**BAB II**

**KAJIAN PUSTAKA**

1. **Kajian Penelitian Terdahulu**

Pada bagian ini akan dipaparkan mengenai kajian terdahulu yang disajikan sebagai bahan acuan dasar dalam penelitian. Berikut akan dijelaskan kajian teori yang mendukung penelitian ini.

Berdasarkan penelitian yang dilakukan oleh Pipin Widyaningsih dan Aprilia Astutiningsih (2016) membahas tentang aplikasi konsultasi masalah kehamilan secara online. Aplikasi ini berbasis web dan dilengkapi dengan sistem pakar untuk membantu para ibu hamil melakukan diagnosa kesehatan berdasarkan gejala yang mereka alami. Metode yang digunakan dalam penelitian adalah metode pengembangan sistem pakar yang terdiri dari penilaian keadaan, akusisi pengetahuan, perancangan, pengujian, dokumentasi, dan pemeliharaan.

Berdasarkan penelitian Rifki Indra Perwira (2012), sistem untuk konsultasi menu diet bagi penderita Diabetes Melitus berbasis aturan adalah salah satu alternatif dari berbagai macam sistem yang sudah pernah dipakai untuk mengatasi permasalahan yang dihadapi. Tujuan rancang bangun sistem berbasis aturan ini adalah agar pemakai dapat melakukan konsultasi terhadap komposis makanan dan diet seimbang untuk membantu proses penyembuhan yang diderita pasien. Perancangan sistem ini menggunakan rule based reasoning yang disimpan dalam basis data dengan menggunakan mesin inferensi kedepan (forward chaining) dan aturan RSCM.

Penelitian selanjutnya dari Eva Yulianti dan Eka Gusriani membahas tentang Rekayasa Perangkat Lunak Konsultasi Penyakit Kehamilan Berbasis Kasus. Hasil dari penelitian berupa aplikasi dekstop mengunakan visual basic 6.0 yang secara sederhana membantu ibu hamil dalam mengetahui penyakit kehamilan berdasarkan gejala-gejala yang dirasakan. Hasil rancangan menunjukkan solusi permasalahan untuk setiap masalah penyakit kahamilan yang berdasarkan kasus-kasus yang ada atau yang sering terjadi.

Terdapat beberapa perbedaan dari penelitian terdahulu yang dipaparkan diatas. Perbedaan penelitian tersebut dapat dilihat pada tabel 2.1.

**Tabel 2.1. Perbedaan Kajian Terdahulu**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Peneliti | Kajian 1 | Kajian 2 | Kajian 3 | Topik yang akan dikembangkan |
| Judul | Aplikasi Sistem Pakar Berbasis Web Untuk Konsultasi Masalah Kehamilan Menggunakan Forward Chaining Dan Production Rule | Sistem Untuk Konsultasi Menu Diet Bagi Penderita Diabetes Mellitus Berbasis Aturan | Rekayasa Perangkat Lunak Konsultasi Penyakit Kehamilan Berbasis Kasus (Case Based Reasoning) Di Puskesmas Gunung Talang Menggunakan Visual Basic 6.0 | Pengembangan Aplikasi Mobile Konsultasi Obat Dengan Metode Penalaran Berbasis Kasus |
| Metode | Forward Chaining dan Production Rule | Rule Based Reasoning | Case Based Reasoning | Case Based Reasoning |
| Hasil | Hasil penelitian berupa aplikasi konsultasi masalah kehamilan secara online yang disajikan secara mudah. Hasil pengujian menggunakan metode black box menunjukkan fungsionalitas sistem dan mesin inferensi dapat bekarja dengan baik | Sistem ini dirancang dengan masukan data-data antropometri yang meliputi tinggi badan, berat badan, jenis kelamin, usia, aktifitas dan kategori badan dengan rule-based dan aturan RSCM untuk perhitungan menu diet | Rekayasa Perangkat Lunak Konsultasi Penyakit Kehamilan Berbasis Kasus diharapkan dapat membantu ibu hamil dalam mengetahui penyakit kehamilan berdasarkan gejala-gejala yang di rasakan ibu hamil dan memberikan informasi kesehatan selama kehamilan | Pasien dapat melakukan konsultasi dengan apoteker, melihat informasi data obat, dan melihat kasus-kasus/hasil konsultasi yang telah dilakukan oleh pasien yang lain |

