Traveler Planner

[1]Pranati Pandey, [2]Purvanshika Bhoot, [3]Bipasha Das

[1]Student, [2]Student, [3]Student
Information Technology Department,
Shri Ramdeobaba College of Engineering and Management, Nagpur, India

Abstract: **'Traveler Planner' aims at strengthening the tourism industry by providing an application that is compatible on almost all platforms using the Quasar framework. The database used here is Firebase with UI/UX work done in Vue.js and backend scripts written in Python. We have incorporated the use of Artificial Intelligence A* algorithm and Machine Learning Decision Tree algorithm to create the most important function of this application- Planner. This planner will give customized sites and locations to visit as the user inputs their preferences. It will also suggest various other spots on the shortest way to their desired locations. This application provides several other utilities like the manual search of places around the world, world clock, weather, checklist and location with 'Find Me' options.**

***Keywords*: Quasar, Vue.js, Firebase, Python, API, AI, ML, Tourist, Tourism.**

I. INTRODUCTION

According to The World Travel and Tourism Council (WTTC), tourism generated $194 bn or 6.8% of India's GDP in 2019 and supported 39.80 million jobs which is 8% of its total employment. The sector is predicted to grow at an annual rate of 6.9% to $460 bn by 2028 which is 9.9% of GDP. Tourism accounts for nearly 40% of all employment here providing livelihoods for thousands in a place like Maldives where other big sectors like corporates can't go. By its very nature Travel & Tourism is labor intensive [1]. Take the hotel sector. It provides on average one employee per room. Add on people indirectly employed too like guides, drivers, gardeners and laundry staff and that number rises to four. And around half of all employees in the hotel, catering and hospitality sector are under 25.

1.1 Problems Encountered by Tourists

The biggest problem with tourists who visit anywhere from anywhere is when some of them (and remember some, not all) don't do their research before arriving and they expect that everything will be as convenient or even as culturally and socially (and even legally) acceptable as it is in their own countries of origin.

Globalization has created less distinctive locales. Due to global standardization, indistinguishable products are found in every country. If a person is traveling in order to learn or have an opportunity to explore the different and unique, then the lack of unique products becomes a tourism challenge. An example is that shopping malls all over the world tend to offer matching products. Also, tourists or travelers can at times deem travel marketing to be false, inadequate or exaggerated. Other challenges include fluctuation in currency exchange, seasonal dependence, regulatory issues, language barriers and lack of skilled human resources who understand the nature of tourism. Finding help from government offices for visas and other documents often becomes difficult due to reasons like language barriers and lack of accurate resources to find them.

1.2 Proposed Solution

As seen above the importance of tourism industry, addressing these problems becomes a vital and important job to decrease the barriers in any way possible. To tackle these problems in an efficient manner, we proposed to build an application, which can give travel predictions and planners according to weather, time, current position and distance. It can also provide a way to locate government offices, local markets and nearby hotels and restaurants. We have built it using Quasar framework hence it can be deployed in multiple platforms like Android, iOS and Desktop simultaneously.

By using our application "Traveler Planner", users might not need to plan their trips. We have used A* algorithm to give the most usage of our application. The user will enter their starting place and their goal place. Using A* algorithm, we have provided the users the best possible path along with other places like restaurants, shops, sites, etc. which comes in that path so that the user can plan their trip accordingly.

In this application we are going to use decision tree for making the customized planner. We created a database and that database will contain all the information which is needed to travel between two cities and will act as a dummy database. then we will apply decision tree algorithm on that data. we will connect the database with this decision tree algorithm and this algorithm will work on that database along with several other parameters that we will give user as options. These are the parameters like what type of places would the user like to go (historic, natural, etc.), number of hours they have or budget that is planned, etc. The decision tree algorithm will work on that data along with these parameters to give personalize result to the user. Hence, we will be providing an automatically made planner through our application, which will let user decide that were would they like to visit. Our application will not force this personalize result on the users since users will also have an option of manual search. APIs integrated will give the real time location and time-weather updates to the users.

II. COMPARATIVE ANALYSIS OF PROPOSED SOLUTION AND EXISTING WORKS

We have prepared an extensive research-based comparison table to highlight several advantages of our applications. The literature we have considered include Design and Implementation of an Online Location Based Services Using Google Maps for Android

Mobile [2], Android Mapping Application [3], Tourist assistant – TAIS [4], Developing a Location Based Tourist Guide Application [5] and many more [6][7][8]. Table 2.1 compares existing methods with the proposed application based on various parameters like platform compatibility, database used, etc.

