BAB II

LANDASAN TEORI

1.1 Landasan/Kerangka Teori

1.1.1 Kebakaran

Menurut *International Labour Organization* (1991) kebakaran adalah suatu kejadian yang tidak diinginkan dan kadangkala tidak dapat dikendalikan, sebagai hasil pembakaran suatu bahan dalam udara dan mengeluarkan energi panas dan nyala api. Setiap kebakaran dapat menimbulkan berbagai macam kerugian seperti kerusakan alat produksi, bahan produksi dan kerugian waktu kerja selama proses produksi. Menurut *National Fire Protection Association* (NFA) kebakaran adalah suatu peristiwa oksidasi yang melibatkan tiga unsur yang harus ada, yaitu bahan bakar yang mudah terbakar, oksigen yang ada dalam udara dan sumber energi atau panas yang berakibat menimbulkan kerugian harta benda, cedera bahkan kematian.

1.1.2 Layanan

Layanan pada dasarnya dapat didefinisikan sebagai aktifitas seseorang, sekelompok atau organisasi baik langsung maupun tidak langsung untuk memenuhi kebutuhan. Menurut Tjiptono (Sunyoto, 2012:236), pelayanan adalah suatu penyajian produk atau jasa sesuai ukuran yang berlaku di tempat produk tersebut diadakan dan penyampaiannya setidaknya sama dengan yang diinginkan dan diharapkan oleh konsumen.

1.1.3 Formula *Haversine*

Formula *Haversine* merupakan sebuah algoritma yang diterapkan secara matematis untuk digunakan dalam navigasi. Sistem navigasi membutuhkan jarak terpendek antara dua titik untuk keperluan sistemnya. Hal tersebut dapat dihasilkan oleh formula *Haversine*. Formula *Haversine* melakukan perhitungan jarak lingkaran besar antara dua titik di permukaan bumi yang didasarkan pada garis bujur dan garis lintang timur. Berikut ini adalah rumus perhitungan formula *Haversine*:

$$\Delta\varphi = \frac{\pi}{180} * (\varphi 1 - \varphi 2)$$

$$\Delta\lambda = \frac{\pi}{180} * (\lambda 1 - \lambda 2)$$

$$a = \sin^2(\Delta\varphi/2) + \cos\varphi 1 \cdot \cos\varphi 2 \cdot \sin^2(\Delta\lambda/2)$$

$$c = 2 \cdot \operatorname{atan2}(\sqrt{a}, \sqrt{1} - a)$$

$$d = R \cdot c$$

Dimana:

φ = *latitude*

λ = *longitude*

c = jarak angular (radian)

d = jarak (km)

R = jari-jari bumi (6.371 km)

Penggunaan formula *Haversine* mengabaikan ketinggian bukit, kedalaman lembah dan bentuk bumi yang tidak sepenuhnya bulat sempurna. Jadi formula ini mengasumsikan bahwa bumi memiliki dataran yang rata dan berbentuk bulat sempurna (Yulianto, Ramadiani, & Kridalaksana 2018).

1.1.4 Algoritma Dijkstra

Algoritma *Dijkstra* merupakan salah satu algoritma yang efektif dalam memberikan lintasan terpendek dari suatu lokasi ke lokasi lainnya. Prinsip dari algoritma *Dijkstra* adalah mencari titik lokasi dengan pencarian dua lintasan yang paling pendek. Pada setiap iterasi, jarak titik yang diketahui (dari titik awal) diperbarui bila terdapat titik baru yang memberikan jarak terpendek. Syarat algoritma ini adalah bobot sisinya yang harus non-negatif (Satyananda, 2012:46).

Algoritma *Dijkstra* ditemukan oleh *Edsger Wybe Dijkstra* pada tahun 1959. Algoritma *Dijkstra* juga dikenal sebagai teknik untuk mengatasi masalah lintasan terpendek yang cukup efisien (Yao, 2016). Algoritma *Dijkstra* termasuk dalam algoritma prinsip *Greedy*. Prinsip *Greedy* artinya memecahkan suatu persoalan dalam tahap pertahap. Prinsip *Greedy* pada algoritma ini adalah memilih sebuah sisi yang berbobot terkecil dan memasukkannya kedalam himpunan solusi (Munir, 2010:413).