1. **Landasan Teori**
2. **Sistem Operasi Android**
3. Definisi Android

Android adalah istilah dalam bahasa inggris yang berarti “Robot yang menyerupai manusia”. Logo “Android” sendiri, dicerminkan seperti sebuah robot berwarna hijau, yang mengacu kepada arti kata android. Android adalah sebuah sistem operasi untuk *Smartphone* dan Tablet. Sistem operasi dapat diilustrasikan sebagai ‘jembatan antar piranti (*device*) dan penggunanya, sehingga pengguna bisa berinteraksi dengan *device*-nya dan menjalankan aplikasi-aplikasi yang tersedia pada *device.* Di dunia personal komputer, sistem operasi yang banyak dipakai adalah Windows, Mac, dan Linux (Satyaputra dan Aritonang, 2014).

1. Android Software Development Kit (SDK)

Android SDK merupakan sebuah tool dan alat bantu API (*Application Programming Interface*) yang diperlukan untuk mengembangkan aplikasi berbasis android yang menggunakan bahasa Java. SDK juga sering disebut sebagai software emulator yang berguna untuk mensimulasikan OS Android pada PC. Saat ini SDK sudah berbentuk bundle, yang didalamnya sudah terdapat Eclipse + ADT (Android Development Tools) Plugin, Android SDK Tools, Android Platform-tools, Platform Android terbaru, dan *Android System Image* terbaru untuk emulator (Satyaputra dan Aritonang, 2014). Berikut beberapa tools pada Android SDK yang terintegrasi ke ADT :

1. Traceview, berfungsi untuk melihat profil eksekusi program aplikasi.
2. Android, menyediakan akses ke android SDK manager dan AVD manager.
3. Hierarchy viewer, berfungsi untuk memvisualisasikan hiearki aplikasi.
4. Pixel Perfect, berfungsi untuk meneliti UI dalam membantu perancangan aplikasi.
5. DDMS, menyediakan fitur debugging seperti screen capture, informasi thread dan hep, dan logcat.
6. Adb, menyediakan akses ke perangkat sistem pembangunan. Beberapa fitur adb yang terintegrasi dengan ADT adalah Eclipse Run Menu, transfer file, device enumeration, dan logcat (DDMS).
7. ProGuard, berfungsi untuk obfuscation, penyusutan, dan optimasi pada pemrograman.
8. Android Development Tools (ADT)

Android Development Tools merupakan plugin yang di desain powerfull dengan lingkungan yang terintegrasi dalam mengembangkan atau membangun apllikasi android memakai Eclipse IDE.



Gambar 2. Tampilan Plugin ADT

Dengan menggunakan plugin ADT dan Android SDK Tools kita dapat membuat aplikasi project android, membuat GUI aplikasi, melakukan debug aplikasi, dan juga membuat package android (.apk) yang digunakan untuk mendistribusikan aplikasi android yang sudah dirancang (Satyaputra dan Aritonang, 2014).

1. Java Development Kit (JDK)

Java Development Kit (JDK) adalah sekumpulan perangkat lunak yang dapat kamu gunakan untuk mengembangkan perangkat lunak berbasis Java, sedangkan JRE adalah sebuah implementasi dari java virtual mesin yang benar-benar digunakan untuk menjalankan program java. Biasaya setiap JDK berisi satu atau lebih JRE dan berbagai alat pengembangan lain seperti sumber compiler java, bundling, debuggers, development libraries dan lain sebagainya (Juansyah, 2015).

1. Android Studio

Android Studio adalah sebuah IDE untuk Android Development yang diperkenalkan *google* pada acara *Google* I/O 2013. Android Studio merupakan pengembangkan dari *Eclipse* IDE, dan dibuat berdasarkan IDE *Java* populer, yaitu *IntelliJ IDEA*. Android Studio merupakan IDE resmi untuk pengembangan aplikasi Android *(*Android Studio, 2016).

