M.TECH (GEOINFORMATICS)

VIVEK ANAND

2021

GEOSPATIAL ANALYSIS BASED AGRICULTURAL TRADE

A DISSERTATION

SUBMITTED IN PARTIAL FULFILMENT OF THE REQUIREMENTS

FOR THE AWARD OF THE DEGREE

OF

MASTER OF TECHNOLOGY

IN

CIVIL ENGINEERING

(With Specialization in Geoinformatics Engineering)

By

VIVEK ANAND

(2K19/GINF/03)

Under the supervision of

DR. (COL) K.C. TIWARI, PROFESSOR

MULTIDISCIPLINARY CENTRE FOR GEOINFORMATICS

DEPARTMENT OF CIVIL ENGINEERING

DELHI TECHNOLOGICAL UNIVERSITY

(Formerly Delhi College of Engineering)

Bawana Road, Delhi – 110042

August 2021

MULTIDISCIPLINARY CENTRE FOR GEOINFORMATICS
DEPARTMENT OF CIVIL ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi – 110042

CANDIDATE'S DECLARATION

I, Vivek Anand, Roll No. - 2K19/GINF/03, Student of M.Tech (Geoinformatics Engineering), hereby declare that the project Dissertation titled "*Geospatial Analysis Based Agricultural Trade"* which is submitted by me to the Multidisciplinary Centre for Geoinformatics, Department of Civil Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of any Degree, Diploma Associateship, Fellowship or other similar title or recognition.

Place: Delhi VIVEK ANAND

Date: 23rd August 2021

MULTIDISCIPLINARY CENTRE FOR GEOINFORMATICS
DEPARTMENT OF CIVIL ENGINEERING
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi – 110042

CERTIFICATE

I hereby certify that the Project Dissertation titled, "GEOSPATIAL ANALYSIS BASED AGRICULTURAL TRADE", which is submitted by Vivek Anand, 2K19/GINF/03, Multidisciplinary Centre for Geoinformatics, Department of Civil Engineering, Delhi Technological University, Delhi in partial fulfillment of the requirement for the award of the degree of Master of Technology, is a record of the project work carried out by the student under my supervision. To the best of my knowledge, this work has not been submitted in part or full for any Degree or Diploma to this University or elsewhere.

Place: Delhi

Date: 23 Aug 21

SUPERVISOR

Dr. (Col) K.C TIWARI, Professor
Multidisciplinary Centre for Geoinformatics
Department of Civil Engineering
Delhi Technological University
Delhi - 110042

ACKNOWLEDGEMENTS

Foremost, I would like to express my sincere gratitude to my supervisor Prof. K.C. Tiwari for guiding me throughout this research work, and for his patience, motivation, enthusiasm, and immense knowledge. His guidance helped me a lot during this research and writing of this thesis. I feel motivated and encouraged every time I attended his meeting. Without his encouragement and guidance this project would not have materialized.

It gives me great pleasure in acknowledging the support and help of all the members of Multidisciplinary Centre of Geoinformatics Lab at Delhi Technological University who were always there to advice and guide me throughout my M. Tech journey. I am thankful to all the staffs and members of Department of Civil Engineering, our HOD, Department of Civil Engineering and the entire fraternity of Delhi Technological University for giving me the opportunity to study at this institute.

I am thankful to all my classmates and friends who made my stay in this institute, an unforgettable and rewarding experience. Finally, I feel great reverence for all my family members and the Almighty, for their blessings and for being a constant source of encouragement.

ABSTRACT

Agriculture has continued to provide a source of income for the majority of India's population, either directly or indirectly. Over time, the structure of Indian agriculture trade has changed significantly but still the methods are very traditional and no advance technologies like Geospatial Data Science or Artificial Intelligence are used for its betterment. India, which is primarily an agricultural economy, is showing signs of progress in agriculture and related businesses. Agriculture has always been regarded as a valuable tool for economic development in India, as many other major sectors of production depend on it. Despite the government's efforts to double the income of the farmers, there is still a difference between a farmer's investment and the return on that investment.

The laws governing agriculture markets in India have been archaic for a long time. The government has found reasons to over-regulate the agriculture industry, such as price controls, export bans, and restrictions on private trade. Agriculture's growth has been hindered by excessive regulation and a lack of freedom, making it one of the most uncompetitive sector. The government announced a complete reform of the country's farm markets after recognizing the underlying flaws with their functioning. The government then passed the three Farm Bills on September 20, amidst great controversy in Parliament.

This study incorporated the new farms laws passed by the Government of India in September 2020 and developed a traditional mandi type model on an online platform with additional geospatial analysis component to remove the existing constraints in the existing flawed agriculture market which is currently regulated by the states under the Agricultural Produce Marketing Committee (APMC). This study will outline how the usage of geospatial data and location intelligence would prove to be a game changer in the Indian agriculture sector. This study develops a WebGIS application where farmers and buyers will come to execute agricultural trade form any location in the country through an online platform with the help of

geospatial analysis-based information provided to them which they were previously unable to conduct due to the restrictions imposed by APMC regulations and lack of any geospatial analysis-based infrastructure for agricultural trade. In this WebGIS application, the farmers and buyers will be provided with principal geospatial analysis based information like shortest distance between their geolocations, location referenced dynamic map pointing the location of each farmer and buyer, and network analysis based optimal route information via interactive maps along with other necessary features of listing and buying options to let the buyers make a calculated decision while buying the agricultural produce from any part of the country which will in turn enable the farmers to sell their agricultural produce to any part of the country hence, empowering them and subsequently increasing their revenue. Furthermore, the provision for the third-party logistics is also provided where they can register themselves and the users can choose the most suitable third-party logistics agency to handle the agricultural trade's transportation.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

I. **COVID 19** – Corona Virus Disease 2019
II. **APMC** – Agricultural Produce Market Committee
III. **APMR -** Agricultural Produce Markets Regulation
IV. **MSP** – Minimum Support Price
V. **FY**- Financial Year
VI. **FC –** Fixed Charges
VII. **FCI –** Food Corporation of India
VIII. **eNAM** – National Agriculture Market
IX. **FDI** – Foreign Direct Investment
X. **GDP** - Gross Domestic Product
XI. **E-Commerce –** Electronic Commerce
XII. **GPS –** Global Positioning System
XIII. **GIS –** Geographical Information System
XIV. **WebGIS –** Web Geographical Information System
XV. **URL** - Uniform Resource Locator
XVI. **HTTP –** Hyper Text Transfer Protocol
XVII. **ORM –** Object relational Mapping
XVIII. **OSM** - Open Street Map
XIX. **MVC**- Model-View-Controller
XX. **HTML** - Hypertext Markup language
XXI. **SQL** - Structured Query language

CHAPTER 1

INTRODUCTION

1.1 GENERAL

Agriculture in India is one of the significant areas in the country that utilizes the greater part of the all-out labor force (53%) of the nation, for example 243 million people[1] . In any case, it adds to 17.5% of the Gross domestic product (at current costs in 2015-16). The portion of populace relying upon horticulture for its work comprises of landowners, sharecroppers who develop a real estate parcel, and agrarian workers who are utilized on these farms. Rural yield has been unpredictable in the course of recent years, with yearly development going from 8.6% in 2010-11, to - 0.2% in 2014-15 and 0.8% in 2015-16 [2]. The farming area in India is assaulted with difficulties at each period of its horticulture cycle.

The agriculture in India broadly consists of the following phases

A. Crop Planning
B. Buying Seeds
C. Planting
D. Growing
E. Harvesting, Packing and Storing
F. Selling

The major problems that persist in India throughout the above-mentioned phases are:

A. Small and fragmented landholdings
B. Quality of Seeds
C. Manures, Fertilizers and Biocides
D. Irrigation
E. Lack of mechanization
F. Inadequate storage facilities
G. Inadequate transport
H. Scarcity of capital
I. Agricultural Marketing

1.2 COVID-19 PANDEMIC

COVID-19 was originally detected as a respiratory tract infection in Wuhan, China, in December 2019. Symptoms include fever, chills, dry cough, exhaustion, and shortness of breath. This rare viral pneumonia has paralyzed the entire world, causing catastrophic health and economic loss. SARS and MERS-CoV are related to the new coronavirus, although the former's impact is more catastrophic, as evidenced by the exponential rise in infected cases. According to the World Health Organization's International Health Regulations (2005), COVID-19 was declared a Public Health Emergency of International Concern by the end of January.