Graf G merupakan graf berlabel yang memuat antar titik dan lintasan terpendek yang dicari adalah v_1 ke v_n. Algoritma *Dijkstra* diawali titik v_1, dimana dalam setiap iterasinya akan mencari satu titik dengan jumlah bobot nilai terkecil dari titik awal. Titik-titik itu disebut tetap dan tidak diperhatikan lagi dalam iterasi-iterasi berikutnya.

Diberikan:

V (G) = $\{v_1, v_2, v_3, ..., v_n\}$

L = Himpunan node V(G) yang sudah terpilih dalam jalur terpendek

$D(v_j)$ = Jumlah bobot jarak terkecil dari v_i ke v_j

$W(v_i, v_j)$ = Bobot garis dari *node* v_i ke *node* v_j

Secara formal, algoritma *Dijkstra* untuk mencari rute terpendek adalah sebagai berikut:

1. L = { };
2. V = $\{v_2, v_3, ..., v_n\}$
3. Untuk i = 2, 3, ... , n lakukan $D(v_i) = W(v_j, v_i)$
4. Apabila $v_n \notin L$ (v_n belum menjadikan titik tetap, maka)
    a. Pilih node $v_k \in V - L$ (bukan node permanen) dimana D (v_k) terkecil, lalu L= L U $\{v_k\}$ (jadikan v_k sebagai node permanen).
    b. Untuk setiap $v_j \in V$ - L maka terapkan:
    Jika $D(v_j) > W(v_k, v_j)$ maka ubah $D(v_j)$ dengan $D(v_k) + W(v_k, v_j)$

Rumus penerapan:

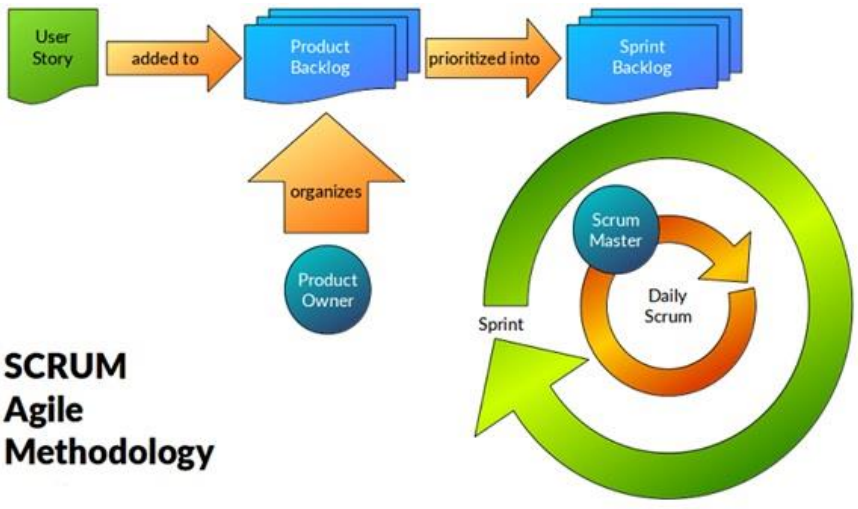
$$D_{(V_j)} = (D_{(V_j)}, D_{(k)} + W_{(V_i,V_j)})$$

Berdasarkan algoritma tersebut, rute terpendek dari node v_1 ke v_n terjadi dengan melewati node-node dalam L secara terurut dan dengan jumlah paling kecilnya yaitu $D(vn)$.

