BAB II
LANDASAN TEORI

2.1 Doa

Doa berasal dari bahasa arab *du'a*. Dalam Quran banyak sekali disebut lafaz *du'a* ini, yang mempunyai arti berbeda-beda antara lafaz satu dengan yang lainnya, antara lain: *al-ibadat*, yakni ibadatnya makhluk (*ibadat al-makhluq li al-khaliq*); *al-istilah* atau *al-istighatsah*, yaitu memohon pertolongan atau bantuan kepada Zat yang Maha kuasa; *al-niga'*: memanggil, yakni panggilan hamba terhadap Allah yang Maha mendengar; *al-su'al*, yakni permintaan atau permohonan dari makhluk yang rendah kepada Khaliq yang Maha tinggi. [1]. Doa dalam Islam memiliki banyak sumber diantaranya yaitu [2]:

1. Doa yang bersumber dari Al-Qur'an adalah doa-doayang diambil dari ayat-ayat suci Al-Qur'an, contohnya: doa mohon ampunan (doa Nabi Yusuf AS) yang terdapat di surat Al-Anbiya' ayat 87.
2. Doa yang bersumber dari Hadist adalah doa-doa yang diambil dari Hadist-hadist Rasulullah SAW yang diriwayatkan oleh para sahabat Rasulullah SAW ,contohnya : doa mendapat kebaikan di dunia dan di akhirat yang diriwayatkan oleh Muttafaq 'Alaih.

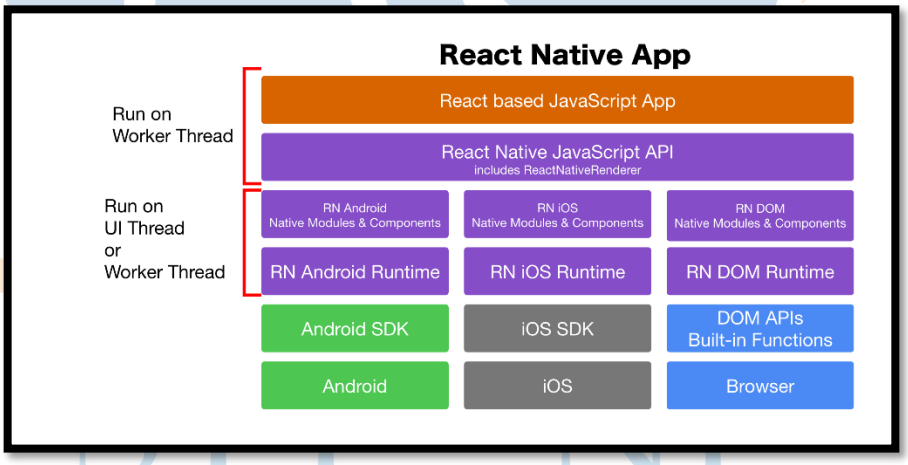
2.2 Android

Android merupakan sistem operasi yang banyak digunakan pada perangkat bergerak yang dewasa ini sangat terkenal dan populer digunakan pada ponsel cerdas. Android juga merupakan *platform* pemrograman yang dikembangkan oleh Google untuk ponsel cerdas dan perangkat seluler lainnya, misalnya tablet. Android bisa berjalan di beberapa macam perangkat yang dikembangkan oleh banyak vendor ponsel cerdas yang berbeda. Android menyertakan paket pengembangan perangkat lunak untuk penulisan kode asli dan perakitan modul perangkat lunak dalam membuat aplikasi bagi pengembang Android. Selain menyediakan paket pengembangan aplikasi Android, Android juga menyediakan pasar untuk mendistribusikan aplikasi yang telah selesai dikembangkan. Dengan lengkapnya

fasilitas yang disediakan oleh Android, dapat dikatakan bahwa secara keseluruhan, Android menciptakan ekosistem sendiri [3].

