

## **BAB II**

### **LANDASAN TEORI**

#### **2.1 Sistem Informasi**

##### **2.1.1 Sistem**

Menurut Sutabri (2005), Secara sederhana, suatu sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel yang terorganisir, saling berinteraksi, saling tergantung satu sama lain, dan terpadu.

Sebuah sistem terdiri atas bagian-bagian atau komponen yang terpadu untuk suatu tujuan. Model dasar dari bentuk sistem ini adalah adanya masukan, pengolahan, dan keluaran. Akan tetapi, sistem ini dapat dikembangkan hingga menyertakan media penyimpanan. Sistem dapat terbuka dan tertutup, akan tetapi sistem informasi biasanya adalah sistem terbuka, artinya sistem tersebut dapat menerima beberapa masukan dari lingkungan luarnya.

Suatu sistem pada dasarnya adalah sekelompok unsur yang erat hubungannya satu dengan yang lain, yang berfungsi bersama-sama untuk mencapai tujuan tertentu.

##### **2.1.2 Informasi**

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasi untuk digunakan dalam proses pengambilan keputusan. Sistem pengolahan informasi mengolah data menjadi informasi atau tepatnya mengolah data dari bentuk tak berguna menjadi berguna bagi penerimanya.

Fungsi utama informasi adalah menambah pengetahuan atau mengurangi ketidakpastian pemakai informasi. Informasi yang disampaikan kepada pemakai mungkin merupakan hasil data yang dimasukkan ke dalam dan pengolahan suatu model keputusan.

Nilai suatu informasi ini didasarkan atas 10 (sepuluh) sifat, yaitu mudah diperoleh, luas dan lengkap, ketelitian, kecocokan, ketepatan waktu, kejelasan, keluwesan, dapat dibuktikan, tidak ada prasangka, dan dapat diukur (Sutabri, 2005).

### **2.1.3 Sistem Informasi**

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar tertentu dengan laporan-laporan yang diperlukan.

Sistem informasi terdiri dari komponen-komponen yang disebut blok bangunan (*building block*), yang terdiri dari blok masukan, blok model, blok keluaran, blok teknologi, blok basis data, dan blok kendali. Sebagai suatu sistem, keenam blok tersebut masing-masing saling berinteraksi satu dengan yang lain membentuk satu kesatuan untuk mencapai sasaran (Sutabri, 2005).

Sistem informasi yang akan dibangun pada penelitian ini adalah sistem informasi untuk membantu proses kegiatan pada program *Islamic Character Building* di SD IT Fathona mulai dari kegiatan penilaian amaliyah harian hingga penghitungan rekapitulasi bulanan dan tahunan.

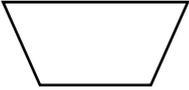
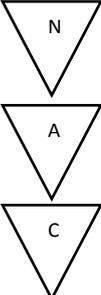
## 2.2 Flowchart

*Flowchart* adalah bagan (*chart*) yang menunjukkan alir (*flow*) di dalam program atau prosedur sistem secara logika (FitGerald *et al.*, 1981:278). Bagan alir digunakan terutama untuk alat bantu komunikasi dan untuk dokumentasi. Pada waktu akan menggambar suatu bagan alir, analis sistem atau pemrogram dapat mengikuti pedoman-pedoman sebagai berikut ini (Jogiyanto, 2005).

- a. Bagan alir sebaiknya digambar dari atas ke bawah dan mulai dari bagian kiri dari suatu halaman.
- b. Kegiatan di dalam bagan alir harus ditunjukkan dengan jelas.
- c. Harus ditunjukkan dari mana kegiatan akan dimulai dan dimana akan berakhirnya.
- d. Masing-masing kegiatan di dalam bagan alir sebaiknya digunakan suatu kata yang mewakili suatu pekerjaan.
- e. Masing-masing kegiatan di dalam bagan alir harus di dalam urutan yang semestinya.
- f. Kegiatan yang terpotong dan akan disambung di tempat lain harus ditunjukkan dengan jelas menggunakan simbol penghubung.
- g. Gunakanlah simbol-simbil bagan alir yang standar.

