Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 3, **Number** 2, November 2023
https://doi.org/10.47709/brilliance.v3i2.3089

Optimization of the Shortest Tsunami Evacuation Route Using Djikstra's Algorithm in Benoa Village

Ida Bagus Kade Puja Arimbawa K[1*], Wayan Sukartiasih[2], Agung Sedayu[3]
[1*,3]Fakultas Teknologi dan Ilmu Kesehatan, Program Studi Sistem Informasi, Universitas Bali Dwipa, Indonesia
[2]Fakultas Humaniora dan Ilmu Sosial, Program Studi Psikologi, Universitas Bali Dwipa, Indonesia
[1*]pj@balidwipa.ac.id, [2]devidevidasi@yahoo.com, [3]day.gplayus@gmail.com

open access

***Corresponding Author**

Article History:
Submitted: 02-11-2023
Accepted: 03-11-2023
Published: 10-11-2023

Keywords:
Djikstra Algorithm, Shortest Path, Tsunami Evacuation

ABSTRACT

Benoa Village has an area of approximately 2.38 km² and a population of 9,569 people in 2020 with a population density of 4,013 people/km^2. This area is included in the list of tsunami-prone areas because the area is located on the edge of the Indian Ocean, which is known as an area with a high level of earthquake and volcanic activity. The 2004 tsunami that hit the coast of the Indian Ocean increased the potential for similar disasters to occur in the area. Determination of the shortest evacuation route in Benoa Village using Djikstra Algorithm. The result obtained is a path from the evacuation starting point vertex to the comfort zone node. Thisvertex represents places and road intersections arranged in the form of a weighted graph (distance) with a total of 51 vertexs, and an Adjacency Matrix is formed which is processed using the C++ Program. The Safe Zone vertex (Grand Hyatt Bali Temporary Meet Point (V50), Hattrick Futsal (V51)) are headed from the evacuation starting point of Serangan Beach (4.49km to V50, 6.94km to V51), Noanui Beach (3.95km to V50, 6.39km to V51), Samuh Beach (2.75km to V50, 5, 19km to V51), Nusa Dua Beach A (3.54km to V50, 5.48km to V51), Nusa Dua Beach B (3.47km to V50, 4.79km to V51), Peninsula Island (4.69km to V50, 4.80km to V51), Megiat Beach (4.81km to V50, 4.15km to V51) and Geger Beach (5.86km to V50, 5.09km to V51).

INTRODUCTION

Tsunami is a natural phenomenon that occurs at sea characterized by very large and destructive sea waves. Tsunamis are formed by sudden changes in the ocean floor such as earthquakes, volcanic eruptions, landslides or even as a result of meteorite impacts. Tsunami waves can reach heights of more than 30 meters and travel at very high speeds, reaching up to 800 kilometers per hour. Tsunamis can cause enormous damage to coastal areas, including flooding, damage to buildings and infrastructure and potentially loss of human and animal life(Badan Meteorologi, 2023).

Benoa Village is a village located in South Kuta District, Badung Regency, Bali, Indonesia. It has an area of approximately 2.38 km² and a population of 9,569 people in 2020 with a population density of 4,013 people/km^2(Badan Pusat Statistik Kabupaten Badung, 2023). This area is included in the list of tsunami-prone areas because the area is on the edge of the Indian Ocean, which is known as an area with a high level of earthquake and volcanic activity (Badan Meteorologi, 2023). The 2004 tsunami that hit the coast of the Indian Ocean increased the potential for similar disasters to occur in the area. Therefore, Benoa Village requires planning in the face of a tsunami disaster to anticipate casualties where the area is a densely populated area as well as a tourist destination that is visited by many foreign and domestic tourists.

The study of tsunami evacuation route selection in tsunami prone areas has been the focus of considerable research by a number of previous researchers. Two of them are the evacuation route selection study in Sanur, Bali (Sani et al., 2013) and a similar study in Padang City (Anggria et al., 2017).

The selection of the best evacuation route can be structured in an optimization study, which is a process to achieve the best result. There are various methods used to solve this problem, one of which is through the application of evolutionary algorithms, specifically genetic algorithms (Iryanto & Ismantohadi, 2017). However, one of the commonly used methods in shortest path selection optimization is using Dijkstra's algorithm.

The use of Dijkstra's algorithm in research on finding evacuation routes or paths in disaster situations has received significant attention in recent years. For example, Pramudya in 2015 applied this algorithm in planning evacuation routes to deal with landslides in Semarang City (Pramudya & Subiyanto, 2015).

E-ISSN : 2807-9035
Volume 3, **Number** 2, November 2023
https://doi.org/10.47709/brilliance.v3i2.3089

From several references that have been described, the utilization of Dijkstra's algorithm has become a commonly used method in finding evacuation routes or paths. Therefore, in this research, the author chose Dijkstra's algorithm as the method used to find the best evacuation route in the face of a potential tsunami in Benoa Village, Bali.

