk pangkalan data (*database*), merawat, dan mempublikasikan Tugas Akhir saya selama tetap mencantumkan nama saya sebagai penulis/pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di: Tangerang Selatan  
Pada Tanggal 18 Februari 2021  
Yang Menyatakan,



(Fachri Alviansyah)

## ABSTRAK

<b>Nama</b>	: Fachri Alviansyah
<b>Program Studi</b>	: Informatika
<b>Judul</b>	: Rancang Bangun Aplikasi Tabungan Kurban Berbasis <i>Mobile</i>
<b>Dosen Pembimbing</b>	: Muhamad Ramli, S.T

Kendala masyarakat Muslim saat mengumpulkan uang untuk membeli hewan kurban terasa begitu sulit, terkadang uang tabungan digunakan untuk keperluan lain sehingga tidak bisa menjaga uang tabungannya dan uang tersebut tidak ada penambahan yang pasti. Tujuan dari tugas akhir ini adalah membuat aplikasi tabungan kurban digital berbasis *mobile* untuk memudahkan masyarakat Muslim membeli hewan kurban. Pembuatan aplikasi Qurbanku menggunakan bahasa pemrograman Dart dan *framework* Flutter. Dalam rancangan *backend* menggunakan REST API sebagai penghubung antar aplikasi dan *server*. Fitur yang digunakan untuk mengelola data berupa *GET*, *POST*, *PUT* dan *DELETE*. REST API dibangun menggunakan bahasa pemrograman PHP (*Hypertext Preprocessor*) dengan *framework* *CodeIgniter*. Implementasi aplikasi Qurbanku terdapat beberapa fitur unggulan yaitu, registrasi menggunakan *email*, menabung, tarik saldo tabungan, dan pembelian hewan kurban.

**Kata Kunci:** Dart, Flutter, Masyarakat Muslim, Qurbanku, REST API, Tabungan

## ABSTRACT

*The constraints of the Muslim community when collecting money to buy Qurban animals are so hard, sometimes the savings money is used for other purposes so that it cannot keep the savings and the money is no definite addition. The purpose of this final task is to create a mobile based digital Qurban savings application to make it easier for Muslims to buy Qurban animals. Qurbanku Application creation using Dart programming language and Flutter framework. In the backend design use the REST API as a link between the application and the server. Features used to manage data in the form of GET, POST, PUT and DELETE. REST API is built using PHP programming language (Hypertext Preprocessor) with CodeIgniter framework. Qurbanku application implementation has several excellent features, namely, registration using email, saving, withdrawing savings balance, and purchasing Qurban animals.*

**Keywords:** Animal, Dart, Flutter, Muslim Community, Qurban, REST API

## DAFTAR ISI

HALAMAN JUDUL .....	i
HALAMAN PERNYATAAN ORISINALITAS .....	ii
HALAMAN PENGESAHAN .....	iii
KATA PENGANTAR .....	iv
HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI .....	v
ABSTRAK .....	vi
DAFTAR ISI .....	vii
DAFTAR GAMBAR.....	ix
DAFTAR TABEL .....	xii
<b>BAB 1. PENDAHULUAN.....</b>	<b>1</b>
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah .....	2
1.3 Tujuan dan Manfaat.....	2
1.4 Ruang Lingkup .....	2
1.5 Metodologi .....	3
1.6 Sistematika Penulisan.....	3
<b>BAB 2. LANDASAN TEORI .....</b>	<b>5</b>
2.1 Kurban (Qurban) .....	5
2.2 Pengertian Tabungan.....	5
2.3 Sistem Operasi Android .....	6
2.4 Pengertian Dart dan Flutter .....	8
2.4.1 Bahasa Pemrograman Dart.....	8
2.4.2 Framework Flutter.....	9
2.5 <i>Representation State Transfer</i> (REST).....	10
2.6 <i>JavaScript Object Notation</i> (JSON).....	11
2.7 UML ( <i>Unified Modeling Language</i> ) .....	14
2.8 Pengertian <i>Hosting</i> .....	16
2.9 Metode <i>Agile</i> .....	17
<b>BAB 3. ANALISIS DAN PERANCANGAN .....</b>	<b>19</b>
3.1 Analisis Masalah .....	19
3.2 Analisis Kebutuhan Teknis .....	19
3.3 Analisis Fungsional .....	20
3.4 Analisis Aplikasi .....	20

3.4.1 <i>Use Case Diagram</i> .....	20
3.4.2 <i>Deployment Diagram</i> .....	21
3.4.3 <i>Activity Diagram</i> .....	22
3.4.4 <i>Sequence Diagram</i> .....	27
3.4.5 <i>Class Diagram</i> .....	31
3.4.6 Struktur Data .....	31
3.5 Perancangan antarmuka Aplikasi .....	33
3.5.1 Rancangan Menu Navigasi.....	33
3.5.2 Rancangan Logo Aplikasi .....	34
3.5.3 Rancangan Registrasi Aplikasi.....	35
3.5.4 Rancangan <i>Login</i> .....	35
3.5.5 Rancangan Menu Beranda .....	36
3.5.6 Rancangan Menu Nabung .....	36
3.5.7 Rancangan Menu Transaksi .....	37
3.5.8 Transaksi <i>Detail</i> .....	38
3.5.9 Rancangan Menu Profil.....	38
3.5.10 Rancangan <i>Form</i> Biodata <i>User</i> .....	39
3.5.11 Rancangan Menu Transaksi Akses <i>Admin</i> .....	40
3.5.12 Rancangan <i>Approve</i> dan <i>Reject</i> Tarik Tunai akses <i>Admin</i> .....	40
3.5.13 Rancangan <i>Approve</i> dan <i>Reject</i> Tabungan akses <i>Admin</i> .....	41
3.5.14 Rancangan Ubah Kata Sandi.....	42
BAB 4. IMPLEMENTASI DAN PENGUJIAN .....	43
4.1 Implementasi .....	43
4.1.1 Implementasi Kebutuhan Teknis.....	43
4.1.2 Implementasi Pengembangan REST API .....	43
4.1.3 Implementasi Aplikasi Qurbanku.....	60
4.2 Pengujian .....	79
4.2.1 Pengujian REST API menggunakan Postman .....	79
4.2.2 Pengujian Aplikasi menggunakan Metode <i>Black Box</i> .....	92
BAB 5. PENUTUP .....	97
5.1 Kesimpulan.....	97
5.2 Saran.....	97
DAFTAR PUSTAKA.....	98

## DAFTAR GAMBAR

Gambar 2.1 <i>Code</i> Bahasa Dart .....	8
Gambar 2.2 <i>Code framework</i> Flutter .....	9
Gambar 2.3 <i>Code JSON</i> .....	12
Gambar 2.4 <i>Object JSON</i> .....	12
Gambar 2.5 <i>Array JSON</i> .....	13
Gambar 2.6 <i>Value JSON</i> .....	13
Gambar 2.7 <i>String JSON</i> .....	13
Gambar 2.8 Angka JSON .....	14
Gambar 3.1 <i>Use Case Diagram</i> .....	21
Gambar 3.2 <i>Deployment Diagram</i> .....	21
Gambar 3.3 <i>Activity Diagram</i> Qurbanku .....	22
Gambar 3.4 <i>Activity Diagram</i> Registrasi .....	23
Gambar 3.5 <i>Activity Diagram</i> <i>Login</i> .....	23
Gambar 3.6 <i>Activity Diagram</i> Menabung .....	24
Gambar 3.7 <i>Activity Diagram</i> Penarikan Saldo Tabungan .....	25
Gambar 3.8 <i>Activity Diagram</i> Pembelian Hewan Kurban .....	26
Gambar 3.9 <i>Sequence Diagram</i> Qurbanku .....	27
Gambar 3.10 <i>Sequence Diagram</i> Registrasi .....	27
Gambar 3.11 <i>Sequence Diagram</i> <i>Login</i> .....	28
Gambar 3.12 <i>Sequence Diagram</i> Menabung .....	28
Gambar 3.13 <i>Sequence Diagram</i> Penarikan Saldo .....	29
Gambar 3.14 <i>Sequence Diagram</i> Pembelian Hewan Kurban .....	30
Gambar 3.15 <i>Class Diagram</i> .....	31
Gambar 3.16 Menu Navigasi.....	33
Gambar 3.17 Logo Aplikasi Tabungan Kurban .....	34
Gambar 3.18 Rancangan Registrasi Aplikasi .....	35
Gambar 3.19 Rancangan <i>Login</i> Aplikasi.....	35
Gambar 3.20 Rancangan Menu Beranda.....	36
Gambar 3.21 Rancangan Menu Nabung.....	36
Gambar 3.22 Rancang Menu Transaksi .....	37
Gambar 3.23 Transaksi <i>Detail</i> .....	38
Gambar 3.24 Rancangan Menu Profil .....	38
Gambar 3.25 Rancangan <i>Form</i> Biodata <i>User</i> .....	39
Gambar 3.26 Rancangan Tarik Tunai dan Pembelian Hewan Kurban.....	40
Gambar 3.27 Rancangan <i>Approve</i> dan <i>Reject</i> Tarik Tunai <i>Admin</i> .....	40
Gambar 3.28 Rancangan <i>Approve</i> dan <i>Reject</i> Tabungan <i>Admin</i> .....	41
Gambar 3.29 Rancangan Ubah Kata Sandi .....	42
Gambar 4.1 <i>Class Auth</i> .....	44
Gambar 4.2 Fungsi <i>Login</i> .....	44
Gambar 4.3 <i>Check Token</i> .....	45
Gambar 4.4 <i>Class User</i> .....	45
Gambar 4.5 Fungsi <i>Create Account</i> .....	46
Gambar 4.6 Fungsi <i>Update User Detail</i> .....	46
Gambar 4.7 Fungsi <i>All User</i> .....	47
Gambar 4.8 Fungsi <i>Edit Password</i> .....	47
Gambar 4.9 <i>Class Tabungan</i> .....	48

Gambar 4.10 Fungsi Data Tabungan <i>User</i> .....	48
Gambar 4.11 Fungsi Data Tabungan <i>Admin</i> .....	49
Gambar 4.12 Fungsi <i>Count</i> Tabungan <i>Admin</i> .....	49
Gambar 4.13 Fungsi <i>Create</i> Tabungan.....	50
Gambar 4.14 Fungsi <i>Count</i> Tabungan <i>User</i> .....	51
Gambar 4.15 Fungsi <i>Approve</i> .....	52
Gambar 4.16 Fungsi <i>Reject</i> .....	53
Gambar 4.17 Fungsi <i>Create</i> Tarik Tunai.....	54
Gambar 4.18 <i>Approve</i> Tarik Tunai.....	55
Gambar 4.19 <i>Reject</i> Tarik Tunai .....	56
Gambar 4.20 <i>Class Product</i> .....	57
Gambar 4.21 Fungsi Data <i>Product</i> .....	57
Gambar 4.22 <i>Class Transaction</i> .....	58
Gambar 4.23 Fungsi <i>Create Transaction</i> .....	59
Gambar 4.24 Tampilan Registrasi .....	60
Gambar 4.25 Validasi Email .....	61
Gambar 4.26 Tampilan <i>Login</i> .....	61
Gambar 4.27 <i>Login</i> Gagal .....	62
Gambar 4.28 Tampilan Beranda <i>Admin</i> .....	63
Gambar 4.29 Tampilan Beranda <i>User</i> .....	63
Gambar 4.30 Menu Nabung .....	64
Gambar 4.31 Media <i>Upload</i> .....	65
Gambar 4.32 <i>Popup</i> Pesan Nabung .....	65
Gambar 4.33 Tampilan Menu Transaksi <i>Off</i> .....	66
Gambar 4.34 Tampilan Tarik Tunai dan Pembelian Hewan Kurban .....	67
Gambar 4.35 Transaksi Detail Pembelian Hewan Kurban.....	67
Gambar 4.36 Tarik Tunai .....	68
Gambar 4.37 Tampilan Menu Profil.....	69
Gambar 4.38 Biodata .....	69
Gambar 4.39 Ubah Kata Sandi .....	70
Gambar 4.40 Riwayat Transaksi <i>User</i> .....	70
Gambar 4.41 Widget Notifikasi Qurbanku .....	71
Gambar 4.42 Notifikasi Sukses .....	72
Gambar 4.43 Notifikasi Gagal .....	72
Gambar 4.44 <i>Login Admin</i> .....	73
Gambar 4.45 Menu Profil .....	73
Gambar 4.46 Memproses Transaksi User .....	74
Gambar 4.47 Tampilan Pencairan Transaksi.....	74
Gambar 4.48 Menu Riwayat Transaksi <i>Admin</i> .....	75
Gambar 4.49 Nasabah Qurbanku .....	76
Gambar 4.50 Memproses Pencairan Saldo Tabungan.....	76
Gambar 4.51 <i>Popup Approve</i> Pencairan Transaksi .....	77
Gambar 4.52 <i>Popup Reject</i> Pencairan Transaksi.....	78
Gambar 4.53 <i>Popup Setuju/Approve Tabungan</i> .....	78
Gambar 4.54 Popup Tolak/ <i>Reject</i> Tabungan.....	79
Gambar 4.55 Postman <i>Login</i> .....	80
Gambar 4.56 Postman <i>Check Token</i> .....	81
Gambar 4.57 Postman <i>All User</i> .....	82

Gambar 4.58 Postman <i>Create Account</i> .....	82
Gambar 4.59 Postman <i>Edit Password</i> .....	83
Gambar 4.60 Postman <i>Update User Detail</i> .....	83
Gambar 4.61 Postman Data Tabungan <i>User</i> .....	85
Gambar 4.62 Postman Data Tabungan <i>Admin</i> .....	85
Gambar 4.63 Postman <i>Count Tabungan Admin</i> .....	86
Gambar 4.64 Postman <i>Create Tabungan</i> .....	86
Gambar 4.65 Postman <i>Count Tabungan</i> .....	87
Gambar 4.66 Postman <i>Approve</i> .....	87
Gambar 4.67 Postman <i>Reject</i> .....	88
Gambar 4.68 Postman <i>Create Tarik Tunai</i> .....	88
Gambar 4.69 Postman <i>Approve Tarik Tunai</i> .....	89
Gambar 4.70 Postman <i>Reject Tarik Tunai</i> .....	89
Gambar 4.71 Postman Data <i>Product</i> .....	90
Gambar 4.72 Postman <i>Create Transaction</i> .....	91

**DAFTAR TABEL**

Tabel 2.1 Versi-versi Android .....	7
Tabel 3.1 <i>User</i> .....	31
Tabel 3.2 <i>User Detail</i> .....	32
Tabel 3.3 Tabungan .....	32
Tabel 3.4 <i>Product</i> .....	32
Tabel 3.5 <i>Transaction Head</i> .....	33
Tabel 3.6 <i>Transaction Detail</i> .....	33
Tabel 4.1 REST API <i>Class Auth</i> .....	80
Tabel 4.2 REST API <i>Class User</i> .....	81
Tabel 4.3 REST API <i>Class Tabungan</i> .....	84
Tabel 4.4 REST API <i>Class Product</i> .....	90
Tabel 4.5 REST API <i>Class Transaction</i> .....	91
Tabel 4.6 Pengujian Aplikasi Menggunakan Metode <i>Black Box</i> .....	92

## **BAB 1**

### **PENDAHULUAN**

#### **1.1 Latar Belakang**

Perkembangan teknologi terasa begitu bersahabat kehadirannya. Berbagai kebutuhan dapat terpenuhi dengan mudah, *smartphone* saat ini berkembang dan sangat diminati oleh masyarakat, terutama masyarakat Muslim karena beragam fitur, daya tarik dan manfaat tersendiri bagi penggunanya, sistem operasi *smartphone* yang sangat diminati adalah sistem operasi *Android*. Berbagai keunggulan dari sistem operasi *Android* salah satunya adalah dapat mengubah tampilan *smartphone* sesuai keinginan *user* dan banyak aplikasi yang sudah tersedia di *Play Store* untuk memudahkan *user* terutama aktivitas beribadah.

Ibadah dalam islam adalah pelaksanaan segala macam perbuatan yang diperintahkan oleh agama untuk mengatur hubungan seseorang dengan Allah SWT dan sebagai ujian terhadap kebenaran dan kekuatan iman dalam kehidupan sehari-hari. Salah satu bentuk ibadah dalam Islam yang membawa spirit sosial dan sangat simbolik untuk kesadaran akan kehadiran Allah SWT dalam hidup manusia adalah ibadah kurban. Ibadah kurban bukan hanya ritual persembahan untuk meningkatkan kualitas spiritual dan bukan cara untuk memperoleh kepuasan batin karena sudah naik ke langit, dengan ibadah kurban masyarakat Muslim dapat memperkuat kepekaan sosialnya (Basyir, 2012).

Masyarakat yang ingin berkurban memiliki keluhan karena uang yang terkumpul tidak menyadari digunakan untuk keperluan lain-lain. Selain itu masyarakat tidak bisa menjaga uang tabungannya, sehingga uang tabungan tidak ada penambahan yang pasti. Dengan cara menabung yang rutin tiap bulannya masyarakat merasa terbantu untuk pembelian hewan kurban di hari raya Idul Adha yang akan datang, akan tetapi masih memiliki beberapa kekurangan yaitu uang tabungan yang sering digunakan maupun di ambil sehingga uang tersebut tidak digunakan untuk berkurban, selanjutnya terkait harga hewan kurban yang tiap tahunnya semakin mahal.

Tabungan kurban digital merupakan solusi alternatif guna mengumpulkan uang dengan cara menabung di aplikasi, uang yang sudah terkumpul di aplikasi tidak bisa di ambil maupun digunakan sampai waktu hari raya Idul Adha, dengan cara ini masyarakat tidak tergoda untuk menggunakan uang tabungan yang sudah terkumpul untuk digunakan keperluan selain kurban.

## 1.2 Rumusan Masalah

Berdasarkan latar belakang dapat dirumuskan masalah untuk Tugas Akhir yaitu, bagaimana membuat aplikasi tabungan kurban digital berbasis *mobile*?

## 1.3 Tujuan dan Manfaat

Tujuan yang akan dicapai dalam tugas akhir ini adalah membuat aplikasi tabungan kurban digital berbasis *mobile* yang dapat digunakan oleh masyarakat Muslim.

Manfaat dari tugas akhir ini adalah:

- a) Adanya alternatif tabungan kurban digital bagi masyarakat Muslim untuk melakukan ibadah kurban.
- b) Membantu masyarakat Muslim mengumpulkan uang menggunakan aplikasi untuk membeli hewan kurban.

## 1.4 Ruang Lingkup

Agar pelaksanaan tugas akhir ini menjadi lebih terarah dan mendapatkan hasil yang lebih spesifik, maka sistem yang dirancang memiliki batasan pada ruang lingkup pembahasan sebagai berikut:

- a) Melakukan registrasi nasabah.
- b) Dapat melakukan transaksi menabung.
- c) Menampilkan saldo dan riwayat transaksi tabungan.
- d) Proses pembayaran melalui transfer bank.
- e) Penarikan saldo dilakukan sebulan menuju hari raya Idul Adha.
- f) Hasil tabungan dapat digunakan membeli hewan kurban melalui aplikasi atau menarik saldo tabungan dan membeli hewan di tempat lain.
- g) Pengelola aplikasi seluruhnya dilakukan oleh *admin*.
- h) Aplikasi diakses menggunakan *smartphone android*.
- i) Aplikasi dirancang menggunakan bahasa pemrograman Dart dan *framework Flutter*.
- j) Cakupan area pembelian hewan kurban di wilayah Tangerang Selatan.

## 1.5 Metodologi

Dalam rancang bangun aplikasi ini, metodologi perancangan aplikasi yang digunakan adalah.

### a) Studi Literatur

Bentuk pencarian informasi dengan cara membaca / mengambil informasi dari makalah, jurnal ilmiah, buku dan juga memanfaatkan internet sebagai sumber informasi yang memiliki keterkaitan dengan *Mobile App, Application Programming Interface (API)* dan *framework Flutter*.

### b) Analisis Kebutuhan dan Perancangan Perangkat Sistem

Melakukan analisis masalah, analisis fungsional dan analisis kebutuhan teknis terhadap kebutuhan sistem, mengidentifikasi kebutuhan informasi berdasarkan hasil observasi yang telah dilakukan dan merancang arsitektur sistem seperti perancangan aplikasi, dan perancangan antarmuka.

### c) Implementasi

Mengimplementasikan hasil rancangan pada tahap perancangan aplikasi menggunakan bahasa pemrograman Dart dan *framework Flutter* untuk tampilan antarmuka, selain itu penghubung antara *mobile* dan *server* menggunakan REST API yang diimplementasikan menggunakan bahasa pemrograman *Hypertext Preprocessor (PHP)* dengan *framework CodeIgniter*, dan implementasi *database* menggunakan *MySQL*.

### d) Pengujian

Pengujian meliputi uji coba aplikasi yang sudah diimplementasikan, fitur utama dari aplikasi yakni, menabung uang atau mengambil uang untuk membeli hewan kurban dan pembelian hewan kurban secara langsung di aplikasi. Hasil pengujian menggunakan aplikasi Postman dan metode *Black Box* untuk di setiap fungsi yang berada di dalam aplikasi.

## 1.6 Sistematika Penulisan

Secara garis besar penulisan laporan tugas akhir ini terbagi dalam beberapa bab yang tersusun antara lain sebagai berikut:

## **BAB I PENDAHULUAN**

Bab ini berisikan latar belakang, rumusan masalah, tujuan, ruang lingkup, metodologi dan sistematika penulisan.

## **BAB II LANDASAN TEORI**

Pada bab ini diuraikan tentang teori yang berhubungan dengan menunjang penulisan tugas akhir.

## **BAB III ANALISIS DAN PERANCANGAN**

Pada bab berisi penjelasan tentang analisis dari sistem yang ingin dibuat, diagram aliran data dan permasalahan yang ada serta solusi rancangan sistem yang diusulkan. Analisis sistem yang dilakukan menggunakan *Unified Modelling Language* (UML) dan perancangan antarmuka.

## **BAB IV IMPLEMENTASI DAN PENGUJIAN**

Pada bab ini dijelaskan tentang hasil implementasi dari analisis dan perancangan sistem yang telah dilakukan sebelumnya dengan hasil implementasi dan memaparkan pengujian terhadap sistem yang telah dikembangkan dengan fase-fase dan metode *Agile* yang relevan.

