Designing futsal match finder application with floyd-warshall algorithm

K Manaf*, W Uriawan, W Agustian, Y A Gerhana, J Jumadi and M A Ramdhani
Department of Informatics, UIN Sunan Gunung Djati Bandung, Bandung, Indonesia

*Corresponding author’s email: khaerulmanaf@uinsgd.ac.id

Abstract. Information about the locations where futsal competition is taking place is needed for the futsal team to meet each other and compete. The purpose of designing Futsal Match Finder is to determine the position of futsal field location and help the shortest path to the field by implementing Floyd-Warshall algorithm. Application system design methodology that used in this research is prototype model. System test results show that the Floyd-Warshall algorithm can help find the shortest path to the location of the opposite futsal team. The futsal opponent search system based on the shortest distance can provide a solution to improve the ability of futsal players who have less relationship with other futsal team.

1. Introduction
Futsal is an abbreviation of the futbol/ football (soccer) and sala (room) of the Spanish language or futebol (Portugal/ Brazil) and salon (France). This sport forms a player to always be ready to receive and feed the ball quickly in the pressure of opposing players. With a narrow field, this game requires high ball possession techniques, collaboration between players, and team cohesiveness. Futsal is a sport that played by two teams, where each team has no more than five players, and one of which is the goalkeeper [1].

In Bandung city, futsal sports become a routine exercise that is usually done by anyone. This can be seen from the enthusiasm of playing futsal performed by people in the Bandung city, either children or adults. Today's futsal academy has been many established in the Bandung city such as the Mayasari, Energico, Jeni, Vamos academies, and so on. The academy was founded aiming to train the quality of playing futsal players. Moreover, with the many futsal fields scattered in the Bandung city making its citizens facilitated to do futsal sports.

Similar previous studies that have been conducted, among others: Opposition Seeker Playing Futsal on the Android Platform [1]; E-Futsal Application with Mobile-GIS and GPS-Based Android Methods [2]; Development of Web-based Geographic Information System for Padang City Futsal Field [3]; Design and Development of Android-based Ready for Battle Futsal Applications [4]; and Futsal Match Training Application Using Laravel Framework [5]. This study aims to design an information system using the Floyd-Warshall algorithm to search for opponents playing futsal based on distance. It will make it easier for each team to determine their opponents, who will be able to efficient the time and costs incurred when making a futsal match with another team.
2. Methods
The system design process in this study uses a prototype design approach. The Prototype model is one of the software development methods that is widely used by developers because besides it is simple also the method can also adjust the user's requirements in detail in developing software. This method is able to offer the best approach in terms of certainty in the efficiency of the algorithm, the ability to adjust itself from a site operation or the forms that must be done by human interaction with the machine [6]. The cycle of the prototype model consists communication process, quick plan process, modelling quick design process, construction of prototype, deployment, delivery, and feedback process [6], [7]. Reliability of system applications in this research, it is used: analytical, logical, conceptual, and operational verification by an expert[8].

3. Result and Discussion
This research uses Prototype as software development method to build information system for finding opponents of futsal competition based on distance using Floyd-Warshall algorithm calculations. Information system processes data in an organized way [9], which attempts to adapt human knowledge to computer, so that computer can solve problems as usually done by experts [10]. Based on many research findings, expert system has excellent ability in decision making, the system has advantages in terms of: more economical [11], broad accessibility [12], improving user understanding [13], good data accessibility[14], time efficiency [15], accuracy [16], providing good data and information [17], used as data storage media [18], supporting decision appropriately [19], more economical [11], broad accessibility [12], improving user understanding [13], and improving productivity [20].

3.1. Floyd-Warshall Algorithm
The algorithm that used in the system is Floyd-Warshall. The Floyd-Warshall algorithm is one variant of dynamic programming, which is a method that performs a solution by looking at the solution that will be obtained as an interrelated decision. This means that the solution is formed from a solution that comes from the previous stage and there are more than one possible solutions [21].