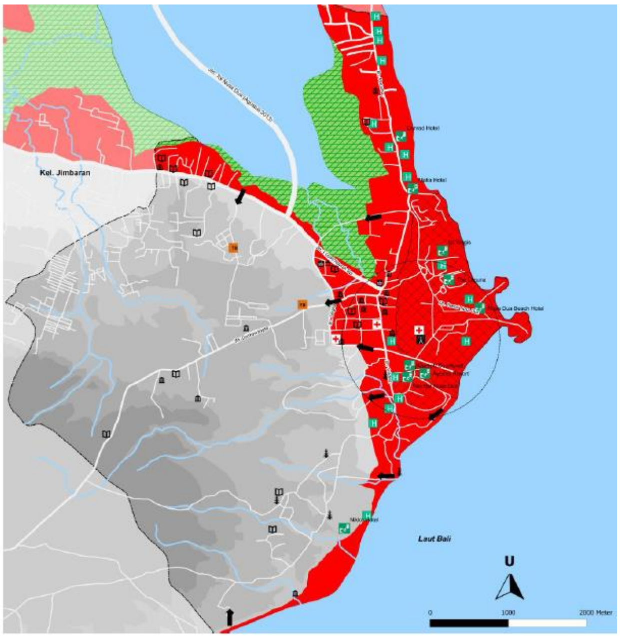

Figure 1. Tsunami Evacuation Map in Benoa Village(Badan Meteorologi, 2023)

LITERATURE REVIEW

The objective of the research conducted by (Darmi & Muntahanah, 2022) is to apply the Dijkstra algorithm in order to ascertain the most efficient evacuation route in the Bengkulu region in the event of an earthquake or tsunami. The development of an application for this route information system would further enhance the effectiveness of the evacuation process. Life-threatening situations will inevitably escalate as the duration of the evacuation process or the evacuation time required increases. A weighted graph is constructed in this investigation (Afandi & Mayasari, 2021) by conceptualizing the node as a road intersection, the edge as the road that connects the nodes, and the weight as a fuzzy output derived from the road's length and breadth. In addition, the evacuation route is determined through the utilization of the Dijkstra Algorithm to compute the minimum number of weights from each intersection to the cluster point. An evacuation route is formulated that takes into account both the breadth and length of the roads. An additional study (Putra & Rohendi, 2017) developed a prototype geographic information system to utilize the DJIKstra algorithm to verify evacuation routes and furnish recommendation information. The system aims to facilitate collective action during emergency situations by providing location and route maps that are accessible at all times and from any location. By doing so, individuals can approximate travel times and promptly evacuate individuals to safer areas. A separate study (Zhu et al., 2022) elucidated that this investigation optimized evacuation route planning under varying flood disaster levels, in light of the scarcity of prior research on the submersion of cities and towns caused by reservoir flooding, the neglect of limitations on pedestrian speed and road factors in prior evacuation route studies, and the inability to effectively combine the path selection problems in the inundated range under different flood disaster levels. Further study (Liu et al., 2023) In times of emergency, selecting the most efficient evacuation route is critical for minimizing evacuation duration. Several algorithms are utilized to optimize evacuation routes, including the network flow algorithm, the ant colony algorithm, and the Dijkstra algorithm.

METHOD

Research Stages

The stages of research are a description of the flow of research carried out from beginning to end. The first stage is to identify problems and literature studies by looking for references related to the research. The next stage is mapping the area and identifying evacuation routes based on data from survey results and marking the location of strategic evacuation posts. Direct distance data between each point on the evacuation route combined with GPS devices and collecting travel time data using a stopwatch. Researchers measured the distance and travel time from each point of the evacuation route to the nearest post. Researchers processed the data that had been obtained and transformed the map

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 3, **Number** 2, November 2023
https://doi.org/10.47709/brilliance.v3i2.3089

into a weighted graph. Researchers then applied the Djikstra's algorithm to find the shortest evacuation route from each point on the evacuation route to the nearest post. At this stage, the researchers utilized C++ in applying the Djikstra's algorithm. Next, the researchers tested and validated the results obtained from the application of the Djikstra's algorithm and finally drew conclusions.