Bagan alir sistem (*system flowchart*) merupakan bagan yang menunjukkan arus pekerjaan secara keseluruhan dari sistem. Bagan ini menjelaskan urutan-urutan dari prosedur-prosedur yang ada di dalam sistem.

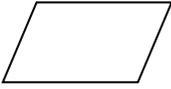
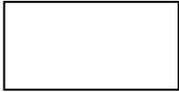
**Tabel 2.1** Simbol-Simbol *Flowchart* Sistem

Simbol	Keterangan
Simbol dokumen 	Menunjukkan dokumen input dan output baik untuk proses manual, mekanik atau komputer.
Simbol kegiatan manual 	Menunjukkan pekerjaan manual
Simbol simpanan offline 	File non-komputer yang diarsip urut angka (numerik)  File non-komputer yang diarsip urut huruf (alphabetical)  File non-komputer yang diarsip urut tanggal (cronological)
Simbol proses 	Menunjukkan kegiatan proses dari operasi program komputer
Simbol garis alir 	Menunjukkan arus dari proses

**Sumber:** (Jogiyanto, 2005:796)

Bagan alir program (*program flowchart*) merupakan bagan yang menjelaskan secara rinci langkah-langkah dari proses program. Bagan alir program dibuat dari derivikasi bagan alir sistem.

**Tabel 2.2** Simbol-Simbol *Flowchart* Program

Simbol	Keterangan
Simbol input/output 	Simbol input/output ( <i>input/output symbol</i> ) digunakan untuk mewakili data input/output
Simbol proses 	Simbol proses digunakan untuk mewakili suatu proses
Simbol garis alir 	Simbol garis alir ( <i>flow lines symbol</i> ) digunakan untuk menunjukkan arus dari proses
Simbol titik terminal 	Simbol titik terminal ( <i>terminal point symbol</i> ) digunakan untuk menunjukkan awal dan akhir dari suatu proses

**Sumber:** (Jogiyanto, 2005:802)

### 2.3 *Unified Modeling Language*

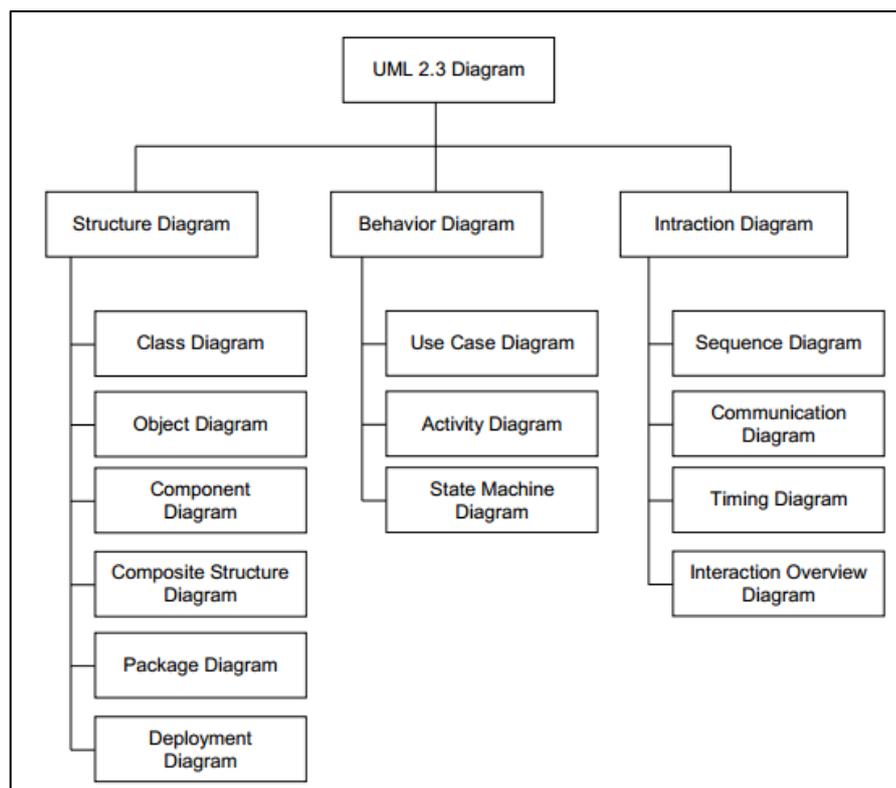
*Unified Modeling Language* (UML) muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak. UML merupakan bahasa visual untuk pemodelan dan komunikasi mengenai sebuah sistem dengan menggunakan diagram dan teks-teks pendukung. UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek.

