BAB II

LANDASAN TEORI

2.1 Sistem Pakar

Sistem pakar merupakan cabang dari *Artificial Intelligence* (AI) yang membuat penggunaan secara luas *knowledge* yang khusus untuk penyelesaian masalah tingkat manusia yang pakar dan juga merupakan bidang ilmu yang sering muncul seiring perkembangan ilmu komputer pada saat ini (Nasution et al., 2017). Menurut Shelly (1990), sistem pakar merupakan suatu sistem komputer yang dibangun agar dapat dilakukan penalaran seperti layaknya seorang pakar pada suatu bidang keahlian tertentu (H. Hayadi, 2018).

Seorang pakar adalah orang yang mempunyai keahlian dalam bidang tertentu, yaitu pakar yang mempunyai *knowledge* atau kemampuan khusus yang orang lain tidak mengetahui atau mampu dalam bidang yang dimilikinya. Ketika sistem pakar dikembangkan pertama kali sekitar tahun 70-an sistem pakar hanya berisi *knowledge* yang ekslusif. Namun demikian sekarang ini istilah sistem pakar sudah digunakan untuk berbagai macam sistem yang menggunakan teknologi sistem pakar. Teknologi sistem pakar ini meliputi bahasa sistem pakar, pemograman dan perangkat keras yang dirancang untuk membantu pengembangan dan pembuatan sistem pakar.

2.1.1 Konsep Umum Dasar Kepakaran

Pengetahuan dari suatu sistem pakar mungkin dapat direpresentasikan dalam sejumlah cara. Salah satu metode yang paling umum untuk pengetahuanadalah dalam bentuk tipe aturan (*rule*) IF…THEN (**Jika…Maka**).

Pada penelitian (B. H. Hayadi, 2017), Turban (1995) menyatakan bahwa konsep dasar dari suatu sistem pakar mengandung beberapa unsur/elemen, yaitu keahlian, ahli, pengalihan keahlian, inferensi, aturan, dan kemampuan menjelaskan.Keahlian merupakan suatu penguasaan pengetahuan dibidang tertentu yang didapatkan dari pelatihan, membaca atau pengalaman. Contoh bentuk pengetahuan yang merupakan keahlian adalah:

a. Fakta-fakta pada lingkup permasalahan tertentu.
b. Teori-teori pada lingkup permasalahan tertentu.
c. Prosedur-prosedur dan aturan-aturan berkenaan dengan lingkup permasalahan tertentu.
d. Strategi-strategi global untuk menyelesaikan masalah.
e. *Meta-knowledge* (pengetahuan tentang pengetahuan)

Menurut Turban dan Arhami (2005,11) dalam penelitian (Yusuf et al., 2016)terdapat tiga orang yang terlibat dalam lingkungan sistem pakar yaitu sebagai berikut:

a. Pakar

   Pakar adalah orang yang memiliki pengetahuan khusus, pendapat, metodeserta kemampuan untuk mengaplikasikan keahlian tersebut guna menyelesaikan masalah.

b. *Knowledge Engineer* (Perekayasa Sistem)

   Knowledge Engineer adalah orang yang membantu pakar dalam menyusun area permasalahan dengan menginterpretasikan dan menginterpretasikan jawaban-jawaban pakar atas pertanyaan yang diajukan, menggambarkan analogi, mengajukan *counter example* dan