## **BAB V PENUTUP**

Pada bab ini berisi tentang kesimpulan yang diperoleh dari hasil pengujian sistem, serta saran perbaikan dan pengembangan sistem ke depannya.

## **BAB 2**

### **LANDASAN TEORI**

#### **2.1 Kurban (Qurban)**

Dalam bahasa arab hewan kurban di sebut juga *udhiyah* atau *adh-dhahiyah* dengan bentuk jamaknya Al-Adhaahi. Kata diambil dari kata *dhuha*. Seakan kata itu berasal dari kata yang menunjukkan waktu diisyaratkan penyembelihan hewan kurban dan dengan kata itu, hari penyembelihan dinamakan *Yaumul Adha*. Kurban disembahkan sebagai bentuk *taqarrub* pada Allah yaitu mendekatkan diri pada-Nya dan bukti nyata Islam adalah agama yang *kaffah* dan sangat memperhatikan hubungan sosial, salah satunya dengan diisyaratkan kurban (Abu Abdullah Muhammad bin Ismail Al Bukhari, n.d.).

Kurban sebagian dari rasa syukur seorang hamba atas nikmat yang telah diberikan Allah SWT kepada-Nya dan dengan ikhlas dia melaksanakan kurban lalu membagikan kepada mereka yang pantas menerimanya. Ada tiga golongan yang berhak menerima daging kurban yaitu, *shohibul kurban* atau keluarga yang berkurban, teman, kerabat dan tetangga di rumah, yang terakhir adalah golongan orang – orang fakir dan miskin.

Kurban atau *udhiyyah* jamak dari *dhahiyah* adalah penyembelihan hewan di pagi hari. Yang dimaksudkan di sini adalah mendekatkan diri atau beribadah kepada Allah SWT dengan cara menyembelih hewan tertentu pada hari raya Idul Adha dan tiga hari *tasyiq* berikutnya yaitu 11, 12, 13 *Dzulhijjah* (Hadi et al., 1994).

Perintah berkurban diwahyukan Allah SWT kepada Nabi Muhammad SAW dan untuk seluruh umat Islam berlaku sampai akhir zaman, perintah berkurban mulai pada tahun kedua hijrah bersamaan dengan perintah mengerjakan shalat *sunnat* dua hari raya Idul Fitri dan Idul Adha.

#### **2.2 Pengertian Tabungan**

Menabung adalah tindakan yang dianjurkan oleh Islam, karena dengan menabung berarti seseorang muslim mempersiapkan diri untuk masa yang akan datang sekaligus untuk menghadapi hal-hal yang tidak di inginkan (Apridayanti, 2013).

Simpanan adalah dana yang dipercayakan oleh nasabah kepada bank syariah berdasarkan akad *wadi'ah* atau akad lain yang tidak bertentangan dengan prinsip syariah dalam bentuk tabungan, giro, atau bentuk lainnya. Sedangkan investasi adalah dana yang dipercayakan oleh nasabah kepada bank syariah berdasarkan akad *mudharabah* atau akad

lain yang tidak bertentangan dengan prinsip syariah dalam bentuk tabungan, deposito, dan bentuk lainnya (Apridayanti, 2013).

Seseorang yang ingin menabung di bank syariah dapat memilih antara akad *wadi'ah* atau *mudharabah*. Tabungan yang menerapkan akad *wadi'ah* tidak mendapat keuntungan dari bank, karena sifatnya hanya titipan. Akan tetapi, bank tidak dilarang jika ingin memberikan semacam bonus atau hadiah (Apridayanti, 2013).

Sedangkan tabungan yang menggunakan akad *mudharabah* adalah mengikuti prinsip – prinsip *mudharabah*, yaitu keuntungan dari dana yang dibagi hasilkan dibagi antara bank dengan nasabah, adanya tenggang waktu antara dana disimpan dan pembagian keuntungan (Apridayanti, 2013).

Tabungan merupakan salah satu bentuk simpanan yang diperlukan oleh masyarakat untuk menyimpan uangnya, karena tabungan merupakan jenis simpanan yang dapat dibuka dengan persyaratan yang sangat mudah. Nasabah hanya menyediakan *Softcopy* KTP, SIM, dan identitas lainnya untuk dapat membuka rekening tabungan. Selain itu, setoran awal buka rekening tabungan juga rendah sehingga mudah terjangkau oleh masyarakat yang ingin menabung.

### 2.3 Sistem Operasi Android

Android adalah sistem operasi perangkat *mobile* berbasis *Linux* yang mencangkup sistem, *middleware*, dan aplikasi. Android merupakan *platform* terbuka bagi para pengembangnya. Awalnya, *Google Inc* membeli *Android Inc*, pendatang baru yang membuat peranti lunak untuk ponsel. Kemudian untuk mengembangkan android, dibentuk *Open Handset Alliance*, konsorsium dari 34 perusahaan peranti keras, peranti lunak dan telekomunikasi, termasuk *Google, HTC, Intel, Motorola, Qualcomm, T-Mobile dan Nvidia* (Supriyadinatha, 2014).

Android merupakan sistem operasi yang berisi *middleware* serta aplikasi-aplikasi dasar. Basis sistem operasi android yaitu kernel *linux* 2.6 yang telah diperbarui untuk *mobile device*. Pengembangan aplikasi android menggunakan bahasa pemrograman *java*, yang mana konsep-konsep pemrograman *java* berhubungan dengan Pemrograman berbasis Objek (OOP). Selain itu pengembangan aplikasi android membutuhkan *Software Development Kit (SDK)* yang disediakan android, SDK ini memberi jalan bagi *programmer* untuk mengakses *Application Programming Interface (API)* pada android (Supriyadinatha, 2014).

Pada saat rilis pertama android versi beta tanggal 5 November 2007, android bersama *Open Handset Alliance* menyatakan mendukung pengembangan standar terbuka pada perangkat seluler. Di pihak lain, *Google* merilis kode-kode android di bawah lisensi *Apache*, sebuah lisensi perangkat lunak dan standar terbuka perangkat seluler. Di dunia ini terdapat dua jenis distributor sistem operasi Android. Pertama yang mendapat dukungan penuh dari *Google* atau *Google Mail Service* (GMS) dan kedua adalah yang benar-benar bebas distribusi nya tanpa dukungan langsung Google atau di kenal sebagai *Open Handset Distribution* (OHD) (Supriyadinatha, 2014).

Sejak pertama kali diperkenalkan pada tahun 2007, android telah mengalami perkembangan yang begitu pesat, ditandai dengan banyaknya versi-versi di sistem operasi android. Tabel 2.1 menunjukkan beberapa versi yang ada beserta tanggal rilis nya (Hansun et al., 2016):

**Tabel 2.1** Versi-versi Android

Versi	API Level	Tanggal Rilis	Code Name
1.0	1	23 Sep 2008	-
1.1	2	09 Feb 2009	-
1.5	3	30 Apr 2009	Cupcake
1.6	4	15 Sep 2009	Donut
2.0/2.0.1/2.1	5/6/7	26 Okt 2009	Eclair
2.2-2.2.3	8	20 Mei 2010	Froyo
2.3-2.3.2/2.3.3-2.3.7	9/10	06 Des 2010	Gingerbread
3.0/3.1/3.2-3.2.6	11/12/13	22 Feb 2011	Honeycomb
4.0-4.0.2/4.0.3-4.0.4	14/15	19 Okt 2011	Ice Cream Sandwitch
4.1-4.1.2/4.2-4.2.2/4.3-4-3-1	16/17/18	27 Juni 2012	Jelly Bean
4.4	19/20	31 Okt 2013	KitKat
5.0-5.0.2/5.1-5.1.1	21/22	12 Nov 2014	Lollipop
6.0-6.0.1	23	05 Okt 2015	Marshmallow
7.0/7.1-7.1.1-7.1.2	24/25	22 Agu 2016	Nougat
8.0/8.1.0	26/27	21 Agu 2017	Oreo
9.0	28	06 Agu 2018	Pie
10	29	13 Sep 2019	Android Q

## 2.4 Pengertian Dart dan Flutter

### 2.4.1 Bahasa Pemrograman Dart

Dart adalah bahasa pemrograman yang dikembangkan oleh Google untuk kebutuhan dalam membuat aplikasi android / *mobile, front-end, web*, IoT (*Internet of Things*), *back-end* (CLI), dan *game*. Dart sudah menerapkan konsep pemrograman berorientasi objek (OOP) dimana struktur kode berada dalam *class* yang di dalamnya berisi *method* maupun variabel. Dart merupakan *C-style syntax* sehingga mekanisme dart mirip seperti bahasa pemrograman *C, java, javascript, dan swift* (Doavers Development Team, 2016).

```
void main(){
    print('hello word');
}
```

**Gambar 2.1** Code Bahasa Dart

Dart merupakan bahasa pemrograman baru yang dikembangkan oleh google pada tahun 2007 namun baru di rilis versi stabilnya (versi 2.0) pada tahun 2018. Dart merupakan bahasa pemrograman yang menarik dimana bahasa ini dapat dilaksanakan dan dapat digunakan untuk menulis skrip sederhana atau aplikasi yang memiliki fitur lengkap (Doavers Development Team, 2016). Berikut adalah beberapa konsep bahasa pemrograman Dart(Dart, n.d.):

- a) Sesuatu yang dapat ditempatkan dalam variabel adalah sebuah *objek*, dan setiap objek adalah turunan dari sebuah *class*. Angka genap, fungsi, dan *null* merupakan objek. Semua objek diwarisi dari *class object* .
- b) Tipe anotasi bersifat opsional karena Dart dapat menyimpulkan tipe pada *code number* menjadi tipe *integer*.
- c) Dart memiliki kondisi ekspresi dan pernyataan bersyarat seperti? *expr1 : expr2* memiliki nilai *expr1* atau *expr2*, dengan pernyataan *if-else* yang tidak memiliki nilai. Pernyataan sering berisi satu atau lebih ekspresi, tetapi ekspresi tidak bisa langsung berisi pernyataan.
- d) Bahasa Dart dapat melaporkan dua jenis masalah, yaitu peringatan dan kesalahan. Peringatan hanyalah indikasi bahwa kode tidak berfungsi, tetapi tidak mencegah program dijalankan. Kesalahan dapat berupa waktu kompilasi atau waktu proses. Kesalahan waktu kompilasi mencegah kode dieksekusi sama sekali. kesalahan runtime menghasilkan pengecualian yang dimunculkan saat kode dijalankan.

## 2.4.2 Framework Flutter

Flutter adalah SDK untuk pengembangan aplikasi *mobile* yang dikembangkan oleh google, sama seperti *react native*, *framework* ini dapat digunakan untuk membuat atau mengembangkan aplikasi *mobile* yang dapat berjalan pada iOS dan Android (Wicaksono, 2019).

```
import 'package:flutter/material.dart';

void main() => runApp(MyApp());

class MyApp extends StatelessWidget {
@override
Widget build(BuildContext context) {
    return MaterialApp(
        title: 'First App',
        home: Scaffold(
            appBar: AppBar(
                title: Text('Welcome to Flutter'),
            ),
            body: Center(
                child: Text('Hello World'),
            ),
        ),
    );
}
}
```

**Gambar 2.2** Code framework Flutter

Versi pertama flutter dikenal sebagai *Sky* dan berjalan pada sistem operasi android. Diresmikan pada perhelatan *Dart Developer Summit* 2015, dengan tujuan untuk mampu membuat grafis secara konsisten pada 120 bingkai per detik (Wicaksono, 2019). Berikut adalah komponen-komponen utama Flutter:

### a) *Flutter Engine*

*Flutter engine*, ditulis terutama dengan bahasa pemrograman C++, memberikan dukungan tingkat rendah menggunakan *library* grafik *Sky* milik Google. Selain itu, *flutter engine* juga berinteraksi dengan perkakas pengembangan perangkat lunak (SDK) spesifik-serambi (*platform-specific*) seperti yang disediakan oleh Android dan iOS (Wicaksono, 2019).

### b) *Foundation library*

*Foundation library*, ditulis dengan bahasa pemrograman dart, menyediakan fungsi dan *class-class* dasar yang digunakan untuk membangun aplikasi

menggunakan Flutter, seperti API untuk berkomunikasi dengan *engine* (Wicaksono, 2019).

### c) *Design Widget spesifik*

*Framework flutter* berisi dua *set widget* yang disesuaikan dengan bahasa desain tertentu. *Widget Material Design* merupakan bahasa desain Google dengan nama yang sama, sedangkan *widget Cupertino* meniru desain iOS milik *Apple* (Wicaksono, 2019).

## 2.5 *Representation State Transfer (REST)*

Salah satu jenis dari *Web Service* adalah REST atau *RESTful (Representation State Transfer)*. REST sendiri memungkinkan sistem *request* dapat mengakses dan memanipulasi teks yang direpresentasikan dari sebuah *web service*. *Web Service API* yang menggunakan REST disebut dengan *RESTful API*. Tidak seperti jenis *Web Service* lainnya, *RESTful API* tidak memiliki standar yang resmi untuk notasinya dikarenakan REST merupakan sebuah arsitektur (Tidwell, 2000).

Metode REST *web service* menerapkan konsep perpindahan antar *state*. *State* yang dimaksud dapat digambarkan apabila *browser* melakukan permintaan suatu *web*, maka *server* akan melakukan pengiriman *state* halaman *web* yang sekarang ke *browser* (Fauziah, 2014). Ide dasar dari metode REST adalah menggunakan metode HTTP untuk menghubungkan aplikasi dibandingkan dengan menggunakan mekanisme yang kompleks, seperti CORBA, RCP dan SOAP (Sutanta & Mustofa, 2012). REST berfokus utama pada interaksi sumber daya dan mengubah *state*, bukan berfokus pada mengirim dan menerima pesan seperti pada *web service* berbasis SOAP (Subhiyakto et al., 2018). Dalam penggunaannya, REST API terbukti lebih cepat dalam transfer data dari pada metode lain yang serupa, dalam hal ini SOAP (*System Object Access Protocol*) (Prabowo Pudjo, 2011).

REST didefinisikan sebagai seperangkat prinsip arsitektur yang digunakan untuk membangun *web service* yang berfokus pada sumber daya sistem, termasuk bagaimana sumber daya *states* ditunjukkan dan di transfer melalui HTTP oleh berbagai *client* yang ditulis ke dalam bahasa-bahasa yang berbeda. Selain itu, REST menuju navigasi melalui *link-link* HTTP untuk melakukan aktivitas-aktivitas tertentu sehingga seakan-akan terjadi perpindahan *state* satu sama lain (Fauziah, 2014).

Metode REST didasari oleh empat prinsip antara lain (Subhiyakto et al., 2018):

1. Identifikasi *resources* untuk *Uniform Resource Identifier* (URI): REST *web service* menyediakan sejumlah *resource* yang mengidentifikasi tujuan dari interaksi dengan para *client*. *Resource* diidentifikasi tujuan dari interaksi dengan para *client*. *Resource* diidentifikasi oleh URI yang dapat mengakomodasi alamat ruang untuk *resource* dan *service discovery*.
2. Keseragaman di dalam *resource*: setiap *resource* di dalam REST dimanipulasi oleh empat macam operasi, yaitu *GET*, *POST*, *PUT* dan *DELETE* untuk membaca, memperbarui, membuat dan menghapus.
3. Penggunaan *hyperlink* untuk interaksi *stateful*: semua interaksi dengan *resource* bersifat *stateless*. Interaksi *stateful* adalah salah satu di mana adanya transfer *state* secara eksplisit. Ada sejumlah cara yang dapat dilakukan untuk melakukan pertukaran *state*, seperti menulis ulang URL. *State* juga dapat ditanamkan di dalam pesan *response*, sehingga dapat digunakan untuk referensi kembali di waktu mendatang.
4. *Message* dengan *self-description*: *resource* pada REST tidak terikat sehingga dapat melakukan akses terhadap berbagai macam konten dengan format PDF, JSON, HTML, XML, JPEG, *Plain Text* dan lainnya. REST didasarkan pada operasi-operasi yang bersifat universal dan dapat digunakan untuk berbagai macam penyimpanan data dan sistem pencarian. Operasi-operasi ini bisa disebut dengan singkatan CRUD (*Create*, *Read*, *Update*, *Delete*).

## 2.6 JavaScript Object Notation (JSON)

JSON (*JavaScript Object Notation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah di terjemahkan dan dibuat *generate* oleh komputer. Format ini dibuat berdasarkan bagian dari bahasa pemrograman *javascript*, standar ECMA-262 Edisi ke-3 Desember 1999. JSON merupakan format teks yang tidak

bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk *C*, *C++*, *C#*, *Java*, *JavaScript*, *Perl* *Python* dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data (Wicaksono, 2019).

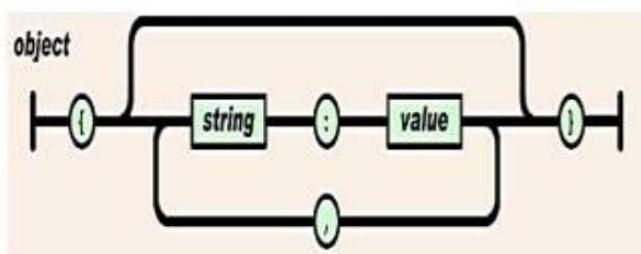
```
{
    "firstname": "Fachri",
    "lastname": "Alviansyah"
}
```

**Gambar 2.3 Code JSON**

Dalam perjalannya, pengembangan untuk mengembangkan sebuah RESTful API menemui kendala dalam penamaan sebuah objek. Notasi objek - objek yang ada haruslah memudahkan pengembang dalam menamainya namun juga harus dikenali oleh komputer. Alasan inilah yang membuat pengembang menggunakan JSON (*JavaScript Object Notation*) sebagai notasi untuk REST *Web Service* dalam aplikasi miliknya. Penggunaan JSON untuk menyokong pembuatan aplikasi *mobile* berbasis REST API juga dianggap lebih baik performanya untuk pengguna *mobile* (Ratulangi et al., 2014). Berikut adalah penjelasan bentuk-bentuk dari JSON:

#### a) Objek

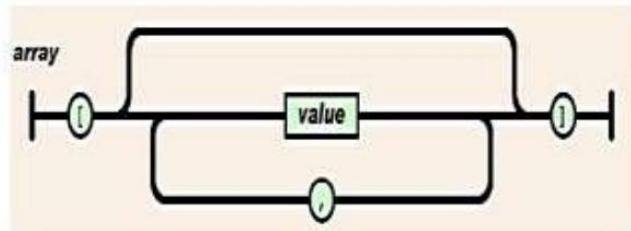
Objek adalah sepasang nama/nilai yang tidak berurutan. Objek dimulai dengan { (kurung kurawal buka) dan diakhiri dengan } (kurung kurawal tutup). Setiap nama diikuti dengan : (titik dua) dan setiap pasang nama/nilai dipisahkan oleh , (koma) (Wicaksono, 2019).



**Gambar 2.4 Object JSON**

### b) Larik

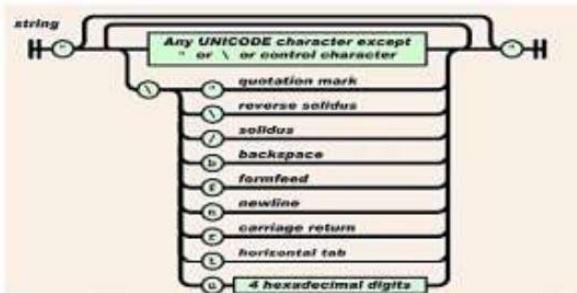
Larik adalah kumpulan nilai yang berurutan. Larik dimulai dengan [ (kurung kotak buka) dan diakhiri dengan ] (kurung kotak tutup). Setiap nilai dipisahkan oleh , (koma) (Wicaksono, 2019).



**Gambar 2.5 Array JSON**

### c) Value

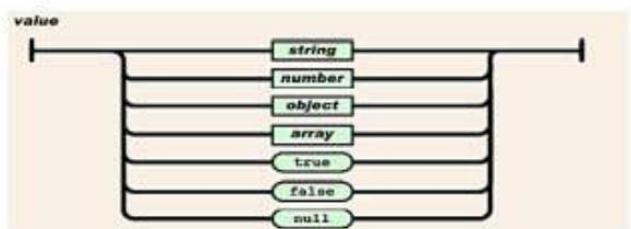
Nilai (*value*) dapat berupa sebuah string dalam tanda kutip ganda, atau angka, *true*, *false*, *null*, sebuah objek dan sebuah larik. Struktur-struktur tersebut dapat disusun bertingkat (Wicaksono, 2019).



**Gambar 2.6 Value JSON**

### d) String

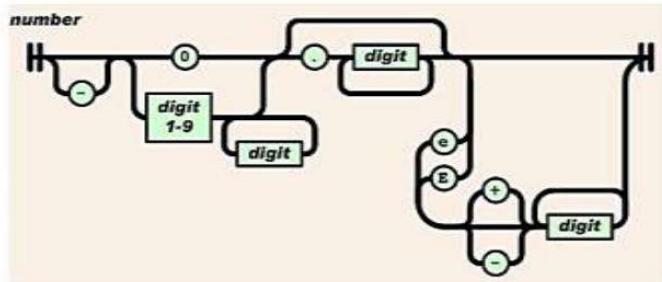
*String* adalah kumpulan dari nol atau lebih karakter *Unicode*, yang dibungkus dengan tanda kutip ganda. Di dalam *string* dapat digunakan *backslash escapes* “\” untuk membentuk karakter khusus. Sebuah karakter mewakili karakter tunggal pada *string*. *String* dangan mirip dengan C atau Java (Wicaksono, 2019).



**Gambar 2.7 String JSON**

### e) Angka

Angka sangat mirip dengan angka di *C* atau *Java*, kecuali format untuk oktal dan heksadesimal tidak digunakan (Wicaksono, 2019).



**Gambar 2.8** Angka JSON

### f) Spasi

Spasi kosong (*white space*) dapat disisipkan di antara pasangan tanda-tanda tersebut, kecuali beberapa detail *encoding* yang secara lengkap dipaparkan oleh bahasa pemrograman yang bersangkutan (Wicaksono, 2019).