The world has gone into virtual lockdown as a result of the virus's unprecedented spread, with numerous countries instituting strict screening of possible cases coming into their territory. The limits and closures of economic operations had a significant influence on all sectors of the economy, including agriculture.

1.3 EFFECTS OF COVID-19 ON AGRICULTURE SECTOR

COVID-19, which originated in Wuhan, China (the epicenter), eventually spread over the world and became a pandemic. As of December 6th, 2020, India had already become a hotspot for the virus, infecting 9.6 million people (14.6 percent of global infection).[3], As a result, gross domestic product fell by 23.9 percent in the first quarter of FY 2020–21.[4]. When compared to a weather shock such as drought or flood, or a trade embargo, a pandemic shock might have a higher impact on economies owing to lost human lives. All of these shocks, without a doubt, have an impact on agricultural systems; nevertheless, pandemic shocks have an impact on all sectors of an economy. Droughts tend to be confined, affecting only the associated industry or stakeholders, whereas pandemics alter food demand and supply, affecting the entire supply chain[5]. Similarly, trade embargo shocks affect a specific industry and can be mitigated in the short term with appropriate policy measures. For example, in the event of a supply shortfall caused by drought, internationally connected wholesalers and merchants acquire from alternative sources to minimize negative consequences[5]. Pandemic impact, on the other hand, might be far-reaching and worse, potentially putting a country into recession. The Indian government imposed a nationwide lockdown on March 25, 2020, hurting the economy, especially agriculture, as a protective and preventative measure. The severity of the lockdown is rated >80 on the global stringency index. Following the pandemic, the agriculture sector experienced good growth (3.4 percent in FY 2020–21 Quarter 1: April to

June), but it was smaller than its immediate before quarter growth (5.9% in FY 2019–20 Quarter 4: January to March), which saw a 2.5 percent fall due to the impact of COVID-19.Although credited to a good crop harvest and a relaxation in agriculture-related activity during the lockdown, agriculture's positive growth in the first quarter (FY 2020–21) did not result in a large gain in farm revenue, although it did register 2.3 percent inflation. [3].

1.4 EFFECTS OF REVERSE MIGRATION ON AGRICULTURE SECTOR

In the intensively cultivated north-western plains of India, reverse labor migration resulted in labor scarcity, which hampered harvesting of winter crops like wheat and pulses (November–March)[6]. Second, movement limitations disrupted supply networks, preventing the continuous flow of inputs and outputs for agricultural activity[7]. Figure 1 shows the reverse migration of labors from farmlands during the lockdown. The supply of perishable goods was particularly affected, posing a threat to the food and nutritional security of the most vulnerable members of society[8]. The Indian food system was able to combat the pandemic because to large buffer supplies of rice and wheat, which were augmented by a record harvest in the 2019–20 crop season[9]. Since May 2020, tones of food grains have been wasted in the Food Corporation of India (FCI) storage structures, according to a government assessment[10].

Fig. 1 Labors from agriculture farms reverse migrating during Lockdown.

Food loss was increased by the COVID-19-induced lockdown at the manufacturing, marketing, and distribution levels, as well as wasted at the consumer level. Food commodities

like as milk, vegetables, and fruits, for example, were wasted at farm level due to a lack of demand and logistics, while milk distribution by the Anand Milk Union Limited (AMUL) - India's largest milk cooperative – was also affected[11].

1.5 RESEARCH GAPS

The research gaps found in this study are:

I. In India, agriculture is a state subject hence, throughout the history, the center-state dispute regarding the agriculture reforms has been evident. Same is the case with the implementation of eNAM. This is due to the fact that State government and powerful intermediaries produce constant reluctance in adoption of this system due to their personal gains at stake and which is backed by everlasting poor infrastructure and lack of transportation facilities and other resources. Hence, introduction of new national level agriculture markets which do not fall under the regulation of APMC, and state governments will help provide competitive price to the Farmer.

II. The new farm laws introduced by Govt. of India aims at providing competitive prices to the Farmer to sell their produce so that they can get better price for the same and this can be systematically achieved by creating more marketplace for the famers which resolves the issue of existing agricultural marketing in India.

III. Existing online marketplace for Agriculture trade does not take into consideration the Location information of the Farmer and Buyers which is an important parameter in any type of E-Commerce. By incorporating the geolocation data, not only the Farmer will have wider range of location to sell their produce but will also help in the procurement and fulfilment phase.

1.6 OBJECTIVES

The objective of this research are as follows:

I. To research and analyze about the existing and current practice of agricultural marketing in India and find out its drawbacks and possible solution.

II. To create a Geospatial analysis based agricultural trade system that will take into consideration the geolocation data of Farmer and buyers and on the basis of geospatial analysis and location intelligence it will facilitate the process of agricultural marketing in India.

CHAPTER 2

REVIEW OF AGRICULTURAL MARKETING IN INDIA

2.1 THE GREEN REVOLUTION

This section gives a review of some of the significant advances that have been taken by the Government of India in order to facilitate a better marketing opportunity for the farmers to sell their produce. Agricultural marketing is the last and maybe the most significant period of the agriculture cycle. The misfortune, however, lies in the fact that farmers incur huge losses in this stage regardless of getting a good harvest with minimum post-harvest losses. It is not the difficulties faced by the farmer during the harvesting of the crop that kills them, it is the insensitivity of the markets that forces them to end it all.

In 1960s Green revolution began in India prompting an increment in food grain production, particularly in Punjab, Haryana, and Uttar Pradesh [12]. It was in this stage that the Agricultural Produce Markets Regulation (APMR) Acts and Minimum Support Price(MSP) came into the picture. The idea of MSP is to secure the farmers in the country in circumstances where crop prices fall radically. Between 1960 and 1970, numerous states in the nation controlled the Agricultural Produce Markets Regulation (APMR) Acts under which a organized marketing structure for agriculture was made which were directed by Agricultural Produce Market Committee (APMC), who were also responsible for the creation of proper infrastructure and framing the rules so that the condition of farmers were uplifted.

2.2 THE APMC ACT AND MSP

There are numerous difficulties faced by farmers because of the limitations forced by the APMC Act. The APMC Act engages the state government to outline geographical regions into different notified market areas', which is headed by a market committee, including merchants and farmers, among others. The act instituted by state governments just makes it unlawful to buy, deal, stockpiling and preparing outside the market yard. This is the greatest drawback which has made this model monopolistic, clearly against the interests of the farmers. APMCs have made the market exceptionally controlled, which is the reason various degrees of intermediaries have arisen.

The APMC Act uses APMC as an arm of the state and the market fees as the tax imposed by the state. Aside from this, there are other formal and casual duties/charges/expenses

that are gathered from the entire scope of functionaries, including traders, commissioning agents, warehousing agents, loading agents, etc. These tolls in the end amount to heavy sums and are majorly incurred by farmers.

According to *Shanta Kumar Committee Report, 2014* just 6% of farmers get MSP, while the leftover 94% depends on the market. As evident, the MSP system is flawed, possibly more than APMC markets. It creates a floor price for the commodity below which it can't be bought from the farmers. Despite the fact that it has shielded farmers against price instability, it still has an enormous amount of shortcomings[13].

Swaminathan committee, 2004 discussed the expense of cultivation at three levels, in particular: *A2* that incorporated all types of money expenditure; *FL* that included assessed cost of work to every individual from the farmer's family; and *C2* which comprises of assessed land lease rent and the amount interest on the money loaned for farming. CACP, then again, adds both A2 and FL but not C2. Farmers contend that they ought to be given MSP by also considering C2. This accounts for a huge difference between the MSP sought by the farmers and the MSP given by the government[14].

However, this point doesn't clarify why only 6% of the farmers get MSP, and a large portion of the produce is sold in APMC markets at a loss. This is clarified by the way that cultivators don't get instant payment under MSP regardless of the sum. It takes weeks and even months for the payment to be credited in their account. Payment urgency forces most of the farmers to sell their grains outside the MSP framework.

Another significant inadequacy lies in the way that the system doesn't cover all harvests, which implies that all farmers don’t have access to it. Additionally, acquisition activities for paddy and wheat differ across states and experience the ill effects of infrastructural barriers.