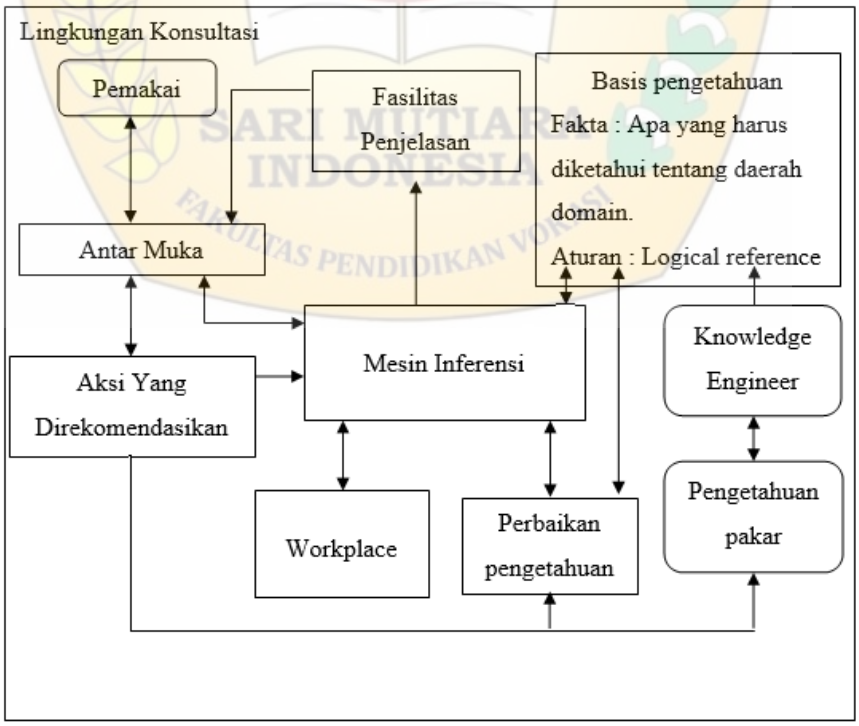
menerangkan kesulitan-kesulitan konseptual.

c. Pemakai

Sistem pakar memiliki beberapa pemakai, yaitu: pemakai bukan pakar, pelajar, pembangun sistem pakar yang ingin meningkatkan dan menambah basis pengetahuan, dan pakar.

2.1.2 Struktur Sistem Pakar

Sistem pakar disusun oleh dua bagian utama, yaitu lingkungan pengembangan (*development environment*) dan lingkungan konsultasi (*consutation environment*) (Yusuf et al., 2016).Lingkungan pengembangan sistem pakar digunakan untuk memasukan pengetahuan pakar ke dalam lingkungan sistem pakar, sedangkan lingkungan konsultasi digunakan oleh pengguna yang bukan pakar guna memperoleh pengetahuan pakar. Komponen–komponen sistem pakar dalam kedua bagian tersebut dapat dilihat didalam gambar berikut ini:



Gambar 2.1 Arsitektur Sistem Pakar

Komponen-komponen yang terdapat dalam sistem pakar adalah seperti yang terdapat pada Gambar 2.1 yaitu *User interface* (antarmuka pengguna), basis pengetahuan, akuisi pengetahuan, mesin inferensi, *workplace*, fasilitas penjelasan, perbaikan pengetahuan.

a. Antarmuka Pengguna

   User interface (UI) merupakan bentuk tampilan grafis yang mempunyai relasi langsung dengan *user* dengan tujuan untuk menghubungkan antara *user* dengan sistem operasi, sehingga komputer tersebut dapat difungsikan (Masturoh et al., 2019). Selain itu antar muka menerima informasi dari sitem dan menyajikannya kedalam bentuk yang dapat dimengerti oleh pemakai. Pada bagian ini terjadi dialog antara program dan pemakai, yang memungkinkan sistem pakar menerima instruksi dan informasi (*input*) dari pemakai, juga memberikan informasi (*output*) kepada pemakai.

b. Basis Pengetahuan

   Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah. Komponen sistem pakar ini disusun atas dua elemen dasar, yaitu fakta dan aturan. Fakta merupakan informasi tentang obyek dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

c. Akuisisi Pengetahuan (*Knowledge Acquisition*)

   Akuisisi pengetahuan adalah akumulasi, transfer dan tranformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan ke

dalam program komputer. Dalam tahap ini *knowledge engineer* berusaha menyerap pengetahuan untuk selanjutnya ditransfer ke dalam basis pengetahuan. Pengetahuan diperoleh dari pakar, dilengkapi dengan buku, basis data, laporan penelitian dan pengalaman pemakai.

d. Mesin Inferensi

Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelsaikan suatu masalah. Mesin inferensi adalah program komputer yang memberikan metodologi untuk penalaran tentang informasi yang ada dalam basis pengetahuan dan dalam *workplace*, dan untuk memformulasikan kesimpulan.

e. *Workplace*

Workplace merupakan area dari sekumpulam memori kerja (*working memory*). *Workplace* digunakan untuk merekam hasil-hasil antara dan kesimpulan yang dicapai. Ada tiga tipe keputusan yang dapat direkam yaitu Rencana : Bagaimana menghadapi masalah Agenda : Aksi-aksi yang potensial yang sedang menunggu untuk dieksekusi. Solusi : Calon aksi yang akan dibangkitkan.