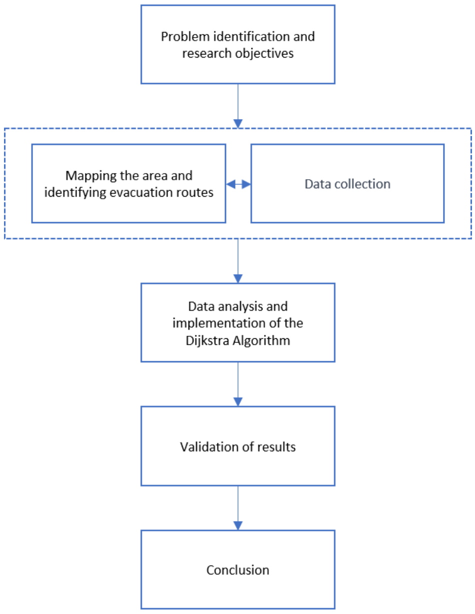

Figure 2. Research Stages

Djikstra's Algorithm

The algorithm is named after its inventor, Edsger Dijkstra, a Dutch scientist. It was developed in 1956 and first publicized in 1959(Dijkstra, 2022). Dijkstra's algorithm is used to solve the problem of finding the shortest path or trajectory, which is the problem of finding the path or trajectory with the smallest total weight, such as distance, between two vertexs in a weighted graph. This algorithm looks for the shortest distance between the start vertex and other vertexs. Therefore, the path formed from the beginning of vertex to the destination vertex has the smallest number of weights.

Here are the steps of Dijkstra's algorithm according to (Akram et al., 2021; Deng et al., 2012; Luo et al., 2020) :

1. **Initialization:** Starting from the start vertex, assign a distance value of zero, and assign infinite values (or large values) to all other vertexs as initial distances. Also specify the start vertex as the current active vertex.
2. **Active Vertex Selection**: Select the vertex with the shortest distance from the beginning of vertex as the current active vertex. Starting from the thebeginning of vertex. This is the vertex with zero distance in the first step.
3. **Update Distance:** For all neighbors of the current active vertex, check if the route through the current active vertex is shorter than the current distance to that neighbor. If yes, update the distance to that neighbor with the new shorter distance.
4. **Mark Active Vertex**: After updating the distances to all neighbors, mark the current active vertex as "visited" or "resolved."
5. **Select New Vertex**: Select the vertex with the shortest distance that has not been "visited" as the current active vertex. It is the node with the shortest distance among all the unvisited vertex.
6. **Repeat**: Repeat steps 3-5 until all vertex have been visited or if the destination vertex has been reached.
7. **Result:** Once all vertex have been visited or the destination vertex has been reached, you have the shortest path from starting vertex to the destination vertex and the total distance.

Adjacency Matrix

This matrix expresses the neighborliness of vertex in the graph. Suppose $G = (V, E)$ is a graph with n vertex, $n \geq 1$. Neighbor matrix G is a rectilinear matrix $n \times n$. If the matrix is called $A = [aij]$, then

$$a_{ij} \begin{cases} 1, & i,j\ neighbor \\ 0, & i,j\ neighbor \end{cases} \qquad (1)$$

E-ISSN : 2807-9035
Volume 3, **Number** 2, November 2023
https://doi.org/10.47709/brilliance.v3i2.3089

Because it only contains 0 and 1, then there are also those who refer to the adjacency matrix as a zero-one matrix. Other than with numbers 0 and 1, matrix elements can also be expressed with the values true (stating 1) and false (stating 0). For weighted graphs, a_{ij} denotes the weight of each edge connecting the vertex i and j.(Khaleel & Al-Shumam, 2020; Sporns, 2022)

RESULT

Data Analysis

In the data collection stage, based on the map in Figure 1, an area was mapped along the coastline in Benoa Village, Badung to determine the startingvertex of the tsunami evacuation route, taking into account the population density and crowded places and to determine the safe zone vertex.

Table 1. Evacuation Starting Points

Vertex	Evacuation Starting Points	Address/Coordinates
V1	Serangan Beach	Jl. Pratama No. 72, Benoa, Badung
V2	Noanui Beach	Jl. Pratama No. 70, Benoa, Badung
V3	Samuh Beach	-8.787117, 115.228341
V4	Nusa Dua Beach (A)	-8.795758, 115.232644
V5	Nusa Dua Beach (B)	-8.800429, 115.235919
V6	Peninsula Island	-8.802739, 115.238965
V7	Mengiat Beach	-8.808445, 115.231053
V8	Geger Beach	-8.815786, 115.226178

Table 2. Safe Zone Points

Vertex	Safe Zone Points	Address/Coordinates
V50	Grand Hyatt Bali Temporary Meet Point	Jl. Bypass Ngurah Rai, Nusa Dua
V51	Hattrick Futsal	Jl. Mayapada No. 888, Benoa

Next, determine the points (road intersections) as vertices in the graph and measure the length of the road as the edge weight according to the Benoa Village map.