1.1.5 Metode Scrum

Menurut Schwaber dan Sutherland scrum adalah sebuah kerangka kerja yang dapat mengatasi suatu masalah komplek yang selalu berubah dan juga dinilai dapat memberikan kualitas produk yang baik sesuai dengan keinginan pengguna secara kreatif dan produktif. Scrum dikembangkan oleh Jeff Sutherland pada tahun 1993 dengan tujuan untuk menjadi metodologi pengembangan dan manajemen yang mengikuti prinsip-prinsip metodologi Agile. Sprint merupakan inti dari metode scrum yang merupakan batasan waktu yang dalam 1 bulan atau kurang dimana sebuah ikremen yang selesai, berfungsi dan berpotensi untuk dikembangkan. Proses sprint biasanya memiliki durasi waktu yang konsisten. Jika proses sprint tahap pertama sudah selesai maka dilanjutkan dengan proses sprint selanjutnya (Lesmana, I Putu Dody, 2019).

Kerangka scrum didasarkan pada seperangkat nilai-nilai, prinsip-prinsip dan praktik yang memberikan dasar yang terorganisasi yang akan menambahkan pelaksanaan untuk mewujudkan praktik scrum. Prinsip yang digunakan scrum dalam mengembangkan perangkat lunak menggabungkan kerangka kegiatan mulai dari *requirements, analysis, design, evolution* dan *delivery* (*Pressman*, 2015). Tahapan metode *Scrum* dapat dilihat pada Gambar 2.1



Gambar 2.1 Metode *Scrum*

1.1.6 Tahapan Metode Scrum

Berikut ini adalah tahapan-tahapan pada metode scrum:

1. ***User Story***

   User story adalah deskripsi secara rinci tentang kebutuhan sistem dalam bentuk bahasa yang tepat dengan mudah dipahami oleh sudut pandang *end user*. *User story* digunakan sebagai acuan untuk membuat *product backlog*.

2. ***Product Backlog***

   Product Backlog adalah daftar urutan segala sesuatu yang dibutuhkan dalam sistem maupun produk. Isi dari *Product Backlog* adalah fitur yang akan diterapkan ke dalam sistem beserta estimasi waktu pengerjaannya. Dokumen ini selalu berubah-ubah secara berkala seiring dengan perkembangan produk agar menghasilkan produk yang layak.

3. ***Sprint***

   Sprint adalah suatu siklus waktu dengan durasi maksimal satu bulan atau kurang. Durasi pada *sprint* sepanjang pengembangan produk tidak berubah. Tujuan dari *sprint* adalah untuk menyelesaikan sesuatu (*Sprint Goal).*

4. ***Daily Scrum***

   Tahap scrum ini bisa dikatakan sebagai evaluasi, karena para anggota tim menyampaikan update pekerjaan harian masing-masing. Berbagai kendala pun bisa didiskusikan disini. Proses *daily scrum* ini dijalankan setiap hari selama *sprint* berlangsung.

5. ***Sprint Review***

   Dalam tahapan ini, setiap anggota tim mendemonstrasikan yang sudah diselesaikan dalam periode satu *sprint*. Dengan kata lain, *sprint review* ini dilakukan setiap satu *sprint* selesai.

1.2 Studi Literature

Studi literature yang terkait dengan penelitian ini antara lain adalah sebagai berikut:

Studi literatur dari Mohamad Rizki Alif Ramdhani pada tahun 2019 yang berjudul "Pencarian Rute Terdekat Pemadam Kebakaran Menuju Lokasi Kebakaran Dengan Formula Haversine Dan Algoritma A*" membahas tentang penggunaan formula *Haversine* dan algortitma A* untuk mencari pos pemadam terdekat dan hidran terdekat serta melakukan optimasi rute pemadam kebakaran terdekat. Dalam studi literature ini, peneliti menunjukkan bahwa formula *Haversine* berhasil menghitung jarak antara lokasi kebakaran dan semua pos pemadam kebakaran yang ada di wilayah Jakarta Timur. Formula *Haversine* juga berhasil mencari letak hidran yang berada diradius satu km dari lokasi kebakaran. Rute terdekat dari pos pemadam kebakaran menuju lokasi kebakaran berhasil ditemukan menggunakan algoritma A*. Hasil perhitungan formula *Haversine* dapat dijadikan tolak ukur untuk menentukan 3 pos pemadam kebakaran terdekat mana yang akan dicari rute terdekatnya.