f. Fasilitas Penjelasan

Fasilitas penjelasan adalah komponen tambahan yang akan meningkatkan kemampuan sistem pakar. Komponen ini menggambarkan penalaran sistem kepada pemakai.

g. Perbaikan Pengetahuan

Pakar memiliki kemampuan untuk menganalisis dan meningkatkan kinerjanya serta kemampuan untuk belajar dari kinerjanya. Kemampuan

tersebut sangat penting dalam pembelajaran terkomputerisasi, sehingga program akan mampu menganalisis penyebab kesuksesan dan kegagalan yang dialaminya.

2.1.3 Rule Sebagai Teknik Representasi Pengetahuan

Sistem *rule* terdiri dari dua bagian, yaitu bagian IF disebut *evidence* (Fakta-fakta) dan bagian THEN disebut *hipotesis* atau kesimpulan.

E : *Evidence* (Fakta-fakta) yang ada

H: *Hipotesis* (Kesimpulan) yang dihasilkan

Secara umum *rule* mempunyai *evidence* lebih dari satu yang dihubungkan oleh kata penghubung *AND* atau *OR*, atau kombinasi keduanya. Tetapi sebaliknya biasakan menghindari pengguna *AND* dan *OR* secara sekaligus dalam satu *rule*.

IF (E1 AND E2 AND E3 ……. AND En) THEN H

IF (E1 OR E2 OR E3 ……. AND En) THEN H

2.1.4 Manfaat Sistem Pakar

Sistem pakar menjadi sangat popular karena sangat banyak kemampuan dan manfaat dalam buku (H. Hayadi, 2018), diantaranya yaitu:

1. Meningkatkan produktivitas, karena sistem pakar dapat berkerja lebih cepat daripada manusia.
2. Membuat seseorang yang awam bekerja seperti layaknya seorang pakar.
3. Meningkatkan kualitas, dengan member nasehat yang konsisten dan mengurangi kesahalan.
4. Mampu menangkap pengetahuan dan kepakaran seseorang.
5. Memudahkan akses pengetahuan seorang pakar.

6. Bisa digunakan sebagai media pelengkap dalam pelatihan. Pengguna pemula yang bekerja dengan sistem pakar akan menjadi lebih berpengalaman kerena adanya fasilitas penjelas yang berfungsi sebagai guru.
7. Meningkatkan kemampuan untuk menyelesaikan masalah kerena sistem pakar mengambil sumber pengetahuan dari banyak pakar.

2.1.5 Kekurangan Sistem Pakar

Selain manfaat, ada juga beberapa kekurangan yang ada pada Sistem Pakar, diantaranya adalah (H. Hayadi, 2018) :

1. Biaya yang sangat mahal untuk membuat dan memeliharanya.
2. Sulit dikembangkan karena keterbatasan keahlian dan ketersedian pakar.
3. Sistem pakar tidak 100% bernilai benar.

2.2 AC (*Air Conditioner)*

Air Conditioner (AC) adalah alat elektronik yang berfungsi dalam mengkondisikan udara untuk dapat membuat sejuk didalam sebuah ruangan tertutup yang disesuikan dengan kondisi tubuh. Penggunaan AC selain membuat udara menjadi sejuk, AC juga dapat meningkatkan kualitas udara dan dapat mengurangi kerusakan asma dan alergi(Mair, 1392).

AC (*AirConditioner*) merupakan sebuah alat perangkat listrik yang digunakan untuk mengkondisikan suhu didalam suatu ruangan. Salah satu prinsip kerja dari AC yaitu dengan menyerap panas dari udara didalam ruangan melalui *indoor* unit, kemudian melepaskan panas tersebut ke luar ruangan melalui *outdoor* unit.Dengan kata lain AC berfungsi sebagai penyejuk udara yang diinginkan (sejuk atau dingin) dan nyaman bagi tubuh. AC juga dapat dikategorikan untuk mengatur suhu udara, sirkulasi, kelembapan, dan kebersihan udara dalam ruangan,

AC juga dapat mempertahankan kondisi udara, baik suhu dan kelembapannya agar didalam ruangan terasa nyaman. Dengan demikian, temperatur udara didalam ruangan akan berangsur-angsur turun sehingga dapat menghasilkan temperatur udara yang diinginkan. Udara didalam ruangan yang terserap disirkulasikan secara terus menerus oleh *blowerindoor* melewati sirip-sirip *evaporator*. Saat melewati *evaporator*, udara yang bertemperatur lebih tinggi dari *evaporator* diserap panasnya oleh bahan pendingin (*refrigeran*), kemudian dilepaskan diluar ruangan ketika aliran *refrigeran* melewati kondensor.