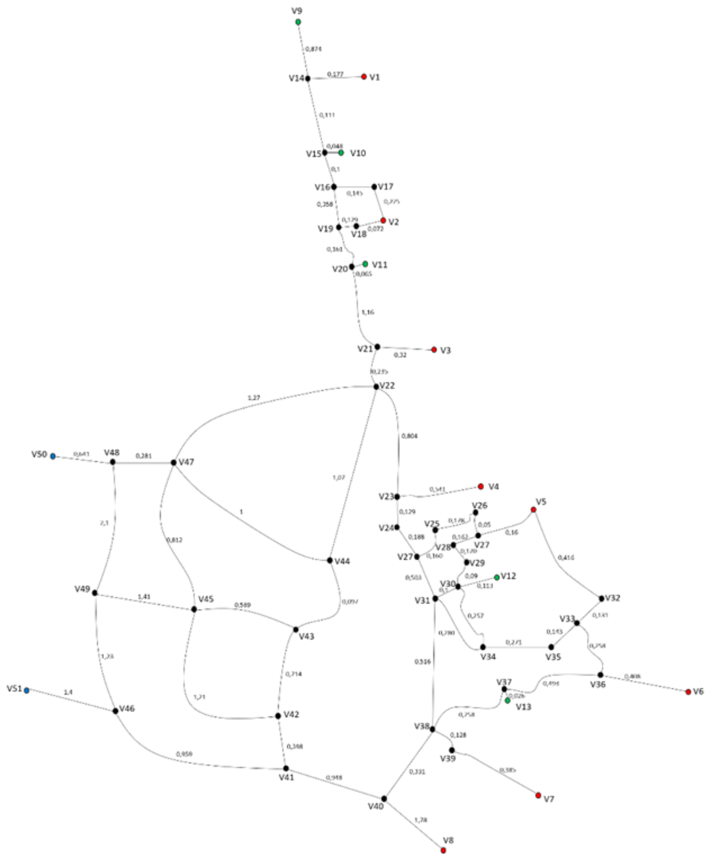

Figure 3. Point (Road Intersections)

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 3, **Number** 2, November 2023
https://doi.org/10.47709/brilliance.v3i2.3089

Implementation of Djikstra's Algorithm

The adjacency matrix is created based on Figure 3 By considering the neighboring vertex and the weight of the edge connecting the vertex. The number of vertex in the graph is 51.

	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12	V13	V14	V15	V16	V17	V18	V19	V20	V21	V22	V23	V24	V25	V26	V27	V28	V29	V30	V31	V32	V33	V34	V35	V36	V37	V38	V39	V40	V41	V42	V43	V44	V45	V46	V47	V48	V49	V50	V51
V1	0	0	0	0	0	0	0	0	0	0	0	0	0	0.177	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.225	0.072	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V3	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.541	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V5	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.16	0	0	0	0	0.416	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.408	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V7	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.385	0	0	0	0	0	0	0	0	0	0	0	0
V8	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.78	0	0	0	0	0	0	0	0	0	0	0
V9	0	0	0	0	0	0	0	0	0	0	0	0	0	0.874	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.048	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V11	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.065	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.113	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V13	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.026	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V14	0.177	0	0	0	0	0	0	0	0.874	0	0	0	0	0	0.111	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V15	0	0	0	0	0	0	0	0	0	0.048	0	0	0	0.111	0	0.1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.1	0	0	0	0.358	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V17	0	0.225	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V18	0	0.072	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.129	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V19	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.358	0	0.129	0	0.161	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V20	0	0	0	0	0	0	0	0	0	0	0.065	0	0	0	0	0	0	0	0.161	0	1.16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V21	0	0	0.32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.16	0	0.235	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V22	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.235	0	0.804	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.07	0	0	1.27	0	0	0	0
V23	0	0	0	0.541	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.804	0	0.129	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V24	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.129	0	0	0	0.188	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V25	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.178	0.16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V26	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.178	0	0.05	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V27	0	0	0	0	0.16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.188	0.16	0.05	0	0.162	0	0	0.503	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V28	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.162	0	0.12	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V29	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.12	0	0.09	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V30	0	0	0	0	0	0	0	0	0	0	0	0.113	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.09	0	0.1	0	0	0.257	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V31	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.503	0	0	0.1	0	0	0	0.28	0	0	0	0.516	0	0	0	0	0	0	0	0	0	0	0	0	0
V32	0	0	0	0	0.416	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.131	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V33	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.131	0	0	0.143	0.258	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V34	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.257	0.28	0	0	0	0.271	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V35	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.143	0.271	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V36	0	0	0	0	0	0.408	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.258	0	0	0	0.493	0	0	0	0	0	0	0	0	0	0	0	0	0	0
V37	0	0	0	0	0	0	0	0	0	0	0	0	0.026	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.493	0	0.258	0	0	0	0	0	0	0	0	0	0	0	0	0
V38	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.516	0	0	0	0	0	0.258	0	0.128	0.331	0	0	0	0	0	0	0	0	0	0	0
V39	0	0	0	0	0	0	0.385	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.128	0	0	0	0	0	0	0	0	0	0	0	0	0
V40	0	0	0	0	0	0	0	1.78	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.331	0	0	0.948	0	0	0	0	0	0	0	0	0	0
V41	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.948	0	0.398	0	0	0	0.959	0	0	0	0	0
V42	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.398	0	0.714	0	1.21	0	0	0	0	0	0
V43	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.714	0	0.097	0.569	0	0	0	0	0	0
V44	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.07	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.097	0	0	0	1	0	0	0	0
V45	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.21	0.569	0	0	0	0.812	0	1.41	0	0
V46	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.959	0	0	0	0	0	0	0	1.23	0	1.4
V47	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.27	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0.812	0	0	0.281	0	0	0
V48	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.281	0	2.1	0.641	0
V49	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.41	1.23	0	2.1	0	0	0
V50	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0.641	0	0	0
V51	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1.4	0	0	0	0	0