The algorithm is found by Warshall to find the shortest route. This algorithm is simple and easy to implement. The Floyd-Warshall algorithm has a directed and weighted input graph (V, E), where V is set of vertex and E is set of edge. Weight of edge E can be given the symbol w (e) as the number of side weights on a path is the total weight of the path. The edge E is allowed to have a negative weight, but it is not allowed for the Wij graph to have a negative weight cycle. This algorithm calculates the smallest weight of all the paths that connect a pair of vertex, and does it all at once for all pairs of vertex to reach the destination with the minimum weight [21].

\[ W_{ij} = (W_{ij}, W_{ik} + W_{kj}) \] \( j \leq i, k \geq 1 \) \( (1) \)

The Floyd-Warshall algorithm for finding the shortest traffic is as follows:

\[
W=W^0 \\
\text{For } k = 1 \text{ to } n \text{ do} \\
\quad \text{For } i = 1 \text{ to } n \text{ do} \\
\quad \quad \text{For } j = 1 \text{ to } n \text{ do} \\
\quad \quad \quad \text{If } W_{ij} > W_{ik} + W_{kj} \text{ then switch } W_{ij} \text{ with } W_{ik} + W_{kj} \\
\quad \quad \text{W* = W}
\]

Where:
- \( W_0 \) = the connectedness matrix of directed graphs with an initial weight;
- \( W* \) = minimal connectivity matrix;
- \( W_{ij} \) = the shortest path from point \( V_i \) to \( V_j \)

In its iteration to find the shortest path, the Floyd-Warshall algorithm forms n matrix, according to the k-iteration, this will cause the process to be slow, especially for large n values. Although the processing time is not the fastest, the Floyd-Warshall algorithm is often used to calculate the shortest
path because of its simplicity. Besides that, the implementation of the Floyd-Warshall algorithm is very easy to create [21].

3.2. System Design
This research is the development result of research that has been done before, in which the previous research discussed about how to search opponents competing in futsal based on distance. And the result will be able to efficiently in the term of manage the time and cost by each futsal team to conduct a futsal competition if they are going to visit another futsal team. In the previous system, there was a shortage where users who have just joined the system do not have a team and have to wait for an invitation from another team or send a confirmation to participate in an existing team. This problem will take a while for new users to join playing futsal.

This research builds an Android-based application which is used by individual user and in its use the user requires an Android-based mobile Smartphone for its operation. Using of mobile phone for the system, based on the advantages of Android-based application programs: fun [22], provide multimedia-based information [23], concise and mobile [24], good accessibility [25], and efficient [26]. Architecture of the futsal match finder system can be seen in Figures 1-2.

![System Architecture](image1)

**Figure 1. System Architecture**

![Application Architecture](image2)

**Figure 2. Application Architecture**

Users are person who can operate the Match Finder application using an Android-based mobile smartphone. Users can create teams as they wish, besides user can also move the home base where the team conducts futsal matches. Then, the user can search for the opposing team and find out the distance to be visited. The user can view the profile of opponent in the application. After getting the information about the location of the opponent who will be invited to compete and see the profile of the opponent, the user can send a notification to compete against the opposing team.

Users can view the distance of each of the futsal team opponents as the calculation result that was done using the Floyd-Warshall algorithm. This result will facilitate the user to find out the location and distance of the futsal team and the data that has been entered will be stored in the database. Users can edit their team personal data. The user can also view the details of the competition held in the Bandung city. Then, the user can view the invitation notification to compete from the other teams who invite to play futsal.
Application architecture design describes the basic framework of applications that will be built that more specifically and structured. The application architecture can be seen in Figure 2 which illustrates a process of input, process, and output of the futsal Match Finder system. For the security requirement of system, the futsal team as a user must be registration before log in or access the application. After getting the login account, the user will log in and proceed the next stage, where the user can view the information about the opposing team profile, the location of the opposing team, the schedule, place of competition for the opposing team, and information about the futsal competition. Furthermore, searching for the opposing futsal team and futsal competition will be processed using the Floyd-Warshall algorithm calculation. After being processed by Floyd-Warshall calculations, the output of this system is in the list of futsal field location based on the closest distance from the user location.