2.3 React Native

React Native adalah *framework* Javascript yang digunakan untuk membuat aplikasi *native* yang mampu berjalan di *platform* Android dan iOS. *Framework* ini berbasis React JS yang merupakan *framework* Javascript buatan dari Facebook yang digunakan untuk membuat tampilan aplikasi yang berfokus pada aplikasi *mobile*. [4]. *Framework* ini dapat membantu dalam membangun aplikasi mobile secara *hybrid*, dimana sebenarnya aplikasi yang dibangun berdasarkan HTML5. Dengan *framework* React Native memungkinkan membangun di dua sistem operasi Android dan IOS sekaligus hanya dengan bahasa pemrograman Javascript tanpa harus menggunakan bahasa asli sistem operasi tersebut, yaitu Java atau Objective-C. Konsep dari React Native divisualisasikan pada gambar di bawah ini.



Gambar 2.1 Konsep React Native

Sumber dari: http://www.reactnative.com/react-native-dom/

Menggunakan *framework* React Native memiliki beberapa keuntungan tersendiri, yaitu [5] :

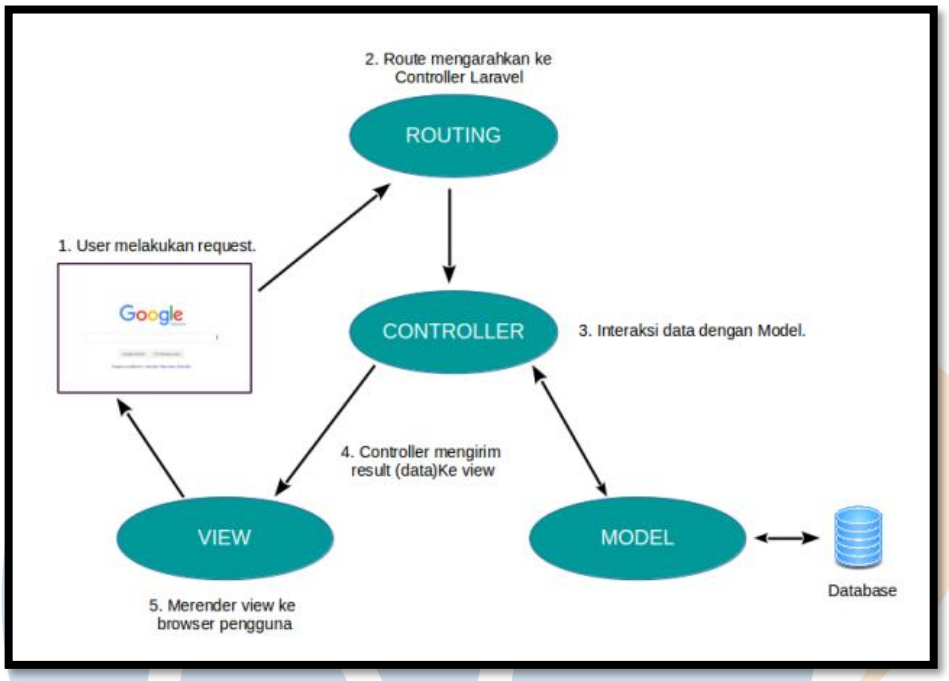
1. Sebagai salah satu alternatif dalam pengembangan aplikasi *mobile*, React Native dapat digunakan untuk mengembangkan aplikasi *mobile* dalam 2 *environment* dalam satu kali proses *development*.
2. Bahasa yang digunakan adalah javascript yang relatif mudah daripada harus belajar Java untuk pengembangan android dan Objective-C atau Swift untuk pegembangan iOS.
3. *Truly Native*, React Native sangat mendukung modul *native* seperti *push notifications, deep linking, native* UI *components* dan lain lain.
4. Dokumentasi yang ada sudah lengkap dan juga sudah banyak *library-library* yang bisa digunakan untuk pengembangan aplikasi sesuai kebutuhan.