2.3 *Certainty Factor* (Faktor Kepastian)

Certainty Factor pertama kali diperkenalkan oleh *Shortliffe* dan Buchanan pada tahun 1975 untuk mengakomodasi ketidakpastian pemikiran (*inexact reasoning*) seorang pakar. Seorang pakar (misalnya dokter) seringkali menganalisis informasi yang ada dengan ungkapan seperti "mungkin", "kemungkinan besar", "hampir pasti". Untuk mengakomodasi hal ini maka digunakan *certainty factor* (CF) guna menggambarkan tingkat keyakinan pakar terhadap masalah yang sedang dihadapi. Dalam penelitian (Zuhriyah & Wahyuningsih, 2019), *Certainty factor* adalah salah satu metode yang digunakan untuk membuktikan apakah suatu fakta itu pasti atau tidak pasti yang berbentuk matrik yang biasanya digunakan dalam mendeteksi sesuatu yang belum pasti.

Didalam metode ini terdapat dua cara yang bisa dilakukan dalam mendapatkan tingkat keyakinan (CF) dari sebuah *rule*, yaitu:

1. Metode *Net belief* yang diusulkan oleh E. H. Shortfille dan B. G. Bachnan

   CF (*Rule*) = MB (H,E)

$$MB(H|E) = \left\{\frac{MAX\,[P\,(H|E),\ P(H)-P(H)]}{MAX[1,0]-P\,(H)}\right\}\ P(H) = 1$$

$$MD(H|E) = \left\{\frac{MIN\,[P\,(H|E),\ P(H)-P(H)]}{MIN[1,0]-P\,(H)}\right\}\ P(H) = 0$$

$$CF\,[H,E] = MB\,[H,E] - MD\,[H,E]$$

Dimana:

CF (*Rule*) : Factor kepastian

MB (H,E) : *Measure of Beliefe* (ukuran kepercayaan) terhadap hipotesis H, jika diberikan *evidence* E (antara 0 dan 1)

MD (H,E) : *Measure of Disbeliefe* (ukuran ketidakpercayaan) terhadap *evidence* E (antara 0 dan 1)

P(H) : Probabilitas kebenaran hipotesis H

P(H|E) : Probabilitas bahwa H benar karena fakta E

2. Dengan cara mewawancarai seorang pakar

Nilai CF (*Rule*) didapat dari interpretasi "*term*" dari pakar, yang diubah menjadi nilai CF tertentu sesuai table berikut :

Tabel 2.1 Nilai Kepastian

No	Uncertain	CF
1	*Definitely not* (Tidak pasti)	-1.0
2	*Almost certainty not* (Hampir pasti tidak)	-0.8
3	*Probability not* (Kemungkinan besar tidak)	-0.6
4	*Maybe not* (Mungkin tidak)	-0.4
5	*Unknown* (Tidak tahu)	-0.2 to 0.2
6	*Maybe* (Mungkin)	0.4
7	*Probability* (Kemungkinan besar)	0.6

8	*AlmostCertainty* (Hampir pasti)	0.8
9	*Definetely* (Pasti)	1.0

2.3.1 Perhitungan *Certainty Factor*

Secara umum, *rule* direpresentasikan dalam bentuk sebagai berikut :

IF E1 AND E2……..AND En THEN H (CF *rule*)Atau

IF E1 OR E2……..OR En THEN H (CF *rule*)

Dimana :

E1….En : Fakta-fakta (*evidence*) yang ada

H : Hipotesis atau konklusi yang dihasilkan

CF *Rule* : Tingkat keyakinan terjadinya hipotesis H akibat adanya 1fakta-fakta E1….En

1. *Rule* dengan *evidence* E tunggal dan hipotesis H tunggal

   IF E_1 **AND** E_2 …….. AND E_n **THEN** H (CF *Rule*)