Studi literatur dari Roddy Yoto Sumary, Paulus Harsadi, Didik Nugroho pada tahun 2020 yang berjudul "Implementasi Algoritma Dijkstra Dan Metode Haversine Pada Penentuan Jalur Terpendek Pendakian Gunung Merapi Jalur Selo Berbasis Android" membahas tentang penggunaan algoritma *Dijkstra* dan Haversine untuk menentukan jalur terpendek pendakian gunung merapi jalur selo. Pada penelitian ini, Haversine digunakan untuk menghitung jarak geografis antara titik simpul di permukaan bumi menggunakan garis lintang (longitude) dan garis bujur (latitude) serta algoritma *Dijkstra* digunakan untuk mencari rute terdekat pendakian. Hasil dari studi literature ini menunjukkan bahwa algoritma *Dijkstra* dan metode Haversine dapat diintegrasikan pada aplikasi berbasis Android untuk membantu pendaki dalam menentukan jalur terpendek saat melakukan pendakian Gunung Merapi Jalur Selo.

Studi literatur dari Nanang Nggufron dan Rochmad Mashuri pada tahun 2019 dengan judul "Pencarian Rute Terbaik Pemadam Kebakaran Kota Semarang Menggunakan Algoritma Dijkstra Dengan Logika Fuzzy Sebagai Penentu Bobot Pada Graf" membahas tentang penggunaan algoritma *Dijkstra* dengan logika *fuzzy* sebagai penentu bobot pada graf untuk mencari rute terbaik pemadam kebakaran di Kota Bandar Lampung. Pada penelitian ini, logika *fuzzy* digunakan untuk

memodelkan kualitas dari input. Logika ini digunakan untuk situasi model dimana pembuatan keputusan dalam lingkup yang kompleks dan sulit untuk melakukan pengembangan model matematis. Sementara algoritma *Dijkstra* digunakan untuk pencarian rute terpendek setelah didapatkan bobot untuk masing-masing ruas jalan. Hasil dari penelitian ini menunjukkan bahwa pengunaan algoritma *Dijkstra* dengan logika *fuzzy* sebagai penentu bobot pada graf dapat meningkatkan efisiensi dalam pencarian rute terdekat pemadam kebakaran di Kota Semarang. Peneliti dalam jurnal ini mengembangkan sistem yang mempertimbangkan beberapa faktor penting, seperti jarak antar pos pemadam kebakaran, kepadatan lalu lintas, dan tingkat kepadatan penduduk pada setiap wilayah. Selain itu, sistem yang dikembangkan juga dapat memberikan alternatif rute terbaik ketika terjadi kemacetan atau kondisi darurat lainnya.

Studi literatur dari Unzilathun Hawa dan Lilik Muzdalifah pada tahun 2023 dengan judul "Solusi Pencarian Rute Terpendek Pemadam Kebakaran Menggunakan Algoritma *Dijkstra*" membahas tentang penerapan algoritma *Dijkstra* pada pencarian rute terpendek untuk pemadam kebakaran. Studi literatur ini mengacu pada beberapa sumber terkait, seperti jurnal yang membahas tentang penerapan algoritma *Dijkstra* pada optimasi rute di daerah perkotaan dan jurnal yang membahas tentang aplikasi algoritma *Dijkstra* pada pemadam kebakaran di kota-kota besar. Hasil dari penelitian ini menunjukkan bahwa algoritma *Dijkstra* dapat diaplikasikan pada pencarian rute terpendek pemadam kebakaran untuk meningkatkan efisiensi dan efektivitas dalam menangani kebakaran. Algoritma *Dijkstra* dapat membantu pemadam kebakaran dalam menentukan rute terpendek yang mempertimbangkan beberapa faktor penting, seperti jarak, waktu tempuh, dan kondisi lalu lintas.