Moreover, the MSP system has prompted mono-cropping practice, for example overproduction of paddy and wheat. It has made the issue excess stocks as well as led to other disastrous impacts. Farmers are cultivating paddy even in those areas which are not appropriate for paddy cultivation.

It is now clear why there exists an imbalance between MSP system and the APMC markets, and why farmers are ultimately suffering with losses at both the ends.

Since, APMC Act constrains farmers to sell their agricultural produce only to the nearest designated APMC market which prompted the exploitation of farmers with respect to price offered, hefty tax imposed and the intermediaries creating a monopsony market. In the mandis, trader (government and private) needs to pay market fee or mandi charges, rural development fees and commissions to middlemen (known as the arhatiya commission). As

indicated by the *Food Corporation of India*, market fee/mandi charge is charged at 8.5% in Punjab and 6.5% in Haryana. Of this, Punjab charges 3% market fee/mandi charge, 3% rural development fee and 2.5% commission. In Haryana, the figure is 2% market expense fees, 2% rural development fee and 2.5% commission.

During wheat procurement in the year 2020-21, a market tax of 3.6% was levied in Rajasthan. Moreover, arhatiya or society commission of 27 rupees for each quintal of wheat bought was additionally paid. There was a market tax of 2.5% for wheat acquirement in Uttar Pradesh and 2.2% in Madhya Pradesh. Table 1 shows the above-mentioned details.

Table 1:Market Fees/Taxes/Charges on Wheat Procurement in Selected States during RMS 2020-21

State	Market Fee/Mandi charge (%)	Commission/other charges (%)	Rural Development fee (%)	Total
Punjab	3.0	2.5	3.0	8.5%
Haryana	2.0	2.5	2.0	6.5%
Rajasthan	1.6	2.0	-	3.6%+FC*
Uttar Pradesh	2.0	0.5	-	2.5%+FC*
Madhya Pradesh	2.0	0.2	-	2.2%+FC*

Source *– Price Policy for Rabi Crops (2020-21), Commission for Agriculture Costs & Prices*

During paddy acquirement in the year 2019-20, market tax of 2.5% was imposed in Uttar Pradesh, 2.2% in Madhya Pradesh, 2.2% in Chhattisgarh, 1.05% in Maharashtra, and 1% in Andhra Pradesh. Alongside this, a general public commission of around 32 rupees for every quintal of paddy is additionally to be paid to the trade board of these states. mandi tax charge was the least in Kerala at 0.07%. In Karnataka, agrarian products draw in a 3.5% commission charge. Table 2 shows the above-mentioned details.

Table 2: State-wise Fees/Taxes/Charges levied on Rice Procurement (2019-20)

State	Market Fee/Mandi charge (%)	Rural Development fee (%)	Commission/Other charges (%)	Total
Andhra Pradesh	1.0	-	-	1%+FC*
Assam	1.0	-	-	1.0%
Chattisgarh	2.0	-	0.2	2.2%+FC*
Haryana	2.0	2.0	2.5	6.5%
Karnataka	-	-	3.5	3.5%
Kerela	-	-	-	0.07%
Maharashtra	1.05	-	-	1.05%+FC*
Madhya Pradesh	2.0	-	0.2	2.2%+FC*
Odisha	2.0	-	-	2%+FC*
Punjab	3.0	3.0	2.5	8.5%
Telangana	1.0	-	-	1%+FC*
Uttar Pradesh	2.0	-	0.5	2.5%+FC*
West Bengal	0.5	-	-	0.5%+FC*

Source–*Price Policy for Kharif Crops (2020-21), Commission for Agriculture Costs &Prices*

2.3 ENAM – ONLINE NATIONAL AGRICULTURE MARKET

The Government on India in 2016, introduced an online agriculture trading platform named National Agriculture Market or eNAM, whose aim was to provide better value to farmers for their agriculture produce and facilitates smooth marketing for the same. There are around 2,477 principal regulated markets (APMCs) and 4,843 sub-market yards controlled by separate APMCs in India [15]. However, approximately only 1,000 APMC mandis have been connected with eNAM. The reason for the slow integration of the APMCs with eNAM is the fact that there is huge resistance for online trading both from the government and various other stakeholders involved in this process, including intermediaries with powerful backgrounds. Speaking about the state governments, since market fee is a huge source of income for them, most of the states have been reluctant in introducing eNAM in their respective states from the beginning.

Majority of the states have rolled out separate improvements in their APMC acts to get markets integrated with eNAM, yet the miserable part is that the traders actually fail to give a competitive price to the farmers for their agrarian produce in their state. There is no proof till date proposing that the new online framework (eNAM) has increased farmers' income altogether in any of the region in India.

Agriculture and horticultural produce production are estimated to be around 590 million tons per year. In 2017-18, however, the traded volume on eNAM was only 10.9 million tons. Farmers sold 22.5 million tons of agricultural commodities worth Rs 58,930 crore through eNAM between December 2018 and December 2019, benefiting 63.75 lakh farmers out of the 12,757,126 who enrolled. [16].

eNAM has failed to address existing APMC issues such as a weak supply and market chain, a scarcity of covered and open auction platforms, common drying yards, grading facilities, and electronic weighbridges, and a lack of an accurate and timely market information system, among others[16].

The overall number of operating holdings (individual farmers) was expected to be 138.35 million, with 159.59 million hectares under cultivation. The holding's average size was judged to be 1.15 hectares. It meant that each farmer has an average of 1.15 hectares of cropland[17]. This means that majority of farmers in India have marginal or small land holdings hence, they are exposed to more uncertainty about the price of their produce and also, they cannot afford any technological investment like precision agriculture.

2.4 THE NEW FARM BILLS, 2020

In September 2020 the Government of India passed three farm bills namely *The Farmers' Produce Trade and Commerce (Promotion and Facilitation) Act, 2020, The Farmers (Empowerment and Protection) Agreement on Price Assurance and Farm Services Act, 2020, The Essential Commodities (Amendment) Bill, 2020*. The government authority guarantees these demonstrations will change Indian agribusiness area and draw in private speculation. As indicated by Indian Agriculture minister, Narendra Singh Tomar these Bills are to guarantee that Farmers gets better return for their produce without being dependent upon the guidelines of mandis and will expand competition and attract private investment which will help in the advancement of homestead framework and create business.

The Farmers' Produce Trade and Commerce (Promotion and Facilitation) Act, 2020 tries to offer opportunity to farmers to sell their produce outside the told APMC market yards

(mandis). It is pointed toward working with profitable costs through serious elective exchanging channels. It will open more choices for farmers, reduce marketing costs, and help them get better prices. It will also help farmers of regions with surplus produce to get better prices and consumers in areas with shortages at lower prices.

The Farmers (Empowerment and Protection) Agreement on Price Assurance and Farm Services Act, 2020, tries to give farmers the option to go into an agreement with agribusiness firms, processors, wholesalers, exporters, or huge retailers for the offer of future cultivating produce at a pre-agreed price. It plans to move the danger of market flightiness from farmers to sponsors. Besides giving them access to modern tech and better inputs, it also seeks to boost farmer income by reducing the cost of marketing.

The Essential Commodities (Amendment) Act, 2020 aims to remove grains, pulses, oilseeds, onion, and potatoes off the list of essential commodities, as well as stock holding limitations, save in ‘exceptional situations' such as war, hunger, extraordinary price rises, and natural calamity. It aims to promote price stability to the farm industry while also attracting private investment/FDI.

2.5 GAPS AND DEFICIENCIES IN THE EXISTING SYSTEM

In India, agriculture is a state subject hence, throughout the history, the centre-state dispute regarding the agriculture reforms has been evident. Same is the case with the implementation of online national agricultural market named eNAM.

This is due to the fact that State government and powerful intermediaries produce constant reluctance in adoption of this system due to their personal gains at stake and which is backed by everlasting poor infrastructure and lack of transportation facilities and other resources. Hence, introduction of new national level online agriculture markets which do not fall under the regulation of APMC, and state governments will help provide competitive price to the farmers.

