REPOSITORY MOBILE LABORATORY

TESIS

Nama	:	Yudha Yogasara
NIM/ NIRM	:	92310029
Pembimbing	:	Prof. Dr. I Wayan Simri Wicaksana. S.Si. M.Eng

GUNADARMA UNIVERSITY
PROGRAM PASCASARJANA

TAHUN 2012

Halaman Pengesahan

Judul Penelitian	:	REPOSITORY MOBILE LABORATORY
Nama Mahasiswa	:	Yudha Yogasara
NIRM	:	92310029
Tanggal Lulus	:	29 Maret 2012

Menyetujui,

Komisi Pembimbing

(Prof. Dr. I Wayan Simri Wicaksana. S.Si. M.Eng)

(Prof. Dr. Yuhara Sukra MSc)

Anggota

(Prof. Dr. Dharma Tintri, SE.Ak, MBA)

Direktur

Abstraksi

Yudha Yogasara, 92310029
REPOSITORY MOBILE LABORATORY.

Kata Kunci : Kata kunci disini

(xiii+ 117+ lampiran)

Setiap perusahaan membutuhkan informasi yang cepat dan akurat. Salah satu informasi yang dibutuhkan ialah informasi aset perusahaan. Pengelolaan aset dalam suatu perusahaan yang terpusat (berbasis web), dapat menjangkau seluruh elemen yang tersebar di berbagai tempat, karena mengurangi waktu proses permintaan, pemeliharaan sampai dengan penerimaan aset. Untuk mengatasi hal ini peneliti merancang suatu sistem informasi manajemen dan penyusutan aset. Tujuan dari penelitian ini untuk membantu kinerja karyawan yang mencakup permintaan kebutuhan, persetujuan permintaan, penerimaan, pemeliharaan, serta penyusutan aset.

Secara umum pendekatan yang dipakai penulis ialah pengumpulan data, analisis serta merancang sistem berdasarkan hasil penelitian yang dilakukan.

Hasil perancangan ini dapat diperolehnya sistem informasi manajemen dan penyusutan aset yang dibutuhkan, sehingga dapat mempercepat proses permintaan, persetujuan permintaan, penerimaan, pemeliharaan, dan penyusutan aset, serta menunjang manajemen dalam mengambil keputusan strategis.

Daftar Pustaka (2006:2011)

Daftar Riwayat Hidup

Penulis dilahirkan di Bogor, pada tanggal 16 Agustus 1987, anak ketiga dari 3 bersaudara, menyelesaikan pendidikan Sekolah Dasar di SDN 03 Karawaci, Tangerang dan tamat pada tahun 1999, Sekolah Pendidikan Menengah Pertama di MTs Mathla'ul Huda Bogor dan tamat pada tahun 2002, serta Pendidikan Menengah Atas di SMA Islamic Village Tangerang dan tamat pada tahun 2005.

Tahun 2005 melanjutkan pendidikan diploma tiga di Universitas Gunadarma dengan mengambil jurusan Manajemen Informatika, dan lulus pada tahun 2008 dengan membuat judul penulisan "Teknik Pembuatan Distro Linux Atunez- ME Berbasis PCLinuxOS". Kemudian penulis melanjutkan pendidikan strata satu di Universitas Gunadarma dengan mengambil jurusan Sistem Informasi, dan lulus pada 2010 dengan membuat judul penulisan "Repository Mobile Laboratory"

Jakarta, Mei 2012

Yudha Yogasara, S. Kom

Kata Pengantar

Segala puji dan syukur penulis naikkan ke hadirat Allah S.W.T yang Maha Kuasa yang telah memberikan berkat, anugerah dan karunia yang melimpah, sehingga penulis dapat menyelesaikan Tesis yang berjudul: “REPOSITORY MOBILE LABORATORY”.

Pada kesempatan ini penulis menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Ibu Prof. Dr. E.S. Margianti, SE, MM, selaku Rektor Universitas Gunadarma yang telah memberikan kesempatan kepada penulis untuk melanjutkan pendidikan dan dalam penyusunan tesis ini.