2.4 Laravel

Laravel adalah PHP *open source framework* yang dibangun dengan *model view controller* dan dibekali dengan berbagai macam sintaks. Sejak diluncurkan tahun 2011, membuat website dengan Laravel banyak digemari oleh berbagai komunitas programmer di Github, sebelum kemudian menyebar ke seluruh dunia. *Framework* ini menyediakan beberapa jenis PHP *library* dan beberapa fungsi lain yang bisa memudahkan dalam menuliskan baris kode. *Framework* Laravel dibuat dengan tujuan memperindah cara untuk membuat website. Laravel terkenal sederhana dan elegan karena pembuatannya memang ditujukan untuk end-user. *Framework* ini juga terkenal dengan dokumentasinya yang lengkap dan selalu diperbarui. Setiap akan ada pembaruan ke versi terbaru selalu ada pembaharuan pada dokumentasi [6].

Laravel mengusung konsep MVC (*Model, View, Controller*) dalam pembangunannya, akan tetapi konsep MVC dalam *framework* laravel sedikit berbeda dari *framework* lainnya yang mengusung konsep ini juga. Di Laravel terdapat *routing* yang menyambungkan antara *request* dari user dan *controller*. Jadi

controller tidak langsung menerima *request* dari user [7]. Alur kerja dari konsep MVC Laravel bisa dilihat pada gambar 2.2.



Gambar 2.2. Alur MVC framework Laravel

Sumber dari: [7]

Terdapat 5 komponen utama pada *framework* Laravel, tiap komponennya memiliki fungsionalitas masing-masing, yaitu [7]:

1. *Routes*, berfungsi sebagai pemberi akses pada setiap *request* sesuai alur yang telah ditentukan.
2. *Controller*, adalah bagian yang menjadi penghubung antara *model* dan *view*. *Controller* memiliki perintah-perintah yang berfungsi untuk memproses bagaimana data ditampilkan dari *Model* ke *View* atau sebaliknya.
3. *Model*, merupakan sekumpulan data yang memiliki fungsi-fungsi untuk mengelola suatu table pada sebuah database. Struktur Pemodelan data pada laravel yakni memiliki fungsi yang terdiri dari *table*, *primary Key* dan *fillable*. Dimana ketiga fungsi tersebut harus di *protected*. Pada bagian

table harus diisi dengan nama *table* yang sesuai pada *database*, di bagian *primary Key* harus diisi sesuai *primary key* pada *table* tersebut dan padabagian *fillable* diisi dengan bagian-bagian yang mencakup dalam *table* tersebut.

4. *View*, merupakan file yang berisi kode html (*HyperText Markup Language*) yang berfungsi untuk menampilkan suatu data ke dalam *browser*. Format *view* pada laravel harus menggunakan istilah *blade*, contohnya seperti: view.blade.php.
5. *Migrations*, merupakan proses perancangan suatu tabel, dalam hal ini *migrations* berfungsi sebagai blueprint *database* atau dapat diistilahkan sebagai penyedia sistem kontrol untuk skema *database*.