   CF (H,E) = CF (E) x CF (*C*)

2. *Rule* dengan *evidence* E ganda dan hipotesis H tunggal

   IF E_1 **AND** E_2 …….. AND E_n **THEN** H (CF *Rule*)

   CF (H,E) = min [CF (E_1), CF (E_2),…., CF(E_n)] x CF (*rule*)

   IF E_1 **OR** E_2 …….. OR E_n **THEN** H

   CF (H,E) = max [CF (E_1), CF (E_2),…., CF(E_n)] x CF (*rule*)

3. Kombinasi dua buah *rule* dengan *evidence* berbeda (E_1dan E_2), tetapi hipotesisnya sama.

   IF E_1 **THEN** H *Rule* 1 CF (H,E_1) = CF_1 = C (E_1) x CF (*rule* 1)

   IF E_2 **THEN** H *Rule* 2 CF (H,E_1) = CF_2 = C (E_2) x CF (*rule* 2)

$$CF(CF_1, CF_2) \begin{cases} CF_1 + CF_2\,(1-CF_1) & \text{Jika } CF_1>0 \text{ dan } CF_2>0 \\ \dfrac{CF_1 + CF_2}{1-\ Min[CF_1, CF_2]} & \text{Jika } CF_1>0 \text{ dan } CF_2<0 \\ CF_1 + CF_2(1 + CF_1) & \text{Jika } CF_1<0 \text{ dan } CF_2<0 \end{cases}$$

2.3.2Kelebihan Dan Kekurangan Metode *Certainty Factor*

1. Kelebihan metode *Certainty Factor* adalah :
   a. Metode ini cocok dipakai dalam sistem pakar yang mengandung ketidakpastian.
   b. Dalam sekali proses perhitungan hanya dapat mengelola dua data saja sehingga keakuratan data dapat terjaga.
2. Kekurangan metode *Certainty Factor* adalah :
   a. Pemodelan ketidakpastian yang menggunakan perhitungan metode *certainty factor* biasanya masih diperdebatkan.
   b. Untuk data lebih dari dua harus dilakukan beberapa kali pengolahan data.

2.4 UML (*Unified Modeling Language*)

*Unified Modeling Language*adalahsebuah metode pemodelan visual yang sering digunakan dalam membangun perancangan dan pembuatan sebuah aplikasi lunak yang berorientasi pada objek. UML ini merupakan salah satu standar penulisan atau semacam *blue print* dimana didalamnya termasuk sebuah bisnis proses, serta penulisan kelas-kelas dalam sebuah bahasa yang spesifik (Prihandoyo, 2018).

UML adalah metode pengembangan perangkat lunak (sistem informasi) dengan metode grafis serta merupakan bahasa untuk visualisasi, spesifiasi, konstruksi serta dokumentasi dari komponen-komponen perangkat lunak dan digunakan untuk pemodelan bisnis.Pemodelan UML berarti menggambarkan yang ada dalam dunia nyata kedalam bentuk yang dapat dipahami denganmenggunakan UML.Berikut proses dalam membuat gambaran UML (*Unified Modeling Language*).

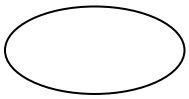
1. *Functional Requirement*

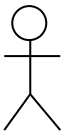
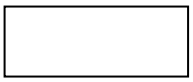
   Functional Requirement adalah dukomen yang berisi rincian kebutuhan user yang menyatakan masalah yang dihadapi user dan solusi yang dibutuhkan.

2. *Use Case Diagram*

   Use Case Diagram digunakan untuk memodelkan dan menyatakan unit fungsi/layanan yang disediakan oleh sistem ke pamakai, *Use Case Diagram* adalah interaksi atau dialog antara sistem dan *actor*, termasuk pertukaran pesan dan tindakan yang dilakukan oleh sistem. Berikut daftar simbol-simbol dari *Use Case Diagram.*

Tabel 2.2 Simbol-simbol *Use Case Diagram*

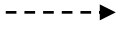
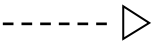
Nama komponen	**Deskripsi**	**Gambar**
Use case	Menerangakan “apa” yang dikerjakan sistem, bukan “bagaimana” sistem mengerjakan	
	Menggambarkan orang, sistem atau	

