**TUGAS AKHIR**

**PERANCANGAN SISTEM INFORMASI LAYANAN JASA *EVENT ORGANIZER* BERBASIS WEB**

Ditulis untuk Memenuhi Sebagian Persyaratan Akademik

Untuk Memperoleh Gelar Strata Satu Sarjana Komputer (S.Kom.)

**Oleh:**

|  |  |  |
| --- | --- | --- |
| **NAMA** | **:** | **Irene Andriani** |
| **NIM** | **:** | **00000013440** |



**PROGRAM STUDI SISTEM INFORMASI**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS PELITA HARAPAN**

**TANGERANG**

**2019**

# BAB I PENDAHULUAN

## 1.1 Latar Belakang

Seiring dengan berkembangnya teknologi dan komunikasi pada era globalisasi ini, internet memegang peran yang penting dalam kehidupan sehari-hari. Dengan adanya internet sebagai sumber informasi yang luas, segala sesuatu pun dapat diperoleh dengan praktis dan cepat. Selain memperoleh informasi, internet juga banyak dimanfaatkan sebagai sarana untuk berbisnis. Banyak orang yang menjual berbagai macam produk ataupun menyediakan berbagai macam jasa melalui internet; salah satu contohnya adalah jasa *event organizer*.

*Event Organizer* (EO) dapat didefinisikan sebagai sebuah profesi yang dijalankan oleh sekelompok orang dengan tugasnya masing-masing yang telah ditetapkan dalam rangka mencapai satu tujuan tertentu. Tujuan tersebut yaitu menghasilkan acara yang sesuai dengan permintaan klien. EO sendiri terdiri dari berbagai macam jenis; ada yang bergerak di bidang olahraga, seni, pameran, maupun acara yang bersifat pribadi seperti pesta pernikahan, ulang tahun, dan acara lainnya [1]. Melihat perkembangan EO yang semakin pesat, peluang untuk melakukan bisnis ini pun cukup terbuka. Jasa-jasa EO pun semakin dibutuhkan, terutama bagi orang-orang di zaman sekarang yang ingin semuanya serba praktis dan tidak ingin mengurusi hal-hal yang rumit.

Dalam penyelenggaraan sebuah acara, banyak hal yang perlu dipertimbangkan seperti lokasi, anggaran, dekorasi, dan hal lainnya yang berkaitan dengan acara. Penggunaan jasa EO yang sesuai sangatlah penting agar acara dapat berjalan dengan lancar dan sukses. Berdasarkan hasil survei yang telah dilakukan terhadap 111 responden dari berbagai provinsi di Indonesia, 36,9% di antaranya pernah menggunakan jasa EO namun 21,6% di antaranya masih menghadapi kesulitan dalam mencari jasa EO. Kesulitan tersebut dikarenakan penyediaan informasi yang masih tergolong minim sehingga sulit mencari EO yang terpercaya dan sesuai dengan acara, tema, dan *budget* (Lampiran 1). Bagi para EO yang baru memulai bisnisnya, tentu juga mengalami kesulitan dalam mendapatkan kepercayaan *user* untuk menggunakan jasanya.

Melihat permasalahan yang ada, riset terhadap media pun dilakukan dan didapatkan beberapa *website* yang secara spesifik hanya menampilkan informasi tentang satu EO. Hal tersebut menyebabkan proses pencarian jasa EO menjadi kurang efektif dimana proses tersebut dilakukan satu per satu melalui internet maupun di berbagai media sosial. Ditemukan juga sebuah *website* bernama sejasa.com yang menyediakan berbagai macam jasa termasuk EO, namun fitur yang disediakan kurang membantu dimana *user* harus menjawab pertanyaan yang telah disediakan terlebih dahulu dan kemudian menunggu para EO untuk menghubunginya. Selain itu, masih belum ditemukan sebuah sistem yang secara komprehensif menampilkan informasi mengenai berbagai jasa EO yang ada di Indonesia.

## 1.2 Perumusan Masalah

Berdasarkan latar belakang diatas, maka penelitian ini memiliki rumusan masalah sebagai berikut:

1. Para *user* yang mengalami kesulitan dalam memilih EO yang tepat dikarenakan banyaknya jasa-jasa EO yang ada di internet maupun di berbagai media sosial.
2. Penyediaan informasi mengenai EO yang ada di Indonesia untuk pencarian jasa EO masih kurang lengkap dan jelas seperti informasi harga dari paket yang disediakan, *portfolio* dari acara yang pernah diselenggarakan, dan juga testimoni dari klien yang pernah menggunakan jasanya.
3. Para EO yang mengalami kesulitan pada tahap awal untuk mempublikasikan jasa yang disediakannya.

## 1.3 Pembatasan Masalah

Penelitian ini memiliki batasan-batasan masalah sebagai berikut:

1. Sistem ini berbasis *web* dan dapat diakses dalam berbagai *browser*.
2. Sistem ini akan menggunakan bahasa pemrograman berupa PHP dan MySQL.
3. Sistem yang dirancang akan menampilkan informasi yang lengkap mengenai jasa-jasa yang disediakan oleh berbagai EO yang ada di Indonesia.
4. Sistem ini dapat menjadi perantara yang mempermudah interaksi antar para EO dan *user*.
5. Sistem tidak mencakup proses pembayaran.