Figure 4. Adjacency Matrix

The next step is to create a shortest path determination program in the C++ Programming Language. Here is a snippet of C++ Syntax Djikstra's Algorithm to determine the shortest path.

```
// Fungsi untuk melakukan algoritma Dijkstra pada graf
void dijkstra(vector<vector<double>>& graph, int start, int V) {
    double dist[V];
    bool sptSet[V];
    int parent[V];

    for (int i = 0; i < V; i++) {
        dist[i] = numeric_limits<double>::max();
        sptSet[i] = false;
        parent[i] = -1;
    }

    dist[start] = 0.0;

    for (int count = 0; count < V - 1; count++) {
        int u = cariJarakTerpendek(dist, sptSet, V);
        sptSet[u] = true;

        for (int v = 0; v < V; v++) {
            if (!sptSet[v] && graph[u][v] > 0.0 && dist[u] != numeric_limits<double>::max() && dist[u] + graph[u][v] < dist[v]) {
                parent[v] = u;
                dist[v] = dist[u] + graph[u][v];
            }
        }
    }
```

```
Fungsi untuk mencari simpul dengan jarak terpendek yang belum dikunjungi
t cariJarakTerpendek(double dist[], bool sptSet[], int V) {
  double minDist = numeric_limits<double>::max();
  int minIndex;

  for (int v = 0; v < V; v++) {
      if (!sptSet[v] && dist[v] <= minDist) {
          minDist = dist[v];
          minIndex = v;
      }
  }

  return minIndex;
```

Figure 5. Syntax

The implementation of Dijkstra's algorithm in C++ syntax above allows users to input the number of vertex, adjacency matrix, and starting vertex to find the shortest path in a weighted graph.

1. The program starts by asking the user to enter the number of vertex (V).
2. Then, the program prompts the user to input an adjacency matrix that represents the weights between vertex in the graph. Each element of the adjacency matrix is the weight of an edge that connects a vertex to another vertex.
3. The user is also asked to enter the starting vertex from which Dijkstra's algorithm will start.
4. Dijkstra's algorithm is executed by calling the function'**dijkstra()'.**This function calculates the shortest path from the start vertex to all other vertices in the graph.
5. The result of Dijkstra's algorithm is printed, including the shortest distance from the starting vertex to all other vertex and their paths.
6. The result is printed with a precision of two decimals using '**fixed'**and **'setprecision(2**)'.

Brilliance
Research of Artificial Intelligence

E-ISSN : 2807-9035
Volume 3, **Number** 2, November 2023
https://doi.org/10.47709/brilliance.v3i2.3089

The program will run Dijkstra's algorithm to find the shortest path from the starting vertex to all other vertex in the graph that has 51 vertexs based on the entered adjacency matrix. The result will be printed in a format that includes the shortest distance from the start vertex to every other vertex along with the paths. Vertex naming starts from $vertex\ 0, vertex\ 1, \ldots, vertex\ 50$.

Table 3. Customization of vertex names as evacuation starting points and Safe Zones in C++

Vertex	Name of Vertex in C++	Information
V1	0	Serangan Beach (Evacuation Starting Point)
V2	1	Noanui Beach (Evacuation Starting Point)
V3	2	Samuh Beach (Evacuation Starting Point)
V4	3	Nusa Dua Beach (A) (Evacuation Starting Point)
V5	4	Nusa Dua Beach (B) (Evacuation Starting Point)
V6	5	Peninsula Island(Evacuation Starting Point)
V7	6	Mengiat Beach (Evacuation Starting Point)
V8	7	Geger Beach (Evacuation Starting Point)
V50	49	Grand Hyatt Bali Temporary Meet Point (Safe Zone)
V51	50	Hattrick Futsal (Safe Zone)