Actor	external entitas/stakeholder yang menyediakan atau menerima inforamasi dari system	
SistemBoundary	Menggambarkan jangkauan sistem	
Association	Menggambarkan bagaimana *actor* terlibat dalam *use case*	
generalization	Dibuat ketika ada sebuah keadaaan yang lain/ perlakuan khusus	
Extend	Perluasan dari use case lain jika kondisi atau syarat terpenuhi	<<Extend>>
Include	Menjelaskan bahwa use case termasuk didalam use case lain	<<Include>>

3. *Class Diagram*

Class Diagram digunakan untuk menampilkan kelas-kelas dan paket-paket didalam sistem. *Class Diagram* memberikan gambaran sistem secara statis dan relasi antar muka. Diagram tersebut membantu untuk mendapatkan struktur sistem sebelum kode ditulis, dan membantu untuk memastikan bahwa sistem adalah desain terbaik. Berikut daftar simbol-simbol dari *Class Diagram*.

Tabel 2.3 Simbol-Simbol *Class Diagram*

Indikator/Gambar	**Deskripsi**
0…1	Kosong atau satu
0…*	Lebih dari sama dengan kosong
1	Lebih sama dengan n, dimana lebih dari satu
1…*	Hanya satu
1…n	Lebih dari sama dengan satu
*	Banyak atau many
N	Hanya N dimana N lebih dari satu
n…*	Lebih dari sama dengan N dimana N lebih dari satu
n…m	Lebih dari sama dengan N dan kurang dari sama dengan M
	Agregasi
	Kunci gabungan
	Turunan
	Relasi
	*Depedencies*
	*Realization*
+	*Public*
#	*Protected*

$	*Static*
/	*Drived*
*	*Abstrak*

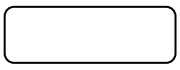
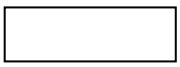
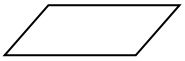
2.4.1 *Flowchart*

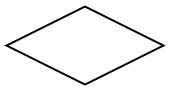
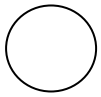
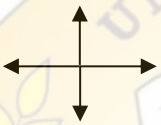
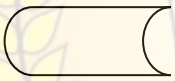
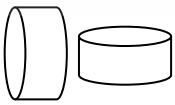
Flowchart (Diagram Alir) atau disebut *Flowchart* merupakan bagan (*Chart*) yang mengarahkan alir di dalam prosedur atau program sistem secara logika. *Flowchart* adalah cara untuk menjelaskan tahap-tahap pemecahan masalah dengan merepresentasikan simbol-simbol tertentu yang mudah dipahami, mudah digunakan dan standar(Syamsiah, 2019).

Flowchart membantu seorang analis dan programmer dalam memecahkan suatu masalah serta dapat membantu dalam menganalisis alternatif – alternatif dalam pengoperasiannya.

Fungsi *Flowchart* untuk menggambarkan sebuah proses agar mempermudah pemahaman dan mudah dilihat berdasarkan urutan langkahnya berdasarkan proses yang satu ke proses yang lainnya. Berikut ini adalah simbol-simbol *flowchart* yang secara umum digunakan sebagai berikut:

Tabel 2.4 Simbol-Simbol *Flowchart*

No	Simbol	Fungsi
1		Terminal, untuk memulai atau mengakhiri suatu program
2		Proses suatu simbol yang menunjukkan setiap pengolahan yang dilakukan
3		Input-output untuk memasukkan data ataupun menunjukkan hasil dari suatu proses

4		Desecion, suatu kondisi yang akan menghasilkan beberapa kemungkinan jawaban atau pilihan
5		Conector, suatu prosedur akan masuk atau keluar melalui simbol ini dalam lembar yang sama.
6		Off-page Connector, merupakan simbol masuk atau keluarnya suatu prosedur pada lembar kertas lainnya
7		Flow, arus dari pada prosedur yang dapat dilakukan atas bawah dan bawah keatas, dari kiri kekanan ataupun dari kanan kekiri
8		Stored data, penyimpanan data secara sementara
9		Predefined process, untuk menyatakan sekumpulan langkah proses yang ditulis sebagai procedure
10		Simbol penyimpanan/storage pada komputer, misalnya menyimpan database