## 1.4 Tujuan dan Manfaat Penelitian

Berdasarkan latar belakang yang telah dijabarkan, maka penelitian ini memiliki tujuan dan manfaat sebagai berikut:

### 1.4.1 Tujuan Penelitian

Berikut tujuan dari penelitian ini:

1. Menghasilkan sistem yang dapat mempermudah *user* dalam mencari jasa EO yang sesuai dengan acara yang ingin diselenggarakan.
2. Menghasilkan sistem berbasis web yang dapat menyediakan informasi mengenai jasa-jasa dari berbagai EO yang ada di Indonesia.
3. Menghasilkan sistem yang dapat menjadi sarana bagi para EO untuk mempublikasikan jasa yang disediakannya.

### 1.4.2 Manfaat Penelitian

Berikut manfaat dari penelitian ini:

1. Sistem dapat mempermudah *user* dalam melihat dan memperoleh informasi yang lengkap dan jelas mengenai jasa yang disediakan oleh berbagai EO.
2. Sistem dapat menjadi sarana untuk mempublikasikan jasa EO yang ada sehingga dapat dikenal secara lebih luas.
3. Dengan adanya sistem ini dapat meningkatkan perkembangan bisnis EO di Indonesia.

## 1.5 Metodologi Penelitian

Berikut adalah beberapa metode penelitian yang akan digunakan dalam melakukan penelitian ini:

### 1.5.1 Metode Pengumpulan Data

Berikut metode pengumpulan data yang akan digunakan dalam penelitian ini:

1. Observasi

Observasi dilakukan untuk mengamati secara langsung bagaimana proses pencarian dan penggunaan layanan jasa EO pada saat ini.

1. Studi Literatur

Studi literatur dilakukan untuk mencari referensi mengenai *website* atau hasil penelitian sebelumnya dari berbagai macam literatur, dan juga mempelajari teori-teori yang berkaitan dengan *tools* yang dapat digunakan dalam menunjang perancangan sistem.

1. Kuesioner

Kuesioner akan dibuat dan kemudian disebarkan kepada beberapa orang untuk mengetahui respon mereka terhadap adanya sistem yang dapat membantu pencarian jasa EO.

### 1.5.2 Metodologi Pengembangan Sistem

Pengembangan sistem dalam penelitian ini akan menggunakan metode *Rapid Application Development* (RAD) atau *Rapid Prototyping*. RAD merupakan sebuah model yang melibatkan pengembangan dan pembangunan *prototype iterative* dimana proses perancangan dan perbaikan sistem dapat dilakukan secara berulang karena adanya respon dari *user.* Metode ini dipilih karena adanya keterbatasan waktu untuk melaksanakan perancangan sistem, selain itu keterlibatan *user* juga dibutuhkan untuk pengembangan sistem yang lebih baik.

## 1.6 Sistematika Penulisan

Sistematika penulisan pada penelitian ini akan memuat uraian secara garis besar dari isi penelitan yang terdiri dari lima bab dan dilengkapi dengan beberapa lampiran. Berikut adalah penjabaran dari masing-masing bab:

**BAB I PENDAHULUAN**

Dalam bab ini akan menjelaskan tentang latar belakang, rumusan masalah, batasan masalah, tujuan dan manfaat penelitian, metode penelitian yang akan digunakan, dan sistematika penulisan.

**BAB II LANDASAN TEORI**

Dalam bab ini akan membahas tentang berbagai teori dan definisi-definisi yang menjadi dasar dalam proses penyusunan penelitian ini.

**BAB III SISTEM SAAT INI**

Dalam bab ini akan dijelaskan gambaran dari analisis sistem saat ini beserta kendala-kendala yang dihadapi.

**BAB IV SISTEM USULAN**

Dalam bab ini akan menjelaskan tentang analisis kelayakan sistem yang diusulkan, pemodelan dari sistem usulan, tahapan implementasi dan pengujian sistem.

**BAB V KESIMPULAN DAN SARAN**

Dalam bab ini akan dipaparkan kesimpulan dari penelitian yang dilakukan dan juga saran-saran yang dapat menjadi bahan pertimbangan untuk pengembangan sistem yang lebih baik.

# BAB II LANDASAN TEORI

## 2.1 *System Development Life Cycle* (SDLC)

SDLC merupakan sebuah metodologi yang digunakan untuk mengembangkan sistem informasi yang dapat mendukung kebutuhan bisnis, melakukan desain terhadap suatu sistem, membangun sistem tersebut, dan menyampaikan sistem yang telah dirancang kepada *user* [2]. SDLC terdiri dari beberapa fase yang dimulai dari fase perencanaan (*planning*), fase analisis (*analysis*), fase desain (*design*), dan fase implementasi (*implementation*).

### 2.1.1 Planning

Fase perencanaan atau *planning* merupakan proses untuk memahami mengapa sistem informasi tersebut harus dibangun dan menentukan bagaimana cara pelaksanaannya yang tepat [2]. Pada fase ini, dilakukan proses identifikasi terhadap nilai bisnis (*business value*), dan kebutuhan-kebutuhan bisnis (*business requirements*) yang dapat mendukung pembangunan sistem sehingga sistem yang akan dibangun dapat berguna bagi organisasi. Proses analisis kelayakan (*feasibility analysis*) juga dilakukan pada fase ini dimana proses tersebut mencakup masalah teknis, ekonomi, dan organisasi yang dapat menentukan kelangsungan proyek.

### 2.1.2 Analysis

Fase analisis atau *analysis* merupakan proses untuk mengetahui siapa yang akan menggunakan sistem, apa saja yang dapat dilakukan oleh sistem, dimana dan kapan sistem tersebut dapat digunakan [2]. Pada fase ini, dilakukan analisa terhadap kondisi sistem yang sedang berjalan pada saat ini (*as-is system*), mengidentifikasi masalah-masalah yang terdapat di dalamnya, dan kemudian mengembangkan sebuah konsep yang dapat digunakan untuk membangun sistem yang baru (*to-be system*).

