BAB II

TELAAH LITERATUR

2.1 Pengertian Sistem Pendukung Keputusan

Sistem Pendukung Keputusan (SPK) atau *Decision Support System* (DSS) adalah sebuah sistem yang mampu memberikan kemampuan pemecahan masalah maupun kemampuan pengkomunikasian untuk masalah dengan kondisi semi terstruktur dan tak terstruktur. Sistem ini digunakan untuk membantu pengambilan keputusan dalam situasi semi terstruktur dan situasi yang tidak terstruktur, dimana tak seorangpun tahu secara pasti bagaimana keputusan seharusnya dibuat (Turban, 2001).

Sistem Pendukung Keputusan (SPK) terdiri dari beberapa tahapan yaitu:

1. Fase Inteligensi

   Intelegensi dalam pengambilan keputusan meliputi *scanning* lingkungan, secara *intermitten* ataupun terus-menerus. Inteligensi mencakup berbagai aktivitas yang menekankan identifikasi situasi atau peluang-peluang masalah.

2. Fase Desain

   Fase Desain meliputi penemuan atau mengembangkan dan menganalisis tindakan yang mungkin untuk dilakukan.

3. Fase Pilihan

   Pilihan merupakan tindakan pengambilan keputusan yang kritis. Fase pilihan adalah fase di mana dibuat suatu keputusan yang nyata dan diambil suatu komitmen untuk mengikuti suatu tindakan tertentu.

4. Fase Implementasi

Implementasi berarti membuat suatu solusi yang direkomendasikan untuk bisa bekerja.

Sistem Pendukung Keputusan (SPK) juga memiliki berbagai komponen yaitu:

1. *Data Management*

Termasuk *database*, yang mengandung data yang relevan untuk berbagai situasi dan diatur oleh *software* yang disebut *Database Management System* (DBMS).

2. *Model Management*

Melibatkan model finansial, statistikal, *management science*, atau berbagai model kualitatif lainnya, sehingga dapat memberikan ke sistem suatu kemampuan analitis, dan manajement *software* yang dibutuhkan.

3. *Communication*

User dapat berkomunikasi dan memberikan perintah pada Sistem Pendukung Keputusan (SPK) melalui subsistem ini. Yang berarti menyediakan antarmuka.

4. *Knowledge Management*

Subsistem *Optional* ini dapat mendukung subsistem lain atau bertindak sebagai komponen yang berdiri sendiri.

2.2 Multiple Attribute Decision Making (MADM)

Multiple Attribute Decision Making (MADM) adalah suatu metode yang digunakan untuk mencari alternatif optimal dari sejumlah alternatif dengan kriteria tertentu. Metode SAW sering juga dikenal sebagai metode penjumlahan yang

memberikan bobot. Konsep dasar metode SAW adalah mencari penjumlahan, kemudian memberikan bobot dari *rating* kinerja pada setiap alternatif pada semua atribut. Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua *rating* alternatif yang ada. Tahapan dalam metode SAW adalah sebagai berikut:

1. Menentukan alternatif A = {A1, A2, ….., Ai}
2. Menentukan kriteria yang akan dijadikan acuan dalam pengambilan keputusan C = {C1, C2, ….., Cj}
3. Menentukan rating kecocokan setiap alternatif pada setiap kriteria.
4. Menentukan bobot preferensi atau tingkatan kepentingan (W) setiap kriteria. W = {W1, W2, W3 Wj}
5. Membuat matriks keputusan berdasarkan kriteria (Cj), kemudian melakukan normalisasi matriks berdasarkan persamaan yang disesuaikan dengan jenis atribut (atribut *benefit* atau atribut *cost*) sehingga diperoleh matriks ternormalisasi r.

2.3 Metode Simple Additive Weighting (SAW)

Metode Simple Additive Weighting (SAW) dikenal dengan istilah metode penjumlahan terbobot. Konsep dasarnya adalah mencari penjumlahan terbobot dari rating inerja pada saat alternative di semua atribut. Metode SAW membutuhkan proses normalisasi matriks keputusan (X) ke suatu skala yang dapat diperbandingkan dengan semua rating alternative yang ada (Raharja, 2020).

Formula untuk melakukan normalisasi tersebut adalah:

$$rij = \begin{cases} \dfrac{X_{ij}}{\underset{i}{Max}\, X_{ij}} \rightarrow \text{Jika j adalah } attribute \text{ keuntungan (benefit)} \\ \dfrac{\underset{i}{Min}\, X_{ij}}{X_{ij}} \rightarrow \text{Jika j adalah } attribute \text{ biaya } (cost) \end{cases} \quad (2.1)$$

Keterangan:

R_{ij} = Nilai rating kinerja ternormalisasi

X_{ij} = Nilai atribut yang dimiliki dari setiap kriteria

Max X_{ij} = Nilai terbesar dari setiap kriteria

Min X_{ij} = Nilai terkecil dari setiap kriteria

Benefit = Jika nilai terbesar adalah terbaik

Cost = Jika nilai terkecil adalah terbaik

Dimana R_{ij} adalah rating kinerja ternormalisasi dari alternatif A_i pada atribut C_j : i = 1, 2, …., m dan j = 1, 2, ….., n. Nilai Preferensi untuk setiap alternative (V_i) diberikan sebagai berikut:

$$V_i = \sum_{j=1}^{n} w_j r_{ij} \quad (2.2)$$

Keterangan:

V_i = Ranking untuk setiap alternatif

W_i = Nilai bobot dari setiap kriteria

R_{ij} = Nilai rating kinerja ternormalisasi

2.4 Framework Angular

Framework adalah fondasi yang dapat digunakan untuk menulis kode dengan menggunakan bahasa penulisan mereka sendiri dan didalam framework sudah terdapat kompiler yang bisa digunakan untuk membangun aplikasi melalui kode yang sudah dituliskan. Secara umum, framework menyediakan lingkungan yang memfasilitasi jenis pemrograman tertentu untuk proyek pengembangan software.(Andy, 2020)

Angular merupakan framework untuk mengembangan frontend yang digunakan untuk membangun aplikasi web modern dengan arsitektur *Single Page Applications* (SPA). Framework ini dikembangkan oleh perusahaan teknologi Google dan bersifat open source. Angular sendiri telah mengalami perkembangan dari versi pertamanya yaitu Angular 1 atau dikenal dengan AngularJS hingga Angular versi 9 yang merupakan versi terakhirnya. Angular versi 2 keatas merupakan pengembangan dari AngularJS yang benar-benar berbeda.