2. Bapak Prof. Dr. Yuhara Sukra MSc, selaku Koordinator Program Pasca Sarjana Universitas Gunadarma.

3. Bapak Prof. Dr. Dharma Tintri, SE.Ak, MBA, selaku Direktur Program Pasca Sarjana Universitas Gunadarma.

4. Bapak Dr. Yuhilza Hanum SSi, SKom, MEng, selaku Ketua Prodi Program Pasca Sarjana Universitas Gunadarma.

5. Bapak Prof. Dr. I Wayan Simri Wicaksana. S.Si. M.Eng, selaku dosen pembimbing serta dosen pengajar yang telah banyak meluangkan waktu, tenaga serta pikiran dalam membimbing dan mengarahkan penulis untuk menyelesaikan penulisan tesis ini.

6. Bapak dan Ibu Dosen Pengajar Universitas Gunadarma yang telah banyak memberikan ilmu pengetahuan kepada penulis.

7. Kedua orang tua dan kakak-kakak tercinta yang selalu mendukung dan terus memberikan motivasi.

8. Seluruh rekan seperjuangan di Universitas Gunadarma yang telah banyak membantu penulis.

9. Semua pihak yang tidak tersebutkan yang telah membantu penyelesaian Tugas Akhir ini, penulis ucapkan juga terima kasih atas segala bantuan dan sarannya.

Sebagai manusia biasa yang tak luput dari kesalahan, maka penulis meminta maaf atas segala kekurangan dan keterbatasan dalam penyusunan Tugas Akhir ini.

Akhir kata, hanya kepada Tuhan jualah segalanya dikembalikan dan penulis sadari bahwa penulisan ini masih jauh dari sempurna, disebabkan karena berbagai keterbatasan yang penulis miliki. Untuk itu penulis mengharapkan kritik dan saran yang bersifat membangun untuk menjadi perbaikan di masa yang akan datang.

Semoga apa yang ada pada penulisan Tesis ini dapat bermanfaat bagi kita semua. Amin.

Jakarta, Mei 2012

Penulis

Contents

List of Figures

List of Tables

Chapter 1

PENDAHULUAN

1.1 Latar Belakang

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc vehicula, risus sit amet fringilla ullamcorper, sem velit fermentum enim, nec viverra quam mauris a mauris. Praesent a tortor ligula, sit amet facilisis arcu. Donec eu lectus ante, eget tincidunt ante. Nam suscipit arcu id felis fermentum sed sollicitudin magna egestas. Curabitur non metus leo. Duis feugiat venenatis orci, in hendrerit velit ornare sed. Nulla facilisi. Donec odio tellus, rutrum vel condimentum et, porta a lectus. Duis volutpat consectetur feugiat.

Pellentesque non est at nibh condimentum feugiat. Donec facilisis molestie odio eget vehicula. Vestibulum sed mauris nunc. Sed pharetra mollis felis, eget consequat libero gravida id. Aenean vel augue mauris. Phasellus consectetur ornare dolor varius congue. Ut blandit, lacus nec rhoncus ultrices, libero tortor porttitor mauris, quis placerat orci ligula at quam. Nullam id tellus ut diam accumsan rhoncus sed in turpis. Duis vulputate euismod convallis. Nam odio ligula, elementum nec euismod scelerisque, porta eget lectus. Mauris purus libero, facilisis id tempor ut, condimentum at ligula. Etiam varius sodales mauris placerat consectetur. Phasellus facilisis pulvinar tempus. Nam id massa metus. Duis tempus sodales augue, non ultrices est egestas in. Aliquam tortor sapien, consectetur id luctus in, commodo a sem.

1.2 Identifikasi Masalah

Aenean mi nibh, sagittis tempus fermentum nec, tristique vitae orci. Suspendisse vitae dui quis mi fringilla luctus et a dolor. Etiam pellentesque nisl quis dui ultrices id interdum dolor ultricies. Suspendisse euismod sodales ipsum et luctus. Aenean in nunc nec diam adipiscing commodo. Nam ornare volutpat ornare. Nunc ultricies, velit nec imperdiet tempus, justo massa ultricies quam, in feugiat sem purus vel urna. In elementum viverra feugiat. Cras urna risus, pulvinar rutrum bibendum non, rutrum eget massa. Quisque tincidunt dictum metus convallis semper. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Vestibulum mollis consectetur neque quis vulputate. Quisque vel luctus nulla. Sed at lacus est, vitae vestibulum magna. Sed interdum hendrerit diam et vestibulum.