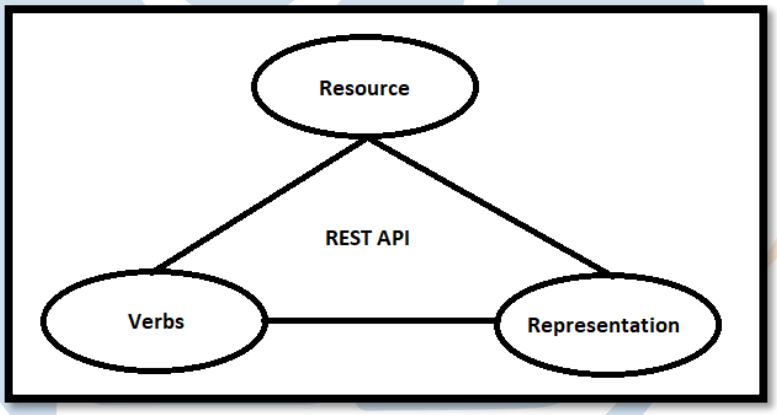
2.5 REST API

API merupakan *software interface* yang terdiri atas kumpulan instruksi yang disimpan dalam bentuk *library* dan menjelaskan bagaimana agar suatu *software* dapat berinteraksi dengan *software* lain. Penjelasan ini dapat dicontohkan dengan analogi apabila akan dibangun suatu rumah. Dengan menyewa kontraktor yang dapat menangani bagian yang berbeda, pemilik rumah dapat memberikan tugas yang perlu dilakukan oleh kontraktor tanpa harus mengetahui bagaimana cara kontraktor menyelesaikan pekerjaan tersebut. Dari Analogi tersebut, rumah merupakan *software* yang akan dibuat, dan kontraktor merupakan API yang mengerjakan bagian tertentu dari *software* tersebut tanpa harus diketahui bagaimana prosedur dalam melakukan pekerjaan tersebut. [8]. Terdapat tiga arsitektur API yang banyak digunakan, salah satunya adalah REST API.

REST (*REpresentational State Transfer*) merupakan sebuah teknik arsitektur software untuk sistem terdistribusi seperti *World Web Wide* . REST tidak memerlukan parsing XML dan tidak memerlukan sebuah header pesan ke dan dari penyedia layanan. Hal ini pada akhirnya menggunakan mengurangi penggunaan *bandwidth* [9].

REST memiliki 3 konsep utama yaitu:

1. *Nous/Resource*, *Noun* merupakan alamat dari suatu data.
2. *Verb*, *verb* merupakan metode yang ingin digunakan terhadap suatu data. metode yang terdapat pada *verb* terdiri dari :
    1. *GET*, Memperoleh data dari alamat yang dituju.
    2. *POST*, Mengirimkan data ke *resource*.
    3. *PUT*, Mengedit data yang diperoleh dari *resource*.
    4. *DELETE*, Menghapus data pada *resource* yang dituju.
3. Representation, ini merupakan format data yang akan dikirim oleh server, bisa berupa JSON atau XML.



Gambar 2.3. Komponen REST API

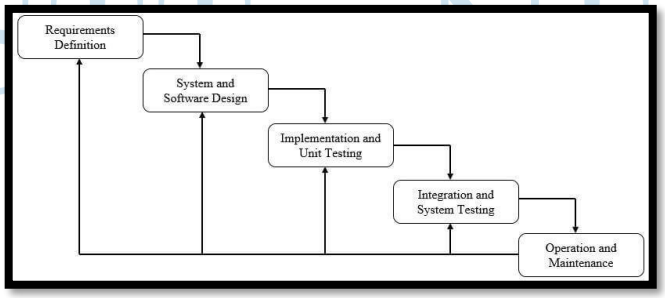
2.6 MYSQL

MySQL adalah salah satu jenis *database* server yang sangat terkenal. Kepopuleran nya disebabkan MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses *database* nya. MySQL termasuk jenis RDBMS (*Relational Database Management System*). Pada MySQL, sebuah *database* mengandung satu atau sejumlah tabel. Tabel terdiri atas sejumlah baris pada setiap baris mengandung satu atau beberapa kolom. Untuk mengelola *database* Mysql ada beberapa cara yaitu melalui *prompt* DOS (*tool command line*) [10]. Alasan MySQL begitu populer digunakan, adalah: [11].

1. Berlisensi *open source*, sehingga anda dapat menggunakannya secara gratis.
2. Merupakan program yang *powerful* dan menyediakan fitur yang lengkap.
3. Menggunakan bentuk standar bahasa data SQL.
4. Dapat bekerja dengan banyak sistem operasi dan dengan bahasa-bahasa pemrograman seperti PHP, PERL, C, C++, JAVA, dan lain-lain.
5. Bekerja dengan cepat dan baik, bahkan dengan data yang banyak.
6. Sangat mudah digunakan dengan PHP untuk pengembangan aplikasi web.
7. Mendukung banyak *database*, sampai 50 juga baris atau lebih dalam suatu tabel.
8. Dapat dikostumisasi sesuai dengan keinginan pengguna.