Table 2.1: Existing Works VS Proposed Method

Basis of Comparison	Existing Methods	Traveler Planner
Platform	Android, iOS	Android, iOS, MAC, Windows
Planner	No	Yes
Traveler Friendly Features	No	Yes
Trivia	No	Yes
Location	Yes	Yes
Date and Time	Yes	Yes
Weather	Yes	Yes
Development Environment	Complex	User Friendly
Application or Website	Application	Both
Meta-data Search	Yes	Yes
User Reviews	No	Yes
Emergency Contact	No	Yes
Database	Physical Servers	Cloud-based

III. METHODOLOGY

We will develop an application that is ready to use on any platform. It will help tourists to get sorted and personalized results as per their inputs in the input bar for the place and duration they feed in the application. Weather updates, world clock, to-do list, the location search bar [9] and maps [10] are some of the integrated functions in our application.

3.1 Technology Stack

The system and software requirements include installing Visual studio for IDE and to run quasar and creating a firebase account on google.

Software Requirements:

i. Windows 7 or higher
ii. Python for backend (To execute AI and ML using Libraries like Pandas and Scikit-learn)
iii. Quasar framework
iv. Firebase Real time Database (Cloud-based Database)

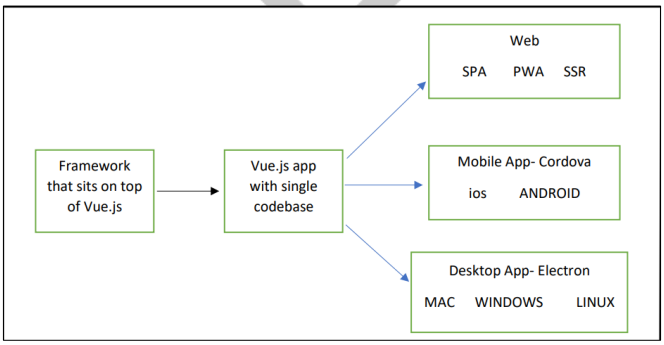
Hardware Components:

i. Processor – Dual Core
ii. Hard Disk – 1 TB
iii. Memory – 4GB RAM

3.2 Action Plan

For achieving this, we have initialized the Quasar framework and downloaded the required packages as per our need.

Figure 3.1: Quasar



Quasar CLI is made up of two packages:

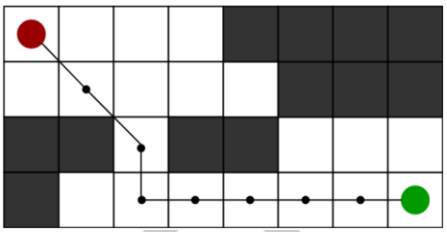
- @quasar/cli: This one is optional and only allows to create a project folder and globally run Quasar commands.
- @quasar/app: This package is the most important and is installed into every Quasar project folder.

Once a project folder has been generated, Quasar CLI will only help in running @quasar/app's commands globally. To ensure full independence from Quasar CLI we can write npm scripts, which is in package.json to run Quasar commands. @quasar /app is specific to each project and will run all the CLI commands.

The backend for our application will be made using Firebase. It is an all-in-one backend solution provided by Google. One can handle authentication and read-write operations on our database using some simple API. In technical language, an API or application-programming interface is an interface that defines interactions between multiple software applications or mixed hardware-software intermediaries. Some examples are web API (JavaScript), Native iOS (Swift/ Objective-C), Native Android (Java/ Kotlin) etc. We have used APIs to display the information and details of tourist cities in our application. Google's geolocation API or alternatively, Telize API can be used to give personalized results without giving in too much of efforts.

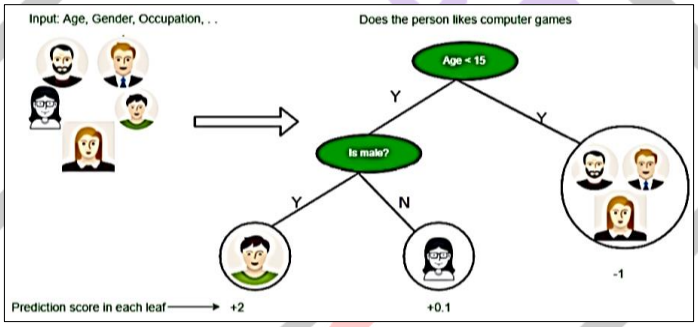
A* Search algorithm (Figure 3.2) is one of the best and popular technique used in path finding and graph traversals. Informally speaking, A* Search algorithms, unlike other traversal techniques, has "brains" which separates it from the other conventional algorithms. This can be achieved using AI [11].