## 2.7 UML (*Unified Modeling Language*)

Merupakan himpunan struktur dan teknik pemodelan desain program berorientasi objek. UML adalah metodologi untuk mengembangkan sistem OOP dan sekelompok perangkat *tool* untuk mendukung pengembangan sistem tersebut. UML juga merupakan bahasa standar untuk mendokumentasikan, menspesifikasi, dan mengembangkan sistem perangkat lunak (Wicaksono, 2019).

Analisis desain berorientasi objek atau disebut OOAD adalah metode analisis yang memeriksa *requirements* dari sudut pandang kelas-kelas dan objek yang ditemui dalam permasalahan yang menggarahkan arsitektur *software* yang didasarkan pada manipulasi objek-objek sistem atau sub sistem. OOAD merupakan cara untuk menjabarkan masalah menggunakan model yang dibuat menurut konsep kenyataan (Wicaksono, 2019). Konsep OOAD mencangkup analisis dan desain sebuah sistem dengan pendekatan objek diantaranya:

### a) *Object*

Sebuah objek memiliki keadaan yang disebut *state*. *State* dari sebuah objek adalah kondisi dari objek atau himpunan keadaan yang menggambarkan objek tersebut. *State* dinyatakan dengan nilai dari atribut objeknya. Atribut adalah nilai

internal suatu objek yang mencerminkan karakteristik objek, kondisi sesaat, koneksi dengan objek lain dan identitas.

**b) *Class***

*Class* adalah himpunan objek yang sejenis atau mempunyai atribut, dan ber-relasi umum dengan objek lain dan semantik umum. *Class* adalah abstraksi dari objek dalam dunia nyata. *Class* menetapkan spesifikasi perilaku dan atribut dari objek tersebut.

**c) *Black Box***

Sebuah objek merupakan kotak hitam, konsep tersebut merupakan dasar dari implementasi objek. *Black box* berisi kode dan data, prosesnya yaitu enkapsulasi dan *message*.

**d) *Asosiasi dan Agregasi***

Asosiasi adalah hubungan yang berkaitan antara sejumlah objek. Sedangkan agregasi adalah bentuk khusus sebuah asosiasi yang menggambarkan seluruh bagian pada suatu objek merupakan bagian dari objek yang lain.

UML juga memberikan standar penulisan sebuah sistem *blue print*, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema *database*, dan komponen-komponen yang diperlukan dalam sistem. Berikut ini merupakan beberapa bagian diagram UML sebagai berikut:

**1. *Use Case Diagram***

*Use Case* menggambarkan eksternal *view* dari sistem yang akan kita buat modelnya. Model *use case* dapat dijabarkan dalam diagram *use case*, diagram *use case* tidak identik dengan model karena model lebih luas dari diagram (Prabowo Pudjo, 2011). *Use case* harus mampu menggambarkan urutan aktor yang menghasilkan nilai terukur.

**2. *Activity Diagram***

Diagram *activity* menunjukkan aktivitas sistem dalam bentuk kumpulan aksi-aksi, bagaimana masing-masing aksi tersebut dimulai, keputusan yang mungkin terjadi hingga berakhirnya aksi. *Activity diagram* juga dapat menggambarkan proses lebih dari satu aksi dalam waktu bersamaan. Diagram *activity* adalah aktivitas-aktivitas, objek, *state*, transisi *state* dan *event*. Dengan kata lain kegiatan diagram alur kerja menggambarkan perilaku sistem untuk aktivitas (WHITTEN et al., 2004).

### **3. Sequence Diagram**

Secara mudah *sequence diagram* adalah gambaran tahap demi tahap, termasuk kronologi (urutan) perubahan secara logis yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case diagram* (WHITTEN et al., 2004).

### **4. Class Diagram**

*Class* sebagian suatu set objek yang memiliki atribut dan perilaku yang sama, kelas kadang disebut kelas objek (Pooley Rob, 2003). *Class* memiliki tiga area pokok yaitu: Nama, *class* harus mempunyai sebuah nama. Atribut, adalah kelengkapan yang melekat pada *class*. Nilai dari suatu *class* hanya bisa diproses sebatas atribut yang dimiliki. Operasi, adalah proses yang dapat dilakukan oleh sebuah *class*, baik pada *class* itu sendiri atau kepada *class* lainnya.

## **2.8 Pengertian Hosting**

*Hosting* merupakan tempat atau jasa internet untuk membuat halaman *website* yang telah dibuat menjadi *online* dan bisa diakses oleh orang lain. Sedangkan *hosting* adalah jasa layanan internet yang menyediakan sumber daya *server-server* untuk disewakan sehingga memungkinkan organisasi atau individu menempatkan informasi di internet berupa HTTP, FTP, *Email*, dan DNS (Kurniansyah & Sinurat, 2020).

Ada beberapa jenis layanan *hosting* yaitu *shared hosting*, VPS atau *Virtual Private Server*, *Dedicated Server*, *Colocation Server*.

- a) *Shared Hosting* adalah *server hosting* bersamaan dengan pengguna lain satu dipergunakan oleh lebih dari satu nama *domain*. Artinya dalam satu *server* tersebut dapat beberapa *account* yang dibedakan antara *account* satu dan lainnya dengan *username* dan *password*.
- b) VPS (*Virtual Private Server*) atau juga dikenal sebagai *Virtual Dedicated Server* merupakan proses virtualisasi dari lingkungan *software* sistem operasi yang dipergunakan untuk *server*. Karena lingkungan ini merupakan lingkungan virtual, hal tersebut memungkinkan untuk menginstal sistem operasi yang dapat berjalan di atas sistem operasi lain
- c) *Dedicated Server* adalah penggunaan *server* yang dikhususkan untuk aplikasi yang lebih besar dan tidak bisa dioperasikan dalam *shared hosting* atau *virtual dedicated server*. Dalam hal ini, penyediaan *server* ditanggung oleh perusahaan *hosting* yang biasanya bekerja sama dengan vendor.

- d) *Colocation Server* adalah layanan penyewaan tempat untuk meletakan *server* yang dipergunakan untuk *hosting*. *Server* disediakan oleh pelanggan yang biasanya bekerja sama dengan vendor.

## 2.9 Metode Agile

*Agile development methods* adalah kumpulan metodologi pengembangan perangkat lunak yang didasarkan pada prinsip-prinsip yang sama atau pengembangan sistem jangka pendek dimana memerlukan adaptasi yang cepat dari pengembang terhadap perubahan dalam bentuk apapun, dalam penggunaanya *agile development* memerlukan inovasi dan tanggung jawab yang baik antar tim pengembang dan klien agar kualitas dari perangkat lunak yang dihasilkan bagus dan seimbang (Abrahamsson et al., 2017).

*Agile development* erat kaitan dengan metodologi dan proses pengembangan perangkat lunak, metodologi pengembangan perangkat lunak sendiri sebuah metodologi yang digunakan untuk membuat struktur, rencana dan kontrol pengerjaan suatu proyek sedangkan proses pengembangan perangkat lunak adalah model-model dan metodologi yang digunakan untuk mengembangkan suatu perangkat lunak (Abrahamsson et al., 2017). Berikut beberapa model pada *agile development*:

### a) *Extreme Programming*

*Extreme Programming* merupakan sebuah proses rekayasa perangkat lunak yang cenderung menggunakan pendekatan berorientasi objek dan sasaran dari metode ini adalah tim yang dibentuk dalam skala kecil sampai medium serta metode ini juga sesuai jika tim dihadapkan dengan *requirement* yang tidak jelas maupun terjadi perubahan-perubahan *requirement* secara cepat (Carolina et al., 2019).

### b) *Adaptive software development*

Teknis atau metode ini digunakan untuk membangun *software* dan sistem yang kompleks. Filosofi yang mendasari adaptif *software development* adalah kolaborasi manusia dan tim yang mengatur diri sendiri. sistem kerja adaptif *software development* adalah *collaboration* dan *learning*.

### c) *Scrum*

*Scrum* merupakan *development method* yang paling populer, kegiatan pada metode ini diantaranya: *Sprint Planning*, *Sprint Review*, *Scrum Meeting*. Di setiap individu akan menjelaskan status tugas tim dan apa yang akan dilakukan hari berikutnya (Abrahamsson et al., 2017).

**d) Dynamic System Development Method (DSDM)**

DSDM menekan pada keterlibatan terus pelanggan, metode ini lebih baik digunakan untuk proyek-proyek yang memiliki keterbatasan waktu dan anggaran. Berikut adalah 5 (Lima) *project* siklus metode DSDM:

1. Studi Kelayakan.
2. Studi Bisnis
3. Perulangan Model Fungsional.
4. Perulangan Perancangan dan Pembuatan.
5. Penerapan.

DSDM memberikan pendekatan berulang tambahan dan memberikan beberapa teknik inti yang disebut *time boxing*, *prototyping*, pengujian, *workshop* dan lain-lain. Tujuan utama dari metode DSDM untuk menjaga proyek serta mengendalikan waktu anggaran (Rusdiana, 2018).

**e) Crystal method**

*Crystal Method* dimulai sebelum *agile* manifesto dan merupakan salah satu pendiri metodologi tangkas. Metode ini memiliki tiga prioritas dan tiga properti, diantaranya:

1. Prioritas: Keselamatan, Efisiensi, *Habitability*.
  2. Properti: *Frequent Delivery*, Peningkatan Reflektif, Komunikasi Tertutup.
- Metode *Crystal* adalah keluarga dari *Adaptive*, *Ultra-light* dan *Stretch-to-fit* metodologi dan lebih difokuskan pada orang dari pada proses atau arsitektur.

**f) Feature Driven Development (FDD)**

*Feature driven development* merupakan sebuah model pengembangan perangkat lunak yang berdasarkan pada fitur yang akan dibuat.

## **BAB 3**

### **ANALISIS DAN PERANCANGAN**

#### **3.1 Analisis Masalah**

Beberapa kalangan masyarakat Muslim yang masih kesulitan untuk mengumpulkan uang membeli hewan kurban karena harganya setiap tahun semakin mahal. Masyarakat Muslim perlu menabung untuk meringankan beban membeli hewan kurban saat menjelang hari raya Idul Adha mendatang, dengan cara menabung masyarakat Muslim bisa mengumpulkan uangnya sedikit demi sedikit untuk membeli hewan kurban.

#### **3.2 Analisis Kebutuhan Teknis**

Untuk membuat aplikasi ini, ada beberapa kebutuhan teknis seperti perangkat keras (*hardware*) dan perangkat lunak (*software*):

##### **a) Perangkat Keras (*Hardware*)**

Perangkat keras (*hardware*) yang digunakan untuk implementasi aplikasi adalah sebagai berikut:

- *Processor* Intel Core i5-7200U CPU @2.50GHz
- *Random Access Memory* (RAM): 8 GB
- *Storage* 1 TB
- *Mouse* Logitech M235
- *Cooler Master X-SLIM II*
- *Samsung A71 2020*
- Koneksi Internet

##### **b) Perangkat Lunak (*Software*)**

Perangkat lunak (*Software*) yang digunakan untuk implementasi aplikasi adalah sebagai berikut:

- |                      |                       |
|----------------------|-----------------------|
| • Windows 10 64 Bit  | • Postman             |
| • Xampp 3.2.4        | • MySQL Workbench 6.3 |
| • MYSQL 5.7.7        | • RESTful API         |
| • Visual Studio Code | • Photoshop CS6       |
| • Android Studio     | • Google Chrome       |

### 3.3 Analisis Fungsional

Analisis kebutuhan fungsional yang terdapat pada aplikasi Tabungan Kurban “Qurbanku” terpecah menjadi dua bagian yaitu *User* dan *Admin*:

#### a) *User*

1. *User* dapat melakukan registrasi akun dengan *email* yang aktif.
2. Setelah *user* melakukan registrasi, *user* dapat melakukan *login* ke aplikasi.
3. *User* dapat melakukan proses menabung.
4. *User* dapat melakukan foto bukti transfer.
5. *User* dapat menampilkan riwayat transaksi.
6. *User* dapat melakukan transaksi penarikan tabungan.
7. *User* dapat melakukan transaksi pembelian hewan kurban.

#### b) *Admin*

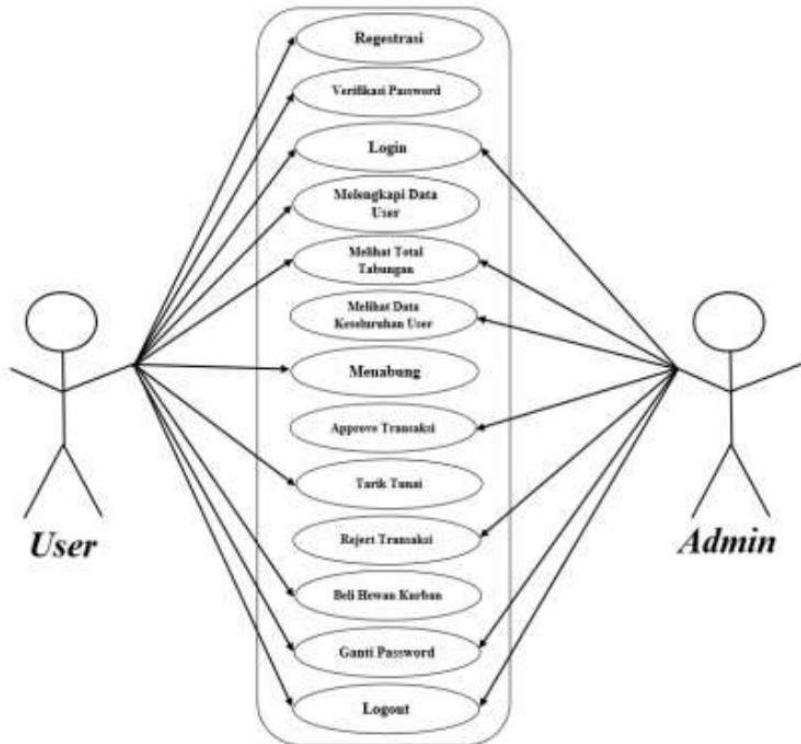
1. *Admin* dapat *login* ke aplikasi.
2. *Admin* dapat menampilkan keuangan tabungan.
3. *Admin* dapat menampilkan data *user*.
4. *Admin* dapat melakukan *approve* transaksi menabung, tarik tunai dan pembelian hewan kurban.
5. *Admin* dapat melakukan *reject* transaksi tabungan, tarik tunai dan pembelian hewan kurban.
6. *Admin* dapat membuka menu transaksi penarikan saldo tabungan jika sudah mendekati hari raya Idul Adha.
7. *Admin* dapat membuka menu fitur transaksi pembelian hewan kurban jika sudah mendekati hari raya Idul Adha.

### 3.4 Analisis Aplikasi

Pada tahap ini dilakukan analisis aplikasi Qurbanku, sistem menggunakan UML (*Unified Modeling Language*) seperti *use case* diagram, *deployment* diagram, *activity* diagram, *sequence* diagram, *class* diagram dan struktur data. Berikut adalah penjelasannya:

#### 3.4.1 *Use Case Diagram*

*Use case* diagram menjelaskan mengenai aktor-aktor yang terlibat dengan perangkat lunak yang dibangun serta proses-proses yang ada di dalamnya.

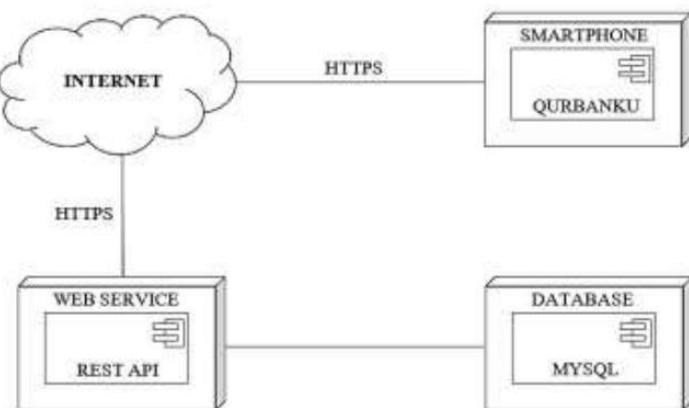


**Gambar 3.1 Use Case Diagram**

Gambar 3.1 Menjelaskan *use case* diagram Qurbanku, terdapat dua aktor yaitu *user* dan *admin* berikut fungsinya: *user* dapat melakukan registrasi, verifikasi *email*, *login*, melengkapi data *user*, melihat total tabungan, menabung, tarik tunai, beli hewan kurban, ganti *password*, *logout* dan *admin* dapat melakukan *login*, melihat total tabung, melihat data keseluruhan *user*, *approve* dan *reject* transaksi, ganti *password* dan *logout*.

### 3.4.2 Deployment Diagram

*Deployment* diagram akan ditunjukkan pada gambar berikut:

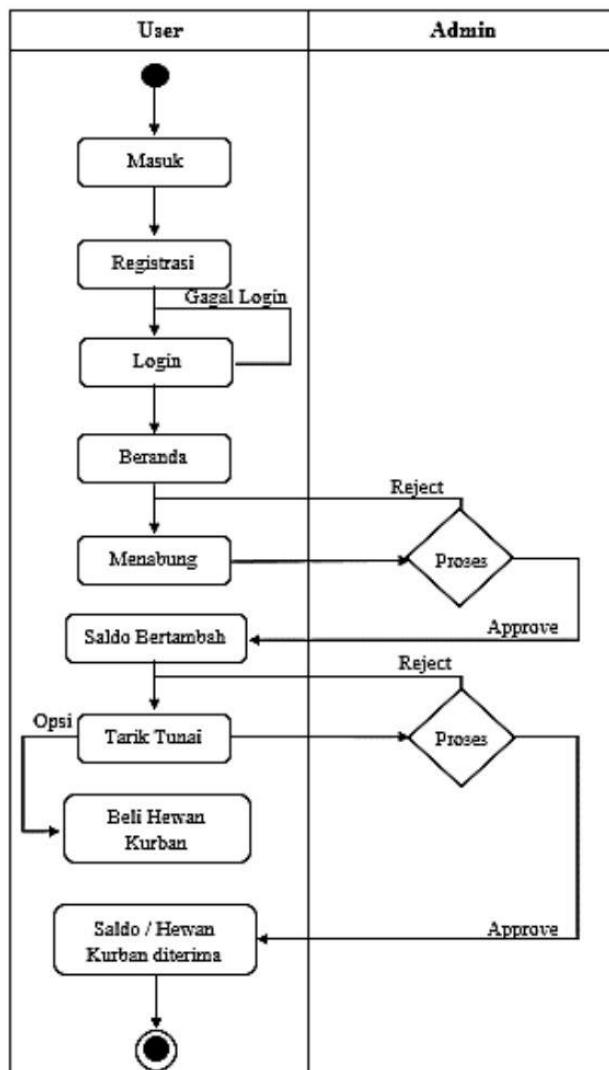


**Gambar 3.2 Deployment Diagram**

Gambar 3.2 Di dalam *deployment diagram* Qurbanku, terdapat beberapa *nodes* dan *artifacts* yaitu, *node smartphone* dengan *artifact* Qurbanku *apps*, *node internet*, *node web service* dengan *artifact* Rest API dan *node database* dengan *artifact* MySQL.

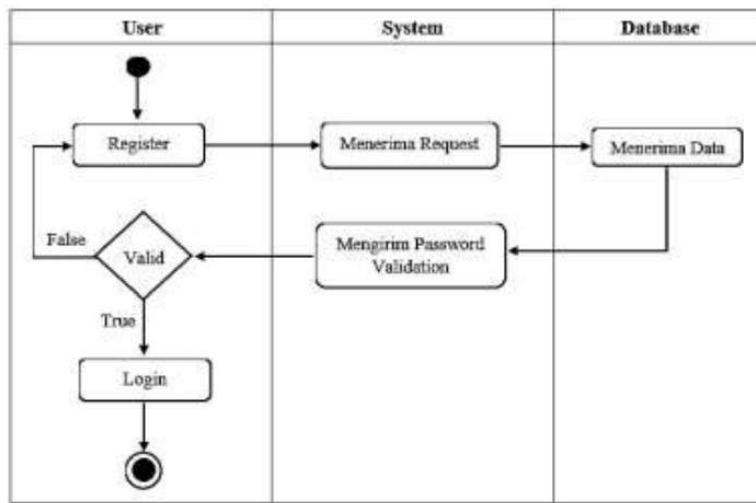
### 3.4.3 Activity Diagram

*Activity* diagram menjelaskan bentuk visual dari alur kerja yang berisi aktivitas dan tindakan, yang juga dapat berisi pilihan, pengulangan dan konkurensi. Berikut penjelasannya:



**Gambar 3.3** *Activity Diagram* Qurbanku

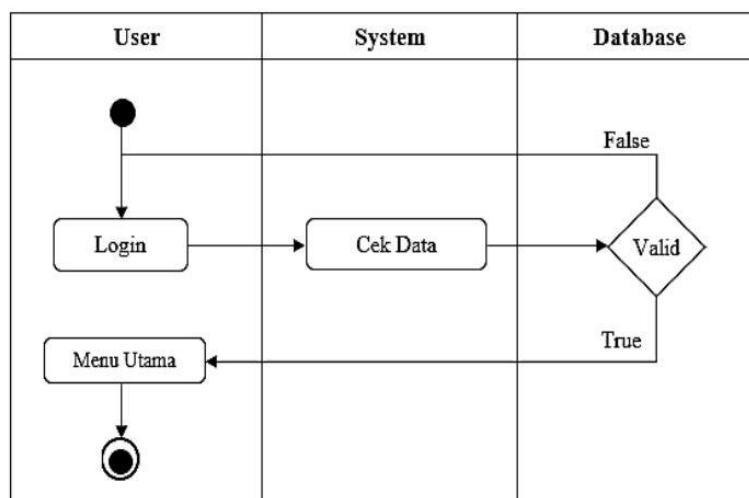
### 3.4.3.1 Activity Diagram Registrasi



Gambar 3.4 Activity Diagram Registrasi

Gambar 3.4 Menjelaskan tentang *activity diagram* registrasi *user*. Saat *user* menjalankan aplikasi, *user* disarankan untuk registrasi dengan cara memasukan nama lengkap dan *email*, aplikasi akan menerima permintaan *user* dan menambahkan data tersebut ke *database*, selanjutnya *user* akan mendapatkan *email* verifikasi *password default*. Jika *valid* benar *user* dapat menggunakan memasukan *password* tersebut dan masuk ke tahap *login*, dan jika *valid* salah *user* tidak akan menerima *email* verifikasi selanjutnya periksa kembali *email* yang dimasukan atau registrasi kembali.