Studi literatur dari Silvia Kartika, Suendri, dan Raissa Amanda Putri pada tahun 2021 dengan judul "Sistem Pencarian Lokasi Dan Rute Terdekat Menggunakan Metode Haversine Formula Pada Aplikasi Donatur Pakaian Berbasis Android" membahas tentang penerapan metode Haversine formula pada pencarian lokasi dan rute terdekat pada aplikasi donatur pekaian berbasis android. Metode Haversine formula mempertimbangkan faktor jarak antar lokasi dengan

menggunakan koordinat geografis untuk menghitung jarak antar titik pada permukaan bumi. Hasil dari penelitian ini menunjukkan bahwa penerapan metode Haversine formula pada aplikasi berbasis lokasi dapat meningkatkan akurasi dan efisiensi dalam pencarian lokasi dan rute terdekat. Selain itu, teknologi android juga memungkinkan aplikasi untuk mengakses data lokasi secara real-time, sehingga pengguna dapat menemukan lokasi dan rute terdekat dengan cepat dan mudah.

Studi literatur dari Ilham Ahbad Syahbana pada tahun 2022 dengan judul "Implementasi Algoritma *Dijkstra* Dalam Pencarian Lintasan Terpendek Dari Kantor Koperasi Darul Mafatih Ulum Menuju Nasabah" membahas tentang langkah-langkah penerapan perhitungan algoritma *Dijkstra* dalam menentukan lintasan terpendek untuk melakukan kunjungan kepada nasabah KSPPS DMU Cabang Malang. Selain itu, peneliti juga melakukan verifikasi data dengan cara melakukan simulasi rute yang telah dihasilkan untuk memastikan keakuratannya. Hasil dari penelitian ini yaitu melakukan kunjungan kepada nasabah KSSPPS CMD Cabang Malang membutuhkan tia hari. Total jarak yang ditempuh selama tiga hari dari awal menuju titik terjauh adalah 19,8 km + 18,9 km + 9,3 km = 48 km. jika setiap harinya melakukan perjalanan pulan pergi dengan asumsi jalan yang dilalui untuk pulang sama dengan jalan pergi maka total jaraknya adalah 28 * 2 = 96 km.

Studi literatur dari Ganet Abyan Habib Nursyam pada tahun 2019 dengan judul "Implementasi Algoritma Dijkstra Untuk Mencari Rute Terpendek Mobil Pemadam Kebakaran Wilayah Kota Yogyakarta" membahas tentang penerapan algoritma *Dijkstra* dalam menentukan rute terpendek mobil pemadam kebakaran menuju tiga tempat berpotensi terjadinya kebakaran di wilayah Kota Yogyakarta dan merepresentasikan jalan raya yang terpilih di wilayah Kota Yogyakarta dalam bentuk graf. Hasil dari penelitian ini adalah algoritma *Dijkstra* berhasil digunakan untuk merepresentasikan jalan yang terpilih di kota Yogyakarta dalam bentuk graf dimulai dengan merepresentasikan lokasi-lokasi dan beberapa persimpangan jalan sebagai titik-titik, beberapa ruas jalan sebagai sisi, mencari panjang sisi-sisi tersebut, kemudian saling dihubungkan dan diberi keterangan sehingga membentuk graf. Selain itu, algoritma *Dijkstra* juga dapat meningkatkan efisiensi dalam mencari rute terdekat.