Existing online marketplace for Agriculture trade does not take into consideration the Location information of the Farmers and Buyers which is an important parameter in any type of E-Commerce. By incorporating the geolocation data, not only the farmers will have wider range of location to sell their produce but will also help in the procurement and fulfilment phase

CHAPTER 3
RELEVANCE OF GEOSPATIAL ANALYSIS IN AGRICULTURAL TRADE

Geospatial analysis makes use of data from a variety of sources, including GPS, position sensors, social media, mobile devices, and satellite imaging, to create data visualisations for better understanding phenomena and identifying trends in complex human-environment relationships. This geo-referenced data can be used to almost any event that occurs on the planet. Maps, graphs, statistics, and cartograms are examples of visualisations that show historical and contemporary shifts. Predictions will be easier and more accurate as a result of this.

Geospatial analysis provides context to traditional data by adding timing and location, allowing for a complete picture of events. Easy-to-recognize visual patterns and maps offer insights that could have been lost in a huge spreadsheet.

The following are some of the advantages of geospatial analytics:

- **Engaging insights** – Seeing data in the context of a visual map makes it easier to comprehend how events are unfolding and how to respond to them.

- **Better foresight** – Being able to see how spatial conditions are changing in real time can help an organisation better prepare for change and plan for the future.

- **Targeted solutions** – Understanding why some areas and countries, such as the United States, are more successful for business than others is aided by seeing location-based data.

In this study, geolocation and distance based geospatial analysis is utilised to provide insights to the farmers and buyers in order to enhance the methods of traditional agricultural trade existing in our country.

The majority of GIS applications rely on distance analysis. Distance is a measurement of how far one thing is from another in its most basic form. The shortest distance between two points can be measured using a straight line. There are, however, other factors to consider. For example, if there is a roadblock along the way, you must take a detour around it, which increases the overall distance travelled. Another factor to consider is the surface roughness,

which might increase the actual distance travelled when the terrain varies from flat to rugged. There are numerous more elements that can influence the amount of work required to move from one spot to another. The steepness of the terrain, whether you'll be travelling with or against the wind, your means of transportation, and the sorts of land cover you'll be traversing are all aspects to consider. If the computations are done using a planar or a geodesic approach, there is also a difference in distance measure.

Distance is the most crucial attribute to get from a set of geocodes (longitude and latitude). In the development and resolution of real-world situations, distance measures are critical[18]. The following are the most commonly used metrics and algorithms for calculating distance:

A. Euclidean Distance

B. The Great Circle Distance

C. Manhattan Distance (Taxicab Distance)

3.1 EUCLIDEAN DISTANCE

The simplest and most obvious manner of describing the distance between two points is the Euclidean distance. This is also known as the Pythagorean distance because it is a formalisation of the "Pythagorean" theorem[19].

Euclidean distance is best for emergency situations where helicopters can fly in a straight line to sites like hospitals since it ignores real-world structures. Another recorded example is trip planning, where you merely need to know which sights are close to each other.

$$d(\boldsymbol{p},\boldsymbol{q}) = \sqrt{\sum_{i=1}^{n}(q_i - p_i)^2}$$

$$where\ \boldsymbol{p},\boldsymbol{q} = two\ points\ in\ Euclidean\ space$$
$$q_i, p_i = Euclidean\ vectors$$

$$n = n - space\ (for\ geospatial\ applications, n = 2, representing\ longitude\ and\ latitude)$$

3.2 THE GREAT CIRCLE DISTANCE

The great circle distance, unlike the Euclidean distance, takes into account the fact that two points are on the surface of a sphere. [20]. Figure 1 shows great circle distance on a globe

The Haversine Formula is used to calculate the Great Circle Distance as follows: [20]:

$$a = sin^2\left(\frac{\Delta lat}{2}\right) + \cos(lat_1) \times \cos(lat_2) \times sin^2\left(\frac{\Delta long}{2}\right)$$

$$c = 2 \times \arctan\left(\frac{\sqrt{a}}{\sqrt{1-a}}\right)$$

$$d = R \times c$$

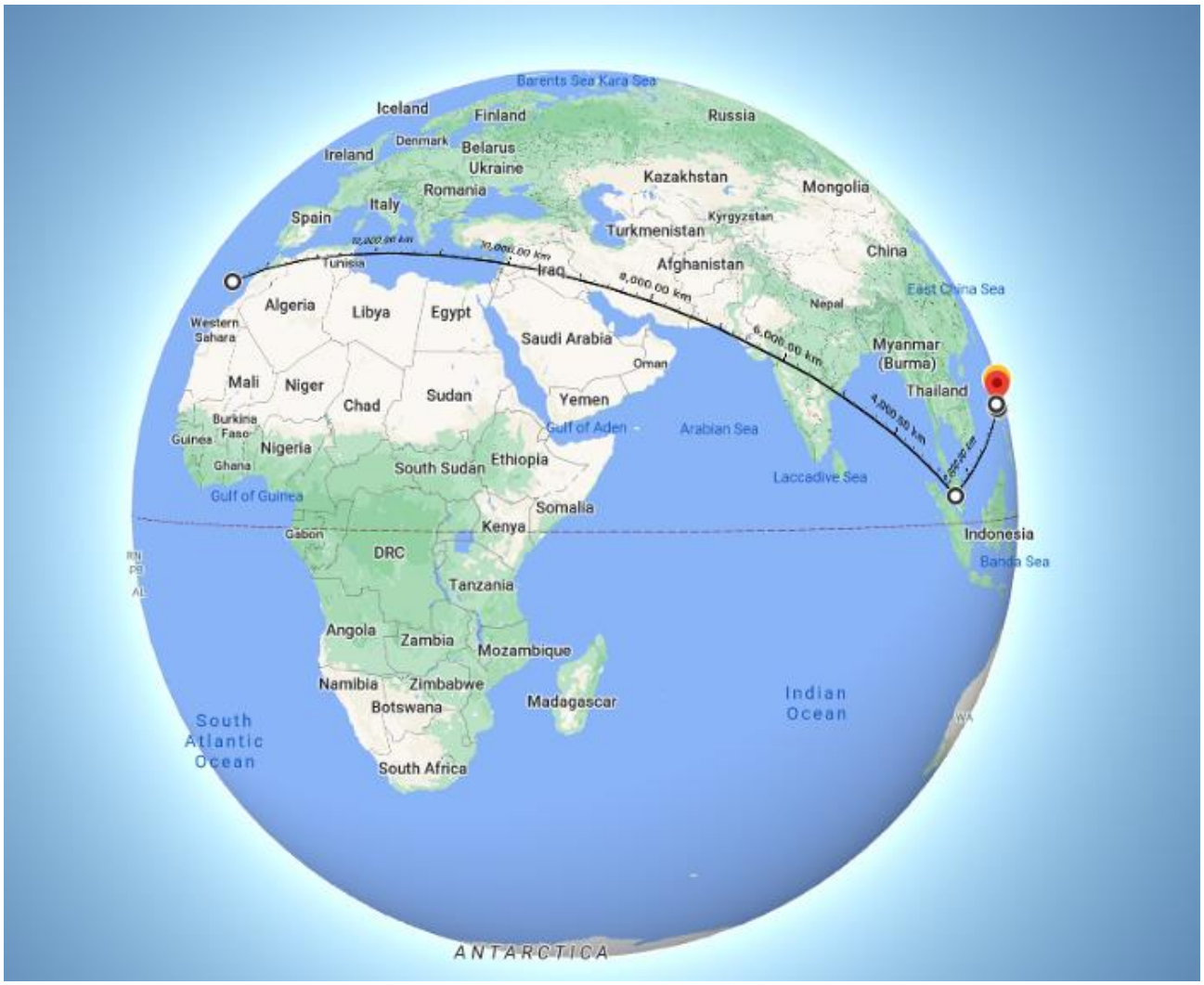

Fig. 2 The Great Circle Distance

3.3 MANHATTAN DISTANCE (TAXICAB DISTANCE)

The Manhattan Distance is a distance between two locations that takes into account the map's perpendicular arrangement. Most GPS units utilise this method to determine distances since it takes into consideration the grid pattern of sites. If driving distance is the variable of concern, this is preferable to both Euclidean and Great Circle distances. This is why it's also known as the taxicab distance, because it's the distance that a taxicab would go in a grid-layout location. Figure 2 explains the Manhattan distance visually[21].