Pada UML 2.3 terdiri dari 13 macam diagram yang dikelompokkan dalam 3 kategori. Berikut penjelasan singkat dari pembagian kategori tersebut.

1. *Structure diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

2. *Behavior diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
3. *Interaction diagrams* yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem (Rosa & Shalahuddin, 2016).

Berikut pembagian kategori dan macam-macam diagramnya.



**Sumber:** (Rosa & Shalahuddin, 2016)

**Gambar 2.1** Bagan UML

Dari 13 diagram yang ada, pada penelitian ini hanya menggunakan 3 diagram yaitu *class diagram*, *use case diagram*, dan *activity diagram*.

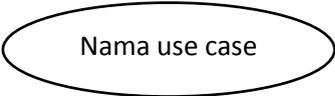
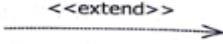
### 2.3.1 Use Case Diagram

*Use case* atau diagram *use case* merupakan pemodelan untuk kelakuan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Secara kasar, *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu.

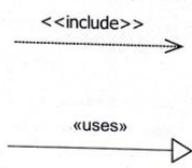
Syarat penamaan pada *use case* adalah nama didefinisikan sesimpel mungkin dan dapat dipahami. Ada dua hal utama pada *use case* yaitu pendefinisian apa yang disebut aktor dan *use case* (Rosa & Shalahuddin, 2016) .

Berikut adalah simbol-simbol yang ada pada diagram *use case* :

**Tabel 2.3** Simbol-Simbol Pada Diagram *Use Case*

Simbol	Deskripsi
<p><i>Use case</i></p> 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor.
<p>Aktor / <i>actor</i></p> 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat diluar sistem informasi yang akan dibuat itu sendiri.
<p>Asosiasi / <i>association</i></p> 	Komunikasi antara aktor dan <i>use case</i> yang berpartisipasi pada <i>use case</i> atau <i>use case</i> memiliki interaksi dengan aktor
<p>Ektensi / <i>extend</i></p> 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> di mana <i>use case</i> yang ditambahkan dapat berdiri sendiri walau tanpa <i>use case</i> tambahan itu;
<p>Generalisasi / <i>generalitation</i></p> 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah <i>use case</i> di mana fungsi yang satu adalah fungsi yang lebih umum dari lainnya.

**Tabel 2.4** Simbol-Simbol Pada Diagram *Use Case* Lanjutan

Simbol	Deskripsi
Menggunakan / <i>include</i> / <i>uses</i> 	Relasi <i>use case</i> tambahan ke sebuah <i>use case</i> di mana <i>use case</i> yang ditambahkan memerlukan <i>use case</i> ini untuk menjalankan fungsinya atau sebagai syarat dijalankan <i>use case</i> ini.

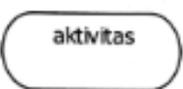
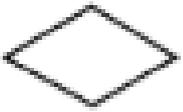
**Sumber:** (Rosa & Shalahuddin, 2016:156-158)

### 2.3.2 *Activity Diagram*

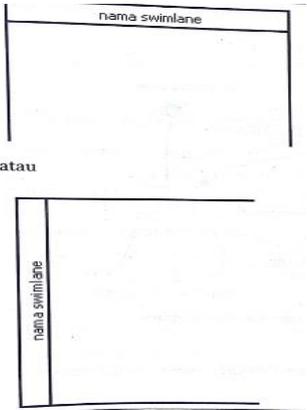
Diagram aktivitas atau *activity diagram* menggambarkan *workflow* (aliran kerja) atau aktivitas dsri sebuah sistem atau proses bisnis atau menu yang ada pada perangkat lunak. Yang perlu diperhatikan di sini adalah bahwa diagram aktivitas menggambarkan aktivitas sistem bukan apa yang dilakukan aktor, jadi aktivitas yang dapat dilakukan sistem (Rosa & Shalahuddin, 2016).