### 2.1.3 Design

Fase desain atau *design* merupakan proses untuk menentukan bagaimana sistem akan beroperasi dengan melihat spesifikasi dalam sistem seperti perangkat keras (*hardware)*, perangkat lunak (*software*), jaringan infrastruktur (*network infrastructure*), tampilan antarmuka (*user interface*), program, *database* dan data-data lain yang diperlukan [2]. Pada fase ini, akan ditentukan siapakah yang membangun sistem dan kemudian dilanjutkan dengan pengembangan desain arsitektur sistem yang mencakup perangkat keras, perangkat lunak, dan jaringan infrastruktur yang digunakan. Selain itu, desain antarmuka dilakukan untuk menentukan bagaimana *user* akan menggunakan sistem yang dibangun. Penentuan akan dimana dan apa saja data yang akan disimpan juga dilakukan pada fase ini.

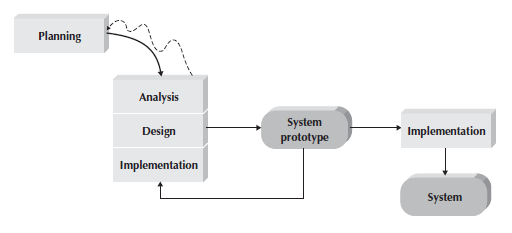
### 2.1.4 Implementation

Fase implementasi atau *implementation* adalah fase terakhir pada SDLC dan merupakan proses dimana sistem telah dibangun [2]. Pada fase ini, sistem yang dibangun akan diuji terlebih dahulu untuk memastikan bahwa sistem dapat bekerja sesuai dengan yang direncanakan. Setelah pengujian sistem, maka akan dilakukan proses instalasi dimana sistem yang lama akan dimatikan dan diganti dengan sistem yang baru dengan menggunakan metode konversi yang ditentukan oleh organisasi.

## 2.2 *Rapid Application Development* (RAD)

Metodologi yang akan digunakan untuk mengembangkan sistem pada penelitian ini adalah RAD. RAD memfokuskan pada proses pengembangan sistem yang dilakukan secara lebih cepat dan kemudian disampaikan kepada *user* [2]. Dengan hal ini, dapat meningkatkan pengertian *user* akan sistem yang dirancang. RAD terbagi lagi menjadi tiga metode yaitu *phased development*, *prototyping* dan *throwaway prototyping*.

Proses perancangan sistem dalam penelitian ini akan menggunakan metode *prototyping*. Dalam metode ini, akan dilakukan proses analisis, desain, dan implementasi secara bersamaan. Ketiga proses tersebut akan dilakukan secara berulang dalam suatu siklus hingga sistem dapat terselesaikan secara menyeluruh [2]. Metode ini memiliki kelebihan yaitu penyediaan sistem yang dapat berinteraksi dengan *user* dilakukan dalam waktu yang singkat, meskipun sistem masih belum dapat digunakan secara luas. Dengan metode ini, *user* juga dapat memastikan pengembangan sistem telah dilakukan dengan baik sehingga tidak ada lagi penundaan dalam pelaksanaan proyek. Gambar 2.1 berikut ini akan menunjukkan bagaimana penerapan dengan menggunakan metode *prototyping*.

[](#_DAFTAR_GAMBAR)

Gambar 2.1 Metode *Prototyping*

Sumber: A. Dennis, B. H. Wixom and D. Tegarden, System Analysis & Design: An Object-Oriented Approach with UML, 5th Edition, USA: Wiley, 2015

## 2.3 *Unified Modelling Language* (UML)

UML adalah sebuah bahasa pemodelan secara visual yang digunakan dalam pengembangan sistem berorientasi objek (*object-oriented*). UML menyediakan berbagai aturan umum mengenai teknik perancangan sistem yang akan digambarkan melalui diagram. Dengan adanya teknik penggambaran diagram, dapat mempermudah proses pemodelan pengembangan sistem dari tahap analisis hingga tahap implementasi [2].

UML mendefinisikan lima belas teknik penggambaran diagram yang digunakan untuk memodelkan suatu sistem, yang kemudian dibagi menjadi dua kategori, yaitu:

1. *Structure diagram;*

Diagram ini digunakan untuk menggambarkan struktur dari suatu sistem, yang terdiri dari *class*, *object*, *package*, *deployment*, *component*, *composite structure*, dan *profile diagram*.

1. *Behavior diagram;*

Diagram ini digunakan untuk menggambarkan perilaku dari suatu sistem, yang terdiri dari *activity*, *sequence*, *communication*, *interaction overview*, *timing*, *behavior state machine*, *protocol state machine*, dan *use-case diagram*.

Teknik pemodelan yang akan digunakan untuk proses pengembangan sistem dalam penelitian ini adalah *use-case diagram*, *activity diagram* dan *class diagram*.

### 2.3.1 Pemodelan Fungsional

Pemodelan fungsional ini menggambarkan proses dari suatu bisnis dan interaksi antar sistem informasi dengan lingkungannya. Dalam pengembangan sistem berorientasi objek, terdapat dua jenis model yang digunakan untuk menggambarkan fungsionalitas dari suatu sistem informasi, yaitu *use-case* dan *activity diagram* [2].