The formula for a Manhattan Distance is as follows:

$$d = |x_1 - x_2| + |y_1 - y_2|$$

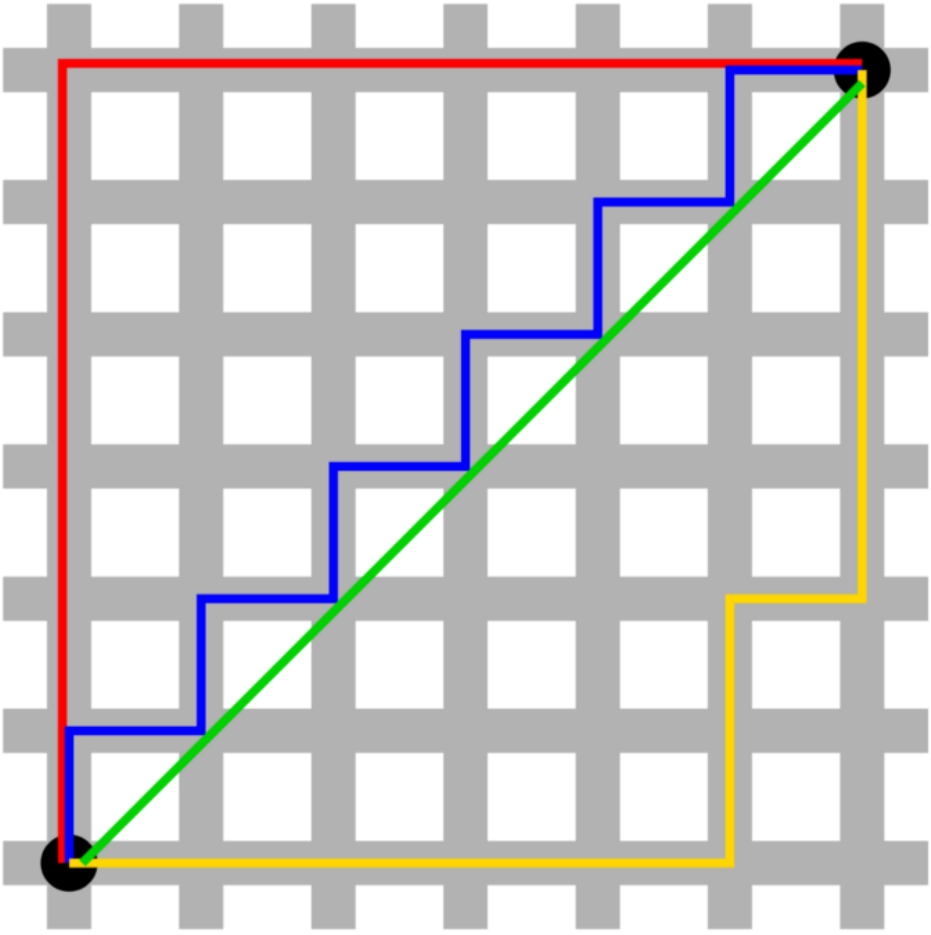

Fig. 3 The green line represents the calculation of Euclidean Distance, the blue line represents the calculations made by Manhattan Distance.

3.4 DIJKSTRA'S ALGORITHM

The shortest path analysis is the essential problem in network analyses, which is an important function of GIS. The shortest distance between the farmer and the buyer is calculated using Dijkstra's technique for network analysis in this study. Dijkstra's algorithm (also known as Dijkstra's Shortest Path First algorithm, or SPF algorithm) is a method for determining the shortest pathways between nodes in a graph, such as road networks. To determine accessibility, the ground realities and barriers such as rivers and mountain ranges are taken into account[22].

Dijkstra's original variant discovered the shortest path between two nodes; however, a more frequent variant fixes a single node as the "source" node and finds shortest pathways from the source to all other nodes in the graph, resulting in a shortest-path tree. Figure 3 represents the Djikatras algorithm in graphical form

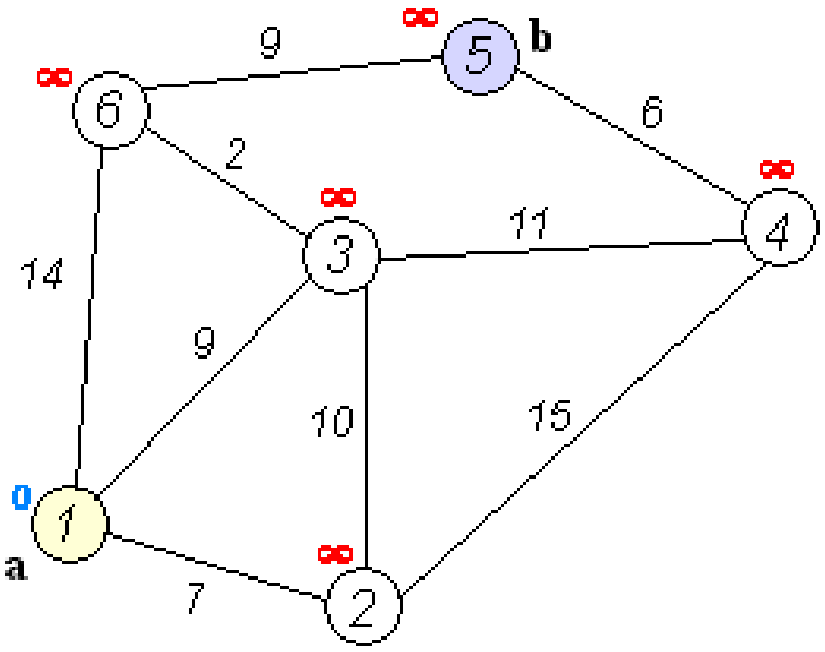

Fig. 4 Dijkstra’s algorithm to find the shortest path between point a & b.

In this study, the address information of both farmers and buyers was geocoded on a geographically referenced map and network analysis was done using Dijkstra’s algorithm and the shortest distance was calculated and displayed on the map when required by the user.

CHAPTER 4

GEOSPATIAL ANALYSIS BASED AGRICULTURAL TRADE

4.1 INTRODUCTION

The second objective of this research is to create a geospatial analysis based agricultural trade system that will take into account an additional component of geographically referenced data above the traditional information required for agricultural trade between farmers and buyers and on the basis of geospatial analysis and location intelligence it will facilitate the process of agricultural marketing in India.

The Government of India launched eNAM, an online national agriculture trade platform. The literature review on eNAM suggests that this platform has miserably failed to deliver on its expectation due to the center-state dispute in its implementation and the involvement of powerful intermediaries which is explained in the previous chapter. Also, it does not take into consideration the Location information of the Farmers and Buyers which is an important parameter in any type of E-Commerce. By incorporating the geolocation data, not only the farmers will have wider range of location to sell their produce but will also help in the procurement and fulfilment phase.

This chapter outlies the process involved in carrying out the second objective, which involves technologies used for the development of the Web GIS application for agricultural trade and the methodology followed. It is covered under the following heads:

A. Data
B. Background and Technologies used
C. Methodology

4.2 DATA

This research required the data to be collected from the farmers broadly about the type, quantity, and quality of the crops they produce which can be traded via this Web GIS Application for agricultural trade and most importantly their geolocation. The specific data to be collected from each user is thoroughly explained in the section 4.4.1 of this chapter. Since, this research was carried out amidst the ongoing COVID 19 pandemic and lockdown in the country therefore the actual data from the farmers and buyer was impossible to collect. The efforts were made to collect the data from competent authorities listed below and the copy of

which is attached in the Appendix 1

1. Bharatiya Kisan Union
2. All India Kisan Sabha
3. All India Kisan Sangharsh Coordination Committee (AIKSCC)
4. Karnataka Rajya Raitha Sangha
5. Ministry of Agriculture of various farming states
6. Department of food supply and chain of the *Food Corporation of India (FCI),*
7. eNAM authorities
8. District Magistrates of various farming districts

Synthetic Data

A dataset of farmer's and buyer's necessary details was generated and used as a proof of concept that this Web GIS Application can function without any difficulties when gone into production. The synthetic data generated for this study is attached in the Appendix 1

4.3 BACKGROUND AND TECHNOLOGIES USED

WebGIS is a web-based version of a geospatial information system (also known as a geographic information system). A server and a client exchange information, with the server being a Geospatial Information System (GIS) server and the client being a web browser, mobile application, or desktop application. Clients can find the server on the web owing to its unique Uniform Resource Locator (URL)[23].WebGIS places GIS in the hands of the public at large. It eliminates the requirement for specialised applications. It serves as a platform for connecting GIS with other business systems, as well as facilitating cross-organizational communication. WebGIS enables businesses to effectively handle all of their geographic data.