2.7 Metode Waterfall

Model *waterfall* adalah model pengembangan perangkat lunak yang paling sering digunakan. Model pengembangan ini bersifat linear dari tahap awal pengembangan sistem yaitu tahap perencanaan sampai tahap akhir pengembangan sistem yaitu tahap pemeliharaan. Tahapan berikutnya tak akan dilaksanakan sebelum tahapan sebelumnya selesai dilaksanakan dan tidak bisa kembali atau mengulang ke tahap sebelumnya. [12]. Sesuai dengan namanya air terjun (*waterfall*), metode ini menjalankan daur pembangunan aplikasi harus dengan langkah yang berurutan dari atas sampai bawah. Alur dari pengembangan metode *waterfall* bisa di lihat pada gambar di bawah ini.



Gambar 2.4. Diagram alur pengembangan metode waterfall

Sumber dari: [12]

Proses pengembangan dengan metode *waterfall* terdiri dari lima fase, yaitu [13] :

1. *Requirement* (analisis kebutuhan). Pengumpulan kebutuhan dengan fokus pada perangkat lunak, yang meliputi:domain informasi,fungsi yang dibutuhkan, unjuk kerja/performansi dan antarmuka. Hasilnya harus di dokumentasi dan di-*review* ke pelanggan.
2. *Design* (Desain / Rancangan), Ada empat atribut untuk program, yaitu:Struktur Data, Arsitektur perangkat lunak, Prosedur Detail, dan Karakteristik Antarmuka. Proses desain mengubah kebutuhan-kebutuhan menjadi bentuk karakteristik yang dimengerti perangkat lunak sebelum dimulai penulisan program. Desain ini harus terdokumentasi dengan baik dan menjadi bagian konfigurasi perangkat lunak.
3. *implementation* (Penerapan), Penterjemahan perancangan ke bentuk yang dapat dimengerti oleh mesin, dengan menggunakan bahasa pemrograman.
4. *Verification* (Integrasi & pengetesan), Setelah kode program selesai *testing* dapat dilakukan. *Testing* memfokuskan pada logika internal dari perangkat lunak, fungsi eksternal dan mencari segala kemungkinan kesalahan dan memeriksa apakah sesuai dengan hasil yang diinginkan.
5. *Maintenance* (Pemeliharaan), Merupakan bagian paling akhir dari siklus pengembangan dan dilakuka setelah perangkat lunak dipergunakan, meliputi kegiatan-kegiatan:
    i. *Corrective Maintenance* : Mengoreksi kesalahan pada perangkat lunak, yang baru terdeteksi pada saat perangkat lunak dipergunaka n. ii)
    ii. *Adaptive Maintenance* : Penyesuaian dengan lingkungan baru, misalnya sistem operasi atau sebagai tuntutan atas perkembangan sistem komputer, misalnya penambahan *printer driver*.
    iii. *Perfektive Maintenance* : Bila perangkat lunak sukses dipergunakan oleh pemakai. Pemeliharaan ditujukan untuk menambah

kemampuannya seperti memberikan fungsi - fungsi tambahan, peningkatan kinerja dan sebagainya.

2.8 UML

Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem [17]. UML terdiri dari beberapa diagram, diantaranya ada beberapa diagram yang paling sering digunakan yaitu [14]:

1. *Use Case* Diagram

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang Ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case* diagram dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem.

2. *Activity* Diagram

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang,bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity* diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity* diagram merupakan *state* diagram khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh Karena itu *activity* diagram tidak menggambarkan *behaviour internal* sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

3. *Class* Diagram

Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class* diagram menggambarkan struktur dan deskripsi *class, package* dan objek beserta hubungan satu sama lain seperti *containment*, pewarisan, asosiasi, dan lain-lain.