Perbedaannya pada penggunaan bahasa pemrogramannya dimana AngularJS menggunakan javascript sedangkan Angular menggunakan typescript. Typescript merupakan bahasa pemrograman berbasis javascript dengan menggunakan konsep *Object-Oriented Programming* (OOP) yang dikembangkan oleh tim Microsoft. Typescript hadir untuk menangani masalah kompleksitas pembuatan aplikasi berskala besar menggunakan javascript. Hal tersebut didukung oleh adanya *class*, *module*, dan *interface* yang menambah fleksibilitas dalam mengambangan aplikasi menjadi lebih sederhana.(Johan, 2020).

Arsitektur Framework Angular terdiri dari:

1. Angular *Component*

Component angular merupakan sebuah *block* pembangunan pada suatu projek angular yang memuat konten dari aplikasi/website. *Component* pada dasarnya adalah kelas yang berinteraksi dengan file html. Sebuah aplikasi angular umunya tersusun atas beberapa *component*. Sebuah *component* terdiri dari file html, css/stylesheet, dan typescript yang saling berhubungan. Logika pada sebuah *component* ditampung didalam file typescript yang menjadi kelas *component*.

2. Angular *Module*

Angular bersifat modular dan memiliki sistem modularitas ngModule. *Module* di angular berfungsi untuk mengendalikan *library* yang digunakan pada projek atau pada *component*.

3. *Template*

Angular memiliki dua hal penting yaitu *input* dan *output*. *Input* dari sebuah *template* berupa *property* dari komponen maupun fungsi yang memberikan nilai balikan sedangkan *output* berupa *string* yang dapat masuk ke nilai *attribute*, *property*, maupun html.

- Interpolasi

  Didefinisikan menggunakan tanda {{}} yang mengembalikan *string* pada tampilan.

  Contoh: {{title}}

- *Template Expression*

  Memiliki tanda yang sama dengan Interpolasi yaitu {{}} yang berguna

untuk memberikan nilai pada sebuah *property*.

Contoh: img src=”{{logo}}”

4. *Directive*

Directive berfungsi untuk menampilkan *template* berdasarkan nilai suatu ekspresi. Terdapat 3 *directive* pada angular:

- *Component Directive*

  Directive yang berkaitan dengan penentuan template html yang akan diproses dan digunakan.

  Contoh: user.component.ts.

- *Structural Directive*

  Directive ini berkaitan dengan manipulasi elemen html.

  Contoh: ngFor dan ngIf.

- *Attribute Directive*

  Directive ini berkaitan dengan mengubah penampilan pada elemen html.

5. *Data Binding*

Angular memungkinkan untuk menghubungkan logika pada kelas *component* dengan elemen html. Terdapat dua jenis *data binding*:

- *Event Binding*

  Digunakan untuk mengontrol suatu *event* terhadap suatu elemen html.

  Contoh: onClick($event)

- *Property Binding*

  Digunakan untuk menghubungkan data dari kelas *component* ke suatu elemen html.

Contoh: id=”{{myId}}”

6. *Metadata*

Metadata digunakan untuk mendekorasi kelas sehingga dapat mengkonfigurasi perilaku yang diharapkan dari suatu kelas tersebut.

7. Service dan *Dependency Injection*

Angular menyediakan kelas khusus bernama service yang dapat digunakan untuk mengolah data maupun logika yang tidak terkait dengan *component* tertentu dan dapat digunakan secara *global*.

8. *Routing*

Routing diangular merupakan bagian ngModule yang menyediakan layanan yang memfasilitasi penentuan navigasi antar *component* dengan Menyusun *route* berdasarkan *path* dan *component* yang berjalan pada *path* tersebut.

2.5 Firebase

Firebase merupakan salah satu layanan dari Google yang memudahkan para *app developer* dalam mengembangkan aplikasi mereka. Firebase termasuk kedalam kategori BAAS (*Backend As A Service*).(Guntoro, 2020).

Firebase memiliki fitur yang bisa kita gunakan untuk mengembangkan aplikasi Android, iOS, *Web*, dan lainnya. Saat menggunakan firebase untuk membuat aplikasi kita dapat menggunakan beberapa fitur yaitu:

1. *Authentication*

   Sebagian besar aplikasi ingin mengetahui identitas penggunanya sehingga nanti aplikasi dapat menyimpan data pengguna secara aman di*cloud* dan memberikan pengalaman *personal* yang sama disetiap perangkat pengguna.

2. *Hosting*

   Layanan *hosting* konten *web* yang berkelas produksi untuk parap pengembang aplikasi.

3. *Cloud Storage*

   Fitur ini dibuat untuk para pengembang aplikasi yang ingin menyimpan dan menampilkan konten buatan pengguna seperti *image* dan *video*.

4. *Realtime Database*

   Database yang di*host* di*cloud*. Data akan disimpan sebagain JSON kemudian disinkronkan secara *realtime* kesetiap *client* yang sudah terhubung.