Figure 3.2: A* Algorithm Implementation



Decision Trees (Figure 3.3) are a type of Supervised Machine Learning. This means the user explains what the input is and what the corresponding output is in the training data. In this, data is continuously split according to a certain parameter. The goal of using a Decision Tree is to create a training model that can use to predict the class or value of the target variable by learning simple decision rules inferred from training data. In Decision Trees, for predicting a class label for a record we start from the root of the tree [12].

Figure 3.3: Decision Tree Implementation



For maps, firstly, we enable and create an API key to your project. Now we have enabled the Google Maps JavaScript API and generated an API key. Now along with the above, we enable Google Maps Places API too. This is important to give the user a dropdown with all places available in the Google maps. Add the plugins to quasar project. Go to the quasar root folder, open with terminal, and add following plugins.

Figure 3.4: Important plugins to be added for Maps

```
If you using yarn
yarn add vue-browser-geolocation

if you using NPM
npm install vue-browser-geolocation
```

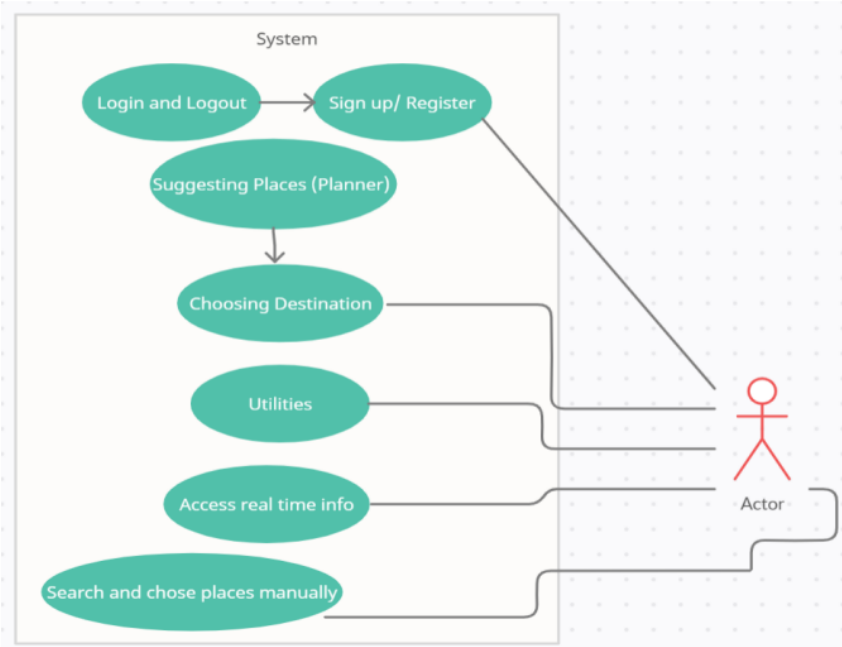
```
If you using yarn
yarn add vue2-google-maps

if you using NPM
npm install vue2-google-maps
```

In all, we have used two already available APIs to incorporate map like functions in our project. A file called index.tempate.html in quasar file directory will be formed. Go to the file and put the required JS file in head section. Now create a new page or component and create the required template. Now we set the page template add all the important scripts in order to run the Google Maps. Google Maps API is now available to run on our application.

IV. PROPOSED SYSTEM

The basic framework of our proposed system has been depicted with the help of the UML diagrams in Figure 4.1 and Figure 4.2. These diagrams show that how can a user interact with the system and how will the data flow in our application, respectively.

Figure 4.1: Use Case Diagram for System



The use case diagram shown in Figure 4.1 suggests how can a user interact with our system. If the user is not registered, they can sign-up fresh, and their data will be stored in our database. Next time, user can login using same credentials. The users can then access utilities like weather, world clock, create checklists with alarm and timer with proper notification system. Users can access the real-time information available on the internet using Google APIs integrated in our system. This real-time information includes reviews, trivia, etc. The users can search and chose places to visit manually. The manual search results are also updated in real-time as they are retrieved from the internet. The most important part of our application, which is the customized planner, can be access by the user which can be seen in the drawer, which is in the left side of our application. This planner will ask user some questions, and based on the answers, the planner will suggest places using the decision tree algorithm.