### 3.4.3.2 Activity Diagram Login

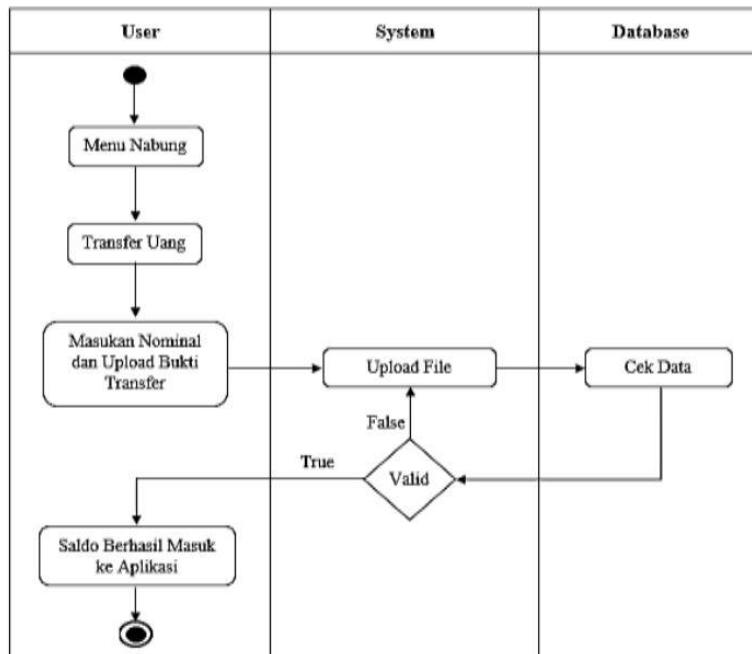


Gambar 3.5 Activity Diagram Login

Gambar 3.5 Menjelaskan *activity diagram* *login*, setelah melakukan registrasi selanjutnya *user login* menggunakan *email* yang didaftarkan dan *password default* yang

dikirim melalui *email*, sistem akan memeriksa data di *database*. Jika *valid* benar *user* masuk ke menu utama dan jika tidak *valid* periksa kembali *email* dan *password*.

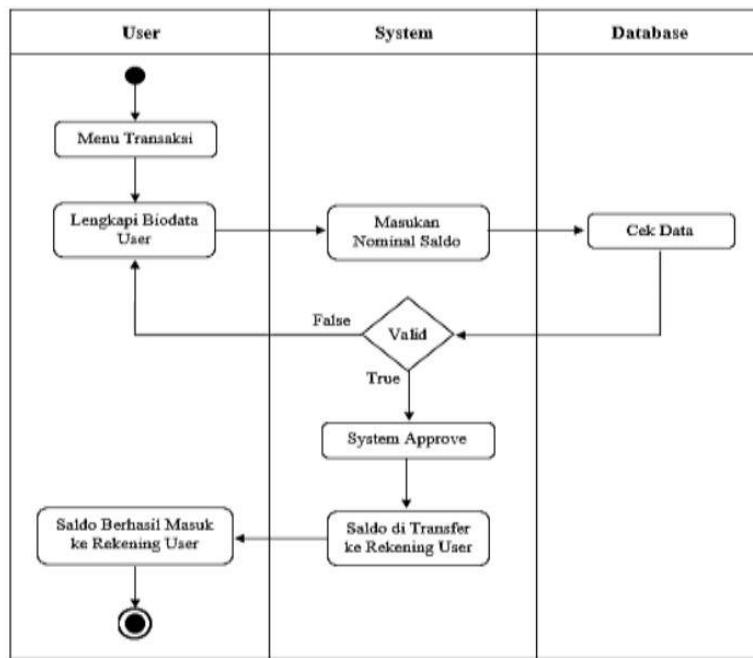
### 3.4.3.3 Activity Diagram Menabung



**Gambar 3.6** Activity Diagram Menabung

Gambar 3.6 Menjelaskan *activity* diagram menabung, langkah utama yang dilakukan *user*, pertama masuk ke menu Nabung, selanjutnya *user* transfer uang ke nomor rekening yang tersedia di aplikasi dengan minimal saldo transfer Rp.50.000.00 (*Lima Puluh Ribu rupiah*) dan simpan bukti transfer, selanjutnya *user* memasukan nominal saldo dan *upload* bukti transfer. Sistem akan meninjau dan memproses data tersebut. Jika *valid* berhasil proses menabung selesai dan menunggu untuk *approve* data dari *admin*, jika data tidak valid silahkan masukan kembali nominal dan *upload* bukti transfer.

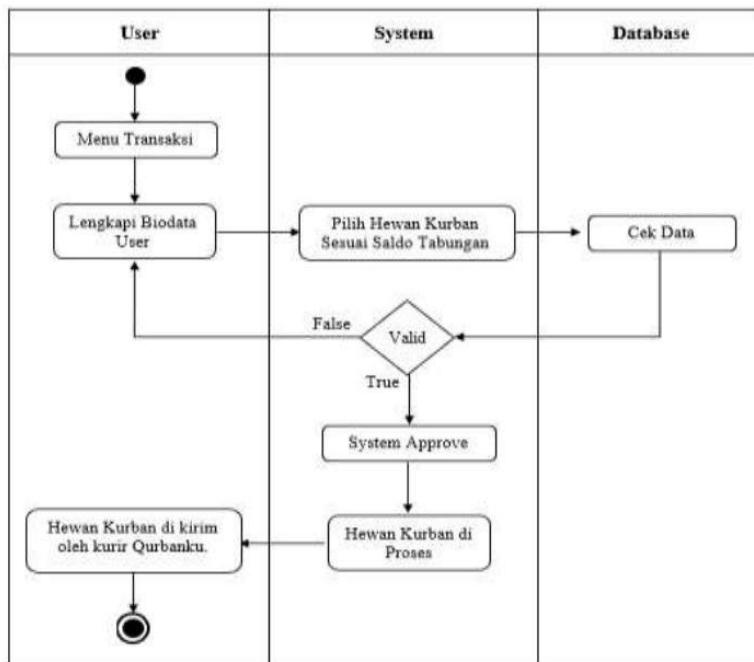
### 3.4.3.4 Activity Diagram Transaksi Penarikan Saldo



**Gambar 3.7** Activity Diagram Penarikan Saldo Tabungan

Gambar 3.7 Menjelaskan *activity* diagram transaksi penarikan saldo tabungan, di dalam fitur atau fungsi transaksi aplikasi Qurbanku *user* wajib menunggu sampai satu bulan sebelum hari raya Idul Adha yang akan datang. Jika waktunya telah tiba maka fitur transaksi bisa digunakan. Berikut alur aktivitasnya: *User* membuka menu Transaksi, dan *user* diwajibkan melengkapi data diri agar proses dapat dijalankan, selanjutnya *user* memasukan nominal saldo yang ingin di tarik/ambil untuk kebutuhan membeli hewan kurban, selanjutnya data dikirim ke *database* dan di cek. Jika valid berhasil data yang dikirim akan di *approve* sistem dan saldo akan di transfer oleh *admin* atau sistem, selanjutnya saldo sudah masuk ke rekening *user*. Jika tidak valid *user* disarankan untuk memasukan kembali data diri dan lakukan cara yang sama.

### 3.4.3.5 Activity Diagram Transaksi Pembelian Hewan Kurban

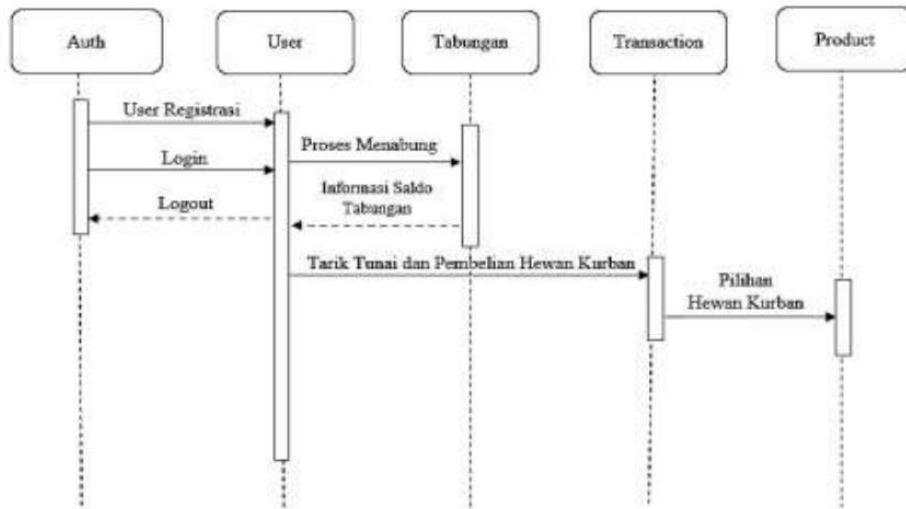


**Gambar 3.8** Activity Diagram Pembelian Hewan Kurban

Gambar 3.8 Menjelaskan *activity* diagram transaksi pembelian hewan kurban, di dalam fitur atau fungsi transaksi aplikasi Qurbanku *user* wajib menunggu sampai satu bulan sebelum hari raya Idul Adha yang akan datang. Jika waktunya telah tiba maka fitur ini bisa digunakan. Berikut alur aktivitasnya: *User* membuka menu Transaksi, dan *user* diwajibkan melengkapi data diri agar proses dapat dijalankan, selanjutnya *user* dapat memilih hewan kurban sesuai dengan nominal tabungan *user*, selanjutnya data dikirim ke *database* dan di cek. Jika valid/berhasil data yang dikirim akan di *approve* sistem dan hewan kurban di persiapkan untuk dikirim ke pemilik hewan kurban atau *user*. jika tidak valid *user* disarankan memeriksa kembali data diri dan melakukan cara yang sama.

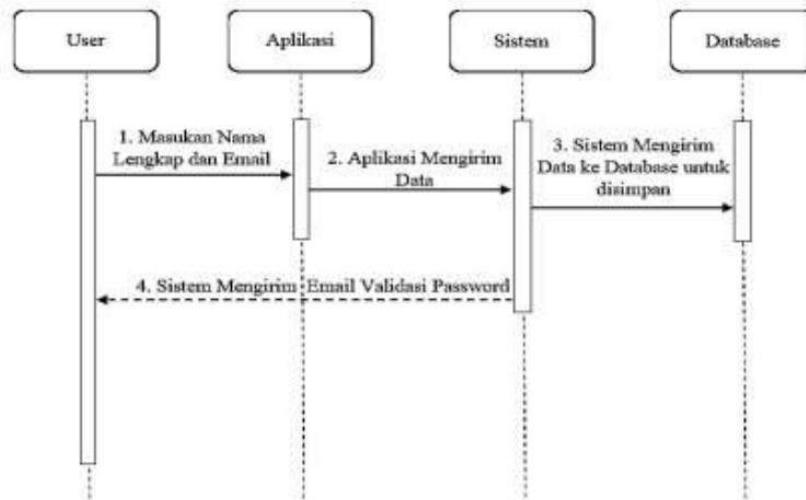
### 3.4.4 Sequence Diagram

*Sequence* diagram menjelaskan diagram urutan untuk menunjukkan interaksi objek yang diatur dalam urutan waktu. Berikut penjelasannya:



Gambar 3.9 Sequence Diagram Qurbanku

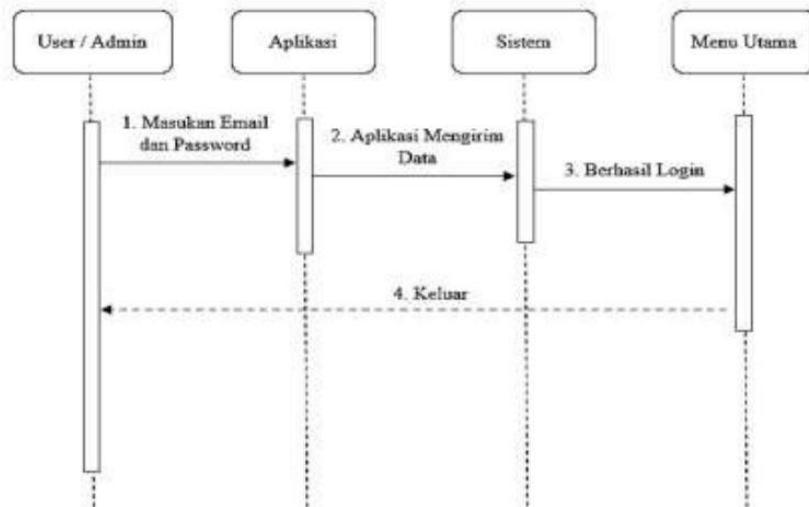
#### 3.4.4.1 Sequence Diagram Registrasi



Gambar 3.10 Sequence Diagram Registrasi

Gambar 3.10 Mengambarkan pesan yang dikirim dari beberapa objek ke objek lainnya. Dalam proses registrasi *user* menjalankan aplikasi dan memasukan nama lengkap dan *email* yang sudah aktif, lalu aplikasi mengirim data ke sistem dan sistem mengirim data untuk disimpan ke *database*, selanjutnya sistem mengirimkan *email validasi password* ke *user*.

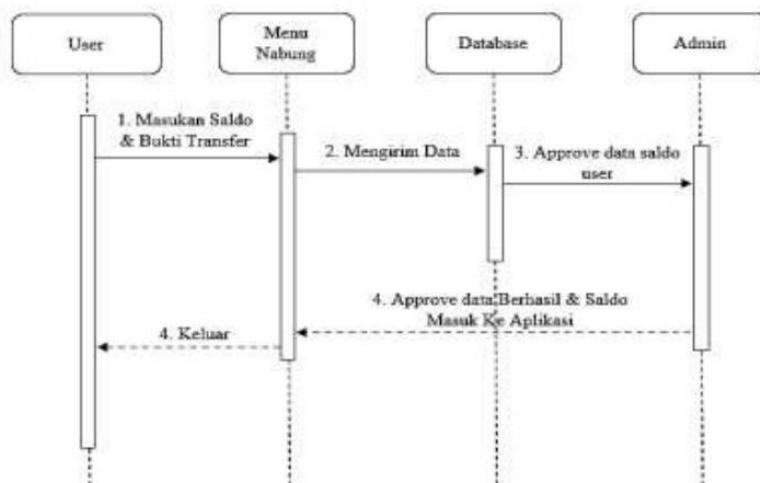
### 3.4.4.2 Sequence Diagram Login



**Gambar 3.11 Sequence Diagram Login**

Gambar 3.11 Menggambarkan pesan yang dikirim dari beberapa objek ke objek lainnya. Dalam proses *login*, *user* dan *admin* menjalankan aplikasi lalu memasukan *email* dan *password* yang sudah didaftarkan lalu aplikasi mengirim data ke sistem jika berhasil masuk ke menu beranda, jika tidak berhasil masuk akan ada pesan *error*.

### 3.3.4.3 Sequence Diagram Menabung

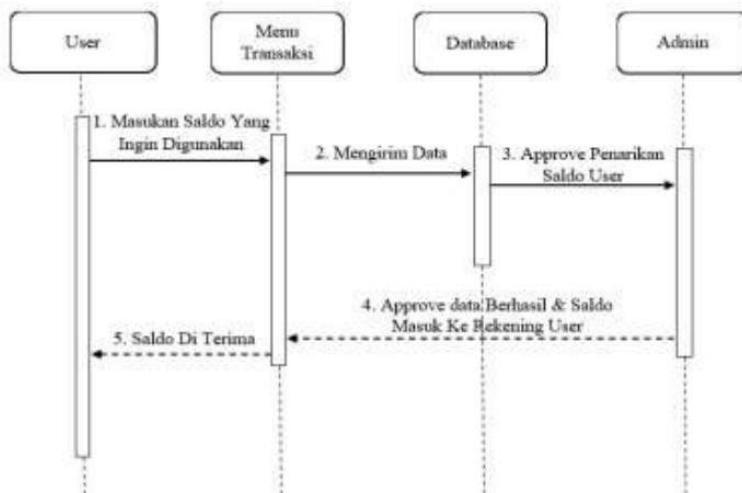


**Gambar 3.12 Sequence Diagram Menabung**

Gambar 3.12 Menggambarkan pesan yang dikirim dari beberapa objek ke objek lainnya. Dalam proses menabung *user* diwajibkan melakukan transfer ke bank yang tertera di aplikasi, selanjutnya *user* memasukan nominal saldo dan bukti transfer yang

sudah di transfer, dan data dikirim melalui *database*, lalu *admin* akan menyetujui data yang sudah masuk dan saldo bertambah di nominal aplikasi.

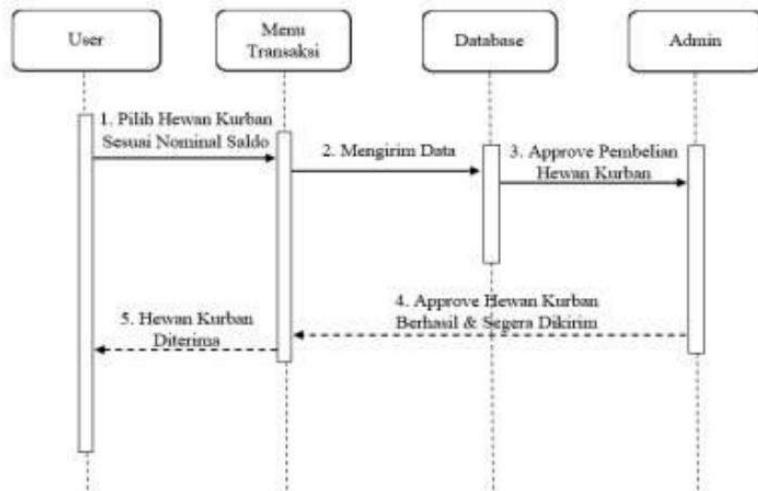
#### 3.3.4.4 Sequence Diagram Transaksi Penarikan Saldo



**Gambar 3.13 Sequence Diagram Penarikan Saldo**

Gambar 3.13 Menggambarkan pesan yang dikirim dari beberapa objek ke objek lainnya. Dalam proses transaksi penarikan saldo tabungan *user* diminta untuk menunggu satu bulan sebelum hari raya Idul Adha yang akan datang, setelah menunggu selanjutnya *user* bisa menjalankan menu transaksi dan dapat melakukan penarikan saldo tabungan. Dalam proses penarikan saldo *user* memasukan nominal saldo yang ingin digunakan untuk kebutuhan kurban, selanjutnya sistem mengirim data ke *database* dan *admin* akan *approve* penarikan saldo, setelah di *approve* saldo masuk ke rekening *user* dan saldo diterima oleh *user*.

### 3.3.4.5 Sequence Diagram Transaksi Pembelian Hewan Kurban

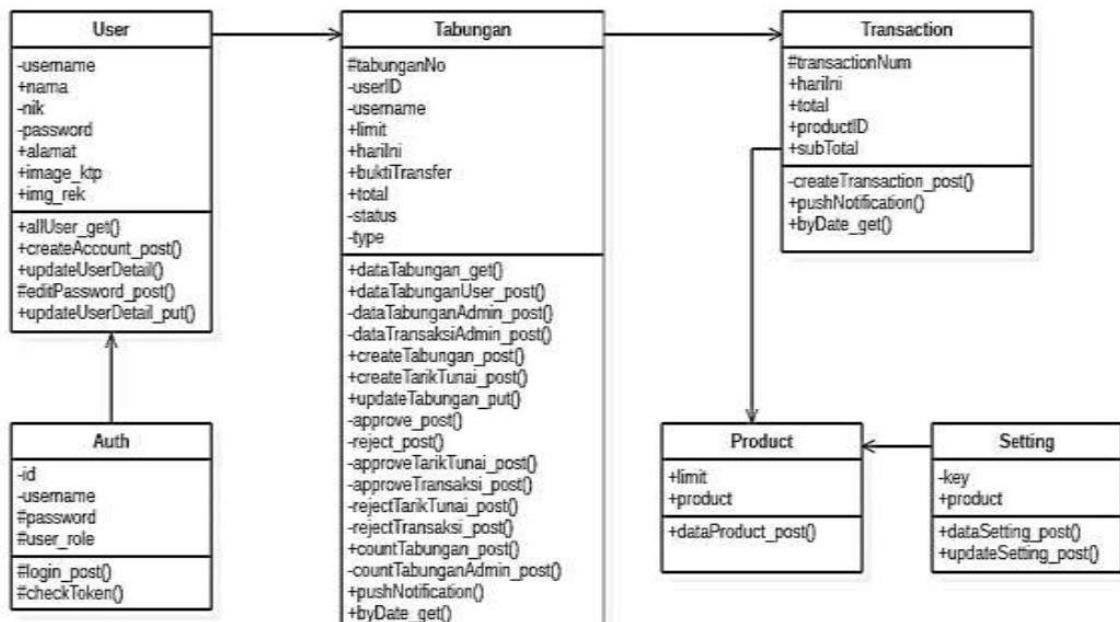


**Gambar 3.14** Sequence Diagram Pembelian Hewan Kurban

Gambar 3.14 Menggambarkan pesan yang dikirim dari beberapa objek ke objek lainnya. Dalam proses transaksi pembelian hewan kurban *user* diminta untuk menunggu satu bulan sebelum hari raya Idul Adha yang akan datang, setelah menunggu selanjutnya *user* bisa menjalankan menu transaksi dan dapat melakukan pembelian hewan kurban. Dalam proses pembelian hewan kurban *user* dapat memilih hewan kurban sesuai dengan nominal saldo tabungan, selanjutnya sistem mengirim data ke *database* dan *admin* akan *approve* pembelian hewan kurban tersebut, setelah di *approve* pengelola akan menyiapkan hewan kurban untuk dikirim dan hewan kurban diterima oleh pemilik hewan kurban/*user*.

### 3.4.5 Class Diagram

*Class* diagram menjelaskan untuk memetakan struktur sistem tertentu untuk memodelkan *class*, atribut, operasi serta hubungan antar objek dan menggambarkan deskripsi dari *class*, atribut, dan objek untuk terhubung satu sama lain.