The program output will include 51 lines (one line for each vertex other than the starting vertex), which will list the shortest distance from the start vertex to each vertex as well as its shortest path. The distances will be displayed with a precision of two decimals.

```
Masukkan simpul awal: 0
Simpul 0 ke simpul 1 memiliki jarak 0.95 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 17 -> 1
Simpul 0 ke simpul 2 memiliki jarak 2.39 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 2
Simpul 0 ke simpul 3 memiliki jarak 3.65 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 3
Simpul 0 ke simpul 4 memiliki jarak 3.58 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 4
Simpul 0 ke simpul 5 memiliki jarak 4.80 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 4 -> 31 -> 32 -> 35 -> 5
Simpul 0 ke simpul 6 memiliki jarak 4.92 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 30 -> 37 -> 38 -> 6
Simpul 0 ke simpul 7 memiliki jarak 6.52 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 30 -> 37 -> 39 -> 7
Simpul 0 ke simpul 8 memiliki jarak 1.05 dengan jalur: 0 -> 13 -> 8
Simpul 0 ke simpul 9 memiliki jarak 0.34 dengan jalur: 0 -> 13 -> 14 -> 9
Simpul 0 ke simpul 10 memiliki jarak 0.97 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 10
Simpul 0 ke simpul 11 memiliki jarak 3.91 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 11
Simpul 0 ke simpul 12 memiliki jarak 4.69 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 30 -> 37 -> 36 -> 12
Simpul 0 ke simpul 13 memiliki jarak 0.18 dengan jalur: 0 -> 13
Simpul 0 ke simpul 14 memiliki jarak 0.29 dengan jalur: 0 -> 13 -> 14
Simpul 0 ke simpul 15 memiliki jarak 0.39 dengan jalur: 0 -> 13 -> 14 -> 15
Simpul 0 ke simpul 16 memiliki jarak 1.17 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 17 -> 1 -> 16
Simpul 0 ke simpul 17 memiliki jarak 0.88 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 17
Simpul 0 ke simpul 18 memiliki jarak 0.75 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18
Simpul 0 ke simpul 19 memiliki jarak 0.91 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19
Simpul 0 ke simpul 20 memiliki jarak 2.07 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20
Simpul 0 ke simpul 21 memiliki jarak 2.30 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21
Simpul 0 ke simpul 22 memiliki jarak 3.11 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22
Simpul 0 ke simpul 23 memiliki jarak 3.23 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23
Simpul 0 ke simpul 24 memiliki jarak 3.58 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 24
Simpul 0 ke simpul 25 memiliki jarak 3.47 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 25
Simpul 0 ke simpul 26 memiliki jarak 3.42 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26
Simpul 0 ke simpul 27 memiliki jarak 3.58 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27
Simpul 0 ke simpul 28 memiliki jarak 3.71 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28
Simpul 0 ke simpul 29 memiliki jarak 3.79 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29
Simpul 0 ke simpul 30 memiliki jarak 3.90 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 30
Simpul 0 ke simpul 31 memiliki jarak 4.00 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 4 -> 31
Simpul 0 ke simpul 32 memiliki jarak 4.13 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 4 -> 31 -> 32
Simpul 0 ke simpul 33 memiliki jarak 4.05 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 33
Simpul 0 ke simpul 34 memiliki jarak 4.27 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 4 -> 31 -> 32 -> 34
Simpul 0 ke simpul 35 memiliki jarak 4.39 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 4 -> 31 -> 32 -> 35
Simpul 0 ke simpul 36 memiliki jarak 4.67 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 30 -> 37 -> 36
Simpul 0 ke simpul 37 memiliki jarak 4.41 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 30 -> 37
Simpul 0 ke simpul 38 memiliki jarak 4.54 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 30 -> 37 -> 38
Simpul 0 ke simpul 39 memiliki jarak 4.74 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 22 -> 23 -> 26 -> 27 -> 28 -> 29 -> 30 -> 37 -> 39
Simpul 0 ke simpul 40 memiliki jarak 4.58 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 43 -> 42 -> 41 -> 40
Simpul 0 ke simpul 41 memiliki jarak 4.18 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 43 -> 42 -> 41
Simpul 0 ke simpul 42 memiliki jarak 3.47 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 43 -> 42
Simpul 0 ke simpul 43 memiliki jarak 3.37 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 43
Simpul 0 ke simpul 44 memiliki jarak 4.04 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 43 -> 42 -> 44
Simpul 0 ke simpul 45 memiliki jarak 5.54 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 43 -> 42 -> 41 -> 40 -> 45
Simpul 0 ke simpul 46 memiliki jarak 3.57 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 46
Simpul 0 ke simpul 47 memiliki jarak 3.85 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 46 -> 47
Simpul 0 ke simpul 48 memiliki jarak 5.45 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 43 -> 42 -> 44 -> 48
Simpul 0 ke simpul 49 memiliki jarak 4.49 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 46 -> 47 -> 49
Simpul 0 ke simpul 50 memiliki jarak 6.94 dengan jalur: 0 -> 13 -> 14 -> 15 -> 18 -> 19 -> 20 -> 21 -> 43 -> 42 -> 41 -> 40 -> 45 -> 50
```

Figure 6. C++ Program Output for the starting vertex : 0

The same process is repeated as many times as the number of initial vertex as listed in Table 3. The results obtained are as follows.