1. *Use-case diagram;*

*Use-case diagram* merupakan diagram yang menggambarkan secara sederhana fungsi-fungsi utama yang ada dalam sistem dan bagaimana *user* atau pengguna lainnya yang terkait dapat berinteraksi dengan sistem tersebut. Penggunaan *use-case diagram* ini memudahkan pengidentifikasian fitur-fitur yang diinginkan oleh *user* sehingga tercipta *requirements* yang dapat menunjang perancangan sistem. Elemen-elemen dalam *use-case diagram* dapat dilihat pada tabel 2.1 berikut ini.

Tabel 2.1 Elemen-elemen *Use-case diagram*

|  |  |  |
| --- | --- | --- |
| Nama Elemen | Fungsi | Notasi |
| *Actor* | Menggambarkan tokoh atau sistem yang memperoleh manfaat dan berada di luar subyek. *Actor* dapat diasosiasikan dengan *actor* lain menggunakan *specialization* atau *superclass association*. |  |
| *Use case* | Merepresentasikan fungsi utama dari sistem. |  |
| *Subject boundary* | Merepresentasikan ruang lingkup dari subyek. |  |
| *Association relationship* | Menghubungkan *actor* dengan satu atau banyak *use case*. |  |
| *Include relationship* | Merepresentasikan penyertaan fungsi dari satu *use case* dengan *use case* lain. |  |
| *Extend relationship* | Merepresentasikan perluasan dari satu use case dengan perilaku yang bersifat *optional*. |  |
| *Generalization relationship* | Merepresentasikan *specialized use case* dengan *use case* yang lebih umum. |  |

Sumber: A. Dennis, B. H. Wixom and D. Tegarden, System Analysis & Design: An Object-Oriented Approach with UML, 5th Edition, USA: Wiley, 2015

1. *Activity diagram;*

*Activity diagram* merupakan diagram yang memodelkan perilaku dalam suatu proses bisnis yang terlepas dari obyek. Diagram ini akan menggambarkan kegiatan utama dan hubungan antar kegiatan-kegiatan tersebut dalam suatu proses. *Activity diagram* digunakan untuk mengetahui kegiatan apa saja yang harus dilakukan oleh *actor* atau *user* dalam suatu sistem. Elemen-elemen dalam *activity diagram* dapat dilihat pada tabel 2.2 berikut ini.

Tabel 2.2 Elemen-elemen *Activity diagram*

|  |  |  |
| --- | --- | --- |
| Nama Elemen | Fungsi | Notasi |
| *Action* | Merepresentasikan suatu perilaku yang sederhana. |  |
| *Activity* | Merepresentasikan suatu kegiatan. |  |
| *Object node* | Merepresentasikan objek yang terhubungan dengan serangkaian *object flow*. |  |
| *Control flow* | Menggambarkan urutan dari eksekusi kegiatan. |  |
| *Object flow* | Menggambarkan alur suatu objek dari satu kegiatan ke kegiatan lain. |  |
| *Initial node* | Menggambarkan awal dimulainya sebuah kegiatan. |  |
| *Final-activity node* | Menghentikan semua *control flow* atau *object flow* dalam suatu kegiatan. |  |
| *Final-flow node* | Menghentikan *specific control flow* atau *object flow*. |  |
| *Decision node* | Merepresentasikan suatu kondisi untuk memastikan *control flow* atau *object flow* menuju pada satu jalur. |  |
| *Merge node* | Mengumpulkan kembali *decision path* yang berbeda dengan menggunakan *decision node*. |  |
| *Fork node* | Membagi perilaku menjadi serangkaian alur kegiatan yang berjalan secara paralel atau bersamaan. |  |

Tabel 2.3 Elemen-elemen *Activity diagram* (Lanjutan)

|  |  |  |
| --- | --- | --- |
| *Join node* | Mengumpulkan kembali serangkaian alur kegiatan yang berjalan secara paralel atau bersamaan. |  |
| *Swimlane* | Memecah sebuah activity diagram ke dalam baris dan kolom untuk menetapkan suatu kegiatan individual kepada individu atau objek yang bertanggung jawab. |  |

Sumber: A. Dennis, B. H. Wixom and D. Tegarden, System Analysis & Design: An Object-Oriented Approach with UML, 5th Edition, USA: Wiley, 2015

### 2.3.2 Pemodelan Struktural

Pemodelan struktural ini menggambarkan objek yang akan digunakan dan diciptakan oleh suatu sistem bisnis dengan mengilustrasikan orang, tempat, atau informasi-informasi yang disimpan dan bagaimana hal-hal tersebut terkait satu sama lain. Tujuan pemodelan struktural ini adalah untuk menciptakan istilah-istilah yang dapat digunakan oleh *system analyst* dan *user* agar dapat berkomunikasi secara lebih efektif [2]. Salah satu pemodelan struktur yang akan digunakan dalam penelitian ini adalah *class diagram*.

1. *Class diagram;*

*Class diagram* merupakan sebuah model statis yang memperlihatkan *class* dan hubungan antar *class* yang ditetapkan dalam sistem. Diagram ini akan menggambarkan *class*, beserta perilaku, status dan hubungan yang terjalin antar *class*. Elemen-elemen pada *class diagram* dapat dilihat pada tabel 2.3 berikut ini.