Berikut adalah simbol-simbol yang ada pada diagram aktivitas.

**Tabel 2.5** Simbol-Simbol Pada Diagram Aktivitas

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan 	Asosiasi percabangan di mana jika ada pilihan aktivitas lebih dari satu

**Tabel 2.6** Simbol-Simbol Pada Diagram Aktivitas Lanjutan

Simbol	Deskripsi
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
Swimlane 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi

**Sumber:** (Rosa & Shalahuddin, 2016:162-163)

### 2.3.3 Class Diagram

*Class diagram* atau diagram kelas menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki atribut dan metode atau operasi.

Diagram kelas dibuat agar pembuat program atau *programmer* membuat kelas-kelas sesuai rancangan di dalam diagram kelas agar antara dokumentasi perancangan dan perangkat lunak sinkron.

Kelas-kelas yang ada pada struktur sistem harus dapat melakukan fungsi-fungsi sesuai dengan kebutuhan sistem sehingga pembuat perangkat lunak atau *programmer* dapat membuat kelas-kelas di dalam program perangkat lunak sesuai dengan perancangan diagram kelas (Rosa & Shalahuddin, 2016).

Berikut adalah simbol-simbol yang ada pada diagram kelas:

**Tabel 2.7** Simbol-Simbol Pada Diagram Kelas

Simbol		Deskripsi
kelas		Kelas pada struktur sistem
	nama_kelas	
	+atribut	
	+operasi()	
antarmuka atau <i>interface</i> 		Sama dengan konsep <i>interface</i> dalam pemrograman berorientasi objek
asosiasi atau <i>association</i> 		Relasi antarkelas dengan makna umum, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
asosiasi berarah ( <i>directed association</i> ) 		Relasi antarkelas dengan makna kelas yang satu digunakan oleh kelas lain, asosiasi biasanya juga disertai dengan <i>multiplicity</i>
Generalisasi 		Relasi antarkelas dengan makna generalisasi-generalisasi (umum khusus)
kebergantungan atau <i>dependency</i> 		Relasi antarkelas dengan makna ebergantungan antarkelas
Agregasi atau <i>aggregation</i> 		Relasi antarkelas dengan makna semua-bagian ( <i>whole-part</i> )

**Sumber:** (Rosa & Shalahuddin, 2016:146-147)

## 2.4 Entity Relationship Diagram

Model *Entity Relationship* diperkenalkan pertama kali oleh P.P. Chen pada tahun 1976. Model ini dirancang untuk menggambarkan persepsi dari pemakai dan berisi objek-objek dasar yang disebut entitas dan hubungan antar entitas-entitas tersebut yang disebut *relationship*. Pada model ER ini semesta data yang ada di

dalam dunia nyata ditransformasikan dengan memanfaatkan perangkat konseptual menjadi sebuah diagram, yaitu diagram ER (*Entity Relationship*).

*Entity Relationship Diagram* (ERD) adalah pemodelan data utama yang membantu mengorganisasikan data dalam suatu proyek ke dalam entitas-entitas dan menentukan hubungan antar entitas beserta atributnya (Setyaningrum, 2013).

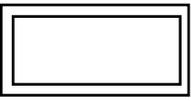
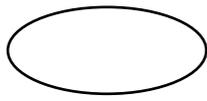
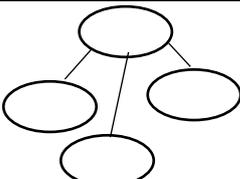
Untuk membuat ERD kita memerlukan entitas, relasi, dan atribut.