Gambar 3.15 *Class Diagram*

### 3.4.6 Struktur Data

Analisis struktur data aplikasi yang telah disusun sesuai dengan fungsi dan kebutuhan aplikasi, terdapat 6 tabel yang terdiri dari Tabel *User*, *User Detail*, *Tabungan*, *Product*, *Transaction Head*, dan *Transaction Detail*. Berikut penjelasan masing-masing tabel struktur data:

Tabel 3.1 *User*

Nama	Tipe Data	Panjang Data	Keterangan
username	varchar	30	Primary key
password	varchar	32	
authkey	text	-	
status	integer	5	
user_role	integer	5	

**Tabel 3.2 User Detail**

Nama	Tipe Data	Panjang Data	Keterangan
id	integer	9	Primary key
username	varchar	30	Foreign key
nama	varchar	30	
NIK	bigint	16	
tglLahir	date	-	
JK	char	1	
alamat	text	-	
noTlp	varchar	13	
noRek	varchar	16	
bankId	integer	9	
fotoKtp	varchar	50	
fotoTabungan	varchar	50	

**Tabel 3.3 Tabungan**

Nama	Tipe Data	Panjang Data	Keterangan
tabunganNo	varchar	12	Primary key
tanggalTabungan	datetime	-	
username	varchar	30	Foreign key
buktiTransfer	varchar	30	
total	decimal	18,2	
status	int	5	
type	int	5	
createBy	varchar	30	
createDate	datetime	-	
updateBy	varchar	30	
updateDate	datetime	-	

**Tabel 3.4 Product**

Nama	Tipe Data	Panjang Data	Keterangan
id	int	11	Primary key
productName	varchar	50	
image	varchar	50	
total	decimal	18,2	
createBy	varchar	30	
createDate	datetime		
updateBy	varchar	30	
updateDate	datetime		

**Tabel 3.5 Transaction Head**

Nama	Tipe Data	Panjang Data	Keterangan
transactionNum	varchar	12	Primary key
transactionDate	datetime		
username	varchar	50	Foreign key
total	decimal	18,2	
status	Integer	11	
createBy	varchar	30	
createDate	datetime		
updateBy	varchar	30	
updateDate	datetime		

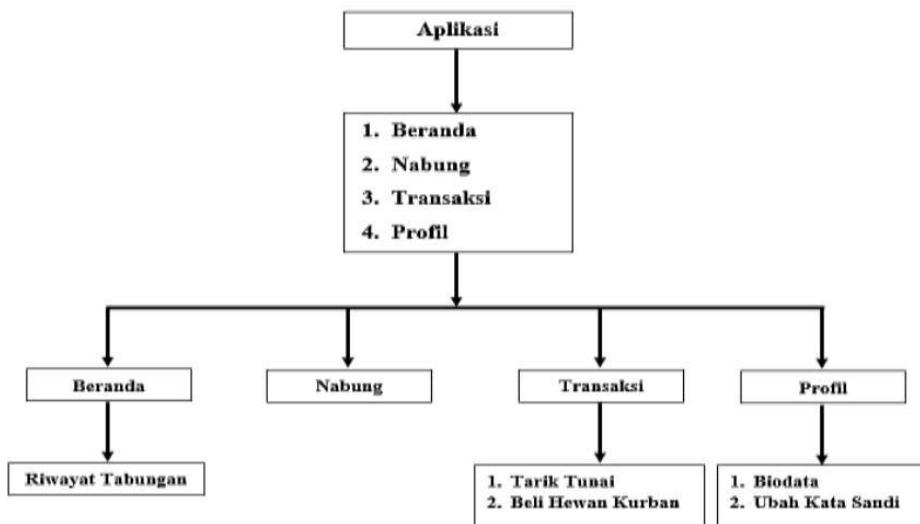
**Tabel 3.6 Transaction Detail**

Nama	Tipe Data	Panjang Data	Keterangan
id	int	11	Primary key
transactionNum	varchar	50	Foreign key
productID	Integer	11	
subTotal	decimal	18,2	

### 3.5 Perancangan antarmuka Aplikasi

Setelah menganalisis kebutuhan aplikasi selanjutnya masuk ke tahap perancangan antarmuka, adapun beberapa rancangan tampilan aplikasi dibuat dalam bentuk kerangka/*layout* untuk menjadi acuan pada saat membuat aplikasi:

#### 3.5.1 Rancangan Menu Navigasi

**Gambar 3.16** Menu Navigasi

Berikut adalah penjelasan menu navigasi pada gambar 3.16:

1. Mulai, diawali mengakses halaman utama.
2. Terdapat 4 (Empat) Menu di dalam aplikasi Qurbanku yaitu, Beranda, Nabung, Transaksi, Profil
3. Masing–masing tiap menu yang dipilih maka akan mengakses ke halaman aplikasi tersebut.
4. Menu beranda terdapat riwayat transaksi.
5. Menu Nabung, *user* dapat melakukan aktivitas menabung.
6. Menu Transaksi terdapat dua fungsi yaitu, tarik tunai tabungan dan pembelian hewan kurban.
7. Menu profil terdapat dua fungsi, *update* biodata *user* dan ubah kata sandi.

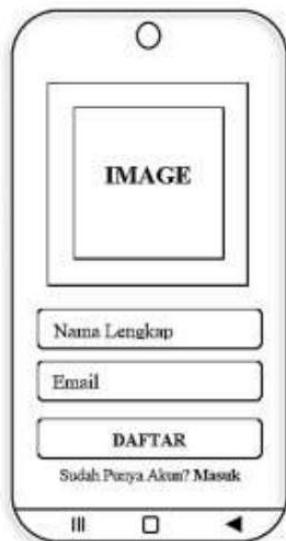
### 3.5.2 Rancangan Logo Aplikasi



**Gambar 3.17 Logo Aplikasi Tabungan Kurban**

Gambar 3.17 Arti logo Qurbanku terlihat lingkaran menyerupai koin dan di dalam lingkaran terdapat hewan kambing agar terlihat ikonik untuk aplikasi tabungan kurban, warna hijau dalam kepercayaan umat Islam artinya kesucian dan terdapat nama aplikasi di bawahnya agar masyarakat Muslim lebih mengenal aplikasi tersebut.

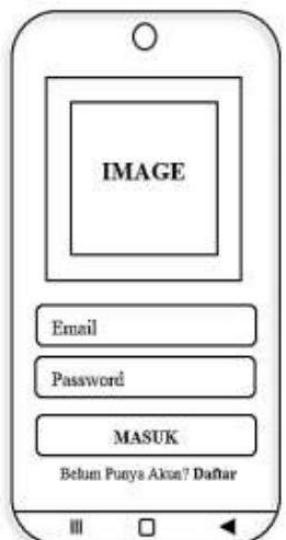
### 3.5.3 Rancangan Registrasi Aplikasi



**Gambar 3.18** Rancangan Registrasi Aplikasi

Gambar 3.18 Rancangan registrasi, *user* wajib memasukan nama lengkap dan *email* lalu tekan *button* daftar maka aplikasi secara otomatis akan memberikan kode verifikasi *password* berupa enam angka digit melalui *email user*.

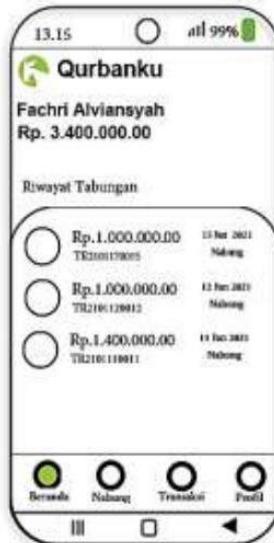
### 3.5.4 Rancangan Login



**Gambar 3.19** Rancangan Login Aplikasi

Gambar 3.19 Rancangan *login* aplikasi, Setelah cek *email*, masukan *username/email* dan salin *password* yang sudah dikirim melalui *email user*, lalu tekan tombol masuk.

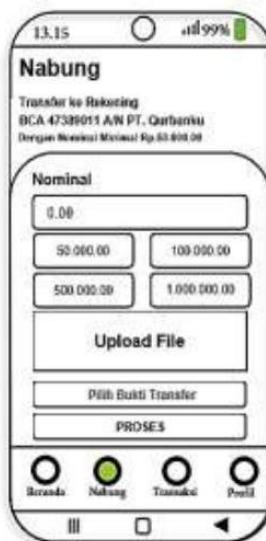
### 3.5.5 Rancangan Menu Beranda



**Gambar 3.20** Rancangan Menu Beranda

Gambar 3.20 Rancangan menu beranda aplikasi Qurbanku, *user* dapat mengetahui informasi total tabungan dengan nominal Rp.3.400.000.00,- (*Tiga Juta Empat Ratus Rupiah*) dan riwayat transaksi, dan terdapat empat *button padding navbar* untuk menu-menu selanjutnya.

### 3.5.6 Rancangan Menu Nabung



**Gambar 3.21** Rancangan Menu Nabung

Gambar 3.21 Rancangan menu nabung aplikasi Qurbanku, *user* diwajibkan untuk transfer ke nomor rekening yang disarankan dengan minimal saldo transaksi Rp.50.000.00,- (*Lima Puluh Ribu Rupiah*) lalu simpan bukti transfer. Langkah selanjutnya *user* dapat menuliskan nominal saldo atau tekan tombol saldo *default* yang

sudah di transfer dan masukan/*upload* bukti transfer. Lalu tekan tombol “PROSES” dan tunggu sampai *admin* memeriksa dan menyetujui data transfer tersebut. Jika berhasil saldo sudah masuk ke aplikasi di menu Beranda.

### 3.5.7 Rancangan Menu Transaksi



**Gambar 3.22 Rancang Menu Transaksi**

Gambar 3.22 Rancangan menu transaksi aplikasi Qurbanku jika sudah mendekati sebulan menuju hari raya Idul Adha, di dalam rancangan menu tersebut *user* dapat melakukan transaksi penarikan saldo tabungan dengan cara menekan tombol tarik tunai. Selain itu ada transaksi pembelian hewan kurban secara langsung melalui aplikasi dengan cara memilih hewan kurban dan harga yang sesuai dengan nominal tabungan.

### 3.5.8 Transaksi Detail



**Gambar 3.23 Transaksi Detail**

Gambar 3.23 Menjelaskan rancangan *detail* atau pembelian hewan kurban, setelah user memilih hewan domba dengan nominal harga Rp.3.000.000,00,- (*Tiga Juta Rupiah*) secara otomatis saldo tabungan user Rp.3.400.000,00,- (*Tiga Juta Empat Ratus Rupiah*) akan dikurangi dengan nominal pembelian hewan kurban dan terdapat sisa saldo tabungan Rp.400.000,00,- (*Empat Ratus Rupiah*). Sisa saldo tabungan dapat digunakan untuk melakukan tarik tunai atau membeli hewan kurban kembali di tahun yang akan datang. Jika sudah di kalkulasi dan saldo mencukupi tekan tombol pesan.

### 3.5.9 Rancangan Menu Profil



**Gambar 3.24 Rancangan Menu Profil**

Gambar 3.24 Rancangan menu profil aplikasi Qurbanku, di dalam menu profil terdapat tampilan nama dan *email* pemilik / *user* selain itu terdapat dua *widget button* Biodata dan Ubah Kata Sandi, di dalam *button* Biodata *user* wajib melengkapi data diri agar proses transaksi dapat dilakukan dan *button* yang kedua untuk mengganti Kata Sandi, *user* wajib mengubah Kata Sandi *default*.

### 3.5.10 Rancangan Form Biodata User



**Gambar 3.25** Rancangan Form Biodata User

Gambar 3.25 Rancangan form biodata *user*, setelah registrasi *user* dapat melengkapi data-data yang sudah disediakan, agar proses penarikan dan pembelian hewan kurban dapat dilakukan. Jika *user* tidak melengkapi biodata maka transaksi tarik tunai dan pembelian hewan kurban akan di *reject/tolak* oleh *admin*.

### 3.5.11 Rancangan Menu Transaksi Akses Admin



**Gambar 3.26** Rancangan Tarik Tunai dan Pembelian Hewan Kurban

Gambar 3.26 Rancangan menu *admin* transaksi tunai aplikasi Qurbanku, pada saat mendekati sebulan menjelang hari raya Idul Adha, *admin* diperbolehkan mengakses tombol *switch* yang ada di sudut kanan atas, setelah mengaktifkan, *user* wajib melengkapi data diri dan dapat mencairkan saldo tabungannya untuk kebutuhan membeli hewan kurban lalu *admin* akan memproses permintaan tersebut dengan cara memilih transaksi dan masuk ke halaman *approve/reject* tarik tunai.

### 3.5.12 Rancangan Approve dan Reject Tarik Tunai akses Admin



**Gambar 3.27** Rancangan Approve dan Reject Tarik Tunai Admin

Gambar 3.27 Rancangan menu *admin approve/setuju dan reject/tolak transaksi tarik tunai*, Setelah *admin* masuk ke menu transaksi dan selanjutnya memilih transaksi yang akan di proses, jika *user* telah melengkapi biodata, *admin* dapat menyetujui transaksi tersebut dan jika data *user* belum lengkap maka *admin* berhak menolak transaksi tarik tunai.

### 3.5.13 Rancangan *Approve* dan *Reject* Tabungan akses Admin



**Gambar 3.28** Rancangan *Approve* dan *Reject* Tabungan Admin

Gambar 3.28 Rancangan menu *admin approve/setuju dan reject/tolak transaksi transfer tabungan user*, sebelum disetujui cek kembali bukti transfer dan saldo masuk rekening, jika sudah melawati proses pengecekan tabungan bisa di *approve* dan jika tidak ada foto bukti transfer atau foto buram *admin* berhak untuk menolak transaksi, dan *user* kembali memasukan data atau foto bukti transfer pastikan gambar tidak pecah atau buram.

### 3.5.14 Rancangan Ubah Kata Sandi



**Gambar 3.29** Rancangan Ubah Kata Sandi

Gambar 3.29 Rancangan menu fungsi ubah kata sandi aplikasi tabungan Qurbanku, *user* dan *admin* dapat melakukan fungsi tersebut dengan cara memasukan kata sandi baru dan mengonfirmasi Kata Sandi lalu tekan tombol Simpan dan kata sandi berhasil diubah.

## **BAB 4**

### **IMPLEMENTASI DAN PENGUJIAN**

#### **4.1 Implementasi**

Tahapan implementasi dilakukan dengan cara menerapkan hasil rancangan aplikasi android yang telah di dapat dari tahap analisis dan perancangan.

##### **4.1.1 Implementasi Kebutuhan Teknis**

Pada proses implementasi aplikasi Qurbanku membutuhkan perangkat keras dan perangkat lunak agar proses menjalankan aplikasi dapat dilakukan, berikut penjelasannya:

###### **a) Perangkat Keras (*Hardware*)**

Dalam implementasi aplikasi Qurbanku ini, ada beberapa kebutuhan teknis perangkat keras (*hardware*) agar aplikasi berjalan dengan baik dan optimal. Perangkat keras yang digunakan dalam implementasi aplikasi adalah sebagai berikut:

- *Smartphone* Android
- Chipset Snapdragon 730G
- RAM 4 GB
- Kamera *Handphone* 32 MP
- Koneksi internet

###### **b) Perangkat Lunak (*Software*)**

Dalam implementasi aplikasi Qurbanku ini, ada beberapa kebutuhan teknis perangkat keras (*software*) agar aplikasi bisa berjalan dengan baik dan optimal. Perangkat lunak yang digunakan dalam implementasi aplikasi adalah sebagai berikut:

- Aplikasi Qurbanku
- Minimal Sistem Operasi Android 7.0 Nougat
- *Email*
- *Web Hosting*
- Postman
- Xampp 3.2.4

##### **4.1.2 Implementasi Pengembangan REST API**

Dalam rancangan *backend* aplikasi menggunakan REST API sebagai penghubung antara aplikasi dan *server*. REST API dibangun menggunakan bahasa pemrograman PHP

(*Hypertext Preprocessor*) dengan *framework* CodeIgniter. Berikut adalah skrip Kode *Class* dan fungsinya:

#### 4.1.2.1 Implementasi Class Auth

Dalam implementasi *class Auth* atau *Authentication* terdapat beberapa fungsi diantaranya, fungsi *login*, *login failed*, dan *check token*. Berikut skrip kode *Class Auth* dan fungsinya:

```
class Auth extends REST_Controller
{
    private $secretkey = 'c0Mb1nAtioNTravel';

    public function __construct()
    {
        header('Access-Control-Allow-Origin: *');
        parent::__construct();
        $this->load->library('form_validation');
```

**Gambar 4.1 Class Auth**

##### a) Fungsi Login

```
public function login_post()
{
    $date = new DateTime();
    $username = $this->post('username', TRUE);
    $password = $this->post('password', TRUE);

    $data_login = $this->M_Auth->checkUser($username, $password);

    if ($data_login) {
        $payload['data_user'] = $data_login;
        $payload['iat'] = $date->getTimestamp();
        $payload['exp'] = $date->getTimestamp()

        $token = JWT::encode($payload, $this->secretkey);
        $this->M_Auth->updateAuth($token, $data_login->username);
        $this->response([
            'status' => TRUE,
            'fullName' => $data_login->nama,
            'username' => $data_login->username,
            'token' => $token
        ], REST_Controller::HTTP_OK);
    } else {
        $error = "User not found";
        $this->loginFailed($username, $password, $error);
    }
}
```

**Gambar 4.2 Fungsi Login**

### b) Fungsi Check Token

```
public function checkToken()
{
    $jwt = $this->input->get_request_header('Authorization');
    $get_auth = $this->M_Auth->getAuth($jwt);
    if ($get_auth) {
        try {
            $decode = JWT::decode($jwt, $this->secretkey,
                array('HS256'));
            $username = $decode->data_user->username;

            if ($this->M_Auth->isValidToken($username) > 0) {
                $res['username'] = $username;

                return $res;
            }
        } catch (Exception $e) {
            exit('Wrong Token');
        }
    } else {
        return false;
    }
}
```

**Gambar 4.3** *Check Token*

#### 4.1.2.2 Implementasi Class User

Di dalam implementasi *class user* terdapat beberapa fungsi diantaranya, fungsi *create account*, *update user detail*, *all user*, dan *edit password*. Berikut skrip kode *Class User* dan fungsi-fungsinya:

```
class User extends Auth
{
    public function __construct()
    {
        header('Access-Control-Allow-Origin: *');
        header('Access-Control-Allow-Headers: Authorization');
        parent::__construct();
        $this->load->model('M_User');
        $this->data_global = $this->checkToken();
    }
}
```

**Gambar 4.4** *Class User*

### a) Fungsi *Create Account*

```

function createAccount_post()
{
    $data = array(
        'username' => $this->post('username', TRUE),
        'nama' => $this->post('nama', TRUE),
    );
    $modelCheck = $this->M_User->checkUser($data['username']);

    if (!$modelCheck) {
        $model = $this->M_User->createAccount($data);
        $res['status'] = $model;
        $res['message'] = 'Email berhasil didaftarkan, Silahkan cek
                           email anda untuk verifikasi.';
        $this->response($res, Auth::HTTP_OK);
    } else {
        $this->failed('Email Sudah Terdaftar');
    }
}

```

**Gambar 4.5** Fungsi *Create Account*

### b) Fungsi *Update Detail*

```

function updateUserDetail_put()
{
    $username = $this->input->update('username');
    $alamat = $this->input->update('alamat');
    $data = array(
        'nama' => $this->put('nama', TRUE),
        'NIK' => $this->put('NIK', TRUE),
        'JK' => $this->put('JK', TRUE),
        'alamat' => $this->put('alamat', TRUE),
        'noTlp' => $this->put('noTlp', TRUE),
        'bankId' => $this->put('bankId', TRUE),
        'fotoKtp' => $this->put('fotoKtp', TRUE),
        'fotoTabungan' => $this->put('fotoTabungan', TRUE),
    );

    $alamat = $this->M_User->editUserDetail($data, $username);
    $res['status'] = TRUE;
    $res['data'] = $data;
    $this->response($res, Auth::HTTP_OK);
}
}

```

**Gambar 4.6** Fungsi *Update User Detail*

### c) Fungsi All User

```
function allUser_get()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $get_data = $this->M_User->getAllUser();

        if ($get_data) {
            $res['status'] = TRUE;
            $res['data'] = $get_data;
            $this->response($res, Auth::HTTP_OK);
        } else {
            $this->failed("Not Found");
        }
    }
}
```

**Gambar 4.7** Fungsi All User

### d) Fungsi Edit Password

```
function editPassword_post()
{
    $username = $this->input->post('username');
    $data = array(
        'password' => md5($this->post('password', TRUE))
    );
    $model = $this->M_User->editPassword($data, $username);
    $res['status'] = $model;
    $res['message'] = 'Password berhasil diubah';
    $res['data'] = $model;
    $this->response($res, Auth::HTTP_OK);
}
```

**Gambar 4.8** Fungsi Edit Password

#### 4.1.2.3 Implementasi Class Tabungan

Di dalam implementasi *class* tabungan terdapat beberapa fungsi diantaranya, fungsi data tabungan *user*, fungsi data tabungan *admin*, *count* tabungan *admin*, *count* tabungan *user*, *create* tabungan, *update* tabungan, *create* tarik tunai, *approve* tarik tunai dan *reject* tarik tunai. Berikut skrip kode *class* Tabungan dan fungsi-fungsinya:

```

class Tabungan extends Auth
{
    function __construct()
    {
        header('Access-Control-Allow-Origin: *');
        header('Access-Control-Allow-Headers: Authorization');
        parent::__construct();
        $this->load->model('M_Tabungan', 'tabungan');
        $this->data_global = $this->checkToken();
    }
}

```

**Gambar 4.9 Class Tabungan****a) Fungsi Data Tabungan User**

```

public function dataTabunganUser_post()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $username = $this->post('username');
        $limit = $this->post('limit');
        $tabungan = $this->tabungan->getTabunganUsername($username,
            $limit);

        if ($tabungan) {
            $this->response([
                'status' => true,
                'data' => $tabungan,
            ], REST_Controller::HTTP_OK);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Data Not Found',
            ], REST_Controller::HTTP_NOT_FOUND);
        }
    }
}

```

**Gambar 4.10 Fungsi Data Tabungan User**

**b) Fungsi Data Tabungan Admin**

```
public function dataTabunganAdmin_post()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $limit = $this->post('limit');
        $tabungan = $this->tabungan->getTabunganAdmin($limit);

        if ($tabungan) {
            $this->response([
                'status' => true,
                'data' => $tabungan,
            ], REST_Controller::HTTP_OK);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Data Not Found',
            ], REST_Controller::HTTP_NOT_FOUND);
        }
    }
}
```