Tabel 2.4 Elemen-elemen *Class* *diagram*

|  |  |  |
| --- | --- | --- |
| Nama Elemen | Fungsi | Notasi |
| *Class* | Merepresentasikan suatu jenis orang, tempat, atau benda yang informasinya akan disimpan oleh sistem. |  |
| *Attribute* | Merepresentasikan *properties* yang mendeskripsikan keadaan suatu objek. |  |

Tabel 2.5 Elemen-elemen *Class* *diagram* (Lanjutan)

|  |  |  |
| --- | --- | --- |
| *Operation* | Merepresentasikan tindakan atau fungsi yang dapat dilakukan oleh *class*. |  |
| *Association* | Merepresentasikan hubungan antara beberapa *class* atau sebuah *class* dengan dirinya sendiri. |  |
| *Generalization* | Merepresentasikan hubungan yang sejenis antara beberapa *class*. |  |
| *Aggregation* | Merepresentasikan bagian hubungan yang logis antara beberapa *class* atau sebuah *class* dengan dirinya sendiri. |  |
| *Composition* | Merepresentasikan bagian hubungan yang *physical* antara beberapa *class* atau sebuah *class* dengan dirinya sendiri. |  |

Sumber: A. Dennis, B. H. Wixom and D. Tegarden, System Analysis & Design: An Object-Oriented Approach with UML, 5th Edition, USA: Wiley, 2015

## 2.4 *Eight Golden Rules of Interface Design*

*Eight golden rules* adalah sebuah prinsip desain yang dapat digunakan sebagai panduan dalam proses perancangan *user interface* dalam suatu *website*. Berikut adalah delapan aturan tersebut [3]:

1. *Strive for consistency*

Dalam perancangan *user interface* suatu *website*, konsistensi sangat dibutuhkan dimana tindakan yang serupa dapat diakses dengan cara yang sama, hal-hal yang berkaitan dikelompokkan bersama, dan *messaging* menggunakan struktur yang seragam.

1. *Enable frequent user-to-use shortcuts*

Seiring dengan meningkatnya frekuensi penggunaan, menyebabkan munculnya keinginan *user* untuk mengurangi tindakan yang perlu dilakukan dalam sistem namun laju interaksi pada sistem tetap meningkat. Pemberian respon dalam waktu yang singkat dapat meningkatkan ketertarikan *user* untuk mengakses *website*.

1. *Offer informative feedback*

Memberikan *feedback* yang informatif setiap *user* melakukan interaksi, dengan tujuan agar *user* dapat mengetahui bahwa perintahnya telah direspon oleh sistem.

1. *Design dialogue to yield closure*

Bagian dari pemberian feedback yang informatif dimana ketika *user* telah menyelesaikan suatu tindakan maka sistem akan memberitahu *user* bahwa tindakan tersebut telah terselesaikan. Hal tersebut dapat mengurangi kekhawatiran *user* dalam menggunakan *website*.

1. *Prevent errors*

Melakukan desain antarmuka yang baik untuk mencegah terjadinya kesalahan yang dilakukan oleh *user* dalam menjalankan sistem. Ketika *user* membuat kesalahan, sistem dapat mendeteksi kesalahan tersebut dan menyediakan instruksi yang sederhana, berguna dan spesifik untuk mengatasinya.

1. *Permit easy reversal of actions*

Mempermudah *user* untuk kembali ke halaman sebelumnya dan membatalkan tindakan yang dapat menyebabkan terjadinya kesalahan.

1. *Support internal locus of control*

*User* dapat mengatur tampilan antarmuka dalam sistem sesuai dengan preferensinya sendiri dan sistem dapat merespon setiap *input* dari *user*. Dengan hal ini, dapat tercipta *user experience* yang baik terhadap *website* tersebut.

1. *Reduce short-term memory load*

Tampilan didesain secara sederhana agar dapat mengurangi beban ingatan jangka pendek yang dapat dialami *user* dalam mengingat data apa saja yang harus diinput dalam sistem.

## 2.5 *Relational Database Management System* (RDBMS)

*Database* merupakan sebuah kumpulan data yang terkait secara logis dan deskripsi dari data tersebut, yang dirancang untuk memenuhi kebutuhan informasi dari suatu organisasi. Di dalam sebuah *database* terdapat *entity*, *attribute*, dan *relationship*. *Entity* merupakan sebuah objek atau benda yang dapat dibedakan dari objek lainnya. *Attribute* merupakan properti yang mendeskripsikan beberapa aspek dari *entity*. *Relationship* adalah hubungan antar *entity* [4].

*Database Management System* (DBMS) adalah sebuah sistem *software* yang memungkinkan *user* untuk mendefinisikan, menciptakan, mempertahankan, dan memiliki kontrol akses terhadap *database*. Fasilitas yang disediakan oleh DBMS adalah sebagai berikut [4]:

1. *Data Definition Languag*e membebaskan *user* untuk menentukan tipe data, struktur, dan kendala pada data yang akan disimpan di *database*.
2. *Data Manipulation Language* membebaskan *user* untuk melakukan *insert*, *update*, *delete*, dan memperoleh data pada *database*. DML menyediakan fasilitas berupa bahasa *query* yang dapat digunakan untuk melakukan permintaan terhadap *database* dan sistem informasi. Bahasa *query* yang paling umum adalah *Structured Query Language* (SQL).
3. Penyediaan akses kontrol pada *database*.

*Relational Database Management System* (RDBMS) merupakan sebuah sistem yang didasarkan oleh model relasional untuk mengatur sekumpulan data yang kemudian dipresentasikan melalui tabel. Data relasional memiliki struktur-struktur sebagai berikut [4]:

1. *Relation* adalah sebuah tabel yang berisi kolom dan baris.
2. *Attribute* adalah kolom dari sebuah relasi.
3. *Domain* adalah kumpulan nilai yang dapat digunakan untuk satu atribut atau lebih.
4. *Tuple* adalah suatu baris dari sebuah relasi.
5. *Degree* adalah jumlah atribut yang ada dalam sebuah relasi.
6. *Cardinality* adalah jumlah *tuple* yang ada dalam sebuah relasi.
7. *Relational database* adalah kumpulan relasi yang telah ternormalisasi dengan nama relasi yang berbeda-beda.