Nunc et erat at nisi cursus venenatis. Pellentesque dictum luctus nisl consectetur consequat. Curabitur id pulvinar diam. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent hendrerit ultrices ante sit amet elementum. Nulla eu ante id sem rutrum fermentum a vitae augue. In sed velit vitae eros imperdiet vehicula. Curabitur nisi nunc, adipiscing a ultricies eget, cursus at ipsum. Proin adipiscing blandit nisi, vitae sollicitudin ante ultricies eu. Maecenas in luctus ante. Ut eu sem nisi. Vivamus tempor erat eros. Nunc et tortor in enim vulputate eleifend a ut eros. Aenean tempus elementum massa, ac scelerisque nulla adipiscing vel. Nulla facilisi.

1.3 Rumusan Masalah

1.4 Tujuan Penelitian

Proin eu leo quis risus congue aliquam quis nec mauris. Donec cursus pretium diam, eget sagittis massa aliquet ut. Suspendisse potenti. Vestibulum adipiscing lorem et urna imperdiet interdum ornare eros aliquet. Aliquam quis ligula in libero condimentum pulvinar. Integer a ante metus. Sed eu quam eu lorem cursus posuere. Duis sed ligula leo, a scelerisque elit. Nam pretium, leo in rhoncus scelerisque, felis lacus imperdiet mi, vitae semper lectus nisl volutpat mi. Vivamus id augue mi, quis lacinia enim. Proin tincidunt mattis lacus eu sollicitudin. Maecenas varius diam quis ipsum pharetra a venenatis dui elementum.

1.5 Manfaat Penelitian

Chapter 2

LITERATURE REVIEW

2.1 System Development

Software Development Life Cycle (SDLC) describes the methodology by which the development of any software takes place. Before SDLC the process of developing the software was taken as informal activity with no formal rules and standards. This may lead to the various problems such as delay in development, cost overrun, and low software quality. The introduction of the SDLC gives the precise standard and the steps for the development of the software. SDLC overcome all the problems which are there before the introduction of the SDLC [Saini and Kaur, 2014].

2.1.1 Software Process

A process is a collection of activities, actions, and tasks that are performed when some work product is to be created. An activity strives to achieve a broad objective (e.g., communication with stakeholders) and is applied regardless of the application domain, size of the project, complexity of the effort, or degree of rigor with which software engineering is to be applied. An action (e.g., architectural design) encompasses a set of tasks that produce a major work product (e.g., an architectural design model). A task focuses on a small, but well-defined objective (e.g., conducting a unit test) that produces a tangible outcome [Pressman, 2010].

A process framework establishes the foundation for a complete software engineering process by identifying a small number of framework activities that are applicable to all software projects, regardless of their size or complexity. In addition, the process framework encompasses a set of umbrella activities that are applicable across the entire software process. A generic process framework for software engineering encompasses five activities [Pressman, 2010]:

1. Communication

   Before any technical work can commence, it is critically important to communicate and collaborate with the customer (and other stakeholders. The intent is to understand stakeholders' objectives for the project and to gather requirements that help define software features and functions.

2. Planning

   Any complicated journey can be simplified if a map exists. A software project is a complicated journey, and the planning activity creates a “map” that helps guide the team as it makes the journey. The map—called a software project plan—defines the software engineering work by describing the technical tasks to be conducted, the risks that are likely, the resources that will be required, the work products to be produced, and a work schedule.

3. Modeling

   Whether you’re a landscaper, a bridge builder, an aeronautical engineer, a carpenter, or an architect, you work with models every day. You create a “sketch” of the thing so that you’ll understand the big picture—what it will look like architecturally, how the constituent parts fit together, and many other characteristics. If required, you refine the sketch into greater and greater detail in an effort to better understand the problem and how you’re going to solve it. A software engineer does the same thing by creating models to better understand software requirements and the design that will achieve those requirements.

4. Construction

   This activity combines code generation (either manual or automated) and the testing that is required to uncover errors in the code.

5. Deployment

   The software (as a complete entity or as a partially completed increment) is delivered to the customer who evaluates the delivered product and provides feedback based on the evaluation.