1. Entitas (*Entity*). Entitas adalah sesuatu apa saja yang ada di dalam sistem, nyata maupun abstrak di mana data tersimpan atau di mana terdapat data. Entitas disajikan dalam bentuk persegi panjang. Entitas diberi nama dengan kata benda dan dapat dikelompokkan dalam empat kelas, yaitu *role* (peran), *events* (kejadian), *locations* (lokasi), *tangible things/concepts* (sesuatu yang tidak nyata / konsep) (Setyaningrum, 2013:22).
2. Relasi (*Relationship*). Relasi adalah penghubung antara suatu entitas dengan entitas yang lain dan merupakan bagian yang sangat penting dalam mendesain basis data (Setyaningrum, 2013:23).
3. Atribut. Atribut adalah karakteristik dari entitas atau relasi, yang menyediakan penjelasan detail tentang entitas atau relasi tersebut. Nilai atribut merupakan suatu data aktual atau informasi yang disimpan pada suatu atribut di dalam suatu entitas atau relasi. Atribut digambarkan dalam bentuk oval (Setyaningrum, 2013:24).

Berikut ini merupakan simbol-simbol yang terdapat dalam ERD dengan

Chen :

**Tabel 2.8** Simbol-Simbol ERD Notasi Chen

Simbol	Nama	Simbol	Nama
	Entitas atau <i>entity</i>		Atribut <i>multi-value</i>
	<i>Weak entity</i>		Atribut kunci
	<i>Relationship</i> atau relasi		Atribut derivatif
	Atribut		Atribut komposit

**Sumber:** (Setyaningrum, 2013)

Tahap pertama pada desain sistem informasi menggunakan model ER adalah menggambarkan kebutuhan informasi atau jenis informasi yang akan disimpan dalam basis data. Teknik pemodelan ini dapat digunakan untuk menggambarkan setiap ontologi (yaitu gambaran dan klasifikasi dari istilah yang digunakan dan hubungan antar informasi) untuk wilayah tertentu.

Secara umum metodologi ERD sebagai berikut:

1. Menentukan entitas
2. Menentukan relasi
3. Menggambar ERD sementara
4. Mengisi kardinalitas
5. Menentukan kunci utama (*primary key*)
6. Menggambar ERD berdasarkan kunci
7. Menentukan atribut
8. Memetakan atribut

9. Menggambar ERD dengan atribut
10. Memeriksa hasil

## **2.5 Mobile-D**

Untuk mengatasi tantangan dalam pengembangan aplikasi mobile, telah dikembangkan sebuah pendekatan dari metode *agile* yang disebut *mobile-D*. Pendekatan ini didasari dari *Extreme Programming*, *Crystal Methodologies*, dan *Rational Unified Process*.

Pendekatan *mobile-D* dioptimalkan untuk pengembangan aplikasi *mobile* dengan anggota tim yang kurang dari 10 orang dengan tujuan membangun aplikasi *mobile* yang berfungsi penuh dalam jangka waktu yang singkat, yaitu kurang dari 10 minggu. *Mobile-D* telah dikembangkan oleh tiga perusahaan pengembang perangkat lunak *mobile*. Pendekatan ini berhasil mendapatkan sertifikat CMMI level 2.

Pendekatan *mobile-D* sudah diuji secara empiris dan dikembangkan lebih lanjut dalam empat studi kasus dalam *ENERGI (Industry-Driven Experimental Software Engineering Initiative) laboratory* di VTT, *The Technical Research Centre of Finland* (Abrahamsson, *et al*, 2004).

## **2.6 Alat Bantu Perangkat Lunak**

### **2.6.1 Framework Ionic**

*Framework ionic* adalah sekumpulan teknologi yang dikembangkan untuk membangun aplikasi *mobile hybrid* yang *powerfull*, cepat, mudah dan juga memiliki tampilan yang menarik. Ionic menggunakan AngularJS sebagai

*framework* berbasis web dan menggunakan Cordova untuk membangun aplikasi *mobile*.

*Framework* ionic merupakan kerangka kerja yang menyediakan sejumlah kontrol antarmuka pengguna yang umum digunakan di aplikasi *mobile*. Ionic juga memiliki *Command Line Tool* (CLI) yang membantu pengembang dalam mengembangkan aplikasi. *Tools* CLI ini dapat membantu untuk memulai proyek, *preview*, *build*, dan *deploy* aplikasi (Anditya dan Ilhami, 2015).