Dalam pemodelan relasional, tidak diperbolehkan adanya duplikasi *tuple* dalam sebuah relasi pada *database*. Setiap *tuple* dalam sebuah relasi harus diidentifikasi menggunakan *relational keys*. *Relational keys* terdiri dari atribut-atribut sebagai berikut [4]:

1. *Superkey* adalah sebuah atau sekumpulan atribut yang secara unik mengidentifikasi *tuple* dalam sebuah relasi.
2. *Candidate key* adalah sebuah *superkey* yang memiliki nilai unik dan dapat dipilih sebagai *primary key* dalam suatu tabel. Dalam sebuah relasi, akan ada kemungkinan munculnya beberapa *candidate key*. Ketika sebuah *key* terdiri dari dua atau lebih atribut, maka *key* tersebut dinamakan *composite key*.
3. *Primary key* adalah sebuah *candidate key* yang dipilih untuk mengidentifikasi *tuple* dalam sebuah relasi secara unik. *Candidate key* yang tidak terpilih menjadi *primary key* dinamakan *alternate key*.
4. *Foreign key* adalah sebuah atau sekumpulan atribut dalam sebuah relasi yang memiliki kesamaan dengan *candidate key* dari beberapa relasi lainnya. *Foreign key* ini merupakan kunci penghubung.

## 2.6 PHP: *Hypertext Preprocessor*

PHP merupakan bahasa pemrograman berbentuk *script* yang dapat menciptakan kedinamisan dalam sebuah *website* [5]. PHP diciptakan pertama kali oleh Rasmus Lerdorf pada tahun 1994. Pada awalnya, PHP merupakan singkatan dari *Personal Home Page*. Namun seiring dengan perkembangannya yang cepat dan baik, singkatan PHP berubah menjadi *Hypertext Preprocessor*.

PHP lebih berfokus pada *server-side scripting* yang merupakan sebuah teknik pemrograman *web* dimana bahasa dan perintah dalam program dapat dijalankan melalui *web server* [6]. PHP dapat digunakan di berbagai *major operating systems*, termasuk Linux, Unix, Microsoft Windows, MacOS dan sistem operasi lainnya. Berbagai *web server* seperti Apache, IIS (*Internet Information Server*), dan PWS (*Personal Web Server*) juga telah mendukung penggunaan PHP. Dengan menggunakan PHP, *user* mendapatkan kebebasan untuk memilih sebuah sistem operasi dan *web server*. Selain itu, *user* juga dapat menentukan jenis pemrograman apa yang ingin digunakan yaitu antara pemrograman prosedural atau pemrograman berorientasi objek, ataupun campuran keduanya. Fitur signifikan lain yang dimiliki oleh PHP adalah dukungan yang luas pada berbagai basis data seperti MySQL, PostgreSQL, Oracle, dan basis data lainnya [7].

## 2.7 JavaScript

JavaScript merupakan bahasa pemrograman berorientasi objek yang memungkinkan proses penerjemahan dan kompilasi kode pada elemen-elemen yang ada dalam sebuah dokumen HTML (*Hypertext Markup Language*) [5]. JavaScript digunakan untuk memberikan fitur yang interaktif dalam sebuah *website* seperti animasi, tombol yang dapat diklik, menu *pop-up*, dan banyak lagi. Dengan menggunakan JavaScript, dapat menghasilkan sebuah *website* yang interaktif dan dinamis sehingga mempermudah *user* untuk menggunakan *website* tersebut.

Penggunaan bahasa pemrograman JavaScript memiliki berbagai fitur umum yang memungkinkan terjadinya hal-hal berikut ini [8]:

1. Dapat menyimpan nilai-nilai (*values*) di dalam variabel. Ketika *user* telah memberikan *input* atas perintah yang diberikan oleh sistem, maka sistem akan menyimpan *input* tersebut ke dalam variabel yang ada.
2. Dapat beroperasi pada potongan teks yang lebih dikenal sebagai *strings* dalam *programming*.
3. Dapat merespon berbagai kejadian yang terjadi pada sistem dengan menjalankan kode. Adanya penyediaan *button* yang dapat mempermudah sistem untuk mendeteksi adanya interaksi yang dilakukan oleh *user* sehingga sistem dapat langsung melanjutkan ke tahap berikutnya.

## 2.8 *Testing*

*Testing* merupakan teknik yang digunakan untuk memverifikasi dan memvalidasi kualitas dari suatu sistem. *Testing* dilakukan bukan untuk menunjukkan bahwa sistem terbebas dari *error*, melainkan *testing* dilakukan dengan tujuan untuk menemukan kesalahan yang terdapat pada sistem [2].

*Black box testing* merupakan salah satu teknik pengujian yang bertujuan untuk menguji suatu sistem dari segi fungsionalitasnya. Pada *black box testing*, penguji tidak memiliki akses pada bagian *internal* sistem. Penguji hanya perlu mengetahui bagaimana *output* dan *input* dapat dikeluarkan oleh sistem tanpa memahami bagaimana proses yang terjadi di dalamnya. *Black box testing* dilakukan berdasarkan sudut pandang *user* dimana penentuan *requirements* dilakukan oleh *user*. Keuntungan penggunaan teknik pengujian ini adalah membantu untuk mengidentifikasi ambiguitas atau inkonsistensi *requirements* yang terdapat pada sistem [9].