This research involves the application of Web GIS in the development of the geospatial analysis based agricultural trade system. The various technologies used for its development are listed below.

4.3.1 ASP.NET Framework

ASP.NET is a web development platform that includes a programming paradigm, a comprehensive software infrastructure, and a number of services for developing robust web applications for both PCs and mobile devices[24].

ASP.NET is built on top of the HTTP protocol, and it makes use of HTTP commands and policies to establish bidirectional communication and cooperation between the browser

and the server[24].

4.3.2 Entity Framework ORM

In computer science, object–relational mapping (ORM) is a programming approach that uses object-oriented programming languages to transform data between incompatible type systems. This effectively provides a "virtual object database" that can be accessed from the programming language itself. It automates the conversion of data from relational database tables to objects that may be used in application code. ORMs give a relational database a high-level abstraction[25].

Microsoft's Entity Framework is an open-source ORM framework for.NET applications. It allows developers to work with data using domain-specific objects rather than the underlying database tables and columns where the data is kept. When working with data, developers can use the Entity Framework to work at a higher level of abstraction, allowing them to construct and manage data-oriented applications with less code than traditional programmes. Fig 4. Illustrates where the Entity Framework fits into this Web GIS application.

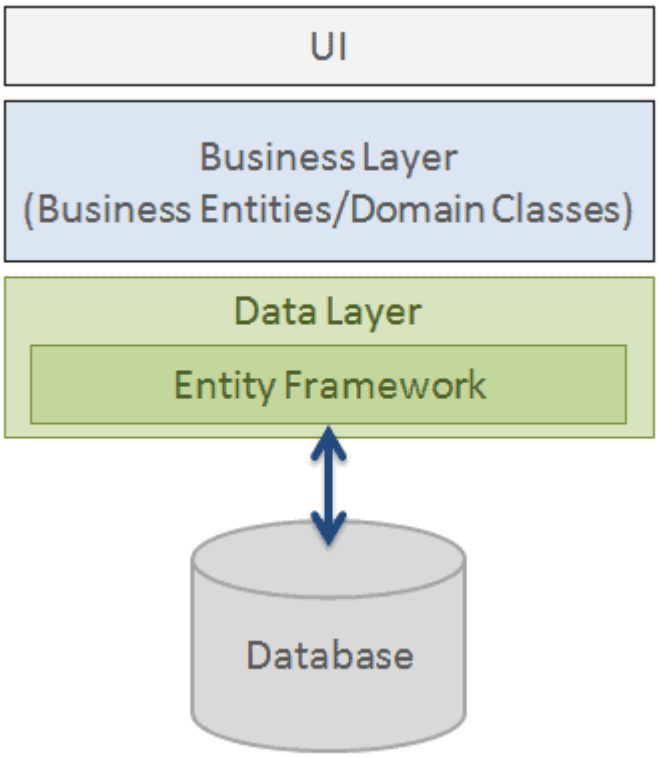

Fig. 5 Entity Framework ORM representation

4.3.3 ASP.NET MVC

Model-View-Controller (MVC) architecture is the foundation of the ASP.NET MVC web framework. Model–view–controller is a software design pattern that divides related program logic into three interrelated pieces and is often used for building user interfaces. This is done to distinguish internal information representations from how information is presented to and accepted by users[24].

The Model-View-Controller (MVC) architectural pattern divides an application into three logical components: model, view, and controller. Each of these components is designed to handle specific parts of application development.

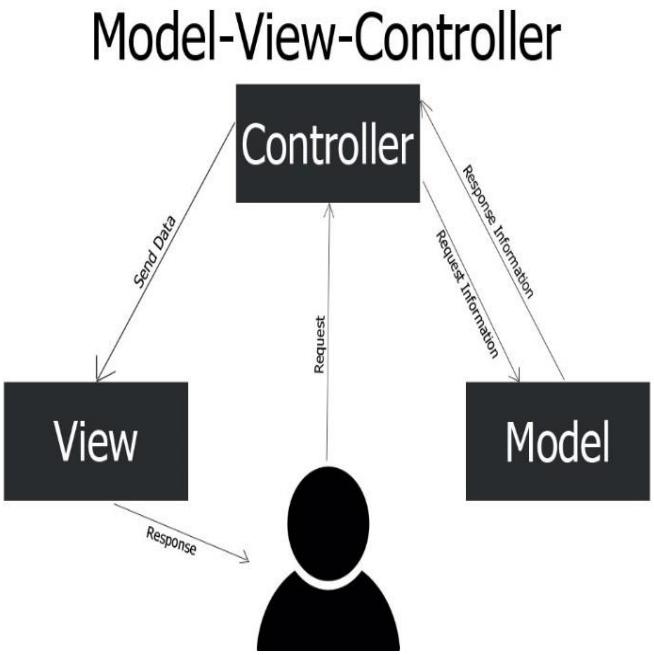

Fig. 6 MVC System Design

Model -All data-related functionality that the user engages with is represented by the Model component. This can be the data being transmitted between the View and Controller components or any other data related to business logic. A customer object, for example, will retrieve customer information from a database, change it, and either update or output the data back to the database.

View-All of the application's UI functionality is handled by the View component. The Customer view, for example, will comprise all of the UI components that the final user interacts with, such as text boxes, dropdowns, and so on.

Controller- Controllers serve as a link between the Model and View components, processing all business logic and incoming requests, manipulating data using the Model, and interacting with Views to produce the final output. The Customers controller, for example, will handle all interactions and inputs from the Customer View and use the Customer Model to update the database. The Customer data will be seen using the same controller.

4.3.4 MS SQL Server

Microsoft SQL Server is a relational database management system that Microsoft has created. It is a database server, which is a software product whose principal role is to store and retrieve data as required by other software applications, which may run on the same computer or on a networked computer[26].

In this study, MS SQL Server is utilized to develop the geodatabase, which will be used to hold both spatial and non-spatial characteristics. Information regarding the physical location and shape of geometric objects is represented by spatial data. These might be simple point locations or more complex items like countries, roads, and lakes.

The geometry data type and the geography data type are two spatial data types supported by SQL Server.

a) Data in a Euclidean (flat) coordinate system is represented by the geometry type.

b) Data in a round-earth coordinate system is represented by the geography type.

In SQL Server, both data types are implemented as.NET common language runtime (CLR) data types.

4.3.5 Open Street Map

The Open Street Map (OSM) project is a collaborative effort to develop a free, editable world map. The key outcome of the project is the geo-data that underpins the map. The inception and growth of OSM was spurred by restrictions on the use and availability of map data in many parts of the world, as well as the introduction of low-cost portable satellite navigation equipment[27]. OSM data can be used for a variety of things, including creating paper and electronic maps, geocoding addresses and place names, and route planning[27].

All the geospatial analysis in this research is visually represented on the map using open street map. It was chosen for this study since it is a free and open-source project.

4.4 METHODOLOGY

This study aims at the creation of a Geospatial Analysis Based Trade System between the Farmers and Buyers, this is done by creating a traditional Mandi type model on an online platform where farmers and buyers will come to execute the trade but with an additional Geospatial component which will take into the account the Geolocation information of the users to provide the users an enhanced way to carry out agricultural trade. This geospatial component which is not considered previously in any kind of agriculture trade will prove to be a vital aspect as all the E-commerce industries are realizing the power of geographically referenced data.

The purpose of this model is to help the small farmers, buyers and agricultural businessman in our country to locate each other and carry out trade of agricultural products in different parts of the country which they were previously unable to execute due to lack of any geospatial based infrastructure for agriculture trade.

This chapter outlines the methodology followed for the development of proposed geospatial analysis-based trade system. This will be discussed under following heads:

A. Database Design

B. System Architecture

C. Use case diagrams

D. Workflow Architecture

4.4.1 Database Design

Based on the assessment of the geospatial analysis-based trade system, it is realised that the database will consists of two types of attribute:

I. ***Spatial attribute***, which consists of geolocation attribute of various end-users of the system, the processed geospatial information and the maps generated on the basis of geospatial analysis.