Figure 4.2: Dataflow Diagram of System

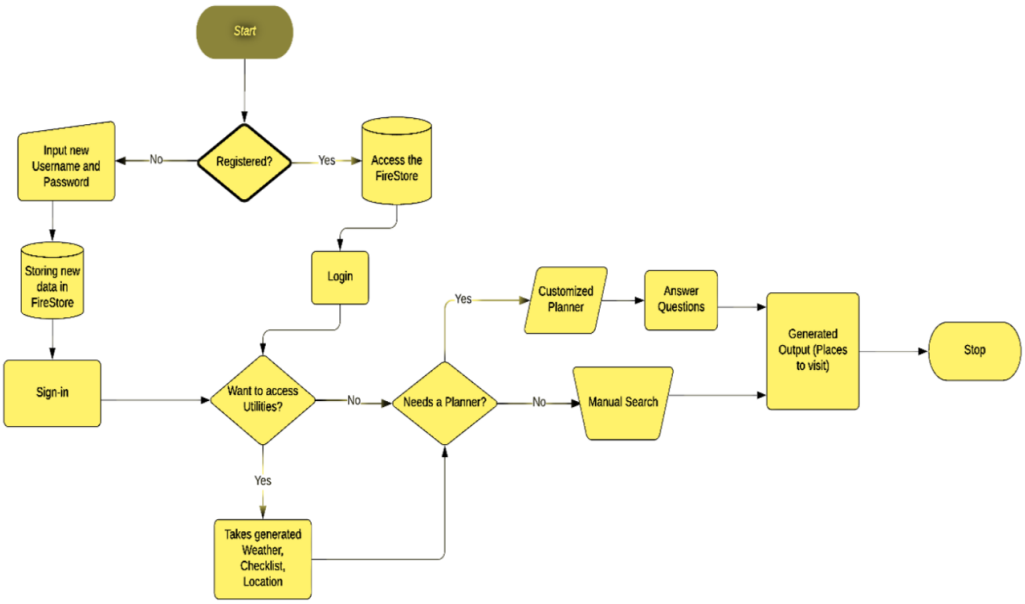


Figure 4.2 shows how the data will flow within our application to achieve the use cases discussed for Figure 4.1. The system shown in Figure 4.1 and 4.2 is to be implemented using Quasar, Firebase and python scripts by first setting up quasar and database, then connecting them both. Thereafter using python scripts for the planner and then connecting it with the frontend Quasar framework.

V. RESULTS

As we connect our firebase and quasar as stated above, we can now access a full-fledged Login/Logout or registration page. After login, we have a lot of functionalities to choose from. We can create our own checklists with reminder and timer. We have a notification reminder for our application, weather updating from real time dataset using Google API, "Find Me" using the same google API and searching places across the world. 'Planner' named feature will be the most important part of our entire project. The main function of this planner will be to suggest the user various places to visit based on different inputs the user gives. The application will ask the user to give their current location and time/ day availability. Based on these inputs we are preparing various parameters of outputs. The application will ask user parameters like the type of site they would like to visit, their budget, etc. with name of places and parameters associated with them, all stored in a database. The database can be updated manually any time. The first part of planner suggests various places to the user by using decision tree algorithm of machine learning using the parameters mentioned. The second part of planner suggest various places and other itinerary sites or shops between initial location and final location using A* algorithm of artificial intelligence. Thus, we have used artificial intelligence and machine learning in our application which can give self-made suggestions to the users, thus, decreasing the efforts by users to plan their own trips. The application will also contain a map where one can find nearby hotels, restaurants, government offices, etc.