## 2.9 Sistem Informasi

Sistem informasi merupakan kumpulan dari komponen yang saling berkaitan dan memiliki tujuan yang sama yaitu untuk memperoleh, mengolah, menyimpan, dan menyebarkan informasi sehingga dapat membantu pengambilan keputusan dalam organisasi. Komponen-komponen tersebut adalah berbagai jenis teknologi informasi seperti komputer, *software*, *database*, *communication system*, dan internet [10]. Sistem informasi tidak berfokus pada bagian teknis dalam teknologi informasi, melainkan sistem informasi berfokus pada penggunaan teknologi informasi dalam menyediakan informasi yang dapat digunakan untuk memenuhi kebutuhan *user*. Dengan adanya sistem informasi yang dapat memberikan informasi yang akurat dan memiliki *value*, tentu akan mempermudah *user* maupun organisasi dalam melaksanakan tugas tertentu sehingga tujuan yang diinginkan dapat lebih mudah tercapai.

## 2.10 *Event Organizer* (EO)

*Event* merupakan sebuah acara yang diselenggarakan dengan tujuan untuk memperingati hal-hal penting atau bermakna yang terjadi sepanjang hidup manusia baik secara individu maupun kelompok yang dapat berikatan dengan adat, budaya, tradisi, dan agama [11].

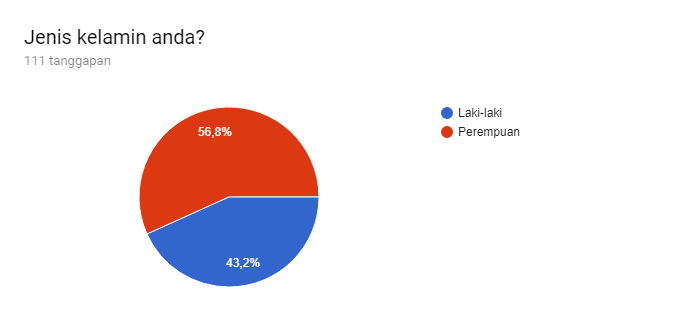
Seiring perkembangannya zaman, muncul banyak sekali ide dalam menyelenggarakan sebuah *event*. *Event* tidak hanya digunakan untuk memperingati peristiwa-peristiwa yang bersifat personal, melainkan dapat digunakan sebagai media untuk mempromosikan kegiatan-kegiatan bisnis dalam sebuah organisasi. Namun dalam rangka mencapai tujuan tersebut, diperlukan sebuah perencanaan yang baik agar *event* dapat berjalan dengan lancar.

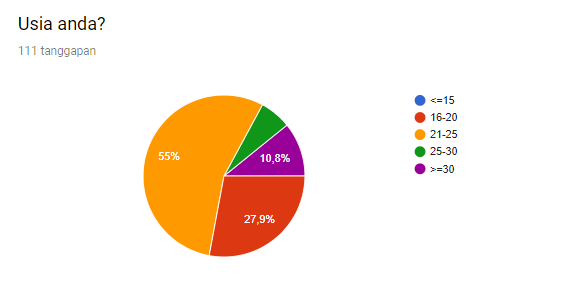
*Event organizer* (EO) adalah sebuah profesi yang memerlukan adanya pertemuan antar sekelompok orang dengan tujuan perayaan, pendidikan, pemasaran dan reuni. Sekelompok orang tersebut akan melakukan serangkaian kegiatan untuk mewujudkan sebuah acara [12]. Tugas seorang EO adalah untuk membantu mewujudkan permintaan klien dengan menghasilkan sebuah acara yang sesuai dengan yang diharapkan klien. Penggunaan EO dapat memberikan banyak kelebihan bagi klien yaitu menghemat waktu, membantu pemilihan tema dan dekorasi yang tepat, dan membantu anggaran menjadi lebih terkendali. EO terdiri dari berbagai macam jenis; ada yang bergerak di bidang olahraga, seni, pameran, maupun acara yang bersifat pribadi seperti pesta pernikahan, ulang tahun, dan acara lainnya [1].