2.9 Blackbox Testing

Black box Testing merupakan Teknik pengujian perangkat lunak yang berfokus pada spesifikasi fungsional dengan mengabaikan struktur kontrol sehingga perhatiannya difokuskan pada informasi domain. *Black box Testing* memungkinkan pengembang *software* untuk membuat himpunan kondisi input yang melatih seluruh syarat-syarat fungsional suatu program [15]. Beberapa keuntungan yang diperoleh dari jenis testing ini antara lain :

1. Kesederhanaan: tes ini mudah dilakukan, karena seseorang berfokus pada *input* dan *output*. Penguji tidak perlu tahu bagaimana sistem bekerja secara internal, atau kode sumbernya, yang tidak dapat diakses. Dengan demikian, metode ini juga tidak mengganggu.
2. Kejelasan: waktu persiapan tes ini sangat singkat, karena sedikit pengetahuan tentang sistem yang dibutuhkan. Membuat dan menguji scenario membutuhkan sedikit waktu, karena ia mengikuti jalur pengguna, yang relatif sedikit, tergantung pada ukuran sistem.
3. Ketidakberpihakan: kita di sini mengikuti sudut pandang "pengguna" dan bukan sudut pandang "pengembang". Hasil pengujian netral: sistem berfungsi, atau tidak. Tidak ada kemungkinan kontestasi, seperti penggunaan satu proses daripada yang lain, tergantung pada pendapat pengembang.

2.10 Usability Testing

Usability berasal dari kata *usable* yang secara umum berarti dapat digunakan dengan baik. Sesuatu dapat dikatakan berguna dengan baik apabila kegagalan dalam penggunaannya dapat dihilangkan atau diminimalkan serta memberi manfaat dan kepuasan kepada pengguna [16].

Dalam interaksi antara manusia dengan komputer, *usability* atau juga disebut "ketergunaan" berkaitan dengan kemudahan dan keterbacaan informasi sekaligus pengalaman navigasi yang *user-friendly*. Pembahasan mengenai *interface* (antarmuka) yang *user-friendly* biasanya digunakan untuk halaman website atau perangkat lunak (*software*) agar dapat digunakan secara lebih efisien, mudah, dan memberikan pengalaman yang menyenangkan [17]. Salah satu metode untuk menguji suatu *Usability* dari perangkat lunak adalah *System Usability Scale* (SUS). Sebuah konsep pengujian *usability* yang diperkenalkan oleh John Brooke yaitu *System Usability Scale* Merupakan sebuah skala *usability* yang *reliabel* dan murah yang dapat digunakan untuk mengevaluasi *usability* sebuah sistem secara global. SUS berdasarkan pada skala kuesioner Likert dengan pertanyaan yang telah distandarisasi yang dapat memberikan nilai rata-rata usability dan kepuasan pengguna dengan skala 0–100 [18].

No.	Pernyataan John Brooke
1.	Saya pikir bahwa saya akan ingin lebih sering menggunakan website ini
2.	Saya menemukan bahwa website ini tidak perlu dibuat serumit ini
3.	Saya pikir website mudah untuk digunakan
4.	Saya pikir bahwa saya akan membutuhkan bantuan dari orang teknis untuk dapat menggunakan website ini
5.	Saya menemukan berbagai fungsi di website ini terintegrasi dengan baik
6.	Saya pikir ada terlalu banyak ketidaksesuaian di dalam website ini
7.	Saya bayangkan bahwa kebanyankan orang akan mudah untuk mempelajari website ini dengan sangat cepat
8.	Saya menemukan website ini sangat rumit untuk digunakan
9.	Saya merasa sangat percaya diri untuk menggunakan website ini
10.	Saya perlu belajar banyak hal sebelum saya bisa memulai menggunakan website ini