3.3. Testing of Floyd-Warshall Method for Futsal Match Finder Application

Testing scenario to see whether the Floyd-Marshall algorithm is able to give best recommendation futsal field location and futsal competition is conducted by comparing between the manual calculations an system calculation. In the case, the distance of each futsal location point in the Bandung city was calculated. Figure 4 shows the results of calculations to determine the distance of the futsal field location in Bandung City.
that will be selected in the city of Bandung using Floyd-Warshall method can be seen in Figures 5-9. And the testing result show that Floyd-Warshall has been implemented well in the system.

\[
W_{ij} = \min(W_{ik}, W_{kj})
\]

\[
W[1][3] = \min (3.6, 3.1 + \infty) = \text{Distance} \rightarrow 3.6
\]

\[
W[2][4] = \min (\infty, 3.1 + \infty) = \text{Distance} \rightarrow \infty
\]

\[
W[2][5] = \min (3.1, 7.7 + \infty) = \text{Distance} \rightarrow 10.8
\]

\[
W[3][2] = \min (3.6, 3.1 + 3.1) = \text{Distance} \rightarrow 3.6
\]

\[
W[3][4] = \min (2.8, \infty + \infty) = \text{Distance} \rightarrow 2.8
\]

\[
W[3][5] = \min (\infty, \infty + 7.7) = \text{Distance} \rightarrow \infty
\]

\[
W[4][2] = \min (\infty, \infty + 3.1) = \text{Distance} \rightarrow \infty
\]

\[
W[4][3] = \min (\infty, \infty + \infty) = \text{Distance} \rightarrow \infty
\]

\[
W[4][5] = \min (\infty, \infty + 7.7) = \text{Distance} \rightarrow 4.7
\]

\[
W[5][2] = \min (\infty, 7.7 + 3.1) = \text{Distance} \rightarrow 10.8
\]

\[
W[5][3] = \min (\infty, 7.7 + \infty) = \text{Distance} \rightarrow \infty
\]

\[
W[5][4] = \min (4.7, 7.7 + \infty) = \text{Distance} \rightarrow 4.
\]

| Distance (1st Iteration) | 1 (A) | 2 (B) | 3 (C) | 4 (D) | 5 (E) |
|--------------------------|-------|-------|-------|-------|-------|
| 1 (A)                    | 0     | 3.1   | ∞     | 7.7   |       |
| 2 (B)                    | 3.1   | 3.6   | ∞     | 10.8  |       |
| 3 (C)                    | ∞     | 3.6   | 2.8   | 14.4  |       |
| 4 (D)                    | ∞     | ∞     | 2.8   | 4.7   |       |
| 5 (E)                    | 7.7   | 10.8  | 14.4  | 4.7   | 0     |

**Figure 5. 1st Iteration**

| Distance (2nd Iteration) | 1 (A) | 2 (B) | 3 (C) | 4 (D) | 5 (E) |
|--------------------------|-------|-------|-------|-------|-------|
| 1 (A)                    | 0     | 3.1   | 6.7   | 7.7   |       |
| 2 (B)                    | 3.1   | 3.6   | ∞     | 10.8  |       |
| 3 (C)                    | 6.7   | 3.6   | 2.8   | 14.4  |       |
| 4 (D)                    | ∞     | ∞     | 2.8   | 4.7   |       |
| 5 (E)                    | 7.7   | 10.8  | 14.4  | 4.7   | 0     |

**Figure 6. 2nd Iteration**

| Distance (3rd Iteration) | 1 (A) | 2 (B) | 3 (C) | 4 (D) | 5 (E) |
|--------------------------|-------|-------|-------|-------|-------|
| 1 (A)                    | 0     | 3.1   | 6.7   | 7.7   |       |
| 2 (B)                    | 3.1   | 3.6   | ∞     | 10.8  |       |
| 3 (C)                    | 6.7   | 3.6   | 2.8   | 14.4  |       |
| 4 (D)                    | ∞     | ∞     | 2.8   | 4.7   |       |
| 5 (E)                    | 7.7   | 10.8  | 14.4  | 4.7   | 0     |