2.1.2 Rapid Application Development

RAD is a process which accelerates the cycle of development of an application. The phases are similar to waterfall but the ’chunks’ are smaller. The emphasis in this model is on fast iterations through the cycle. Prototypes are designed, developed and evaluated with users, involving them in the process and correcting the design. The model is particularly suited to projects in rapidly changing environments where the team needs to adapt to different situations [Jenkins, 2008]. It is possible to develop quality products faster, thus valuabe resources can be saved. This methodology consists of the following three phases [Noertjahyana, 2002]:

1. Requirement Planning

2. RAD Design Workshop

3. Implementation

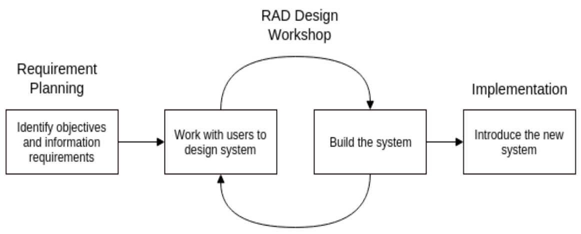


Figure 2.1 – Rapid Application Development Phases [Noertjahyana, 2002]

2.2 Manga

Bearing this in mind, the word “manga” has come to have two meanings outside Japan. Some use it to designate Japanese “comics,” the socio-cultural objects, and often the industry and community surrounding them. However, others use “manga” to name this visual language itself — loosely conceived of as an “aesthetic style”. Since the conflation of these ideas can be confusing and inappropriate, in this piece “manga” will be used in the first sense — to designate a socio-cultural artifact — while referring to the system of graphic expression as “Japanese Visual Language” or JVL. While JVL is the graphic system of communication, “manga” is the socio-cultural context in which it appears most [Cohn, 2007].

2.3 Navigation Structure

Navigation structure is a plot that is used in applications that will be made. Before preparing multimedia applications in software, determining the flow to be used in applications that are made should be done first. The basic form of navigation structures commonly used in the manufacturing process there are four kinds of multimedia applications, namely linear navi- gation structure, non-linear, hierarchy, and composite [Mahendra, 2009].

2.3.1 Linear Navigation Structure

Linear navigation structure is a structure that has a series of stories sequentially. This structure displays layer one by one sequentially, according to the rules [Mahendra, 2009].

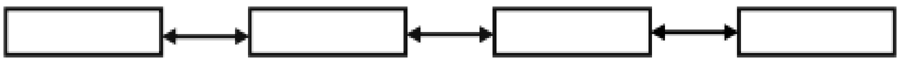


Figure 2.2 – Linear Navigation Structure [Mahendra, 2009]

2.3.2 Nonlinear Navigation Structure

Non-linear navigation structure (not sequential) is a development of lin- ear navigation structure, only in this structure is allowed to make branching. Branching on nonlinear structure is different from branching in a hierarchical structure. In non-linear structure, position of all the pages are the same, so it is not known a master or slave page [Mahendra, 2009].

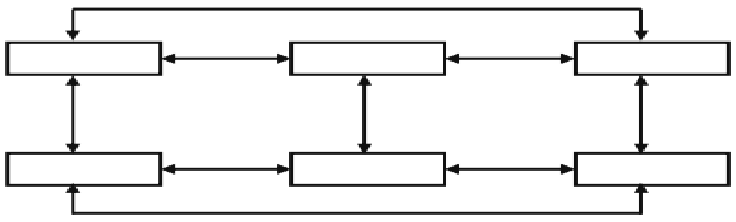


Figure 2.3 – Nonlinear Navigation Structure [Mahendra, 2009]

2.3.3 Hirarchical Navigation Structure

Hierarchy navigation structure is often called branched navigation struc- ture, which is a branching structure that relies on the data or images for display on a layer with a certain criteria. Display on the main menu called the master page (the main page), it has a page called branching slave page (page support) and if selected will be the second page, and so on [Mahendra, 2009].

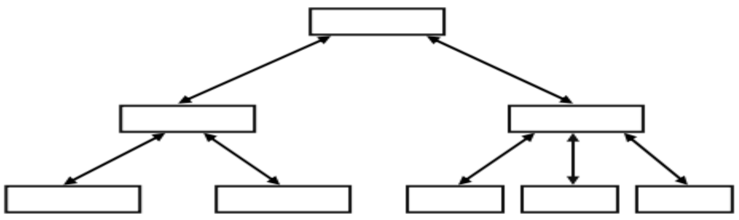


Figure 2.4 – Hierarchical Navigation Structure [Mahendra, 2009]

2.3.4 Composite Navigation Structure

Composite navigation structure is a combination of the previous structure and also called the free navigation structure; the point is if a display requires branching then made branching. This structure is most widely used in the creation of multimedia applications [Mahendra, 2009].