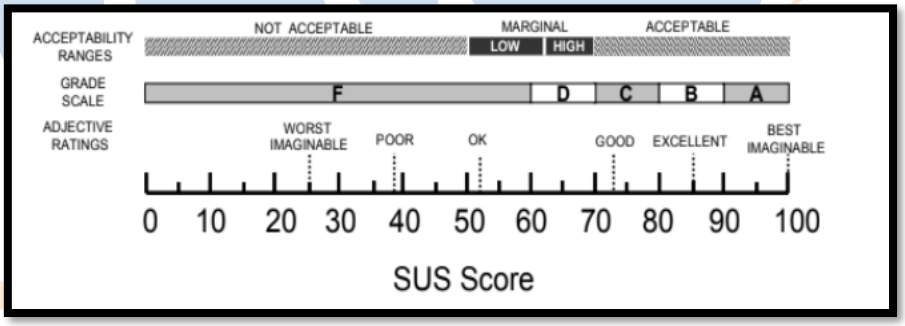
Gambar 2.5. Kuesioner John Brooke

Sumber dari: [18]

Penilaian SUS adalah sebagai berikut [18]:

1. Skala yang digunakan adalah sangat tidak setuju *(strongly disagree*) sampaisangat setuju (*strongly agree*) bernilai 1 sampai 5.
2. Untuk pernyataan bernomor ganjil dihitung dengan cara : nilai dari respon pengguna dikurangi dengan nilai 1.
3. Untuk pernyataan bernomor genap dihitung dengan cara : nilai 5 dikurangi dengan nilai dari respon pengguna.
4. Jumlahkan nilai respon yang telah dihitung pada poin 2 dan 3 diatas, dan kalikan hasilnya dengan nilai 2.5. Hasil perhitungan ini akan mengkonversi rentang nilai menjadi antara 0–100.

SUS dapat diinterpretasikan ke dalam rating sifat (*adjective rating*) untuk lebih memperjelas tingkat *usability* suatu sistem yang kemudian diterjemahkan kedalam tingkat penerimaan pengguna terhadap suatu sistem (*acceptability range*) untuk menentukan sistem dapat diterima atau tidak oleh pengguna. Gambar dari adjective rating bisa dilihat di bawah ini [18].



Gambar 2.6. SUS Score

Sumber dari: [18]

Pada gambar di atas menunjukkan bagaimana rating sifat membagi penilaian dari *software* yang diuji menjadi enam tingkatan, yaitu sangat buruk, buruk, cukup, bagus, sangat bagus, dan sempurna. Tiap tingkatan memiliki nilainya masing - masing, dimana jika penilaian kurang dari 60 poin maka aplikasi yang dibangun dianggap tidak dapat diterima.

2.11 Penelitian Terkait

Saat ini penelitian yang membahas tentang pengembangan aplikasi kumpulan doa sudah banyak dilakukan, salah satunya seperti yang dilakukan oleh Meriska Defriani dan Asep Saepudin pada tahun 2020 yang berjudul "RANCANG BANGUN APLIKASI KUMPULAN DO'A". Pada penelitian tersebut menggunakan metode waterfall pada pengembangannya dan dibangun untuk perangkat smartphone android menggunakan bahasa pemrograman Java. Fitur yang dibangun pada penelitian tersebut adalah list doa dan doa favorit. Penelitian lain yang membahas aplikasi doa dapat dilihat pada tabel di bawah ini.

Tabel 2.1 Penelitian Terkait

	Judul	Penulis	Tahun	Kesimpulan
a	Perancangan Aplikasi Doa Dan Wirid Harian Muslim Berbasis Android [19]	Nurdiansah dan Abdul Ibrahim	2016	Berdasarkan hasil yang didapatkan pada pengujian perangkat lunak yang menggunakan metode White Box, maka dapat disimpulkan bahwa aplikasi Doa dan Wirid Harian Muslim yang kami rancang dikatakan bebas dari kesalahan logika dengan Independent Path, Region dan Cyclomatic Complexity bernilai 10.
b	Aplikasi Pembelajaran Doa Harian Untuk Anak Usia Dini Berbasis Android [20]	Bodi Santoso dan Okky Pebriyani	2017	Berdasarkan hasil pengujian dari berbagai golongan, menyatakan bahwa pembelajaran doa