### **2.6.2 AngularJS**

*AngularJS* adalah proyek *open-source* dari Google yang telah sangat populer di kalangan pengembang aplikasi web. Angular membantu para pengembang dalam menuliskan baris-baris kode program secara cepat dan memiliki struktur yang baik. *Framework* berbasis web seperti *AngularJS* memungkinkan membangun aplikasi kompleks di browser (Anditya dan Ilhami, 2015).

### **2.6.3 Cordova**

*Cordova* merupakan layer yang mengatur komunikasi antara jendela browser dan *native* API. *Cordova* menyediakan banyak sekali fitur dan plugin yang dapat kita gunakan dalam mengembangkan aplikasi. *Cordova* adalah proyek *open-source* Apache yang memiliki komunitas yang cukup besar. Adobe merupakan pengembang utama *framework* ini. *Cordova* berada di bawah lisensi Apache 2.0 (Anditya dan Ilhami, 2015).

#### **2.6.4 PHP**

Jika diartikan, sebenarnya PHP memiliki beberapa pandangan dalam mengartikannya, akan tetapi kurang lebih PHP dapat diambil arti sebagai *PHP: Hypertext Preprocessor*. Ini merupakan bahasa yang hanya dapat berjalan pada server dan hasilnya dapat ditampilkan pada client. PHP adalah bahasa program yang berbentuk skrip yang diletakkan di dalam server web (Nugroho, 2009:113).

#### **2.6.5 MySQL**

MySQL (*My Structured Query Language*) adalah sebuah program pembuat dan pengelola *database* atau yang sering disebut DBMS (*DataBase Management System*). MySQL juga merupakan program pengakses *database* yang bersifat jaringan, sehingga dapat digunakan untuk aplikasi *Multi User* (banyak pengguna). Kelebihan lain dari MySQL adalah menggunakan bahasa query (permintaan) standar SQL (*Structured Query Language*). SQL adalah suatu bahasa permintaan yang terstruktur, SQL telah distandarkan untuk semua program program pengakses *database* (Nugroho, 2009:91).

#### **2.6.6 Web Service**

*Web service* (kadang disebut layanan aplikasi adalah layanan-layanan yang disediakan *web server* yang memungkinkan peranti elektronik bisa memanfaatkannya. Elemen *web service* berupa WSDL, SOAP, dan UDDI (Kadir & Triwahyuni, 2013:331-332).

1. WSDL. *Web Service Definition Language* (WSDL) adalah bahasa berbasis XML yang menjelaskan layanan-layanan web dan cara untuk mengaksesnya.
2. SOAP. *Simple Object Access Protocol* (SOAP) adalah protokol berbasis XML yang memungkinkan pertukaran informasi melalui HTTP.
3. UDDI. *Universal Description, Discovery And Integration* (UDDI) adalah suatu kerangka yang tidak bergantung platform yang ditujukan untuk menyimpan informasi mengenai layanan-layanan web, menjadi direktori antarmuka layanan-layanan web yang dijabarkan dengan WSDL, dan berkomunikasi melalui SOAP.

## **2.7 Penelitian Sebelumnya**

Menurut Rizki dkk (2014) dalam jurnalnya yang berjudul Implementasi Metode *Mobile-D* untuk Pembuatan Ensiklopedia Kebudayaan Kota Palembang Berbasis Android menyatakan bahwa dengan mengimplementasikan *Mobile-D* dapat membuat eksiklopedia kebudayaan kota Palembang berbasis android dengan lebih terstruktur. Juga dengan menerapkan *Mobile-D*, Rizki dkk dapat menyelesaikan pembuatan eksiklopedia dengan waktu yang singkat karena dengan *Mobile-D* rizki dkk dapat merencanakan kegiatan yang dilakukan pada tahap selanjutnya.