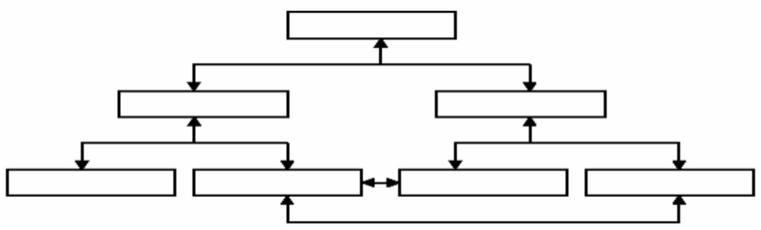


Figure 2.5 – Composite Navigation Structure [Mahendra, 2009]

2.4 Unified Modeling Language

The Unified Modeling Language (UML) is a general-purpose visual modeling language that is used to specify, visualize, construct, and document the artifacts of a software system. It captures decisions and understanding about systems that must be constructed. It is used to understand, design, browse, configure, maintain, and control information about such systems. The modeling language is intended to unify past experience about modeling techniques and to incorporate current software best practices into a standard approach [Rumbaugh et al., 2005].

The UML captures information about the static structure and dynamic behavior of a system. A system is modeled as a collection of discrete objects that interact to perform work that ultimately benefits an outside user. The static structure defines the kinds of objects important to a system and to its implementation, as well as the relationships among the objects. The dynamic behavior defines the history of objects over time and the communications among objects to accomplish goals. Modeling a system from several separate but related viewpoints permits it to be understood for different purposes [Rumbaugh et al., 2005].

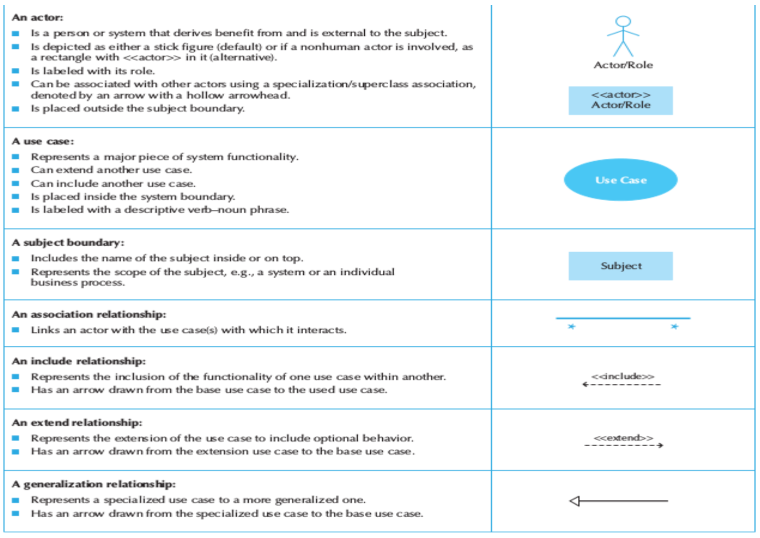
2.4.1 Use-Case Diagram

Use case diagram is one of the UML diagrams that is used to represent the functionality of the system. It shows the functionality that the system will provide and the users who will communicate with the system to use that functionality. The use case diagram includes the nota-tions such as actors, use cases, communication association and the system or subsystem boundary. The use case diagrams provide interactions between roles known as actors and system to achieve a certain goal. Human or external system both are assumed as actors in definition of use case diagrams. Use cases define function nality of the system from a users' perspective and are used to document the system scope. Usually, use cases are used at a higher level in systems engineering as compared to their use in software engineering. Syntax use-case diagram can be seen in Figure 2.6.

2.4.1.1 Actor

An actor is an idealization of a role played by an external person, process, or thing interacting with a system, subsystem, or class. An actor characterizes the interactions that a class of outside users may have with the system. Different users may be bound to the same actor and therefore represent multiple instances of the same actor definition [Rumbaugh et al., 2005].

Each actor participates in one or more use cases. It interacts with the use case by exchanging messages. The internal implementation of an actor is not relevant in the use case; an actor may be characterized sufficiently by a set of attributes that define its state. An actor is drawn as a small stick person with the name below it [Rumbaugh et al., 2005].