Figure 5.1: Login/ Register Section

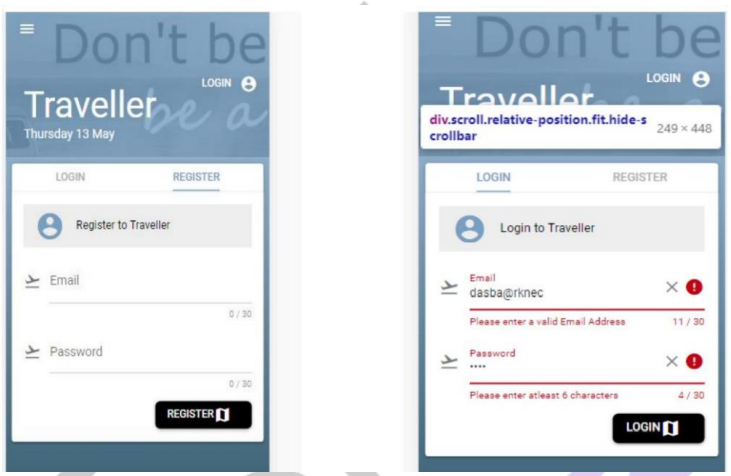


Figure 5.2: Weather, Date, Time and Fine Me

Figure 5.3: Manual Search Bar with Left Drawer

Figure 5.4: Suggestions in Hotels, Restaurants, Places to Visit, etc.

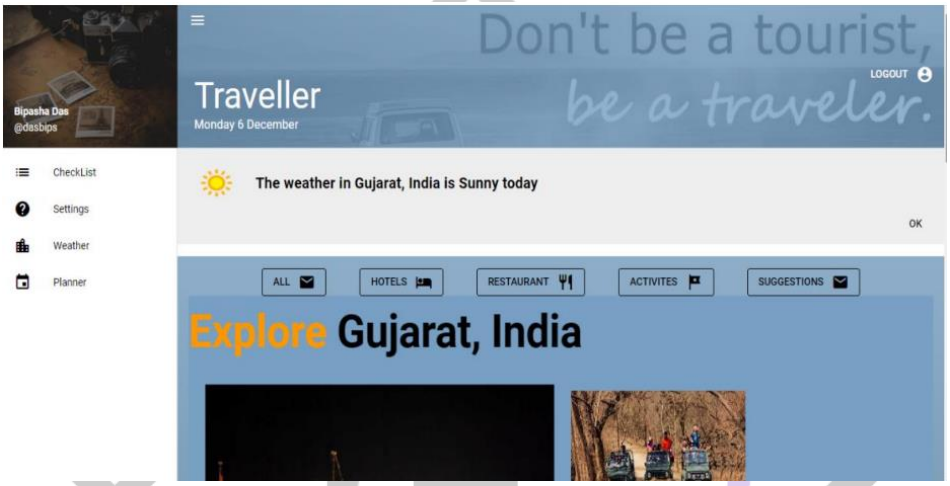


Figure 5.5: Suggesting Places According to Distance when given Current Location in Input

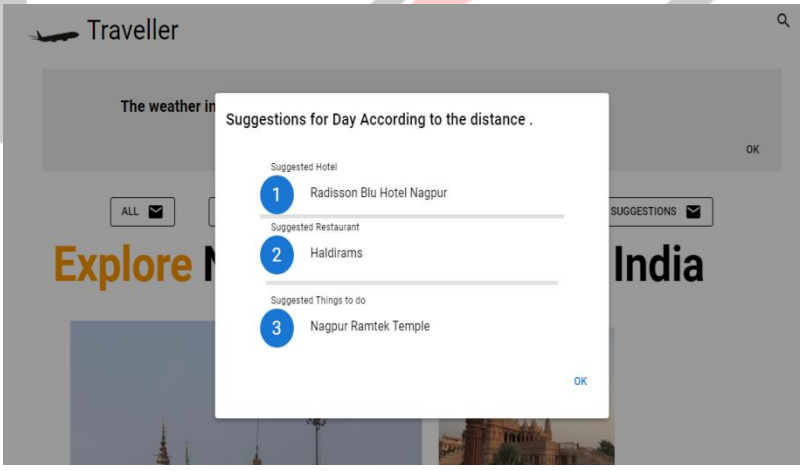


Figure 5.6: Demo Customizable Planner Using ML Decision Tree

Figure 5.7: A* Algorithm Implementation

The users can register or sign-in into our application, as shown in Figure 5.1, hence, proper authentication has been incorporated. As seen in Figure 5.2, weather, date and time along with reminders can be set; accordingly, weather, date and time are coordinated with internet data. The side drawer in Figure 5.4 shows various utilities for user's convenience, like checklists and planner. This application has been incorporated with timely notifications as well. Manual searches (Figure 5.3) with various options and suggestions are available. The next code outputs (Figure 5.6, Figure 5.7) show the Decision Tree algorithm implementation, A* algorithm implementation and manual search backend code, respectively, to give suggested places along a route and customized planner according to user preferences. The Decision tree output shows (not to users) the probability of what can the application show if the user give certain input.