Android studi sendiri dikembangkan berdasarkan Intellij IDEA yang mirip dengan Eclipse disertai dengan ADT plugin (Android Development Tools). Android Studio memiliki fitur :

1. Project berbasis pada Gradle Build.
2. Refactory dan pembenahan bug yang cepat.
3. Tools baru yang bernama “Lint” diklaim dapat memonitor kecepatan, kegunaan, serta kompetibelitas aplikasi dengan cepat.
4. Mendukung Proguard And App-signing untuk keamanan.
5. Memiliki GUI aplikasi android lebih mudah.
6. Didukung oleh Google Cloud Platform untuk setiap aplikasi yang dikembangkan.
7. **Web Service**
8. Definisi Web Service

Web service adalah sekumpulan application logic beserta objek-objek dan metode-metode yang dimilikinya, yang terletak di suatu server yang terhubung ke internet sehingga dapat diakses menggunakan protocol HTTP dan SOAP (Simple Object Access Protocol). Dalm penggunaannya, web service dapat digunakan hanya untuk memeriksa data user yang login ke sebuah website ataupun untuk digunakan pada transaksi perbankan online yang rumit.

Tujuan dari teknologi ini adalah untuk memudahkan beberapa aplikasi komponennya untuk saling terhubung dengan aplikasi lain dalam sebuah organisasi maupun diluar organisasi menggunakan standar yang tidak terikat platform (platform-neutral) dan tidak terikat akan bahasa pemrograman yang digunakan. Ha tersebut dapat terjadi karena XML standar yang didukung oleh banyak perusahaan besar didunia, yang digunakan untuk bertukar data (Juansyah, 2015).

1. Java Script Object Nonation (JSON)

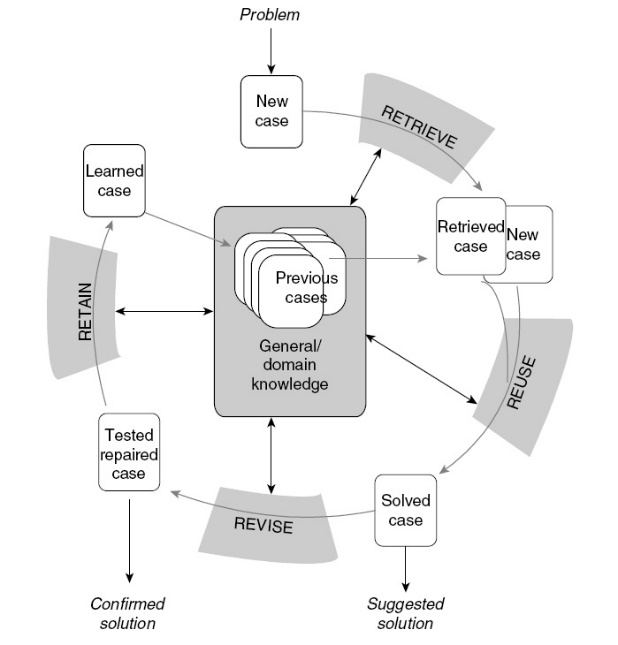
Java Script Object Nonation adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (generate) oleh komputer. Format ini dibuat berdasarkan bagian dari bahasa pemrograman Javascript. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh programmer keluarga C termasuk C, C++, C#, Java, Javascript, Perl, Phyton, dll.

1. XML

XML merupakan dasar terbentuknya web service yang digunakan untuk mendeskripsikan data. Pada level paling detail web service secara keseluruhan dibentuk diatas XML. Fungsi utama dari XML adalah komunikasi antar aplikasi, integrasi data, dan komunikasi aplikasi eksternal dengan partner luaran. Dengan standarisasi XML, aplikasi-aplikasi yang berbeda dapat dengan mudah berkomunikasi antar satu dengan yang lain (Deviana, 2011).