An actor: ■ Is a person or system that derives benefit from and is external to the subject. ■ Is depicted as either a stick figure (default) or if a nonhuman actor is involved, as a rectangle with <<actor>> in it (alternative). ■ Is labeled with its role. ■ Can be associated with other actors using a specialization/superclass association, denoted by an arrow with a hollow arrowhead. ■ Is placed outside the subject boundary.	Actor/Role <<actor>> Actor/Role
A use case: ■ Represents a major piece of system functionality. ■ Can extend another use case. ■ Can include another use case. ■ Is placed inside the system boundary. ■ Is labeled with a descriptive verb–noun phrase.	Use Case
A subject boundary: ■ Includes the name of the subject inside or on top. ■ Represents the scope of the subject, e.g., a system or an individual business process.	Subject
An association relationship: ■ Links an actor with the use case(s) with which it interacts.	* *
An include relationship: ■ Represents the inclusion of the functionality of one use case within another. ■ Has an arrow drawn from the base use case to the used use case.	<<include>>
An extend relationship: ■ Represents the extension of the use case to include optional behavior. ■ Has an arrow drawn from the extension use case to the base use case.	<<extend>>
A generalization relationship: ■ Represents a specialized use case to a more generalized one. ■ Has an arrow drawn from the specialized use case to the base use case.	

Figure 2.6 – Use-case diagram syntax [Dennis et al., 2005]

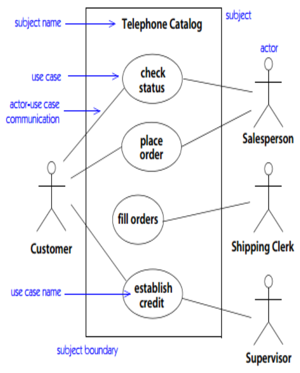


Figure 2.7 – Use-case Diagram

2.4.1.2 Activity Diagram

An activity is operation sequence from start to end the system done and per activity can be transformed on data or process [Azizi, 2011]. Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In UML, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control [Constantianus and Suteja, 2005].

2.5 Python

Python is a high level programming language (high level language) developed by Guido van Rossum in 1989 and introduced the first time in 1991. Python was born of the desire to simplify the programmer to complete tasks quickly. Python is designed to provide ease very remarkable for programmers both in terms of time efficiency, and ease of program development and in terms of compatibility with the system. Python can be used to create stan- dalone program and programming scripts [R.H.Sianipar and Wadi, 2015].

Implementation Python under an open source license that makes Python can be used and distributed freely, even for commercial purposes. Python provides libraries for desktop development, and the development of educa- tional applications. PyBiblio an application for education related to many different sources. In addition, to software development, gaming and dimensional graphics and programming computer network. In addition to the support of the web and internet Python, Python provides support for programming level lower computer network, such as network interface socket.

2.6 Django

Django is a high-level Python web application framework designed to support the development of dynamic websites, web applications, and web services. It is designed to promote rapid development and clean, pragmatic design and build high-performing, elegant web application quickly. The main advantages of Django [Hourieh, 2008]:

1. Tight Integration between Components

   First of all, Django provides a set of tightly integrated components; all of these components have been developed by the Django team themselves. Django was originally developed as an in-house framework for managing a series of news-oriented websites. Later its code was released on the Internet and the Django team continued its development using the Open Source model. Because of its roots, Django's components were designed for integration, reusability and speed from the start.

2. Object-Relational Mapper

   Django's database component, the Object-Relational Mapper (ORM), provides a bridge between the data model and the database engine. It supports a large set of database systems, and switching from one engine to another is a matter of changing a configuration file. This gives the developer great flexibility if a decision is made to change from one database engine to another.

3. Clean URL Design

   he URL system in Django is very flexible and powerful; it lets you define patterns for the URLs in your application, and define Python functions to handle each pattern. This enables developers to create URLs that are both user and search engine friendly.

4. Automatic Administration Interface

   Django comes with an administration interface that is ready to be used. This interface makes the management of your application's data a breeze. It is also highly flexible and customizable.

5. Advanced Development Environment

   In addition, Django provides a very nice development environment. It comes with a lightweight web server for development and testing. When the debugging mode is enabled, Django provides very thorough and detailed error messages with a lot of debugging information. All of this makes isolating and fixing bugs very easy.

6. Multi Lingual Support

   Django supports multi-lingual websites through its built-in internationalization system. This can be very valuable for those working on websites with more than one language. The system makes translating the interface a very simple task.

The concepts of Django are similiar enough to a Model View Controller design, but Django is described as a Model Template View (MTV) framework. The Django models are modified slightly to better represent the structure of web applications. The View layer includes business logic and request handling, Templates determine the appearance of the websitehe, and the Model and object relational mapping layer defines data and relationships that are stored in the database [Ward, 2009].