VI. CONCLUSION AND FUTURE SCOPE

This project discusses a system which will help achieving an application which is compatible on all possible platforms. "Traveler Planner" is an application which aims at strengthening the tourism industry and avoids hassle of planning the entire trip by themselves. Other features like manual searching of places and sites gives an extra advantage and strength to this application. The AI and ML functionality makes this application even stronger. A traveler can find the best route to the decided location where they want to visit with different other amenities which falls on the way.

The future scope of this paper can extend up to and beyond the inclusion of cloud deployment of this application. The entire application can be incorporated to work automatically all the time, without the need to run the scripts manually.

VII. ACKNOWLEDGMENT

The satisfaction that accompanies the successful completion of any task would not have been possible without the kind support and help of many individuals. We take this opportunity to express our profound gratitude and deep regards to our project guide Dr. Rakesh Kadu for his exemplary guidance, monitoring and constant encouragement throughout the course of the work.

We also thank Head of the Information Technology Department, Dr. Padma Adane for providing us with all the facilities to pursue our project and for his support and encouragement during project.

We are also grateful to the college for giving us the opportunity to work with them and providing us the necessary resources for the project. We are also thankful to all the staff members of the department, who helped us directly or indirectly in our endeavor and shown keen interest by providing their encouragement.

REFERENCES

[1] STATISTICS OF TOURISM BY GOVT. OF INDIA: https://www.investindia.gov.in/sector/tourism-hospitality

[2] Mohsin, Khalid & Aldabbagh, Omar. (2014). "Design and Implementation an Online Location Based Services Using Google Maps for Android Mobile"

[3] Abdalwhab, Abdalwhab & Almahmoud, Ahmed & Ahmed, Wigdan. (2014), "Android Mapping Application", Computer Science & Information Technology. 4. 11-22.10.5121/csit.2014.4402.

[4] A. Smirnov, A. Kashevnik, N. Shilov, N. Teslya and A. Shabaev, "Mobile application for guiding tourist activities: tourist assistant - TAIS," Proceedings of 16th Conference of Open Innovations Association FRUCT, 2014, pp. 95-100, doi: 10.1109/FRUCT.2014.7000931.

[5] Todd Simcock, Stephen Peter Hillenbrand and Bruce H. Thomas, "Developing a Location Based Tourist Guide Application", ResearchGate, Conference: ACSW Frontiers 2003, 2003 ACSW Workshops - the Australasian Information Security Workshop (AISW) and the Workshop on Wearable, Invisible, Context-Aware, Ambient, Pervasive and Ubiquitous Computing (WICAPUC), Adelaide, South Australia, February 2003

[6] Jannatul Ferdaus, Hang Nguyen, Shamima Nasrin, "Android Application: Travel Guide", Operating System, Android, ResearchGate, Research: September 2015, DOI:10.13140/RG.2.1.4865.4569

[7] Frederico Carvalho Vieira, Adrião Duarte Dória Neto and José Alfredo Ferreira Costa, "An Efficient Approach to the Travelling Salesman Problem Using Self-Organizing Maps", International Journal of Neural Systems Vol. 13, No. 02, pp. 59-66 (2003)

[8] Pooja Singal, R.S. Chhillar, "Dijkstra Shortest Path Algorithm using Global Positioning System", International Journal of Computer Applications (0975 – 8887) Volume 101– No.6, September 2014

[9] Isfandyari-Moghaddam, Alireza. (2007). "Web metasearch engines - A comparative study on search capabilities using an evaluation check-list", Online Information Review. 31. 300-309. 10.1108/14684520710764087.

[10] G. Li and W. Geng, "Research and Design of Meta-search Engine Oriented Specialty", 2010 International Symposium on Intelligence Information Processing and Trusted Computing, 2010, pp. 204-207, doi: 10.1109/IPTC.2010.62.

[11] Nilsson, N. J. (1980). Principles of Artificial Intelligence. Palo Alto, California: Tioga Publishing Company. ISBN 978-0-935382-01-3.

[12] Janikow, C. Z. (1998). "Fuzzy decision trees: issues and methods". IEEE Transactions on Systems, Man and Cybernetics, Part B (Cybernetics). 28 (1): 1–14. doi:10.1109/3477.658573. PMID 18255917.