2.5 Aplikasi Pendukung

2.5.1 *Microsoft Visual Basic*

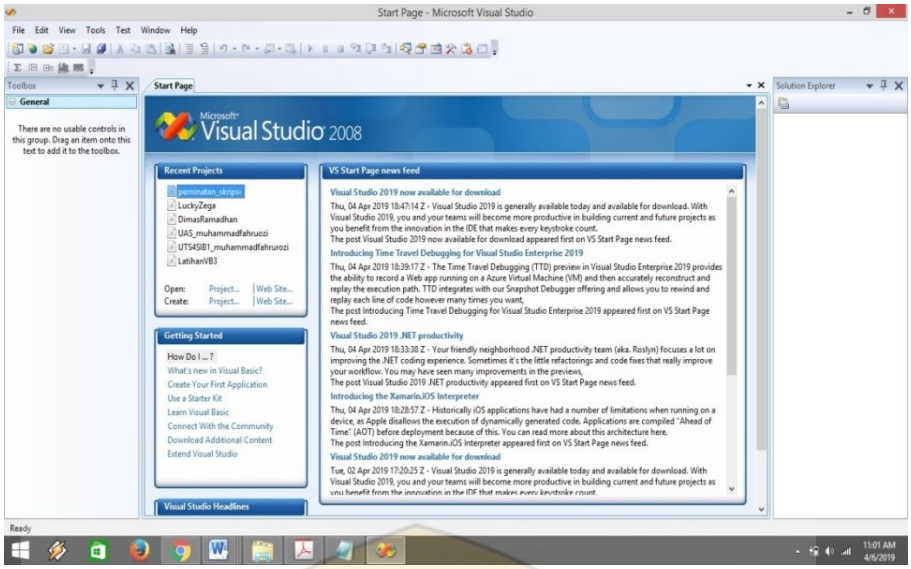
*Microsoft Visual Basic*adalahsebuah bahasa pemrograman komputer yang berorientasi objek yang diimplementasikan oleh .NET Framework 2.0 dan merupakan evolusi dari bahasa *Visual Basic* klasik (Tewari, 2020).

Pada penelitian(Irviani & Oktaviana, 2017) menjelaskan menurut (Jung, G, David, 2000) *MicrosoftVisualBasic* merupakan sebuah bahasa pemrograman yang menawarkan *IntegratedDevelopmentEnvironment* (IDE) *visual* untuk membuat program perangkat lunak berbasis sistem operasi *MicrosoftWindows* dengan menggunakan model pemrograman (COM).

Visual Studio ini telah mengalami perubahan versi mulai dari *Visual Studio* 6.0, *Visual Studio* 2005, *Visual Studio* 2006, dan yang terakhir adalah *Visual Studio* 2008. Namun pada *Visual Studio* 2008, penulis tidak melihat satu bahasa pemrograman yang termasuk bagian dari *Visual Studio*, yaitu *Microsoft Visual FoxPro*.

Microsoft Visual Basic 2008 setara dengan *Microsoft Visual Basic* 9.0, yang memiliki kelebihan-kelebihan, yaitu *suport* dengan bahasa *query Language-Integreted Query* (LINQ) dan *suport* dengan *database Microsoft SQL Server Compact* 3.5. Selain itu, kelebihan lain adalah memiliki *Object Relational Designer* (O/R *Designer*) untuk membantu mengedit LINQ ke SQL yang akan dihubungkan dengan *database* dan *fiture* lain, seperti WPF (*Windows Presentation Foundation*) dan WCF (*Windows Communication Foundation*).