II. ***Non-Spatial attribute,*** which consists of attribute related to farmers, buyers and the logistic company like type of crops, its amount etc.

Further, Spatial data should have a geographical reference system, which for India is Indian Geodetic Datum which uses the Everest spheroid as reference surface which defines the

shape of the Earth and the projection system used is UTM WGS 84. The non-spatial data should be systematically arranged.

The database design for this work can be covered under following heads:

A. Database for basic farmer's information

B. Database for basic buyer's information

C. Database for third party logistic' information

D. Database for crops added by the farmer

A. Database for basic farmer's data

The farmers are the main pivot of this study and to facilitate the online agriculture trade system between the farmer and buyer, a database dedicated for storing the spatial and non-spatial attribute of the farmer is created which will take the following data from the farmer:

- Email id
- Name
- Contact No.
- Date of Birth
- Gender
- Aadhaar Number
- PAN No.
- Address:
  - Village
  - District
  - State
  - Pin code
- Geolocation (Lat/Long)

B. Database for basic buyer's data

The buyer is the endpoint of the system who will purchase the agricultural produce based on his requirements from the farmer, a database dedicated for storing the spatial and non-spatial attribute is created by taking the following data from the buyer:

- Email id
- Name
- Contact No.
- Date of Birth
- Gender
- Aadhaar Number
- PAN No.
- Address:
  - Village
  - District
  - State
  - Pin code
- Geolocation (Lat/Long)

C. Database for third party logistic' data

This segment of the geospatial analysis based online agricultural trade system will act as the connection point between the farmer and the buyer. The third party logistic is responsible to facilitate the execution of sale/purchase/trade between the farmers and buyers. A database dedicate for storing the necessary data from the Third Party Logistic company is created that will take the following information from the Third Party Logistic side:

- Contact Details:
- Name
- Email Id
- Phone
- Address
- Geo Location

Order Fulfillment Method

- Transportation Type – i) Refrigerated Transportation
  ii) Non-Refrigerated Transportation
  iii) Both
- Shipping Method – i) Roadways
  a. Vehicle type
  b. Capacity
  ii) Railways
  iii) Airways

Rate Catalogue

- Rate per Kilogram (Rs)
- Rate per Kilometers (Rs)

D. Database for items to be traded by the farmer

A crop is defined as a plant or animal product that can be harvested and grown extensively for revenue or livelihood [28]. The farmer after plantation and cultivation of his crops goes to sell his produce in the nearest mandi available to him and to make this system applicable in this study, a database for storing the information about the crops that the farmer has for trade is created that will take the following information from the farmer:

- Crop Name
- Trade Price (per Kg)
- Quality Type
- Maximum Trade Quantity
- Image of the Crop
- Crop Description

The Fig X pictorially depicts all the entities stored in the database explained above and their relationship with each other.

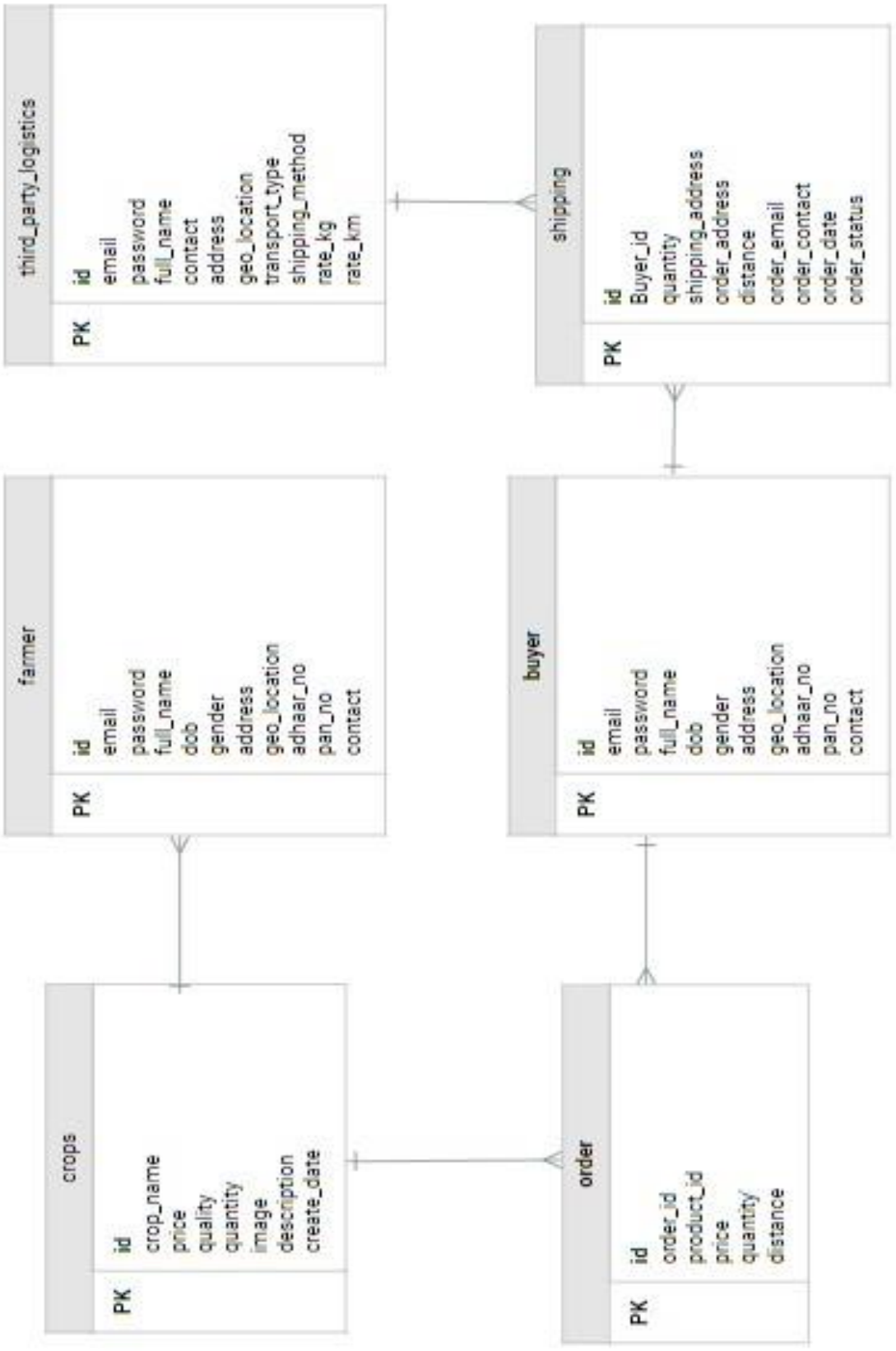

Fig. 7 Database Entity Relationship Diagram

4.4.2 System Architecture

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements.

This system basically consists of five modules:

A. Admin

B. Farmer

C. Buyer

D. Third Party Logistics

E. Services

F. Financial Module

A. Admin

The admin module acts as the super user of the system which hold various functionalities like:

I. Activating and deactivating Farmer account

II. Activating and deactivating Buyer account

III. Activating and deactivating Third Party Logistic account

IV. Regulating the application

V. Admin can view all the list of buyer, farmer and the crops available for trade.

VI. Admin will be able to make any necessary change to the system in case of any discrepancy

B. Farmer

Farmer module consist of the functionalities that the farmer can perform which includes:

I. Adding of crops

II. Editing/deleting of crops

III. Inventory management

IV. Viewing received orders

C. Buyer

Buyer module consist of the functionalities that the buyer can perform which includes:

I. Viewing the crops available for trade

II. Executing trade

III. Choosing the logistic service

IV. Viewing placed orders

D. Third Party Logistics

Third party logistic module consist of the functionalities that the logistic end can perform which includes:

I. Viewing the request.

II. Carrying out the logistic of the order.

E. Services Module

This module lists out the Services that the users will get in this geospatial based agricultural trade system:

I. Geospatial analysis-based information.

II. Alert Services (SMS and Email

I. Geospatial Analysis-Based Information.

The geosptial services provided by this GIS based application are:

1 Provide a location referenced map-based information to the buyer which points out all the location from where the farmers are available to sell a particular crop.