1. ***Case Based Reasoning***
2. Definisi *Case Based Reasoning*

Penalaran berbasis kasus atau PBK adalah salah satu metode pendekatan berbasis pengetahuan untuk mempelajari dan memecahkan masalah berdasarkan pengalaman pada masa lalu. Pengalaman yang lalu dikumpulkan dan disimpan dalam tempat yang disebut Basis Kasus. Basis kasus adalah kumpulan kasus-kasus yang pernah terjadi. *Case Based reasoning* merupakan teknik penyelesaian masalah berdasarkan knowledge pengalaman yang lalu. Aamodt dan Plaza (1994) menggambarkan tipe CBR sebagai suatu proses melingkar yang terdiri dari *The Four Res* yaitu :



Gambar 2. Siklus CBR (Aamodt dan Plaza, 1994)

1. *Retrieve*adalah menemukan kembali kasus yang paling mirip dengan kasus baru yang akan dievaluasi.
2. *Reuse* adalah menggunakan kembali informasi atau pengetahuan yang telah tersimpan pada basis kasus untuk memecahkan masalah kasus.
3. *Revise* adalah memperbaiki solusi yang diusulkan.
4. *Retain* adalah menyimpan pengetahuan yang nantinya akan digunakan untuk memecahkan masalah kedalam basis kasus yang ada.
5. Kemiripan (*Similarity*)

Kemiripan (*similarity*) adalah langkah yang digunakan untuk mengenali kesamaan atau kemiripan antara kasus-kasus yang tersimpan dalam basis kasus dengan kasus yang baru. Kasus dengan nilai similarity paling besar dianggap sebagai kasus yang “paling mirip”. Nilai similarity berkisar antara 0 sampai 1 (Akmal dan Winiarti, 2014). Berikut ini merupakan persamaan untuk mencari nilai kemiripan (*similarity*).

Untuk mencari nilai kemiripan (*similarity*) digunakan persamaan 2.1.

Ti =

Keterangan :

Ti = nilai kesamaan dengan kasus ke-i.

nX1-nXn = banyaknya kesamaan subobjek nXn.

N = banyaknya elemen pada basis kasus.

1. **Unified Modeling Language (UML)**

UML adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun (Dharwiyanti, 2003).

UML mendefinisikan diagram-diagram antara lain :

1. Diagram Use Case

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah use case merepresentasikan sebuah interaksi antara aktor dengan sistem. Use case merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-create sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

Use case diagram dapat sangat membantu bila kita sedang menyusun requirement sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang test case untuk semua feature yang ada pada sistem (Dharwiyanti, 2003).

1. Diagram Aktivitas

Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir. Activity diagram juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Sebuah aktivitas dapat direalisasikan oleh satu use case atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara use case menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas (Dharwiyanti, 2003).

1. **Rekayasa Perangkat Lunak**

Perangkat lunak adalah instruksi-instruksi (program komputer) yang ketika dijalankan menyediakan fitur-fitur, fungsi-fungsi, dan kinerja-kinerja yang dikehendaki (Presman, 2012).

Menurut IEEE [IEE93a] rekayasa perangkat lunak pada dasarnya merupakan aplikasi dari suatu pendekatan yang sistematik, disiplin, dan dapat diukur pada pengembangan, operasi, dan perawatan perangkat lunak. Terdapat suatu kerangka kerja proses pada rekayasa perangkat lunak yang terdiri dari lima aktivitas.

1. Komunikasi

Sebelum pekerjaan teknis apapun dimulai, sangatlah penting untuk berkomunikasi dan berkolaborasi dengan pelanggan. Maksudnya adalah untuk memahami tujuan-tujuan *stakeholder* atas proyek perangkat lunak yang sedang dikembangkan dan mengumpulkan kebutuhan-kebutuhan yang akan membantu mengartikan fitur-fitur perangkat lunak beserta fungsi-fungsinya.

1. Perencanaan

Alur yang rumit dapat disederhanakan jika terdapat suatu peta. Suatu proyek perangkat lunak pada dasarnya merupakan suatu alur yang rumit, dan kegiatan perencanaan perangkat lunak tersebut menciptakan suatu “peta” yang membantu membimbing tim perangkat lunak ketika mereka melakukan suatu perjalanan. Peta perangkat lunak tersebut yang disebut sebagai rencana proyek perangkat lunak yang menggambarkan tugas-tugas teknis yang harus dilakukan, risiko-risiko yang mungkin muncul, sumber daya yang akan dibutuhkan, produk-produk kerja yang harus dihasilkan, dan jadwal-jadwal kerja.