**Figure 7. 3rd Iteration**

| Distance (4th Iteration) | 1 (A) | 2 (B) | 3 (C) | 4 (D) | 5 (E) |
|--------------------------|-------|-------|-------|-------|-------|
| 1 (A)                    | 0     | 3.1   | 6.7   | 9.5   | 7.7   |
| 2 (B)                    | 3.1   | 3.6   | 6.4   | 10.8  |       |
| 3 (C)                    | 6.7   | 3.6   | 2.8   | 14.4  |       |
| 4 (D)                    | 9.5   | 6.4   | 2.8   | 4.7   |       |
| 5 (E)                    | 7.7   | 10.8  | 14.4  | 4.7   | 0     |

**Figure 8. 4th Iteration**
3.4. Application Testing

Operating the system can be done by checking all of the functions in the system whether it is running well or not, whether the application run as expected or not. Besides testing whether the implementation of the Floyd-Warshall method is applied well or not on the futsal Match Finder application. The functionalities of the application are also tested using black-box testing. Black-box testing is a software testing method to test the functionality of the software [6].

A black box is a testing method that observes the results of the execution and checks the functionality of the software. This test method based on what is executed through the interface without knowing the process that occurs in software in detail (code or logic structure of software). Testing to see a list of futsal field locations is a one of functional test. The example of Field Location List testing can be seen in Table 1. The test results prove that the functionalities of application can run as expected.

**Table 1. Application Black Box Testing Result**

| Code   | Scenarios                              | Expected Result                                      | Real Result                                      | Decision |
|--------|----------------------------------------|------------------------------------------------------|--------------------------------------------------|----------|
| FMF.01 | User access to field form              | Application shows field form                         | Application success shows field form              | √        |
| FMF.02 | View list of futsal field location     | Application shows list of futsal field location      | Application shows list of futsal field location   | √        |
| FMF.03 | User access the futsal team form       | Application shows futsal team form                   | Application success shows futsal team form        | √        |
| FMF.04 | View list of futsal team               | Application shows list of futsal team                | Application shows list of futsal team             | √        |
| FMF.05 | User choose one of futsal team         | Application shows the information of opposing futsal team | Application shows the information of opposing futsal team | √        |
| FMF.06 | User access the futsal schedule form   | Application shows futsal schedule form               | Application success shows futsal schedule form    | √        |
| FMF.07 | View list of opposing futsal team      | Application shows list of opposing futsal team schedule | Application shows list of opposing futsal team schedule | √        |
| FMF.08 | User access the notification form      | Application shows notification form                  | Application success shows notification form       | √        |
| FMF.09 | View list of notification from opposing futsal team | Application shows list of notification from opposing futsal team | Application shows list of notification from opposing futsal team | √        |
| FMF.10 | User access the futsal competition information form | Application shows futsal competition information form | Application success shows futsal competition information form | √        |
| FMF.11 | View list of futsal competition information | Application shows list of futsal competition information | Application shows list of futsal competition information in detail | √        |