**Gambar 4.11 Fungsi Data Tabungan Admin**

**c) Fungsi Count Tabungan Admin**

```
public function countTabunganAdmin_post()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $result = $this->tabungan->countTabunganAdmin();

        if ($result) {
            $this->response([
                'status' => true,
                'data' => $result,
            ], REST_Controller::HTTP_OK);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Data Not Found',
            ], REST_Controller::HTTP_NOT_FOUND);
        }
    }
}
```

**Gambar 4.12 Fungsi Count Tabungan Admin**

**d) Fungsi *Create Tabungan***

```

public function createTabungan_post()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $byDate = $this->byDate_get();
        $hariIni = date("Y-m-d H:i:s");
        $data = [
            'tabunganNo' => $byDate,
            'tanggalTabungan' => $hariIni,
            'username' => $this->post('username'),
            'buktiTransfer' => $this->post('buktiTransfer'),
            'total' => $this->post('total'),
            'status' => $this->post('status'),
            'transferApproval' => $this->post('transferApproval'),
            'createBy' => $this->post('createBy'),
            'createDate' => $hariIni,
            'updateBy' => $this->post('updateBy'),
            'updateDate' => $hariIni
        ];
        if ($this->tabungan->createDataTabungan($data) > 0) {
            $this->response([
                'status' => true,
                'message' => 'Data Berhasil di Input'
            ], REST_Controller::HTTP_CREATED);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Gagal masukan data'
            ], REST_Controller::HTTP_BAD_REQUEST);
        }
    }
}

```

**Gambar 4.13** Fungsi *Create Tabungan*

**e) Fungsi *Count Tabungan User***

```
public function countTabungan_post()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $username = $this->post('username');
        $result = $this->tabungan->countTabungan($username);

        if ($result) {
            $this->response([
                'status' => true,
                'data' => $result,
            ], REST_Controller::HTTP_OK);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Data Not Found',
            ], REST_Controller::HTTP_NOT_FOUND);
        }
    }
}
```

**Gambar 4.14** Fungsi *Count Tabungan User*

### f) Fungsi Approve

```

public function approve_post()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $update = $this->post('tabunganNo');
        $userID = $this->post('userID');
        $username = $this->post('username');
        $data = [
            'status' => $this->tabungan::$STATUS_APPROVE
        ];

        if ($this->tabungan->updateApprove($data, $update) > 0) {
            $arrayDataUser = array(
                "notification" => [
                    "title" => "Pemberitahuan !",
                    "body" => "Tabungan dengan nomor " . $this->post
                               ('tabunganNo') . " Diterima"
                ],
                "data" => [
                    "tabunganNo" => $this->post('tabunganNo'),
                    "status" => $this->tabungan::$STATUS_APPROVE
                ],
                "to" => "/topics/$userID"
            );

            $this->pushNotification($arrayDataUser);
            $this->response([
                'status' => true,
                'message' => 'Data Berhasil Diapprove',
            ], REST_Controller::HTTP_CREATED);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Data Gagal Diapprove',
            ], REST_Controller::HTTP_BAD_REQUEST);
        }
    }
}

```

**Gambar 4.15** Fungsi Approve

### g) Fungsi *Reject*

```

public function reject_post()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $update = $this->post('tabunganNo');
        $userID = $this->post('userID');
        $data = [
            'status' => $this->tabungan::$STATUS_REJECT
        ];

        if ($this->tabungan->updateReject($data, $update) > 0) {
            $arrayDataUser = array(
                "notification" => [
                    "title" => "Pemberitahuan !",
                    "body" => "Tabungan dengan nomor " . $this->post
                               ('tabunganNo') . " Diterima"
                ],
                "data" => [
                    "tabunganNo" => $this->post('tabunganNo'),
                    "status" => $this->tabungan::$STATUS_REJECT
                ],
                "to" => "/topics/$userID"
            );

            $this->pushNotification($arrayDataUser);
            $this->response([
                'status' => true,
                'message' => 'Data Berhasil Direject'
            ], REST_Controller::HTTP_CREATED);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Data Gagal Direject'
            ], REST_Controller::HTTP_BAD_REQUEST);
        }
    }
}

```

**Gambar 4.16** Fungsi *Reject*

### **h) Fungsi *Create Tarik Tunai***

```

public function createTariktunai_post()
{
    if (!(!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $tabunganNo = $this->byDate_get();
        $hariIni = date("Y-m-d H:i:s");

        $total = str_replace(".", ",",
            str_replace(",",
                "", $this->
                post('total')));

        $data = [
            'tabunganNo' => $tabunganNo,
            'tanggalTabungan' => $hariIni,
            'username' => $this->post('username'),
            'buktiTransfer' => '',
            'total' => $total,
            'type' => $this->tabungan::$TIPE_TARIK_TUNAI,
            'status' => $this->tabungan::$STATUS_PENDING,
            'createBy' => $this->post('username'),
            'createDate' => $hariIni,
            'updateBy' => $this->post('username'),
            'updateDate' => $hariIni
        ];
    }

    if ($this->tabungan->createDataTabungan($data) > 0) {
        $arrayDataAdmin = array(
            "notification" => [
                "title" => "Pemberitahuan !",
                "body" => "Permintaan Tarik tunai dengan nomor
                    $tabunganNo"
            ],
            "to" => "/topics/administrator"
        );

        $this->pushNotification($arrayDataAdmin);
        $this->response([
            'status' => true,
            'message' => 'Data Berhasil di Input',
        ], REST_Controller::HTTP_CREATED);
    } else {
        $this->response([
            'status' => false,
            'message' => 'Gagal masukan data'
        ], REST_Controller::HTTP_BAD_REQUEST);
    }
}
}

```

**Gambar 4.17** Fungsi *Create Tarik Tunai*

### i) Fungsi *Approve Tarik tunai*

```

public function approveTariktunai_post()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $update = $this->post('tabunganNo');
        $userID = $this->post('userID');
        $data = [
            'status' => $this->tabungan::STATUS_APPROVE
        ];

        if ($this->tabungan->updateApprove($data, $update) > 0) {
            $arrayDataUser = array(
                "notification" => [
                    "title" => "Pemberitahuan !",
                    "body" => "Tarik tunai dengan nomor " . $this->
                        post('tabunganNo') . " Diterima"
                ],
                "data" => [
                    "tabunganNo" => $this->post('tabunganNo'),
                    "status" => $this->tabungan::STATUS_APPROVE
                ],
                "to" => "/topics/$userID"
            );

            $message = "Tarik tunai dengan nomor " . $this->post
                ('tabunganNo') . " Diterima";
            $this->pushNotification($arrayDataUser, $userID,
                $message);
            $this->response([
                'status' => true,
                'message' => 'Data Berhasil Diapprove',
            ], REST_Controller::HTTP_CREATED);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Data Gagal Diapprove',
            ], REST_Controller::HTTP_BAD_REQUEST);
        }
    }
}

```

**Gambar 4.18** *Approve Tarik Tunai*

### j) Fungsi *Reject Tarik Tunai*

```

public function rejectTariktunai_post()
{
    if (!(!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $update = $this->post('tabunganNo');
        $userID = $this->post('userID');
        $data = [
            'status' => $this->tabungan::$STATUS_REJECT
        ];

        if ($this->tabungan->updateReject($data, $update) > 0) {
            $arrayDataUser = array(
                "notification" => [
                    "title" => "Pemberitahuan !",
                    "body" => "Tarik tunai dengan nomor " . $this-
                        >post ('tabunganNo'). " Ditolak"
                ],
                "data" => [
                    "tabunganNo" => $this->post ('tabunganNo'),
                    "status" => $this->tabungan::$STATUS_REJECT
                ],
                "to" => "/topics/$userID"
            );

            $message = "Tarik tunai dengan nomor " . $this->post
                ('tabunganNo'). " Ditolak";
            $this->pushNotification($arrayDataUser, $userID,
                $message);
            $this->response([
                'status' => true,
                'message' => 'Data Berhasil Direject'
            ], REST_Controller::HTTP_CREATED);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Data Gagal Direject'
            ], REST_Controller::HTTP_BAD_REQUEST);
        }
    }
}

```

**Gambar 4.19** *Reject Tarik Tunai*

#### 4.1.2.4 Implementasi *Class Product*

Di dalam implementasi *Class Product* terdapat fungsi diantaranya, fungsi data *Product*. Berikut skrip kode *Class Product* dan fungsi-fungsinya:

```
class Product extends Auth
{
    public $fcmKey = 'AAAARlm1fPI:APA91bEmVS6oUY_TTmG4UV_V7JJc6D
                      Raecyv7603wnMuupXqjkFMeyCZGEV472bNk_0KwBGHXV2C4
                      m0CoTK2HDv0amEFGA76_q1i53wUY1zpLhLZJCTvFM_Y-
                      Yi2dszKGCEv-79wrMTc';

    function __construct()
    {
        header('Access-Control-Allow-Origin: *');
        header('Access-Control-Allow-Headers: Authorization');
        parent::__construct();
        $this->load->model('M_Product', 'product');
        $this->load->helper(array('form', 'url'));
        $this->data_global = $this->checkToken();
        $this->load->library('upload');
    }
}
```

**Gambar 4.20 Class Product**

#### a) Fungsi Data *Product*

```
public function dataProduct_post()
{
    if (!$this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $limit = $this->post('limit');
        $product = $this->product->getProduct($limit);

        if ($product) {
            $this->response([
                'status' => true,
                'data' => $product,
            ], REST_Controller::HTTP_OK);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Data Not Found',
            ], REST_Controller::HTTP_NOT_FOUND);
        }
    }
}
```

**Gambar 4.21 Fungsi Data Product**

#### 4.1.2.5 Implementasi *Class Transaction*

Di dalam implementasi *Class Transaction* terdapat fungsi diantaranya, fungsi *create transaction*. Berikut skrip kode *Class Transaction* dan fungsi-fungsinya:

```
class Transaction extends Auth
{
    public $fcmKey = 'AAAAR1mlfPI:APA91bEmVS6oUY_TTmG4UV
                      _V7JJc6DRaecyv7603wnMuupXqjkFMeyCZGEV472bNk_0Kw
                      BGHXV2C4m0CoTK2HDv0amEFGA76_qli53wUY1zpLhLZJCTv
                      FM_Y-Yi2dszKGCEv-79wrMTc';

    function __construct()
    {
        header('Access-Control-Allow-Origin: *');
        header('Access-Control-Allow-Headers: Authorization');
        parent::__construct();
        $this->load->model('M_User');
        $this->load->model('M_Tabungan', 'tabungan');
        $this->load->model('M_Transaction', 'transaction');
        $this->load->helper(array('form', 'url'));
        $this->data_global = $this->checkToken();
        $this->load->library('upload');
    }
}
```

**Gambar 4.22 Class Transaction**

### a) Fungsi *Create Transaction*

```

public function createTransaction_post()
{
    if (! $this->data_global) {
        $this->failed("Anda tidak memiliki akses!");
    } else {
        $transactionNum = $this->byDate_get();
        $hariIni = date("Y-m-d H:i:s");
        $total = str_replace(".", ",",
            str_replace(",",
                "", $this->
                post('total')));

        $data = [
            'transactionNum' => $transactionNum,
            'transactionDate' => $hariIni,
            'username' => $this->post('username'),
            'total' => $total,
            'status' => $this->transaction::STATUS_PENDING,
            'createBy' => $this->post('username'),
            'createDate' => $hariIni,
            'updateBy' => $this->post('username'),
            'updateDate' => $hariIni
        ];

        $detail = $this->post('data');
        foreach ($detail as $key) {
            $subTotal = str_replace(".", ",",
                str_replace(",",
                    "", $key['total']));
            $dataDetail = [
                'transactionNum' => $transactionNum,
                'productID' => $key['id'],
                'subTotal' => $subTotal,
            ];
            $this->transaction>
                createDataTransactionDetail($dataDetail);
        }

        if ($this->transaction->createDataTransaction($data) > 0 ) {
            $arrayDataAdmin = array(
                "notification" => [
                    "title" => "Pemberitahuan !",
                    "body" => "Transaksi masuk dengan nomor
                        $transactionNum"
                ],
                "to" => "/topics/administrator"
            );
            $this->pushNotification($arrayDataAdmin);
            $this->response([
                'status' => true,
                'message' => 'Data Berhasil di Input',
            ], REST_Controller::HTTP_CREATED);
        } else {
            $this->response([
                'status' => false,
                'message' => 'Gagal masukan data'
            ], REST_Controller::HTTP_BAD_REQUEST);
        }
    }
}

```

**Gambar 4.23** Fungsi *Create Transaction*

#### 4.1.3 Implementasi Aplikasi Qurbanku

Implementasi antarmuka adalah tahapan pembuatan tampilan *mobile* pada aplikasi Qurbanku diantaranya menu, registrasi, *login*, beranda, menabung, transaksi, profil dan lain-lain.

##### 4.1.3.1 Implementasi *Interface* Pendaftaran, *Login User*

Pada tampilan *interface* berikut terdapat beberapa aktivitas di dalam aplikasi yang dijelaskan sebagai berikut:

###### a) Registrasi



**Gambar 4.24** Tampilan Registrasi

Gambar 4.24 Merupakan tampilan awal dari aplikasi Qurbanku, *user* diwajibkan untuk melakukan registrasi diri dengan cara memasukan nama lengkap dan *email* lalu tekan tombol mendaftar.

**b) Cek Validasi Email**



**Gambar 4.25 Validasi Email**

Gambar 4.25 Selanjut *user* akan menerima *email* dari Qurbanku, salin *password* validasi yang sudah dikirim, contoh: **381127** lalu masukan *password* di halaman *login* aplikasi.

**c) Login**



**Gambar 4.26 Tampilan Login**

Gambar 4.26 Adalah tampilan *login* aplikasi, *user* yang sudah mendaftar dapat *login* dengan memasukan *email* dan *password* yang sudah dikirim melalui *email*. Perhatikan contoh berikut: *Username* fachrialviansyah@gmail.com dan Kata Sandi **381127** yang tertera di gambar 4.26. Selanjutnya tekan tombol masuk dan *user* akan masuk ke halaman beranda. Lihat Gambar 4.29.

d) ***Login Gagal***



**Gambar 4.27 Login Gagal**

Gambar 4.27 Kondisi jika *user* salah memasukan *email* dan kata sandi, maka akan keluar notifikasi *Error* dan diberikan pesan “*Periksa Kembali Kata Sandi*”.

#### 4.1.3.2 Implementasi Menu Beranda

Pada tampilan *interface* berikut terdapat beberapa aktivitas di dalam aplikasi yang dijelaskan sebagai berikut:

**a) Beranda Admin**



**Gambar 4.28** Tampilan Beranda Admin

Gambar 4.28 Merupakan tampilan beranda *admin*, dan terdapat nominal total tabungan keseluruhan dan terdapat dua daftar tabungan yang memiliki Kode TR2101250003 dan TR2101250002 yang sedang diproses oleh *admin*. Proses *approve* dan *reject* terdapat pada gambar 4.45

**b) Beranda User**



**Gambar 4.29** Tampilan Beranda User

Gambar 4.29 Merupakan tampilan Beranda *user*, di dalam tampilan beranda *user* terdapat beberapa informasi seperti nominal saldo tabungan dan riwayat transaksi tabungan, di dalam riwayat transaksi terdapat tiga gambar dan warna yang memiliki aksi yang berbeda. Perhatikan gambar berikut: Kode transaksi TR2101250007 transaksi di *reject/tolak*, transaksi TR2101250006 dan TR2101250006 di *approve/setuju*, dan transaksi TR2101250004 sedang di proses oleh *admin*

#### **4.1.3.3 Implementasi *Interface* Aktivitas Menabung**

Pada tampilan *interface* berikut terdapat beberapa aktivitas di dalam aplikasi yang dijelaskan sebagai berikut:

**a) Nabung**



**Gambar 4.30** Menu Nabung

Gambar 4.30 Selanjutnya di menu Nabung *user* dapat melakukan proses menabung, langkah pertama *user* transfer terlebih dahulu uang ke nomor rekening yang telah disediakan, setelah melakukan transfer, *User* dapat memasukan nominal jumlah saldo sesuai yang di transfer, selanjutnya *upload* bukti transfer di gambar 4.31 dan yang terakhir tekan tombol proses.

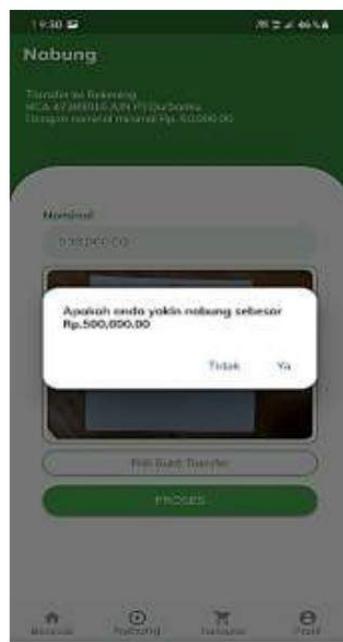
**b) Media *Upload* Bukti Transfer**



**Gambar 4.31 Media *Upload***

Gambar 4.31 Merupakan tampilan *popup* pilih bukti transfer, *user* diberikan dua pilihan *upload* melalui kamera *handphone* atau menggunakan folder pribadi, format yang digunakan (.jpg).

**c) *Popup* Menu Nabung**



**Gambar 4.32 *Popup* Pesan Nabung**

Gambar 4.32 Setelah melakukan proses menabung sistem akan mengeluarkan pesan peringatan, selanjutnya tekan tombol (Ya).

#### 4.1.3.4 Implementasi *Interface* Menu Transaksi

Pada tampilan *interface* berikut terdapat beberapa aktivitas di dalam aplikasi yang dijelaskan sebagai berikut:

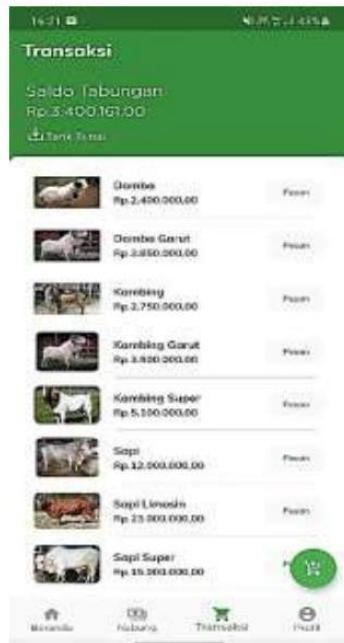
##### a) Tampilan transaksi off



**Gambar 4.33** Tampilan Menu Transaksi *Off*

Gambar 4.33 Menampilkan menu transaksi, jika belum mendekati sebulan menjelang hari raya Idul Adha, *user* belum bisa menggunakan fitur ini/menarik uang tabungan, guna untuk mencegah *user* menggunakan uang tabungan untuk keperluan selain berkurban.

**b) Tampilan Transaksi Tarik Tunai dan Pembelian Hewan Kurban**



**Gambar 4.34** Tampilan Tarik Tunai dan Pembelian Hewan Kurban

Gambar 4.34 Jika sudah mendekati sebulan hari raya Idul Adha fitur ini sudah dapat digunakan *user* untuk menarik saldo tabungan atau membeli langsung hewan kurban, sebelum melakukan transaksi ini *user* disarankan untuk melengkapi data diri di menu profil.

**c) Transaksi Detail Pembelian Hewan Kurban**



**Gambar 4.35** Transaksi Detail Pembelian Hewan Kurban

Gambar 3.35 Saat *user* memilih hewan kurban yang ingin dibeli selanjutnya *user* masuk ke halaman transaksi *detail*, di halaman ini *user* bisa mengetahui total pembelian hewan kurban yang sudah dipesan, tekan tombol pesan untuk melanjutkan transaksi, dan tunggu sampai *admin approve* transaksi. Jika tekan tombol batal, transaksi akan dibatalkan.

#### d) Transaksi Tarik Tunai



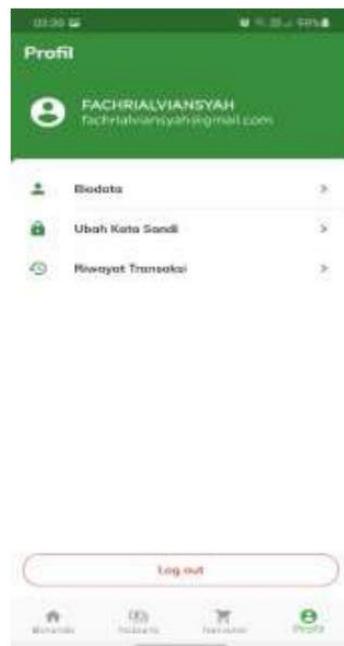
**Gambar 4.36 Tarik Tunai**

Gambar 4.36 Selain membeli hewan kurban *user* juga dapat mencairkan uang tabungannya untuk membeli hewan kurban di tempat lain. Transaksi dapat dilakukan dengan cara memasukan nominal saldo dan tekan tombol proses. Jika proses di *reject* oleh *admin*, periksa kembali *form* biodata *user*.

#### 4.1.3.5 Implementasi *Interface* Menu Profil

Pada tampilan *interface* berikut terdapat aktivitas di dalam aplikasi yang dijelaskan sebagai berikut:

### a) Profil



**Gambar 4.37** Tampilan Menu Profil

Gambar 4.37 Menjelaskan tentang menu profil *user*, di dalamnya terdapat tiga fungsi yaitu, melihat dan memperbarui biodata *user*, ubah kata sandi dan riwayat transaksi *user* lalu ada tombol *logout* atau keluar aplikasi.

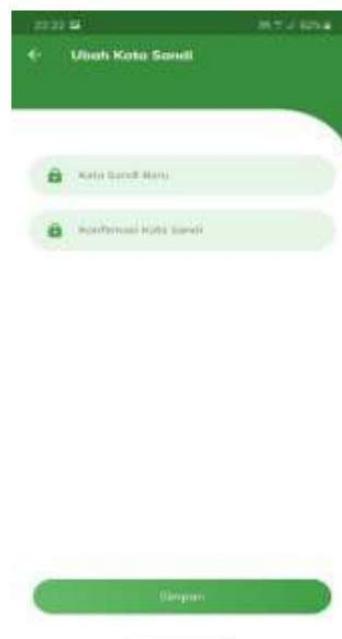
### b) Biodata



**Gambar 4.38** Biodata

Gambar 3.38 Tampilan *form* biodata dalam menu profil, user disarankan dapat memperbarui data diri pribadi seperti NIK, No Rekening, Foto KTP, Foto Rekening, Alamat, agar aplikasi dapat dijalankan dengan sempurna. Selanjutnya tekan tombol *update*.