Tabel 2.1 Studi Literatur

Judul, Penulis, Tahun	Jumlah & Atribut	Topik	Lokasi	Algoritma	Data source	Akurasi
Pencarian Rute Terdekat Pemadam Kebakaran Menuju Lokasi Kebakaran Dengan Formula Haversine Dan Algoritma A* (Mohamad Rizki Alif Ramdhani, 2019).	Ada 5 attribut: Koordinat, rute, jarak, lokasi, nama pos.	Membahas tentang penggunaan formula *Havserine* dan algoritma A* untuk mencari pos pemadam terdekat dan hidran terdekat.	Jakarta Timur	*Haversine* dan A*	Website & Google Maps	88,9 %
Implementasi Algoritma Dijkstra Dan Metode Haversine Pada Penentuan Jalur Terpendek Pendakian Gunung Merapi Jalur Selo Berbasis Android (Roddy Yoto Sumaryo, Paulus Harsadi, Didik Nugroho, 2020).	Ada 4 attribut: Jarak, Lokasi, Koordinat, Rute	Membahas tentang penggunaan algoritma *Dijkstra* dan formula *Havserine* untuk menentukan jalur terpendek pendakian gunung Merapi jalur selo.	Yogyakarta	Dijkstra dan Haversine	Website & Google Maps	77,8%
Pencarian Rute Terbaik Pemadam Kebakaran Kota Semarang Menggunakan Algoritma Dijkstra Dengan Logika Fuzzy Sebagai Penentu Bobot	Ada 4 attribut: Graf, Rute, Lokasi, Jarak	Membahas tentang penggunaan algoritma *Dijkstra* dan logika *fuzzy* sebagai penentu bobot pada graf untuk mencari rute	Kota Bandar Lampung	Dijkstra	Website & Google Maps	80%

Pada Graf (Nanang Nggufron, Rochmad, Mashuri, 2019).		terbaik pemadam kebakaran di Kota Bandar Lampung.				
Solusi Pencarian Rute Terpendek Pemadam Kebakaran Menggunakan Algoritma Dijkstra (Unzilathun Hawa, Lilik Muzdalifah, 2023).	Ada 4 attribut: Rute, Jarak, Nama Pos, Daerah Kebakaran	Membahas tentang penerapan algoritma *Dijkstra* pada pencarian rute terpendek untuk pemadam kebakaran.	Kabupaten Tuban, Jawa Timur	Dijkstra	Website & Google Maps	85%
Sistem Pencarian Lokasi Dan Rute Terdekat Menggunakan Metode Haversine Formula Pada Aplikasi Donatur Pakaian Berbasis Android (Silvia Kartika, Suendri, Raissa Amanda Putri, 2021).	Ada 6 attribut: Koordinat, Jarak, Lokasi, Tanggal, Keterangan, Alamat	Membahas tentang penerapan metode *Haversine* formula pada pencarian lokasi dan rute terdekat pada aplikasi donatur pakaian berbasis android.	Kota Medan	Haversine	Website & Google Maps	-
Implementasi Algoritma *Dijkstra* Dalam Pencarian Lintasan Terpendek Dari Kantor Koperasi Darul Mafatih Ulum Menuju Nasabah (Ilham Ahbab	Ada 5 attribut: Titik lokasi, Alamat, Rute, Lintasan, Koordinat	Membahas tentang langkah-langkah penerapan algoritma *Dijkstra* dalam menentukan lintasan terpendek untuk melakukan	Kota Malang	Dijkstra	Website Geografis & Google Maps	-

Syahbana, 2022)		kunjungan kepada nasabah KSPPS DMU cabang Malang.				
Implementasi Algoritma Dijkstra Untuk Mencari Rute Terpendek Mobil Pemadam Kebakaran Wilayah Kota Yogyakarta (Ganet Abyan Habib Nursyam, 2019)	Ada 5 attribut: Titik lokasi, Alamat, Rute, Lintasan, Koordinat	Membahas tentang penerapan algoritma *Dijkstra* dalam menentukan rute terpendek mobil pemadam kebakaran menuju tiga tempat berpotensi terjadinya kebakaran dan merepresentasikan jalan raya yang terpilih di wilayah Kota Yogyakarta dalam bentuk graf.	Yogyakarta	Dijkstra	Website Geografis & Google Maps	82%
Lampung Smart Service Pelayanan Kebakaran Menggunakan Algoritma Di Kota Bandar Lampung Berbasis Android	Ada 8 attribut : Titik koordinat, Lokasi, Alamat, Rute, Waktu, Graf, Path Dijkstra, Deskripsi	Membahas tentang penerapan *Haversine* formula dalam mencari lokasi pemadam kebakaran terdekat dan penerapan algoritma *Dijkstra*	Kota Bandar Lampung	Haversine Formula & Dijkstra	Website Geografis & Google Maps API	?

		dalam mencari rute terpendek dari pos pemadam menuju lokasi kebakaran.				