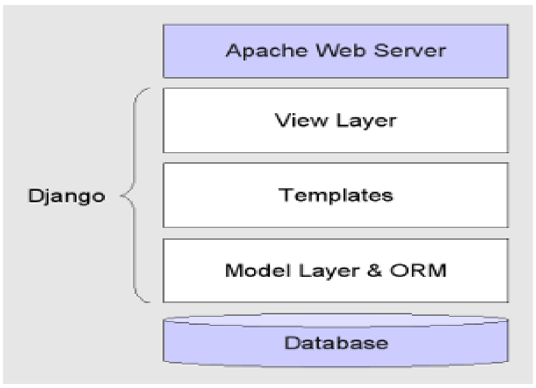


Figure 2.8 – Django Layers [Ward, 2009]

2.7 HTML

HTML is pure text file that can be created with any text editor. HTML is the language for describing the structure of Web pages. HTML gives the means to [Rabbani, 2015]:

- Publish online documents with headings, text, tables, lists, photos, etc.
- Retrieve online information via hypertext links, at the click of button.
- Design forms for conducting transactions with remote services, for use in searching for information, making reservations, ordering products, etc.
- Include spread-sheets, video clips, sound clips and other systems directly in their documents.

Nevertheless, until now HTML still standing strong as the basis of web languages like PHP, ASP, JSP and others. In fact, in general, the majority of web sites on the Internet was still using HTML as the primary technology [Constantianus and Suteja, 2005]. A basic HTML document consists of four sections defined by four sets of elements. For example, here is the code HTML (saved with the extension .htm or .html) [Brooks, 2007].

```
<html>
<head>
<title> My Website </title>
</head>
<body> How to learn html </body>
</html>
```

2.8 Javascript

JavaScript is not related to Java, although it is a curly-brace language like Java, C#, C++, and many other programming languages. JavaScript is related to ECMAScript, however. Ecma International, working with other organizations, created this standardized scripting language in the ECMA-262 specification and ISO/IEC 16262. The language is widely used for client-side scripting on the web, and you commonly see several well-known variations of ECMAScript such as JavaScript, JScript, and ActionScript. The current release is ECMAScript Edition 5.1 and most common browsers support ECMAScript Edition 5.0 or newer [Johnson, 2013].

2.9 AJAX

Rousseaux and Lhoste [Rousseaux and Lhoste, 2009] described that with the faster evolution in IT tools rapid software prototyping is a best way to meet the demands of users instantly. With a classical software development cycle it is difficult to develop an application so fast which normally takes several months for development. By using RAD functionalities and interfaces can be revised with the passage of time. The term “Ajax” first used in a piece of written which is now online. Ajax is a latest way for the applications of web said by Jesse James Garrett in 2005 for the first time. Garrett named this phrase for the architecture and makes it equal like the new production group of web Applications. These new productions can be Google maps.

Ajax cannot be said as a actual language of programming. This is an advanced design model. It is made up of a lot of connected technologies and thoughts. With the use of Ajax in applications of web we can get back data to the server asynchronously. Ajax helps in retrieve of data with no intervention on display. It refreshes the webpage without reloading it again. Some technologies implicated in Ajax are XHTML, CSS, XML, XSLT and JavaScript. Mostly web sites use AJAX. Like Facebook, Netvibes, Flickera and Google maps. This technology brought more flexibility and interactivity in WebPages. It adds functionalities like desktop to the web sites. Its aim is to almost finish the gaps between web and desktop [Rousseaux and Lhoste, 2009].

2.10 MySQL

MySQL is the world's most popular open source database software, with over 100 million copies of its software downloaded or distributed throughout it's history. MySQL is a Relational Database Management System (RDBMS) that runs as a server providing multi-user access to a number of database. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of prop- rietary agreements. MySQL is a key part of LAMP (Linux, Apache, MySQL, PHP/ Perl/ Python), the fast-growing open source enterprise software stack [Rabbani, 2015].

The main features from MySQL are [Rabbani, 2015]:

1. Written in C and C++.
2. Working in a variety of platforms (eg Mac OS X, Solaris, Sun OS, Unix, Novel Netware, Windows, etc).
3. Provide transactions and no transactions storage engine.
4. The server is available as a separate program to use on the network client or server environment.
5. MySQL has a library that can be embedded in systems that run itself, so that the system can be used on computers that do not has network.