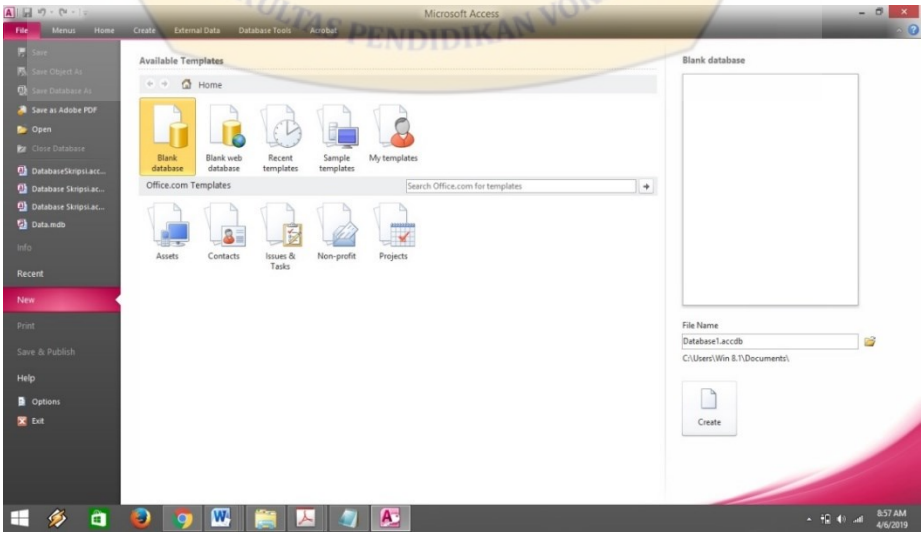
Semua hal yang baru tersebut diatas menambah kelengkapan aplikasi *Microsoft Visual Basic* 2008 dalam membuat media dan dokumen.



Gambar 2.2Tampilan Awal *Microsoft Visual Basic* 2008

2.5.2 *Microsoft Office Access*

Microsoft Office Access adalah salah satu aplikasi *Microsoft Office* yang secara khusus dikembangkan untuk kebutuhan pemograman *database*(Wirawan et al., 2017). Selain itu *Microsoft Office Access* merupakan program *database* yang digunakan untuk mengelolah berbagai jenis data. *Microsoft Acces* memiliki beberapa komponen yang mendukung akan pembuatan *database* atau pangkalan data diantaranya *table*, *field*, *query*, *form*, dan data yang dibutuhkan.

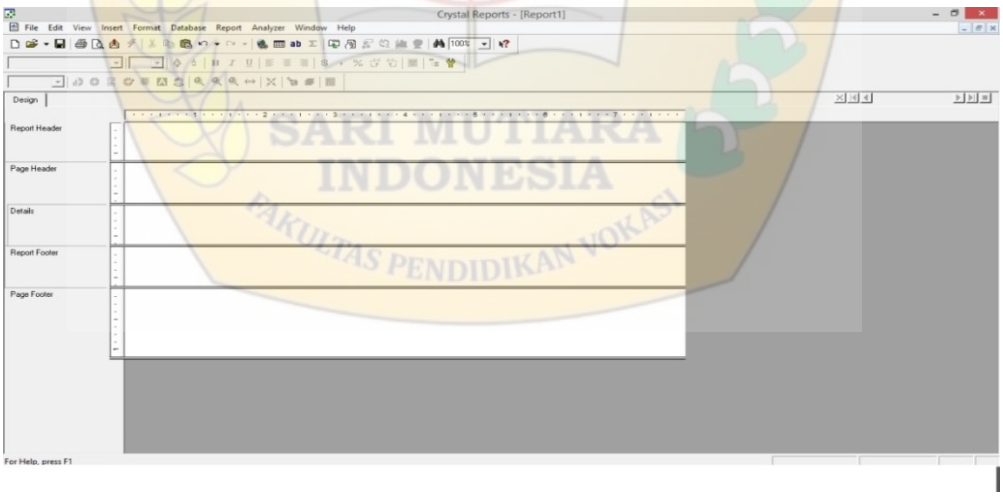


Gambar 2.3 Tampilan Awal *MicrosoftOfficeAccess*

2.5.3 *Crystal Report*

Crystal report merupakan program aplikasi *windows* yang digunakan untuk membuat laporan yang diperlukan oleh sebuah program aplikasi database yang membutuhkan sebuah laporan dari suatu data. Kelebihan dari *Crystal report* adalah mampu mengumpulkan data dari sebuah *database* kedalam bentuk yang lebih baik dan menarik, dapat memperlihatkan data berbentuk grafik, angka dan juga kolom.

Crystal Reports merupakan salah satu paket program yang digunakan untuk membuat, menganalisa, dan menterjemahkan informasi yang terkandung dalam database kedalam berbagai jenis laporan. *Crystal Reports* dirancang untuk membuat laporan yang dapat digunakan dengan berbagai macam bahasa pemrograman berbasis *Windows*, seperti *Visual Basic*, *Visual C/C++, VisualInterdev,* dan *Borland Delphi*.



Gambar 2.4 Tampilan Awal *CrystalReport*