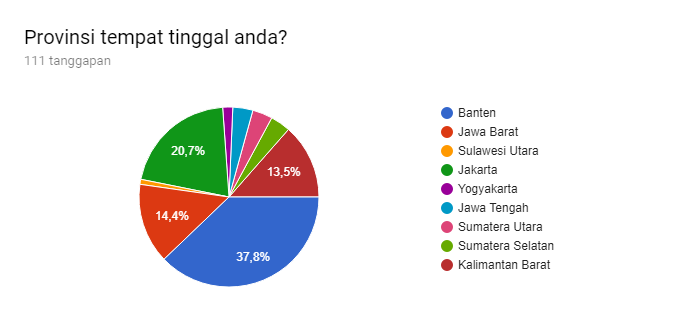
# DAFTAR PUSTAKA

|  |  |
| --- | --- |
| [1] | I. K. Suseno, Cara Pinter Jadi Event Organizer, Yogyakarta: Galang Press, 2005. |
| [2] | A. Dennis, B. H. Wixom and D. Tegarden, System Analysis & Design: An Object-Oriented Approach with UML, 5th Edition, USA: Wiley, 2015. |
| [3] | B. Shneiderman and C. Plaisant, Designing the User Interface: Strategies for Effective Human-Computer Interaction, 4th Edition, Boston: Pearson, 2004. |
| [4] | T. Connolly and C. Begg, Database Systems: A Practical Approach to Design, Implementation, and Management, 6th Edition, England: Pearson, 2015. |
| [5] | R. Nixon, Learning PHP, MySQL, and JavaScript, 4th Edition, USA: O'Reilly, 2015. |
| [6] | "What can PHP do?," PHP, 2001. [Online]. Available: http://php.net/manual/en/intro-whatcando.php. [Accessed 20 Juli 2018]. |
| [7] | K. Tatroe, P. MacIntyre and R. Lerdorf, Programming PHP, 3rd Edition, USA: O'Reilly, 2013. |
| [8] | "What is JavaScript?," MDN Web Docs, [Online]. Available: https://developer.mozilla.org/en-US/docs/Learn/JavaScript/First\_steps/What\_is\_JavaScript. [Accessed 20 Juli 2018]. |
| [9] | S. Nidhra and J. Dondeti, "Black Box and White Box Testing: A Literature Review," *International Journal of Embedded Systems and Applications,* vol. II, no. 2, pp. 1-22, 2012. |
| [10] | S. K. Boel and D. C. Kecmanovic, "What is an Information System?," *Hawaii International Conference on System Sciences,* no. 48, pp. 1-11, 2015. |
| [11] | A. Noor, Manajemen Event, Bandung: Alfabeta, 2013. |
| [12] | J. Goldblatt, Special Events: Creating and Sustaining a New World for Celebration, New York: Wiley, 2013. |

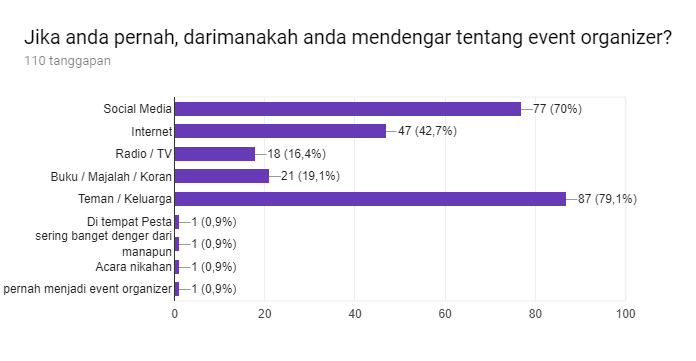
LAMPIRAN 1  
HASIL SURVEI

****

****

****

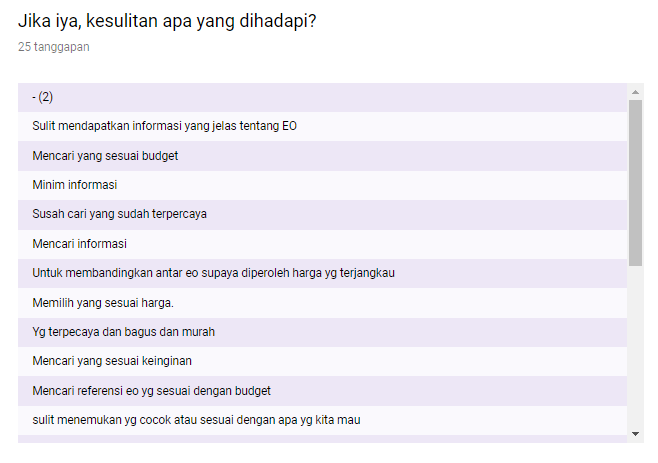
****

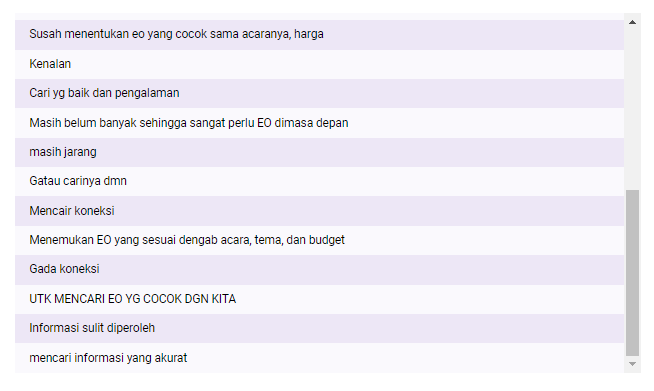
****

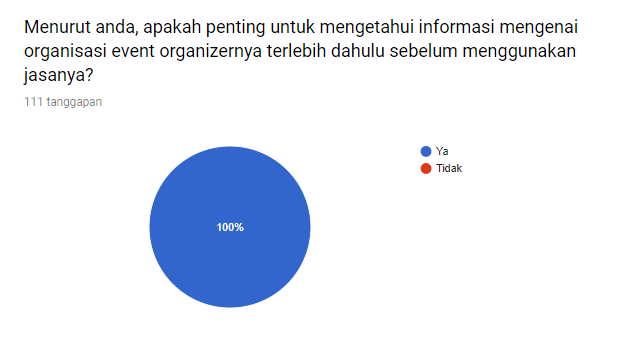
****

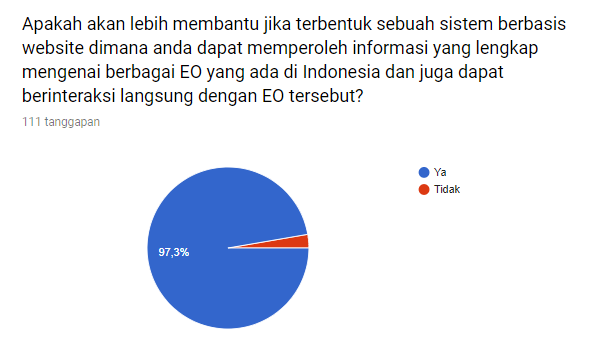
****

****









****