2.11 Testing

Software testing is the process of verification and validation whether a software system or program meets the requirements of business and technical requirements that drive the design and development and how it works as expected and also identify critical errors that are classified based on the level severity in systems that must be corrected [Nidhra and Dondeti, 2012a]. Software testing is a technique often used to verify and validate the quality of the software. Software testing is a procedure for executing a program or system in order to find fault.

2.11.1 Black Box Testing

Black box testing is also called as functional testing, a functional testing technique that designs test cases based on the information from the specifica- tion. Black box testing not concern with the internal mechanisms of a system, these are focus on the outputs generated in response to selected inputs and execution conditions. In black box testing, the software tester should not (or does not) have access to the internal source code itself [Nidhra and Dondeti, 2012b]

Advantages of using Black Box Testing are [Khan and Khan, 2012]:

1. Efficient for large code segment.
2. Tester perception is very simple.
3. Users perspective are clearly separated from developers perspective (programmer and tester are independent of each other).
4. Quicker test case development.

Chapter 3

RESEARCH METHODOLOGY

Chapter 4

IMPLEMENTATION AND TESTING

Chapter 5

KESIMPULAN DAN SARAN

5.1 Kesimpulan

5.2 Saran

Halaman Untuk Daftar Pustaka

DAFTAR PUSTAKA

Somayeh Azizi. *Modeling UML2 Activity Diagram by Using Graph Transformation Systems and Abstract State Machine*. 2011.

David R. Brooks. *An Introduction to HTML and JavaScript: for Scientist and Engineers*. Springer-Verlag London Limited, 2007.

Neil Cohn. Japanese visual language: The structure of manga. 2007.

Frederick Constantianus and Bernard Renaldy Suteja. Analisa dan desain sistem bimbingan tugas akhir berbasis web dengan studi kasus fakultas teknologi informasi. *Jurnal Informatika UKM*, 2005.

Alan Dennis, Barbara Haley Wixom, and David Tegarde. *Systems Analysis & Design With UMLVersion 2.0 (An-Object Oriented Approach)*. John Wiley & Sons, (2nd edition) edition, 2005.

Ayman Hourieh. *Learning Website Development with Django*. Packt Publishing Ltd., (1st edition) edition, April 2008.

Nick Jenkins. *A Software Testing Primer: An Introduction to Software Testing*. 2008.

Glenn Johnson. *Programming In HTML5 With JavaScript And CSS3 (Training Guide)*. Microsoft Press, 2013.

Mohd. Ehmer Khan and Farmeena Khan. A comparative study of white box, black box and grey box testing techniques. *International Journal of Advanced Computer Science and Applications (IJACSA)*, 2012.

A. Mahendra. Struktur navigasi. Online, August 2009. Available: http://oke.or.id/wp-content/plugins/downloads-manager/upload/Struktur[Accessed March 29, 2016].

S Nidhra and J Dondeti. Black box and white box testing techniques. *International Journal for Embedded Systems and Applications*, 2012a.

Srinivas Nidhra and Jagruthi Dondeti. Black box and white box testing techniques. *International Journal of Embedded Systems and Applications (IJESA)*, 2012b.

Agustinus Noertjahyana. Studi analisis rapid development sebagai salah satu alternatif metode pengembangan perangkat lunak. *Jurnal Informatika 3*, 2:pp. 74–79., 2002.

Roger S. Pressman. *Software Engineering: A Practitioner'sApproach*. Mc Graw-Hill, New York, (7th edition) edition, 2010.

Muhammad Huda Rabbani. Website classification system using keyword method and term frequency-inverse document frequency (td-idf) weighting method. *Gunadarma University*, 2015.

R.H.Sianipar and Hamzan Wadi. *Pemrogramman Python Teori dan Implementasi*. Informatika, 2015.

F. Rousseaux and K. Lhoste. Rapid software prototyping using ajax and google map api. in advances in computer-human interactions. *Advances in Computer-Human Interactions, ACHI'09, Second International Conferences on IEEE*, pages pp. 317–323., 2009.

James Rumbaugh, Ivar Jacobson, and Grady Booch. *The Unified Modeling Language Reference Manual*. Addison-Wesley, 2005.

Munish Saini and Kuljit Kaur. A review of open source software development life cycle models. *International Journal of Multimedia and Ubiquituous Engineering*, 2014.

Ian Ward. excess.org: Django 1.1 talk text. Online, 2009. Available: https://excess.org/article/2009/05/django-1-1-talk-text. [Accessed March 29, 2016].

Lampiran