**c) Ubah Kata Sandi**



**Gambar 4.39** Ubah Kata Sandi

Gambar 4.39 *admin* dan *user* dapat menggunakan fitur ubah kata sandi. Dan *user* di wajibkan mengubah kata sandi.

**d) Riwayat Transaksi User**



**Gambar 4.40** Riwayat Transaksi *User*

Gambar 4.40 Merupakan tampilan riwayat transaksi *user*, *user* dapat melihat transaksi yang sudah dilakukan oleh *user*, informasi digambar menunjukkan kode transaksi, nominal transaksi dan tanggal transaksi. Semua terekam di dalam fungsi ini.

#### 4.1.3.6 Implementasi *Interface Notifikasi*

Pada tampilan *interface* berikut terdapat beberapa aktivitas di dalam aplikasi yang dijelaskan sebagai berikut:

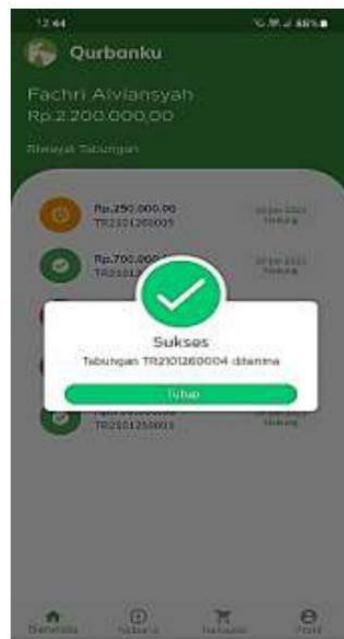
##### a) Notifikasi *Widget Handphone*



**Gambar 4.41** Widget Notifikasi Qurbanku

Gambar 4.41 Merupakan tampilan *widget* notifikasi pemberitahuan *user* dan *admin* di *smartphone* Android.

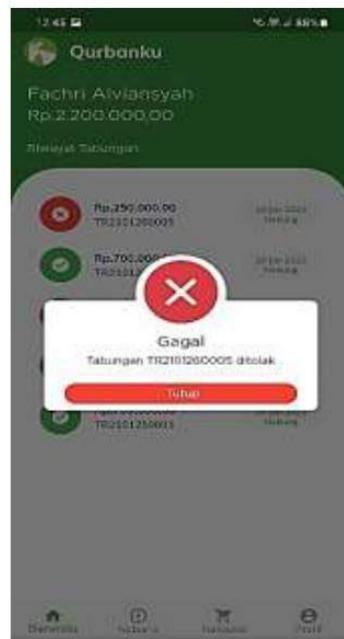
##### b) Notifikasi *Popup Sukses*



**Gambar 4.42 Notifikasi Sukses**

Gambar 4.42 Merupakan tampilan *popup* jika tabungan di *approve/terima* oleh *admin*.

c) Notifikasi *Popup* Gagal



**Gambar 4.43 Notifikasi Gagal**

Gambar 4.43 Merupakan tampilan *popup* jika tabungan di *reject/tolak* oleh *admin*.

#### 4.1.3.7 Implementasi *Interface Admin*

Pada tampilan *interface* berikut terdapat beberapa aktivitas di dalam aplikasi yang dijelaskan sebagai berikut:

**a) Login admin**



**Gambar 4.44 Login Admin**

Gambar 4.44 Pada tampilan *login admin* tidak ada yang berbeda dengan *user*, hanya saja *admin* tidak perlu menggunakan *username* dan registrasi.

**b) Menu Profil Admin**



**Gambar 4.45 Menu Profil**

Gambar 4.45 Merupakan tampilan profil *admin*, *admin* dapat melihat data nasabah/*user* dan melakukan perubahan kata sandi.

**c) Approve dan Reject Tabungan User**



**Gambar 4.46 Memproses Transaksi User**

Gambar 4.46 Menjelaskan proses *admin* melakukan setuju/approve dan tolak/reject transaksi tabungan *user*, selain itu terdapat informasi nomor transaksi, *email* dan nama lengkap *user*. Jika menekan tombol setuju akan menampilkan gambar 4.51 dan jika menekan tombol tolak akan menampilkan gambar 4.52.

**d) Menu Pencairan Transaksi**



**Gambar 4.47 Tampilan Pencairan Transaksi**

Gambar 4.47 Dalam menu pencairan transaksi saldo, *admin* dapat melakukan *approve* atau *reject* data *user* yang sudah melakukan penarikan saldo, dalam gambar diatas menjelaskan dua transaksi dengan *id* TR2102010018 dan TR2102010017 yang menunggu untuk diproses oleh *admin* dan terdapat *widget switch* untuk mengaktifkan menu transaksi *user*, jika tidak aktif tidak dapat digunakan lihat gambar 4.33.

#### e) Menu Riwayat Transaksi Admin



**Gambar 4.48** Menu Riwayat Transaksi *Admin*

Gambar 4.48 Di dalam menu Riwayat *Admin* dapat melihat riwayat transaksi *user* keseluruhan mulai dari transaksi menabung, penarikan saldo, dan pembelian hewan kurban. Informasi yang disampaikan berupa nama *user*, kode transaksi, nominal saldo, tanggal transaksi.

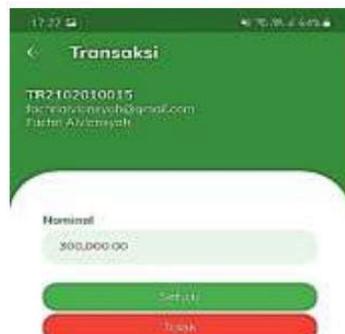
#### f) Nasabah Qurbanku



**Gambar 4.49 Nasabah Qurbanku**

Gambar 4.49 merupakan fungsi di dalam menu profil *admin*, *admin* dapat mengetahui data nasabah/*user* yang sudah melakukan registrasi. Di dalam gambar dapat tertera nama nasabah/*user* dan *username/email user*.

**g) Approve dan Reject Pencairan Transaksi**



**Gambar 4.50 Memproses Pencairan Saldo Tabungan**

Gambar 4.50 Menjelaskan proses *admin* melakukan setuju/approve dan tolak/reject pencairan transaksi saldo tabungan *user*, selain itu terdapat informasi nomor transaksi, *email*, dan nama lengkap *user*. Jika admin menekan tombol setuju/approve lihat gambar 4.53 dan jika menekan tombol tolak/reject lihat gambar 4.54.

**h) *Popup Approve* Pencairan Transaksi**



**Gambar 4.51 *Popup Approve* Pencairan Transaksi**

Gambar 4.51 Merupakan tampilan *popup* setuju/approve jika *admin* memilih tombol *approve*/setuju. Secara otomatis saldo *user* akan berkurang.

**i) *Popup Reject* Pencairan Transaksi**



**Gambar 4.52 Popup Reject Pencairan Transaksi**

Gambar 4.52 Merupakan tampilan *popup* tolak/reject data pencairan transaksi *user*. Jika *admin* memilih tombol tolak/reject, tidak ada aktivitas yang dilakukan.

j) *Popup Approve Tabungan Admin*



**Gambar 4.53 Popup Setuju/Approve Tabungan**

Gambar 4.53 Merupakan tampilan *popup* setuju/approve jika *admin* memilih tombol setuju/approve. Secara otomatis saldo *user* akan bertambah.

**k) Popup Reject Tabungan Admin**



**Gambar 4.54** Popup Tolak/*Reject* Tabungan

Gambar 4.54 Merupakan tampilan *popup* tolak/*reject* data *user* Menabung. jika *admin* memilih tombol tolak/*reject*. Tidak ada aktivitas yang dilakukan.

## 4.2 Pengujian

Pengujian Aplikasi Tabungan Qurbanku dalam Tugas Akhir ini dibagi menjadi dua tahapan, yakni pengujian REST API menggunakan Postman dan pengujian menggunakan metode *Black Box*.

### 4.2.1 Pengujian REST API menggunakan Postman

Pengujian fungsional sistem REST API dilakukan secara keseluruhan. Pengujian diklasifikasi menjadi empat bagian utama yakni *Auth/Authentication*, *User*, dan Tabungan, *Product*. Pengujian semua proses dilakukan menggunakan aplikasi Postman dan berikut hasil pengujian dapat dilihat pada Tabel 4.1, Tabel 4.2, Tabel 4.3, Tabel 4.4.

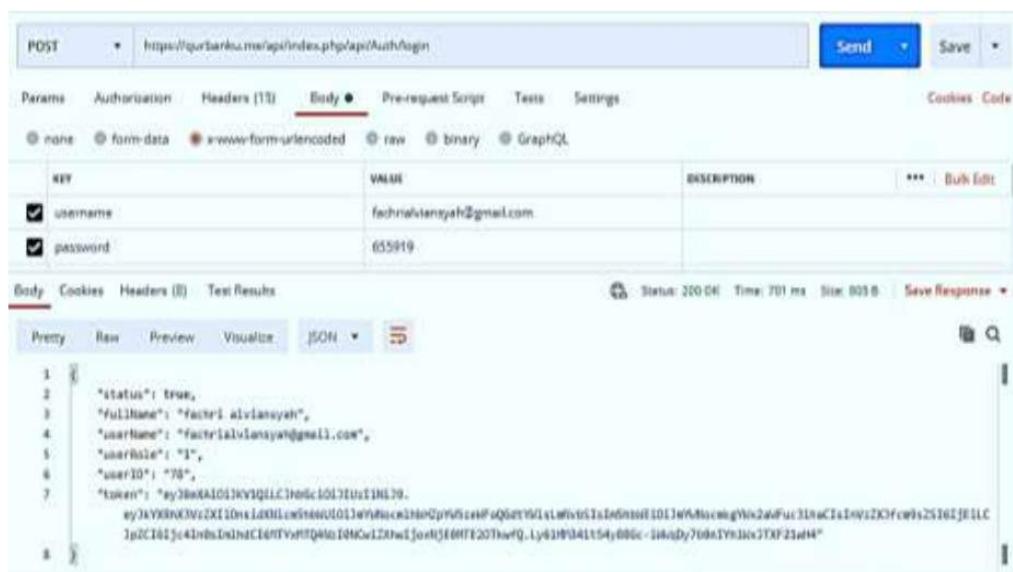
#### **4.2.1.1 Pengujian *Class Auth***

Dalam pengujian *class auth* terdapat dua fungsi yang akan diuji menggunakan aplikasi Postman. Berikut penjelasan fungsi-fungsi dalam *Class Auth* pada tabel 4.1:

**Tabel 4.1 REST API Class Auth**

No	Nama Proses	Request Method	URI	Keterangan
1	login	POST	/login_post	Akses <i>Login</i>
2	checkToken	GET	/login	Menampilkan Token

a) *Login*



**Gambar 4.55 Postman Login**

Gambar 4.55 Menjelaskan tentang pengujian API *login* dengan *method POST* menggunakan aplikasi postman. Terdapat pesan *true*, *full name*, *username*, *token* dan lain-lain. Dapat disimpulkan API berikut berhasil.

### b) Check Token

The screenshot shows a Postman interface with a POST request to <https://qurbanaku.me/api/index.php/api/Auth/login>. The Headers tab is selected, showing an Authorization header with a value of `eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.`. The Body tab shows a JSON response with fields: status (true), fullname ("Fachri Alviqiansyah"), username ("fachrialviqiansyah@gmail.com"), userRole ("1"), userID ("78"), and token (a long string of characters). The status bar at the bottom indicates a 200 OK response with a time of 701 ms and a size of 803 B.

**Gambar 4.56** Postman *Check Token*

Gambar 4.56 Melanjutkan penjelasan tentang pengujian API *login* dengan *method POST* menggunakan aplikasi postman. *User* akan diberikan *token/kode unik* yang nantinya berguna untuk mengamankan suatu transaksi selain itu di *token* juga dapat digunakan untuk *authentication login*. Dapat disimpulkan pengujian API berikut berhasil.

#### 4.2.1.2 Pengujian Class User

Dalam pengujian *class user* terdapat empat fungsi yang akan diuji menggunakan aplikasi Postman. Berikut penjelasan fungsi-fungsi dalam *Class User* pada tabel 4.2:

**Tabel 4.2** REST API *Class User*

No	Nama Proses	Request Method	URI	Keterangan
1	allUser	GET	/allUser	Menampilkan data <i>user</i>
2	createAccount	POST	/createAccount	Membuat akun baru
3	editPassword	POST	/editPassword	Mengubah <i>password</i> lama
4	updateUserDetail	PUT	/updateUserDetail	Menambahkan data <i>user</i>

### a) All User

The screenshot shows a Postman interface with a GET request to `https://qurbaniku.me/api/index.php/api/User/allUser`. The Headers tab is selected, showing an Authorization header with a value of `eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpYXQiOjE1...`. The Body tab shows a JSON response:

```

1 {
2   "status": true,
3   "data": [
4     {
5       "username": "115170000@student.iti.ac.id",
6       "status": "1"
7     },
8     {
9       "username": "admin",
10      "status": "1"
11    }
12 ]

```

The response status is 200 OK, Time: 1129 ms, Size: 973 B.

**Gambar 4.57 Postman All User**

Gambar 4.57 Menjelaskan tentang pengujian API melihat semua *user* dengan *method GET* menggunakan postman. Jika berhasil akan ada pesan status *true* dan *admin* dapat melihat semua data *user*. Dapat disimpulkan pengujian API berikut berhasil.

### b) Create Account

The screenshot shows a Postman interface with a POST request to `https://qurbaniku.me/api/index.php/api/User/createAccount`. The Headers tab is selected, showing a Content-Type of `application/x-www-form-urlencoded`. The Body tab shows form-data fields:

KEY	VALUE	DESCRIPTION
username	fachrialiyah@gmail.com	
nama	fachri aliyah	
total	70001	
status	1	
transferApproval	sukses	

The response status is 200 OK, Time: 134 s, Size: 847 B.

```

1 {
2   "status": true,
3   "message": "Email berhasil didaftarkan, Silahkan cek email anda untuk verifikasi."
4 }

```

**Gambar 4.58 Postman Create Account**

Gambar 4.58 Menjelaskan tentang pengujian API membuat akun dengan *method POST* menggunakan aplikasi postman. *User* dapat memasukan data *username* dan

nama, lalu *user* akan mendapatkan *email* validasi *password*. Dapat disimpulkan API berikut berhasil.

#### c) Edit Password

The screenshot shows the Postman interface with a successful API call. The URL is `https://qurbaniku.me/api/index.php/api/User/editPassword`. The method is set to `POST`. The `Body` tab is selected, showing the following JSON payload:

```

1 [
2   "status": true,
3   "message": "Password berhasil diubah",
4   "data": true
5 ]

```

The response status is `200 OK`, time `566 ms`, and size `514 B`.

**Gambar 4.59 Postman Edit Password**

Gambar 4.59 Menjelaskan tentang pengujian API edit *password* dengan *method* `POST` menggunakan aplikasi postman. *user* memasukan *username* dan *password* baru, jika berhasil akan keluar pesan *password* berhasil diubah dan status *true*. Dapat disimpulkan API berikut berhasil.

#### d) Update User Detail

The screenshot shows the Postman interface with a successful API call. The URL is `https://qurbaniku.me/api/index.php/api/User/updateUserDetail`. The method is set to `POST`. The `Body` tab is selected, showing the following JSON payload:

```

1 [
2   "status": true,
3   "message": "Data Berhasil di Input"
4 ]

```

The response status is `201 Created`, time `1072 ms`, and size `305 B`.

**Gambar 4.60 Postman Update User Detail**

Gambar 4.60 Menjelaskan tentang API *update user detail* dengan *method PUT* menggunakan aplikasi postman. *user* dapat memasukan *username*, NIK, nomor rekening, foto ktp, foto tabungan, jika berhasil akan keluar status *true*. Dapat disimpulkan pengujian API berikut berhasil.

#### 4.2.1.3 Pengujian Class Tabungan

Dalam pengujian *class* tabungan terdapat sepuluh fungsi yang akan diuji menggunakan aplikasi Postman. Berikut penjelasan fungsi-fungsi dalam *Class Tabungan* pada tabel 4.3:

**Tabel 4.3 REST API Class Tabungan**

No	Nama Proses	Request Method	URI	Keterangan
1	dataTabunganUser	POST	/dataTabunganUser	Menambahkan data tabungan <i>user</i>
2	dataTabunganAdmin	POST	/dataTabunganAdmin	Melihat data <i>user</i> yang sudah mendaftar
3	countTabunganAdmin	POST	/countTabunganAdmin	Menghapus data tabungan berdasarkan <i>Id user</i>
4	createTabungan	POST	/createTabungan	Menambahkan transaksi tabungan
5	countTabungan	POST	/countTabungan	Mengubah data tabungan
6	approve	POST	/approve	Menerima data tabungan <i>user</i>
7	reject	POST	/reject	Menolak data tabungan <i>user</i>
8	createTariktunai	POST	/createTariktunai	Melakukan Penarikan Tunai
9	approveTariktunai	POST	/approveTariktunai	Admin Approve Tarik Tunai
10	rejectTariktunai	POST	/rejectTariktunai	Admin Reject Tarik Tunai

### a) Data Tabungan User

The screenshot shows a Postman interface with a GET request to `https://qurbanku.me/api/index.php/api/Tabungan/dataTabungan`. The response status is 200 OK. The JSON data returned is:

```

355
356
357
358
359
360
361
362
363
364
365
366
367
    {
      "tabunganId": "TR2101240023",
      "tanggalTabungan": "2021-01-24 17:36:00",
      "username": "fachrialviansyah@gmail.com",
      "buktiTransfer": "TR2101240023.jpg",
      "total": "533333,00",
      "status": "2",
      "type": "1",
      "createBy": "fachrialviansyah@gmail.com",
      "createDate": "2021-01-24 17:36:00",
      "updateBy": "fachrialviansyah@gmail.com",
      "updateDate": "2021-01-24 17:36:00"
    }
  
```

**Gambar 4.61** Postman Data Tabungan *User*

Gambar 4.61 Menjelaskan tentang pengujian API data tabungan *user* dengan *method GET* menggunakan aplikasi postman. *User* yang sudah melakukan transaksi menabung datanya akan tersimpan dan dapat ditampilkan datanya di postman. Dapat disimpulkan pengujian API berikut berhasil.

### b) Data Tabungan Admin

The screenshot shows a Postman interface with a POST request to `https://qurbanku.me/api/index.php/api/Tabungan/dataTabungan/Admin`. The response status is 200 OK. The JSON data returned is:

```

1
2   "status": true,
3   "data": [
4     {
5       "tabunganId": "TR2101240024",
6       "tanggalTabungan": "2021-01-24 21:57:18",
7       "username": "fachrialviansyah@gmail.com",
8       "buktiTransfer": "TR2101240024.jpg",
9       "total": "5000000,00",
10      "status": "1",
11      "type": "1",
12      "createBy": "fachrialviansyah@gmail.com",
13      "createDate": "2021-01-24 21:57:18",
14      "updateBy": "fachrialviansyah@gmail.com",
15      "updateDate": "2021-01-24 21:57:18"
16    }
  ]
  
```

**Gambar 4.62** Postman Data Tabungan *Admin*

Gambar 4.62 Menjelaskan tentang pengujian API data tabungan *admin* dengan *method POST* menggunakan aplikasi postman. di fungsi ini *admin* dapat

mengetahui data-data *user* yang sudah menabung. Jika benar akan keluar status *true*. Dapat disimpulkan pengujian API berhasil.

#### c) Count Tabungan Admin

KEY	VALUE	DESCRIPTION
Key	Value	Description

```

1
2   "status": true,
3   "data": [
4     {
5       "total": "1655895390439.00"
6     }
7   ]
8
  
```

**Gambar 4.63 Postman Count Tabungan Admin**

Gambar 4.63 Menjelaskan tentang pengujian API total saldo tabungan keseluruhan *user* dengan *method POST* menggunakan aplikasi postman. di fungsi ini *admin* dapat mengetahui jumlah semua saldo dari *user* yang sudah menabung. Jika benar akan keluar status *true*. Dapat disimpulkan pengujian API berhasil.

#### d) Create Tabungan

KEY	VALUE	DESCRIPTION
tabunganNo		
senggatalungan		
username	farhanurroiqah@gmail.com	
isAutoTransfer	img501.jpg	
total	500000	
type		

```

1
2   "status": true,
3   "message": "Data Berhasil di Input"
  
```

**Gambar 4.64 Postman Create Tabungan**

Gambar 4.64 Menjelaskan tentang pengujian API membuat transaksi tabungan dengan *method POST* menggunakan aplikasi postman. *User* dapat memasukan

*email*, nama, *file* bukti transfer, dan jumlah saldo, jika benar status *true*. Dapat disimpulkan pengujian API berhasil.

#### e) Count Tabungan

The screenshot shows a Postman interface with the following details:

- Method:** POST
- URL:** <https://qurbaniku.me/api/index.php/api/Tabungan/countTabungan>
- Body (x-www-form-urlencoded):**

Key	Value	Description
username	fachrialviansyah@gmail.com	
- Response Body (JSON):**

```

1 {
2     "status": true,
3     "data": [
4         {
5             "username": "fachrialviansyah@gmail.com",
6             "total": "501250524241.00"
7         }
8     ]
9 }
```
- Status:** 200 OK | Time: 580 ms | Size: 542 B

**Gambar 4.65** Postman Count Tabungan

Gambar 4.65 Menjelaskan tentang pengujian API *count* tabungan dengan *method POST* menggunakan aplikasi postman. Fungsi ini *user* dapat mengetahui jumlah total saldo yang sudah ditabung. Jika benar akan keluar status *true*. Dapat disimpulkan pengujian API berhasil.

#### f) Approve

The screenshot shows a Postman interface with the following details:

- Method:** POST
- URL:** <https://qurbaniku.me/api/index.php/api/Tabungan/approve>
- Body (x-www-form-urlencoded):**

Key	Value	Description
tabunganNo	TR21012400Q4	
UserID	fachrialviansyah@gmail.com	
- Response Body (JSON):**

```

1 [
2     {
3         "status": true,
4         "message": "Data berhasil Disetujui"
5     }
6 ]
```
- Status:** 201 Created | Time: 1319 ms | Size: 556 B

**Gambar 4.66** Postman Approve

Gambar 4.66 Menjelaskan tentang pengujian API *approve* dengan *method POST* menggunakan aplikasi postman. Fungsi ini *admin* dapat melakukan

*approve*/menyetujui data yang masuk. Jika benar akan keluar status *true*. Dapat disimpulkan pengujian API berhasil.