Menurut Rachman dkk (2015) dalam penelitiannya yang berjudul Implementasi Metode *Mobile-D* Pembuatan Intellectual Abilities Plus Berbasis *Mobile* menyatakan bahwa dengan *Mobile-D* dapat membuat perangkat lunak dengan lebih terstruktur.

Menurut Ependi (2017) dalam penelitiannya yang berjudul *Mobile Application Monitoring* Pengisian Uang Anjungan Tunai Mandiri PT Bank Mandiri Cabang Palembang menyatakan bahwa hasil penelitiannya terdapat kesesuaian dengan yang diinginkan dengan dibuktikan dari hasil pengembangan yang dilakukan secara sistematis dan terstruktur menggunakan *Mobile-D*.

Menurut Oktaviani dan Sa'uda (2017) dalam penelitiannya yang berjudul Aplikasi Simulasi Soal Tes Kemampuan Dasar Akademik dan Tes Kemahiran Berbahasa Inggris menyatakan bahwa dengan mengimplementasikan *Mobile-D*, pembuatan aplikasi simulasi soal ini menjadi lebih terstruktur.

Menurut Tompoh dkk (2016) dalam penelitiannya yang berjudul Rancang Bangun Aplikasi Pemesanan Menu Makanan Restoran Berbasis Android mengatakan bahwa dengan menggunakan *framework* ionic dan *framework* cordova menjadikan proses pembangunan aplikasi menjadi baik dan cepat.

Menurut Syaripul dan Bachtiar (2016) dalam penelitiannya yang berjudul Visualisasi Data Interaktif Data Terbuka Pemerintah Provinsi DKI Jakarta: Topik Ekonomi Dan Keuangan Daerah mengatakan bahwa menggunakan *framework* ionic 2 memiliki keunggulan bisa di-*install* tidak terbatas pada satu jenis *platform* saja. Namun, disebutkan juga adanya kelemahan yang terletak pada *hybrid platform* ionic 2 yang memiliki performa kurang baik untuk beberapa *case* pengguna seperti *crash*.

Menurut Dewanti dan Permana (2017) dalam penelitiannya yang berjudul Pengembangan Aplikasi *Hybrid* Menggunakan Ionic 2 *Framework* Dan Angular 2 mengatakan bahwa ionic sebagai teknologi baru, namun sangat berkembang, adalah sederhana dan mudah digunakan. Ionic 2 memungkinkan pengembang untuk

membuat kode pada TypeScript, HTML dan CSS untuk menghasilkan aplikasi lintas platform *hybrid* yang secara penuh memiliki fungsional bagi pengguna.

Menurut Mustagbal dkk (2015) dalam penelitiannya yang berjudul Pengujian Aplikasi Menggunakan *Blackbox Testing Boundary Value Analysis* (Studi Kasus : Aplikasi Prediksi Kelulusan SNMPTN) mengatakan bahwa metode *black box testing* merupakan salah satu metode yang mudah digunakan karena hanya memerlukan batas bawah dan batas atas dari data yang diharapkan.

Menurut Khan (2011) pada penelitiannya yang berjudul *Different Approaches To Black Box Testing Technique For Finding Errors* mengatakan bahwa *black box testing* merupakan teknik pengujian yang mengabaikan mekanisme internal atau struktur sistem dan berfokus pada output yang dihasilkan sebagai respons terhadap input dan kondisi eksekusi. *Black box testing* dilakukan untuk mengevaluasi kepatuhan terhadap sistem dengan persyaratan fungsional yang ditentukan dan hasil yang diprediksi.

Pada penelitian ini akan dibangun sebuah sistem informasi *Islamic Character Building* pada SD IT Fathona dengan menggunakan *framework ionic* agar dapat digunakan pada dua platform dengan satu kali pengkodean. Sistem informasi ini akan dibangun sesuai dengan metode pengembangan sistem *Mobile-D* agar menghasilkan sistem informasi yang lebih terstruktur. Lalu sebelumnya diproduksi sistem informasi ini akan diuji menggunakan pengujian *boundary value analysis* dari *blackbox* pada tiap-tiap input yang ada pada sistem informasi yang akan dibangun.