**Figure 9. 5th Iteration**

| Distance (5th Iteration) | 1 (A) | 2 (B) | 3 (C) | 4 (D) | 5 (E) |
|--------------------------|-------|-------|-------|-------|-------|
| W₁₁ = min(W₁₁, W₁₂ + W₁₃) | 3.1   | 0     | 3.6   | 6.4   | 10.8  |
| W₁₂ = min(3.1, 7.7 + 10.8) = Distance → 3.1 |
| W₁₃ = min(6.7, 7.7 + 7.5) = Distance → 6.7 |
| W₁₄ = min(9.5, 7.7 + 4.7) = Distance → 9.5 |
| W₁₅ = min(6.7, 7.5 + 7.7) = Distance → 6.7 |
| W₂₁ = min(3.1, 10.8 + 7.7) = Distance → 3.1 |
| W₂₂ = min(3.6, 10.8 + 7.5) = Distance → 3.6 |
| W₂₃ = min(3.6, 10.8 + 7.5) = Distance → 3.6 |
| W₂₄ = min(9.5, 4.7 + 7.7) = Distance → 9.5 |
| W₂₅ = min(6.4, 4.7 + 7.5) = Distance → 6.4 |
| W₃₁ = min(6.7, 7.5 + 7.7) = Distance → 6.7 |
| W₃₂ = min(3.6, 7.5 + 10.8) = Distance → 3.6 |
| W₃₃ = min(3.6, 7.5 + 10.8) = Distance → 3.6 |
| W₃₄ = min(2.8, 7.5 + 4.7) = Distance → 2.8 |
| W₄₁ = min(3.1, 10.8 + 7.7) = Distance → 3.1 |
| W₄₂ = min(3.6, 10.8 + 7.5) = Distance → 3.6 |
| W₄₃ = min(2.8, 7.5 + 4.7) = Distance → 2.8 |
| W₄₄ = min(6.4, 4.7 + 7.5) = Distance → 6.4 |
| W₅₁ = min(6.7, 7.5 + 7.7) = Distance → 6.7 |
| W₅₂ = min(3.6, 7.5 + 10.8) = Distance → 3.6 |
| W₅₃ = min(2.8, 7.5 + 4.7) = Distance → 2.8 |
| W₅₄ = min(6.4, 4.7 + 7.5) = Distance → 6.4 |
| W₅₅ = min(2.8, 4.7 + 7.5) = Distance → 2.8 |
4. Conclusion
Based on the testing results of the futsal Match Finder system using the Floyd-Warshall algorithm, the implementation of the Floyd-Warshall algorithm for the futsal Match Finder system has been successfully built and can be used to determine the opponent to compete in futsal based on the closest distance. The Floyd-Warshall algorithm that used in the futsal Match Finder system has an accuracy that can be used as a benchmark by the user to determine the opponent to compete in futsal based on the distance that displayed in the Futsal Match Finder system. With this good result, the system can be a reference for each futsal team in determining the opponent competes with an estimated distance that is not too far away.

5. References
[1] Subagia G P 2018 Pencari Lawan Bermain Futsal pada Platform Android,” Bandung, 2015.
[2] Agustian A, Rahayu S and Nurlani L 2018 Aplikasi E-Futsal dengan Metode Mobile-GIS dan GPS Berbasis Android Jurnal Teknologi Rekayasa 3 1115-128
[3] Suryamen H, Aminuddin I and Akbar F 2016 Pembangunan Sistem Informasi Geografis Lapangan Futsal Kota Padang Berbasis Web Jurnal Teknologi dan Sistem Informasi 2 1 45-54
[4] Ardiansyah Y and Hartanto A D 2015 Perancangan dan Pembuatan Aplikasi Ready For Battle Futsal Berbasis Android Data Manajemen dan Teknologi Informasi (DASI) 16 1 63
[5] Pramana D and Yuningsih L 2017 Aplikasi Latih Tanding Futsal Menggunakan Framework Laravel. Jurnal Sistem dan Informatika (JSI) 12 1 48-55
[6] Pressman R S 2011 Software Engineering: A Practitioner’s Approach, 7th ed. (New York: McGraw-Hill)
[7] Rosa A S and Shalahuddin M Rekayasa 2014 Perangkat Lunak Terstruktur dan Berorientasi Objek in Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek
[8] Suhendi H Y, Ramdhani M A and Irwansyah F S 2018 Verification Concept of Assesment for Physics Education Student Learning Outcome International Journal of Engineering & Technology (UEA) 7 3.21 321-325
[9] Maylawati D S, Darmalaksana W and Ramdhani M A 2018 Systematic Design of Expert System Using Unified Modelling Language In IOP Conference Series: Materials Science and Engineering 288 1 012047
[10] Ramdhani M A 2013 Metodologi Penelitian untuk Riset Teknologi Informasi (Bandung: UIN Sunan Gunung Djati Bandung)
[11] Rahman A, Slamet C, Darmalaksana W, Gerhana Y A and Ramdhani M A 2018 Expert System for Deciding a Solution of Mechanical Failure in a Car using Case-based Reasoning in IOP Conference Series: Materials Science and Engineering 288 1 012011
[12] Slamet C, Rahman A, Ramdhani M A and Darmalaksana W 2016 Clustering the Verses of the Holy Qur’an Using K-Means Algorithm Asian J. Inf. Technol 15 4 5159–5162
[13] Maylawati D S, Ramdhani M A, Zulfikar W A, Taufik I and Darmalaksana W 2017 Expert system for predicting the early pregnancy with disorders using artificial neural network in 2017 5th International Conference on Cyber and IT Service Management, CITSM 2017
[14] Aulawi H, Ramdhani M A, Slamet C, Ainissyifa H and Darmalaksana W 2017 Functional Need Analysis of Knowledge Portal Design in Higher Education Institution Int. Soft Comput 12 2 132–141
[15] Slamet C, Rahman A, Sutedi A, Darmalaksana W, Ramdhani M A and Maylawati D S 2018 Social Media-Based Identifier for Natural Disaster,” in IOP Conference Series: Materials Science and Engineering 288 1 012039.
[16] Slamet C, Andrian R, Maylawati D S, Darmalaksana W and Ramdhani M A 2018 Web Scraping and Naïve Bayes Classification for Job Search Engine 288 1 1–7
[17] Maylawati D S, Ramdhani M A, Rahman A and Darmalaksana W 2017 Incremental technique with set of frequent word item sets for mining large Indonesian text data 2017 5th Int. Conf. Cyber IT Serv. Manag. CITSM 2017 1–6
[18] Taofik A, Ismail N, Gerhana Y A, Komarujaman K and Ramdhani M A 2018 Design of Smart
System to Detect Ripeness of Tomato and Chili with New Approach in Data Acquisition in IOP Conference Series: Materials Science and Engineering 288 1 012018.