### g) *Reject*

The screenshot shows a POST request to <https://qurbanku.me/api/index.php/api/Tabungan/reject>. The 'Body' tab is selected, showing a JSON payload with two checked fields: 'tabunganNo' (value: TR2101240024) and 'UserID' (value: fachnelviensyah@gmail.com). The response status is 201 Created, and the JSON body is:

```

1 {
2   "status": true,
3   "message": "Data Berhasil Direject"
4 }

```

**Gambar 4.67 Postman Reject**

Gambar 4.67 Menjelaskan tentang pengujian API *reject* dengan *method POST* menggunakan aplikasi postman. Fungsi ini *admin* dapat melakukan *reject*/menolak data yang masuk. Jika benar akan keluar status *true*. Dapat disimpulkan pengujian API berhasil.

### h) *Create Tarik Tunai*

The screenshot shows a POST request to <https://qurbanku.me/api/index.php/api/Tabungan/createTarikTunai>. The 'Body' tab is selected, showing a JSON payload with six checked fields: 'tabunganNo' (value: TR2101310009), 'tanggalTabungan', 'username' (value: fachnelviensyah@gmail.com), 'buktiTransfer', 'sosial', and 'type'. The response status is 201 Created, and the JSON body is:

```

1 {
2   "status": true,
3   "message": "Data Berhasil di Input"
4 }

```

**Gambar 4.68 Postman Create Tarik Tunai**

Gambar 4.68 Menjelaskan tentang pengujian API *create tarik tunai* dengan *method POST* menggunakan aplikasi postman. *User* dapat memasukan nomor

tabungan, *username*, jumlah total saldo, dan *type*, jika berhasil status *true*. Dapat disimpulkan pengujian API berhasil.

### i) *Approve Tarik Tunai*

The screenshot shows a POST request to <https://qurbanlu.me/api/index.php/api/Tabungan/approveTarikTunai>. The 'Body' tab is selected, showing a JSON payload with 'tabunganNo' (TR2101250002) and 'UserID' (fachrialviansyah@gmail.com). The response status is 201 Created, and the JSON body is {"status": true, "message": "Data Berhasil Diapprove"}.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> tabunganNo	TR2101250002	
<input checked="" type="checkbox"/> UserID	fachrialviansyah@gmail.com	

```

1 {
2   "status": true,
3   "message": "Data Berhasil Diapprove"
4 }

```

**Gambar 4.69** Postman *Approve Tarik Tunai*

Gambar 4.69 Menjelaskan tentang pengujian API *approve tarik tunai* dengan *method POST* menggunakan aplikasi postman. *Admin* dapat memasukan nomor transaksi dan *user id*, jika benar status *true*. Dapat disimpulkan pengujian API berhasil.

### j) *Reject Tarik Tunai*

The screenshot shows a POST request to <https://qurbanlu.me/api/index.php/api/Tabungan/rejectTarikTunai>. The 'Body' tab is selected, showing a JSON payload with 'tabunganNo' (TR2101250002) and 'UserID' (fachrialviansyah@gmail.com). The response status is 201 Created, and the JSON body is {"status": true, "message": "Data Berhasil Direject"}.

KEY	VALUE	DESCRIPTION
<input checked="" type="checkbox"/> tabunganNo	TR2101250002	
<input checked="" type="checkbox"/> UserID	fachrialviansyah@gmail.com	

```

1 {
2   "status": true,
3   "message": "Data Berhasil Direject"
4 }

```

**Gambar 4.70** Postman *Reject Tarik Tunai*

Gambar 4.70 Menjelaskan tentang pengujian API *Reject* tarik tunai dengan *method POST* menggunakan aplikasi postman. *Admin* dapat memasukan nomor transaksi dan *user id*, jika benar status *true*. Dapat disimpulkan pengujian API berhasil.

#### 4.2.1.4 Pengujian Class Product

Dalam pengujian *class product* terdapat satu fungsi yang akan diuji menggunakan aplikasi Postman. Berikut penjelasan fungsi-fungsi dalam *class tabungan* pada tabel 4.4:

**Tabel 4.4 REST API Class Product**

No	Nama Proses	Request Method	URI	Keterangan
1	dataProduct	POST	/dataProduct	Menampilkan Penjualan Hewan Kurban.

##### a) Data Product

```

POST https://qurbanku.ma/api/index.php/api/Product/dataProduct
Send Save
Params Authorization Headers (0) Body Pre-request Script Tests Settings Cookies Code
Value: limit: 10
Body Cookies Headers (0) Text Results Status: 200 OK Time: 593 ms Size: 2.23 KB Save Response
Pretty Raw Preview Visualize JSON
1 {
2   "status": true,
3   "data": [
4     {
5       "id": "9",
6       "productName": "Domba",
7       "image": "kambing.jpg",
8       "total": "3000000.00",
9       "createBy": "system",
10      "createDate": "2022-02-01 09:09:09",
11      "updateBy": "",
12      "updateDate": "2022-02-01 09:09:09"
}

```

**Gambar 4.71** Postman Data Product

Gambar 4.71 Menjelaskan tentang pengujian API *data product* dengan *method POST* menggunakan aplikasi postman. *Admin* dapat memasukan *limit* berapa banyak data *product* yang ingin ditampilkan, jika benar status *true*. Dapat disimpulkan pengujian API berhasil.

#### 4.2.1.5 Pengujian Class Transaction

Dalam pengujian *class transaction* terdapat satu fungsi yang akan diuji menggunakan aplikasi Postman. Berikut penjelasan fungsi-fungsi dalam *class transaction* pada tabel 4.5:

**Tabel 4.5 REST API Class Transaction**

No	Nama Proses	Request Method	URI	Keterangan
1	createTransaction	POST	/createTransaction	Melakukan Transaksi Penarikan saldo dan Pembelian Hewan Kurban.

##### a) Create Transaction

The screenshot shows a Postman interface with a POST request to `https://qurbanku.me/api/index.php/api/transaction/createTransaction`. The request method is set to POST. In the Body tab, the content type is set to form-data. The body contains several fields:

- total**: 1500000
- productID**: 3
- subTotal**: 5650000
- username**: fachriaviansyah@gmail.com
- createBy**: fachriaviansyah@gmail.com
- updateBy**: fachriaviansyah@gmail.com
- id**: (empty)

The response status is 201 Created, and the message is "Data berhasil di Input".

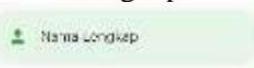
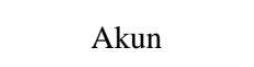
**Gambar 4.72** Postman Create Transaction

Gambar 4.72 Menjelaskan tentang pengujian API *create transaction* tarik tunai dan pembelian hewan kurban dengan *method POST* menggunakan aplikasi postman, *user* dapat memasukan nominal saldo yang ingin ditarik atau digunakan membeli hewan kurban. Selanjutnya jika membeli hewan kurban secara langsung di aplikasi *user* dapat memasukan *id product* dan masukan sub total dan *username*, *create by*, *update by*. Jika berhasil status *true*. dapat disimpulkan pengujian API berhasil.

#### 4.2.2 Pengujian Aplikasi menggunakan Metode *Black Box*

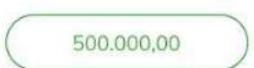
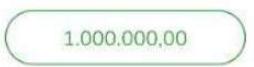
*Black box* testing merupakan tahap pengujian yang mempunyai fokus pada persyaratan fungsional perangkat lunak. Pada *Black Box* testing, cara pengujian dilakukan dengan menjalankan atau mengeksekusi untuk unit atau modul, kemudian diamati apakah hasil dari unit itu sesuai dengan proses yang diinginkan. Hasil pengujian *black box* dapat dilihat pada tabel dibawah ini:

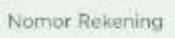
**Tabel 4.6** Pengujian Aplikasi Menggunakan Metode *Black Box*

Kasus dan Hasil Uji (Data Normal)			
Aktivitas Pengujian	Relasi Yang Diharapkan	Hasil Pengujian	Kesimpulan
Tekan Tombol Aplikasi 	Tombol ini digunakan untuk membuka aplikasi	Tombol Berfungsi	[√]
Masukan Email 	User dan Admin wajib mengisikan email untuk Registrasi dan Login	Input Berfungsi	[√]
Masukan Kata Sandi 	User dan Admin wajib mengisikan Kata Sandi untuk Login	Input Berfungsi	[√]
Masukan Nama Lengkap 	User wajib mengisikan Nama Lengkap untuk register	Input Berfungsi	[√]
Tekan Tombol Masuk 	Tombol ini digunakan masuk ke dalam aplikasi	Tombol Berfungsi	[√]
Tekan Tombol Register 	Tombol ini digunakan user untuk mendaftar	Tombol Berfungsi	[√]
Tekan Tombol Buat Akun 	Tombol ini berfungsi untuk masuk ke halaman registrasi	Tombol Berfungsi	[√]

Tekan Tombol Masuk 	Tombol ini berfungsi untuk masuk ke halaman <i>login</i>	Tombol Berfungsi	[√]
Tekan Tombol Eyes 	Tombol ini berfungsi untuk menyembunyikan dan menampilkan Kata Sandi	Tombol Berfungsi	[√]
Tekan Tombol Beranda 	Tombol ini digunakan masuk ke menu beranda	Tombol Berfungsi	[√]
Tekan Tombol Nabung 	Tombol ini digunakan masuk ke menu Nabung	Tombol Berfungsi	[√]
Tekan Tombol Transaksi 	Tombol ini digunakan masuk ke menu transaksi	Tombol Berfungsi	[√]
Tekan Tombol Profil 	Tombol ini digunakan masuk ke menu profil	Tombol Berfungsi	[√]
Tekan Tombol Riwayat 	Tombol ini digunakan admin untuk melihat riwayat transaksi <i>admin</i>	Tombol Berfungsi	[√]
Tombol <i>Switch</i> Tarik Tunai 	Tombol ini digunakan <i>admin</i> ketika penarikan tabungan ingin diaktifkan	Tombol Berfungsi	[√]
Tombol Tarik Tunai 	Tombol ini berfungsi untuk <i>user</i> melakukan tarik tunai tabungan	Tombol Berfungsi	[√]
Notifikasi <i>Reject</i> 	Notifikasi ditampilkan ketika transaksi di tolak	Notifikasi Berfungsi	[√]

Notifikasi Proses 	Notifikasi ditampilkan ketika transaksi sedang diproses	Notifikasi Berfungsi	[√]
Notifikasi <i>Approve</i> 	Notifikasi ditampilkan ketika transaksi berhasil dilakukan	Notifikasi Berfungsi	[√]
Notifikasi <i>Smartphone</i> 	Notifikasi akan keluar di <i>smartphone</i> jika melakukan transaksi	Notifikasi Berfungsi	[√]
Tekan Tombol Buy 	Tombol ini berfungsi untuk pembelian hewan kurban.	Tombol Berfungsi	[√]
Tekan Tombol Pesan 	Tombol ini berfungsi untuk melakukan pemesanan hewan kurban.	Tombol Berfungsi	[√]
Tekan Tombol Batal 	Tombol ini berfungsi untuk melakukan pembatalan transaksi pembelian hewan kurban.	Tombol Berfungsi	[√]
Masukan Nominal Saldo 	<i>Input</i> ini berfungsi untuk memasukan nominal saldo	<i>Input</i> Berfungsi	[√]
Tekan Tombol Bukti Transfer 	Tombol berfungsi untuk membuka kamera / <i>upload</i> bukti transfer saldo	Tombol Berfungsi	[√]
Tekan Tombol Saldo 1 	Tombol berfungsi untuk memasukan nominal Rp.50.000.00 secara <i>default</i> .	Tombol Berfungsi	[√]
Tekan Tombol Saldo 2 	Tombol berfungsi untuk memasukan nominal Rp.100.000.00 secara <i>default</i> .	Tombol Berfungsi	[√]

Tekan Tombol Saldo 3 	Tombol berfungsi untuk memasukan nominal Rp.500.000.00 secara default.	Tombol Berfungsi	[√]
Tekan Tombol Saldo 4 	Tombol berfungsi untuk memasukan nominal Rp.1.000.000.00 secara default.	Tombol Berfungsi	[√]
Tekan Tombol Kamera dan Berkas   Camera  Folder	Tombol berfungsi untuk <i>upload</i> bukti transfer, bisa menggunakan kamera dan file di berkas <i>smartphone</i>	Tombol Berfungsi	[√]
Tekan Tombol Proses  	Tombol berfungsi untuk memproses saldo tabungan yang sudah di <i>input</i>	Tombol Berfungsi	[√]
Tekan Tombol Setuju  	Tombol ini berfungsi untuk <i>admin</i> menyetujui transaksi tabungan <i>user</i> .	Tombol Berfungsi	[√]
Tekan Tombol Tolak  	Tombol ini berfungsi untuk <i>admin</i> menolak transaksi tabungan <i>user</i> .	Tombol Berfungsi	[√]
Tekan Tombol Biodata   Biodata      >	Tombol ini berfungsi untuk melengkapi atau mengubah biodata <i>user</i>	Tombol Berfungsi	[√]
Tekan Tombol Ubah Kata Sandi   Ubah Kata Sandi      >	Tombol ini berfungsi untuk mengubah kata sandi <i>user</i>	Tombol Berfungsi	[√]
Masukan Kata Sandi Baru   Kata Sandi Baru	<i>User</i> dapat mengubah kata sandi baru.	Input Berfungsi	[√]
Masukan Konfirmasi Kata Sandi   Konfirmasi Kata Sandi	Ulangi kata sandi baru untuk di konfirmasi	Input Berfungsi	[√]
Tekan Tombol Simpan  	Tombol ini berfungsi untuk menyimpan data baru	Tombol Berfungsi	[√]

Masukan No KTP 	<i>Input</i> ini berfungsi untuk memasukan no KTP	<i>Input</i> Berfungsi	[√]
Masukan No Rekening 	<i>Input</i> ini berfungsi memasukan no rekening	<i>Input</i> Berfungsi	[√]
Tombol Foto KTP 	Tombol ini berfungsi untuk menambahkan foto KTP	Tombol Berfungsi	[√]
Tombol Foto Rekening 	Tombol ini berfungsi untuk menambahkan foto buku rekening	Tombol Berfungsi	[√]
Tombol Pencarian 	Tombol ini berfungsi mencari/search data nasabah dan transaksi	Tombol Berfungsi	[√]
Tombol Riwayat Transaksi <i>User</i> 	Tombol ini berfungsi untuk menampilkan riwayat transaksi <i>user</i>	Tombol Berfungsi	[√]
Tombol Nasabah 	Tombol ini berfungsi untuk melihat data nasabah/ <i>user</i>	Tombol Berfungsi	[√]
Tombol Logout 	Tombol ini berfungsi untuk keluar aplikasi	Tombol Berfungsi	[√]

## **BAB 5**

### **PENUTUP**

#### **5.1 Kesimpulan**

Kesimpulan dari tugas akhir ini yaitu berhasil membuat aplikasi tabungan kurban digital menggunakan bahasa pemrograman Dart dan *framework* Flutter berbasis *mobile*. Diharapkan dapat membantu masyarakat Muslim untuk mengumpulkan uangnya membeli hewan kurban. Berikut adalah poin-poin kesimpulan lainnya:

- a)** “Qurbanku” adalah nama aplikasi tabungan kurban digital.
- b)** Aplikasi dapat berjalan menggunakan *smartphone* dengan sistem operasi Android.
- c)** Menjalakan transaksi menabung di aplikasi dan setelah mendekati satu bulan hari raya Idul Adha dapat melakukan tarik tunai tabungan atau membeli hewan kurban di aplikasi.

#### **5.2 Saran**

- a)** Untuk pengembangan berikutnya pada aplikasi Qurbanku baiknya di daftarkan dengan pihak ketiga yaitu Otoritas Jasa Keuangan (OJK) agar nasabah lebih aman dalam melakukan proses menabung maupun bertransaksi.
- b)** Transaksi menabung menggunakan *virtual account*.
- c)** Pengembang Qurbanku segera mempublikasikan aplikasi ke Google *Play Store*.

## **DAFTAR PUSTAKA**

- Abrahamsson, P., Salo, O., Ronkainen, J., & Warsta, J. (2017). *Agile software development methods: Review and analysis*. ArXiv.
- Abu Abdullah Muhammad bin Ismail Al Bukhari. (n.d.). Hadist Al-Bukhari.
- Apridayanti, M. (2013). Strategi Pemasaran Produk Tabungan *Qurban* Pada Lembaga Amil Zakat Swadaya Ummah Pekanbaru.
- Basyir,A.A.(2012). *Asas-Asas Muamala* (UUI).  
<http://thesis.umy.ac.id/index.php?opo=bibliography&id=57>
- Carolina, I., Pardede, A. M. H., & Supriyatna, A. (2019). Penerapan Metode *Extreme Programming* Dalam Perancangan Aplikasi Perhitungan Kuota Sks Mengajar Dosen. 3(1), 106–113. <https://doi.org/10.31227/osf.io/se6f9>
- Dart. (n.d.). *Important concepts*. Dart.Dev. <https://dart.dev/guides/language/language-tour#important-concepts>
- Doavers Development Team. (2016). Pengenalan Bahasa Pemrograman Dart untuk para Pemula. <https://www.doavers.com/blog/pengenalan-bahasa-pemrograman-dart-untuk-para-pemula>
- Fauziah, Y. (2014). Aplikasi Iklan Baris *Online* menggunakan Arsitektur REST *Web Service*. *Telematika*, 9(2). <https://doi.org/10.31315/telematika.v9i2.286>
- Hadi, A. M. A. M. bin A. Q. bin A., Munawar, S. Agil Husin, H., & Mukhtar, Ahmad Rifqi, H. (1994). Metode takhrij / Abu Muhammad Abdul Mahdi bin Abdul Qadir bin Abdul Hadi ; alih bahasa, H.S. Agil Husin Munawar, H. Ahmad Rifqi Muchtar (Dina Utama). <https://opac.perpusnas.go.id/DetailOpac.aspx?id=234822>
- Hansun, S., Kristanda, M. M., & Saputra, M. W. (2016). Pemrograman Android dengan Android Studio *IDE*. ANDI.
- Kurniansyah, M. I., & Sinurat, S. (2020). Sistem Pendukung Keputusan Pemilihan *Server Hosting* dan *Domain* Terbaik Untuk *WEB Server* Menerapkan Metode VIKOR. *Jurnal Sistem Komputer Dan Informatika (JSON)*, 2(September), 14–24. <https://doi.org/10.30865/json.v2i1.2450>

- Pooley Rob, W. P. (2003). *Applying UML* (1st ed.). Butterworth-Heinemann. <https://www.elsevier.com/books/applying-uml/pooley/978-0-7506-5683-2>
- Prabowo Pudjo, W. (2011). Menggunakan UML : UML Secara Luas Digunakan untuk Memodelkan Analisis & Desain Sistem Berorientasi Objek (W. Prabowo Pudjo (ed.)). <https://openlibrary.telkomuniversity.ac.id/pustaka/17035/menggunakan-uml-uml-sekara-luas-digunakan-untuk-memodelkan-analisis-desain-sistem-berorientasi-objek.html>
- Ratulangi, U. S., Kasaedja, B. A., Sengkey, R., & Lantang, O. A. (2014). Rancang Bangun *Web Service* Perpustakaan Universitas Sam Ratulangi. *Jurnal Teknik Elektro Dan Komputer*, 3(3), 38–50. <https://doi.org/10.35793/jtek.3.3.2014.5332>
- Rusdiana, L. (2018). *Dynamic Systems Development Method* dalam membangun Aplikasi Data Kependudukan Pada Kelurahan Rantau Pulut. *Jurnal Transformatika*, 16(1), 84. <https://doi.org/10.26623/transformatika.v16i1.859>
- Subhiyakto, E. R., Astuti, Y. P., Ningrum, N. K., Informatika, J. T., Komputer, F. I., Nuswantoro, U. D., & Semarang, K. (2018). 5. 834-2954-1-ED *Egia Rosi Subhiyakto 20% ok 161-166. 03(02)*, 161–166.
- Supriyadinatha, I. M. (2014). *Game Edukasi Puzzle Dewa Dewi Hindu Dan Tokoh Pewayangan Berbasis Android*. 16.
- Sutanta, E., & Mustofa, K. (2012). *Identifying The Needs of Web Service to Data Synchronization Between Information Systems as E-Government Ecosystem at Bantul-Yogyakarta*. *Teknik Informatika - STMIK Bandung*, 2(3), 20–26. <https://repository.ugm.ac.id/id/eprint/33043>
- Tidwell, D. (2000). *Web Services : The Web 's next Revolution*. IBM Developer Works, 1–17. <http://www.ibm.com/developerWorks/>
- WHITTEN, J. L., Bentley, L. D., & Ditman, K. D. (2004). *System analysis and design methods* (1st ed.). <http://balaiyanpus.jogjaprov.go.id/opac/detail-opac?id=1271>
- Wicaksono, A. (2019). BAB 2 Landasan Teori. Aplikasi Dan Analisis Literatur Fasilkom UI, 4–25.

# LAMPIRAN

## RANCANG BANGUN APLIKASI TABUNGAN KURBAN BERBASIS MOBILE

### ORIGINALITY REPORT



### PRIMARY SOURCES

1	<a href="#">repository.uin-suska.ac.id</a>	3%
	Internet Source	
2	<a href="#">www.coursehero.com</a>	2%
	Internet Source	
3	<a href="#">filestream.blogspot.com</a>	2%
	Internet Source	
4	<a href="#">widuri.raharja.info</a>	2%
	Internet Source	
5	<a href="#">Submitted to Universitas Brawijaya</a>	2%
	Student Paper	
6	<a href="#">docobook.com</a>	1%
	Internet Source	
7	<a href="#">ejurnal.stimata.ac.id</a>	1%
	Internet Source	
8	<a href="#">id.123dok.com</a>	1%
	Internet Source	
9	<a href="#">eprints.akakom.ac.id</a>	