Table 4. Evacuation Routes

Evacuation starting points	Safe Zones	Evacuation Routes	Distance (km)
Serangan Beach (V1 – 0)	Grand Hyatt Bali Temporary Meet Point (V50 – 49)	**V1** – V14 – V15 – V16 – V19 – V20 – V21 – V22 – V47 – V48 – **V50**	4,49
	Hattrick Futsal (V51 – 50)	**V1** – V14 – V15 – V16 – V19 – V20 – V21 – V22 – V44 – V43 – V42 – V41 – V45 – **V51**	6,94
Noanui Beach (V2 – 1)	Grand Hyatt Bali Temporary Meet Point (V50 – 49)	**V2** – V18 – V19 – V20 – V21 – V22 – V47 – V48 – **V50**	3,95
	Hattrick Futsal (V51 – 50)	**V2** – V18 – V19 – V20 – V21 – V22 – V44 – V43 – V42 – V41 – V46 – **V51**	6,39
Samuh Beach (V3 – 2)	Grand Hyatt Bali Temporary Meet Point (V50 – 49)	**V3 –** V21 – V22 – V47 – V48 – **V50**	2,75

E-ISSN : 2807-9035
Volume 3, **Number** 2, November 2023
https://doi.org/10.47709/brilliance.v3i2.3089

	Hattrick Futsal (V51 – 50)	**V3 –** V21 – V22 – V44 – V43 – V42 – V41 – V46 – **V51**	5,19
Nusa Dua Beach (A) (V4 – 3)	Grand Hyatt Bali Temporary Meet Point (V50 – 49)	**V4** - V23 – V22 – V47 – V48 – **V50**	3,54
	Hattrick Futsal (V51 – 50)	**V4** – V23 – V24 – V27 – V28 – V29 – V30 – V31 – V38 – V40 – V41 – V46 – **V51**	5,48
Nusa Dua Beach(B) (V5 – 4)	Grand Hyatt Bali Temporary Meet Point (V50 – 49)	**V5** – V27 – V24 – V23 – V22 – V47 – V48 – **V50**	3,47
	Hattrick Futsal (V51 – 50)	**V5** – V27 – V28 – V29 – V30 – V31 – V38 – V40 – V41 – V46 – **V51**	4,79
Peninsula Island (V6 – 5)	Grand Hyatt Bali Temporary Meet Point (V50 – 49)	**V6** – V36 – V33 – V32 – V5 – V27 – V24 – V23 – V22 – V47 – V48 – **V50**	4,69
	Hattrick Futsal (V51 – 50)	**V6** – V36 – V37 – V38 – V40 – V41 – V46 – **V51**	4,80
Mengiat Beach (V7 – 6)	Grand Hyatt Bali Temporary Meet Point (V50 – 49)	**V7** – V39 – V38 – V31 – V30 – V29 – V28 – V27 – V24 – V23 – V22 – V47 – V48 – **V50**	4,81
	Hattrick Futsal (V51 – 50)	**V7** – V39 – V38 – V40 – V41 – V46 – **V51**	4,15
Geger Beach (V8 – 7)	Grand Hyatt Bali Temporary Meet Point (V50 – 49)	**V8** – V40 – V41 – V42 – V43 – V44 – V47 – V48 – **V50**	5,86
	Hattrick Futsal (V51 – 50)	**V8** – V40 – V41 – V46 – **V51**	5,09

Based on Table 4, it is obtained the evacuation route with the shortest distance from the Evacuation Starting Point of Serangan Beach (V1), Noanui Beach (V2), Samuh Beach (V3), Nusa Dua Beach A (V4), Nusa Dua Beach B (V5), Peninsula Island (V6), Mengiat Beach (V7), and Geger Beach (V8) to the Grand Hyatt Bali Temporary Safe Zone (V50) and Hattrick Futsal (V51). However, there are some Evacuation Starting Points that are close to the green vertex in Figure 2, which can evacuate to the 3rd floor of hotel buildings with large capacity. Several hotels in Benoa Village have got their own tsunami evacuation procedures.

DISCUSSION

The research findings presented are not without limitations. Some large hotels with 3-floor buildings can be considered as nearby evacuation points, thus shortening the distance from the initial evacuation spot. However, many of these large hotels already have their own evacuation procedures, necessitating discussions on this matter. The graph depicted in Figure 3 only includes major roads. This research is open for further studies, taking into account smaller roads, which would result in a more intricate graph with a greater number of vertices. This would influence a wider variety of evacuation routes and potentially establish a relatively shorter evacuation distance.