[19] Gerhana Y A, Zulfikar W B, Ramdani A H and Ramdhani M A 2018 Implementation of Nearest Neighbor using HSV to Identify Skin Disease IOP Conf. Ser. Mater. Sci. Eng 288 1 012153.

[20] Zulfikar W B, Jumadi, Prasetyo P K and Ramdhani M A 2018 Implementation of Mamdani Fuzzy Method in Employee Promotion System IOP Conf. Ser. Mater. Sci. Eng. 288 1 012147.

[21] Ningrum F W and Andrasto T 2016 Penerapan Algoritma Floyd-Warshall dalam Menentukan Rute Terpendek pada Pemodelan Jaringan Pariwisata di Kota Semarang J. Tek. Elektro 8 1 21–24.

[22] Aisyah R, Zakiyah I A, Farida I and Ramdhani M A 2017 Learning Crude Oil by Using Scientific Literacy Comics J. Phys. Conf. Ser. 895 1 012011.

[23] Irwansyah F S, Yusuf Y M, Farida I and Ramdhani M A 2018 Augmented Reality (AR) Technology on the Android Operating System in Chemistry Learning IOP Conf. Ser. Mater. Sci. Eng., vol. 288 1 012068.

[24] Sari, Irwansyah F S, Farida I and Ramdhani M A 2017 Using Android-Based Educational Game for Learning Colloid Material Using Android-Based Educational Game for Learning Colloid Material,” J. Phys. Conf. Ser. 895 1 012012.

[25] Helsy I, Maryamah, Farida I and Ramdhani M A 2017 Volta-Based Cells Materials Chemical Multiple Representation to Improve Ability of Student Representation J. Phys. Conf. Ser. 895 1 012010.

[26] Farida I, Helsy I, Fitriani I and Ramdhani M A 2018 Learning Material of Chemistry in High School Using Multiple Representations IOP Conf. Ser. Mater. Sci. Eng. 288 012078.

Acknowledgments
The authors expressed the appreciation and thank you to the Research and Publishing Center of UIN Sunan Gunung Djati Bandung, which has provided funding support for the publication of this article.