Tabel 2.1 Penelitian Terkait

	Judul	Penulis	Tahun	Kesimpulan
				harian untuk usia dini berbasis android, lebih menarik minat anak-anak untuk belajar doa harian karena dalam aplikasi doa harian ini menampilkan warna-warna yang cerah, audio serta gambar-gambar yang mendukung dari setiap bacaan doa, sehingga proses belajar sangat menyenangkan dan tidak membosankan.
c	Pemanfaatan Android Dalam Perancangan Aplikasi Kumpulan Doa [21]	Efmi Maiyana	2018	Dengan adanya pemanfaatan android dalam perancangan aplikasi kumpulan doa ini juga sangat membantu admin dalam menginputkan doa-doa yang diperlukan tanpa batas.
d	Analisa Aplikas Penghafal Doa-doa Islam Berbasis Android untuk Meningkatkan	Misbaul Ulum dan Muhammad Idris	2019	Menghasilkan sebuah aplikasi yang menarik untuk digunakan dan

Tabel 2.1 Penelitian Terkait

	Judul	Penulis	Tahun	Kesimpulan
	Minat Belajar Santri Madrasah Diniyah [22]			dapat digunakan oleh berbagai kalangan. Aplikasi ini menampilkan bacaan doa arab, audio hafalan, dan video hafalan yang dapat diputar dengan baik secara online. Aplikasi ini hanya bisa dibuka secara online karena pengetahuan peneliti dalam pembuatan aplikasi ini masih sangat kurang.
e	Rancang Bangun Aplikasi Kumpulan Do'a Sehari-hari Berbasis Mobile [23]	Meriska Defriani dan Asep Saepudin	2020	Dari sisi tujuan, Aplikasi berhasil dibuat dan memiliki fungsionalitas sesuai dengan yang diharapkan Aplikasi kumpulan do'a sehari-hari berbasis mobile ini dapat memberikan informasi do'a menurut Al-Quran dan menurut hadist juga bisa memainkan suara.

2.12 Perbandingan dengan Penelitian Terkait

Pada tabel dibawah ini terdapat perbandingan teknologi yang digunakan dan fitur yang dibangun pada penelitian ini dan penelitian lain yang membahas tentang aplikasi kumpulan doa.

Tabel 2.2 Perbandingan Penelitian Terkait

Penelitian		**a**	**b**	**c**	**d**	**e**	**Penelitian Ini**
Teknologi		Android, Java	Android, Java	Android, Java	Android, Java	Android, Java	React Native, Laravel, MYSQL, Android
Fitur	**List Doa**	✓	✓	✓	✓	✓	✓
	Pencarian Doa						✓
	Kategori Doa					✓	✓
	Doa Favorit				✓		✓
	Pengingat Doa						✓
	Berbagi Doa						✓
	Kelola Doa			✓			✓
	Kotak Saran						✓
	Audio/Vidio	✓					
	Quiz/ Pembelajaran		✓				

2.13 Perbandingan Aplikasi Beredar dengan Penelitian ini

Saat ini aplikasi kumpulan doa sudah banyak beredar pada *platform* android, berikut tabel perbandingan fitur yang akan dibangun pada penelitian ini dengan aplikasi yang sudah beredar.

Tabel 2.3 Perbandingan Aplikasi Beredar

Aplikasi		Apa Doa nya [1] https://play.google.com/store/apps/details?id=com.rahmadid.apadoanya	Dzikir & Doa [2] https://play.google.com/store/apps/details?id=com.bekalislam.dzikirndoa	Doa Doa [3] https://play.google.com/store/apps/details?id=com.tidakdijual.doadoa	**Penelitian Ini**
Fitur	**List Doa**	✓	✓	✓	✓
	Pencarian Doa	✓	✓		✓
	Kategori Doa	✓	✓	✓	✓
	Doa Favorit	✓		✓	✓
	Pengingat Doa	✓			✓
	Berbagi Doa	✓	✓		✓
	Kelola Doa				✓
	Kotak Saran			✓	✓
	Pengaturan Tulisan	✓			✓