CONCLUSION

Based on the results of the discussion, there are 2 safe zones in terms of tsunami evacuation in Benoa Village. They are Grand Hyatt Bali Temporary (V50) and Hattrick Futsal (V51). The Evacuation Starting Point is determined from population density and frequently visited areas namely Serangan Beach (V1), Noanui Beach (V2), Samuh Beach (V3), Nusa Dua Beach A (V4), Nusa Dua Beach B (V5), Peninsula Island (V6), Mengiat Beach (V7), Geger Beach (V8) towards the Grand Hyatt Bali Temporary (V50) and Hattrick Futsal (V51) Safe Zones. The tsunami evacuation paths in the form of shortest trajectories obtained using the Djikstra's Algorithm and their shortest distances are presented in Table 3 with distances from Serangan Beach (4.49km to V50, 6.94km to V51), Noanui Beach (3.95km to V50, 6.39km to V51), Samuh Beach (2, 75km to V50, 5.19km to V51), Nusa Dua Beach A (3.54km to V50, 5.48km to V51), Nusa Dua Beach B (3.47km to V50, 4.79km to V51), Peninsula Island (4.69km to V50, 4.80km to V51), Megiat Beach (4.81km to V50, 4.15km to V51) and Geger Beach (5.86km to V50, 5.09km to V51).

REFERENCES

Afandi, N., & Mayasari, Z. M. (2021). An evacuation route in bengkulu city based on fuzzy dijkstra algorithm. *Journal of Physics: Conference Series*, *1863*(1), 12007.

Akram, M., Habib, A., & Alcantud, J. C. R. (2021). An optimization study based on Dijkstra algorithm for a network with trapezoidal picture fuzzy numbers. *Neural Computing and Applications*, *33*(4), 1329–1342.

Anggria, S., Syafwan, M., & Efendi, E. (2017). Pemodelan Optimasi Evakuasi Tsunami di Kota Padang. *Contoh*.

Badan Meteorologi, K. dan G. (BMKG) dan G. (2023). *Pedoman Pelayanan Peringatan Dini Tsunami InaTEWS*. Https://Www.Gitews.de/Tsunami-Kit/Id/E3/Perangkat/Pedoman%20Pelayanan%20Peringatan%20Dini%20Tsunami%20InaTEWS%20(2).Pdf.

Badan Pusat Statistik Kabupaten Badung. (2023). *Kabupaten Badung Dalam Angka 2023*. Https://Badungkab.Bps.Go.Id/Publication/2023/02/28/E43a29aa94f24066c1e95c35/Kabupaten-Badung-Dalam-Angka-2023.Html.

E-ISSN : 2807-9035
Volume 3, **Number** 2, November 2023
https://doi.org/10.47709/brilliance.v3i2.3089

Darmi, Y., & Muntahanah, M. (2022). Implementation of Dijkstra's Algorithm to Determine Evacuation Paths in Earthquake and Tsunami Simulations in Bengkulu City Based on Gis. *Jurnal Media Computer Science*, *1*(2), 117–130.

Deng, Y., Chen, Y., Zhang, Y., & Mahadevan, S. (2012). Fuzzy Dijkstra algorithm for shortest path problem under uncertain environment. *Applied Soft Computing*, *12*(3), 1231–1237.

Dijkstra, E. W. (2022). A note on two problems in connexion with graphs. In *Edsger Wybe Dijkstra: His Life, Work, and Legacy* (pp. 287–290).

Iryanto, I., & Ismantohadi, E. (2017). Optimasi Pemilihan Barang Dagangan bagi Pedagang Keliling dengan Algoritma Genetika. *JTT (Jurnal Teknologi Terapan)*, *3*(1).

Khaleel, T. A., & Al-Shumam, A. A. (2020). A study of graph theory applications in it security. *Iraqi Journal of Science*, 2705–2714.

Liu, L., Zhang, H., Shi, J., & Geng, J. (2023). Real-time evacuation route optimization in the fire scenarios of cruise ships. *Simulation Modelling Practice and Theory*, 102843.

Luo, M., Hou, X., & Yang, J. (2020). Surface optimal path planning using an extended Dijkstra algorithm. *Ieee Access*, *8*, 147827–147838.

Pramudya, R. A., & Subiyanto, S. (2015). Penggunaan algoritma dijkstra dalam perencanaan rute evakuasi bencana longsor di kota semarang. *Geoplanning: Journal of Geomatics and Planning*, *2*(2), 93–102.

Putra, I. E., & Rohendi, K. (2017). Implementation of geographic information system with Dijkstra Algorithm Base on mobile application: A model for disaster risk evacuation route in Padang City Indonesia. *Proceedings of the 1st International Conference on E-Commerce, E-Business and E-Government*, 30–34.

Sani, A. F., Tastrawati, N. K. T., & Dwipayana, I. M. E. (2013). Algoritma Floyd Warshall Untuk Menentukan Jalur Terpendek Evakuasi Tsunami di Kelurahan Sanur. *E-Jurnal Matematika*, *2*(1), 1–5.

Sporns, O. (2022). Graph theory methods: applications in brain networks. *Dialogues in Clinical Neuroscience*.

Zhu, Y., Li, H., Wang, Z., Li, Q., Dou, Z., Xie, W., Zhang, Z., Wang, R., & Nie, W. (2022). Optimal Evacuation Route Planning of Urban Personnel at Different Risk Levels of Flood Disasters Based on the Improved 3D Dijkstra's Algorithm. *Sustainability*, *14*(16), 10250.