1. Pemodelan

Aktivitas pemodelan digambarkan layaknya seorang penata taman, pembuat jembatan, seorang rekayasawan dirgantara, seorang tukang kayu, ataukah seorang arsitek, bahkan seorang musisi profesional. Setiap harinya membuat sketsa atau membuat gambaran dari sebuah model-model benda. Seorang rekayasawan perangkat lunak melakukan hal yang sama dengan cara membuat model-model untuk memahami kebutuhan perangkat lunak maupun rancangan-rancangan yang akan memenuhi kebutuhan tersebut.

1. Konstruksi

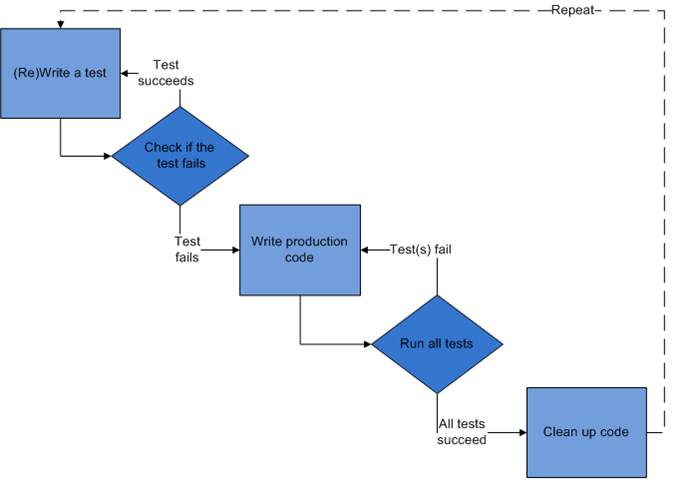
Kegiatan ini menggabungkan pembentukan kode (*code generation*) dan pengujian yang sangat dibutuhkan untuk menemukan kekeliruan-kekeliruan/kesalahan-kesalahan dalam kode program komputer yang dihasilkan sebelumnya.

1. *Deployment*

Pada tahap ini, perangkat lunak disajikan kepada pelanggan yang kemudian akan mengevaluasi produk yang disajikan dan akan memberikan umpan balik berdasarkan evaluasi tersebut.

1. **Test Driven Development**

Test-driven development (TDD) adalah proses pengembangan perangkat lunak yang bergantung pada pengulangan siklus pengembangan yang sangat singkat, dimana programer dituntut untuk memikirkan terlebih dahulu output yang hasilkan oleh method atau fungsi tersebut serta parameter atau input apa saja yang akan dibutuhkan. Kemudian dibuat sebuah *test script*untuk menguji apakah input dan output yang diinginkan telah sesuai. TDD merupakan uji testing yang baik untuk pengembangan Software yang baru pertama kali dikembangkan dan membuat sebuah aplikasi yang lebih stabil karena kesalahan-kesalahan yang terjadi pada setiap fungsi bisa dideteksi dengan lebih cepat. Alur kerja dari penggembangan menggunakan *Test Driven Developmen* (TDD) dapat dilihat pada Gambar 2.2.



Gambar 2. Alur Kerja Test Driven Development (TDD)

Penjelasan alur kerja pengujian menggunakan *Test Driven Development* (TDD), sebagai berikut :

1. Uji kinerja modul.
2. Jika kode modul ini bekerja dengan baik maka uji sukses.
3. Jika tes tidak berhasil maka Pergi ke modul perpustakaan.
4. Jadi pilih modul specfic dari perpustakaan modul.
5. Ulangi langkah pertama dan menguji kode yang dipilih dari modul perpustakaan modul.

Pengembangan software dengan *Test Driven Development* (TDD) dengan mengambil langkah-langkah kemungkinan kecil ketika diperlukan. Hal ini memungkinkan bagi *programmer*  untuk fokus pada pembuataan aplikasi agar dalam pengujian pada aplikasi yang dibuat tidak terdapat *error* (Khan, dkk., 2013)*.*