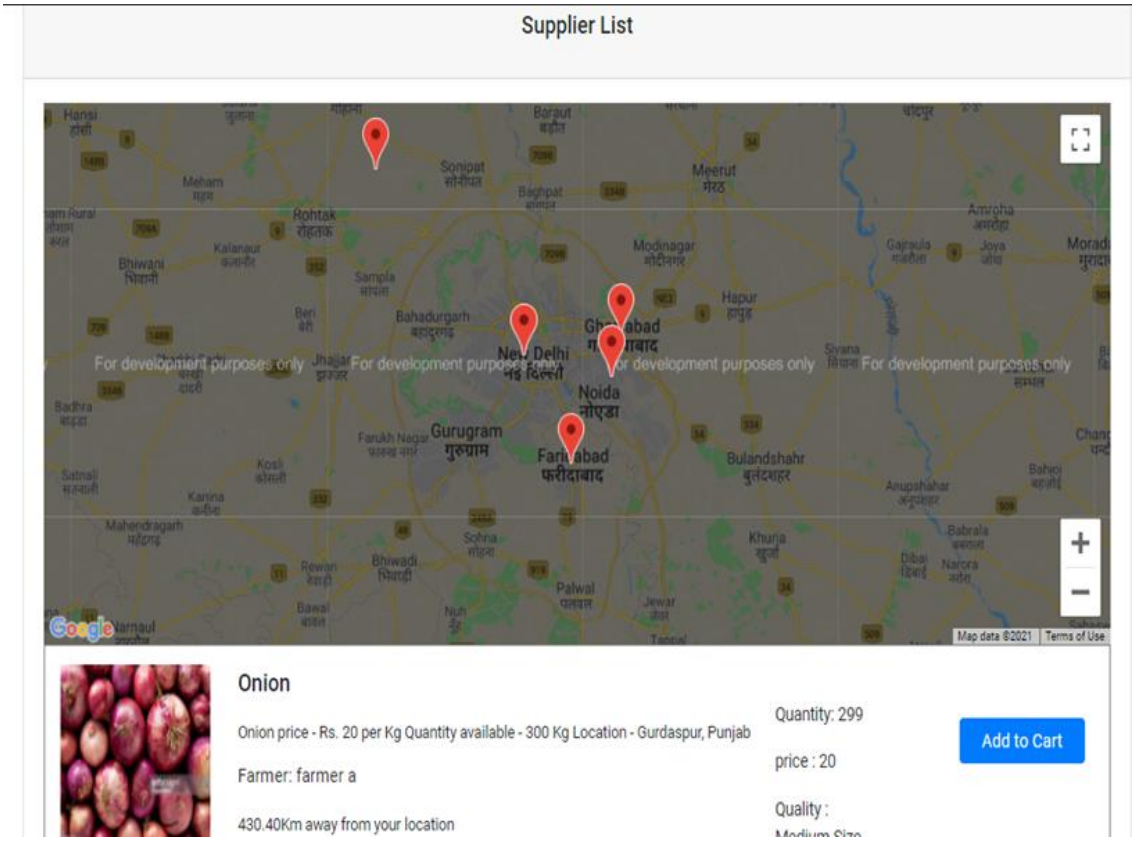

Fig. 8 Seller's Location Referenced Map

2. Using Geospatial network analysis, it will show the shortest distance between the buyer and farmer so that the buyer can take calculated information before placing the order and choosing the optimum third party logistic for its delivery.

3. The distance metric obtained using geospatial analysis will be further used by the third party logistic for carrying out the delivery of the crops.

Fig. 9 Network Analysis Based Information.

II. Alert Services (SMS and Email)

These are the following SMS Services that the users will get:

a) The farmer will get the SMS and Email alert about the orders they have received and its updates.

b) The buyer will get the SMS and Email alert about the orders they have placedand its updates.

c) The third party logistic will get the SMS and Email alert about the orders they have placed.

Fig 9. explains the above details in a pictorial form. It shows how every module interact with each other through the central system of the online mandi. These modules and their assigned actions coupled with their Geolocation information serves as the necessary ground on which this Geospatial analysis-based framework for agriculture trade works.

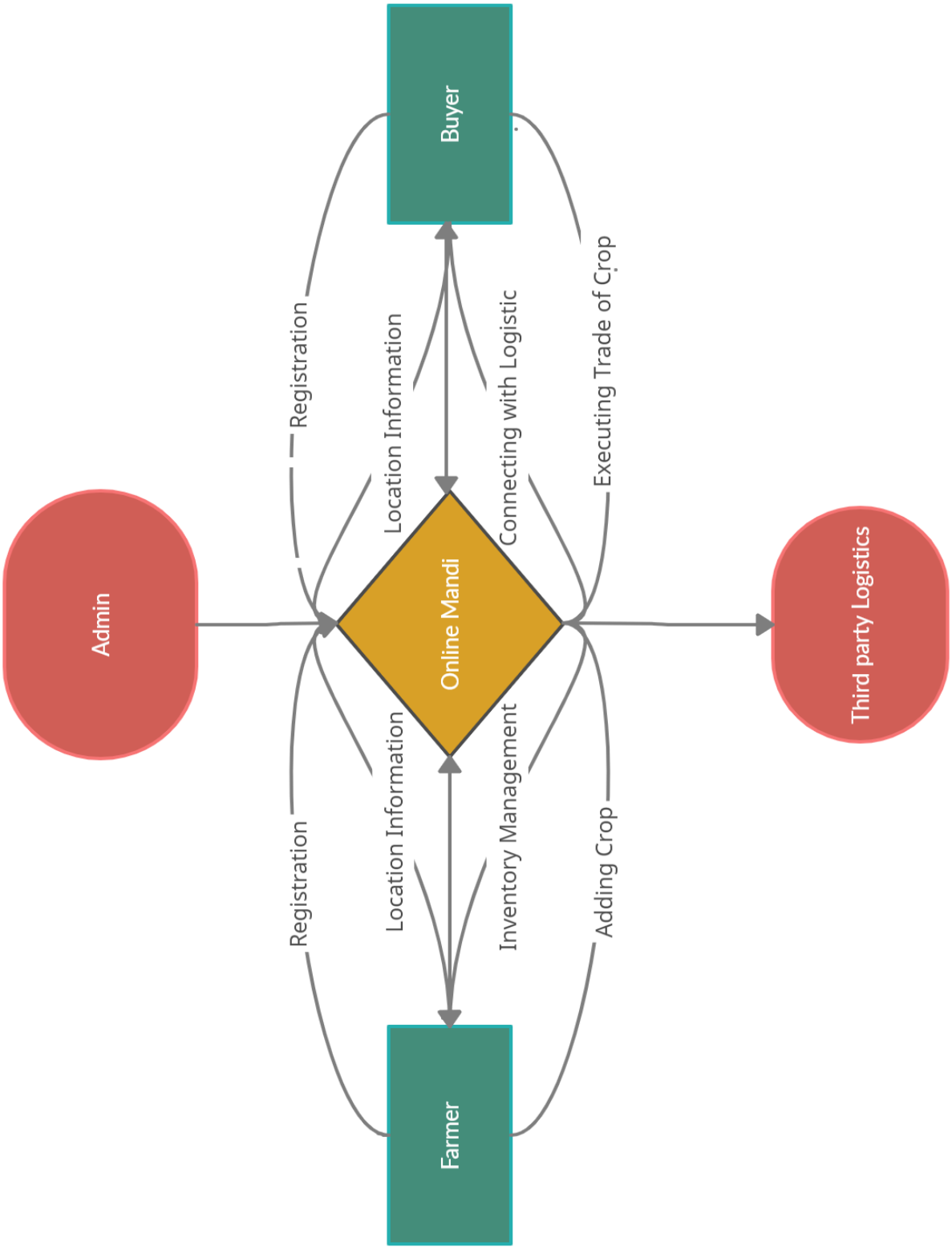

Fig. 10 System Architecture

4.4.3 Use Case Diagram

A use case diagram is a visualization of set of elements and the relationships between them. Use case is a set of scenarios, which defines functionalities of the system from a user's perspective. The main components of a use case diagram include actors, use cases and their relationships. They depict the interaction between actors and system to achieve certain goal. Thus, a use case diagram is important in modelling the behavior of a system.

Fix 10 depicts the use case diagram of this geospatial analysis based agricultural trade system.

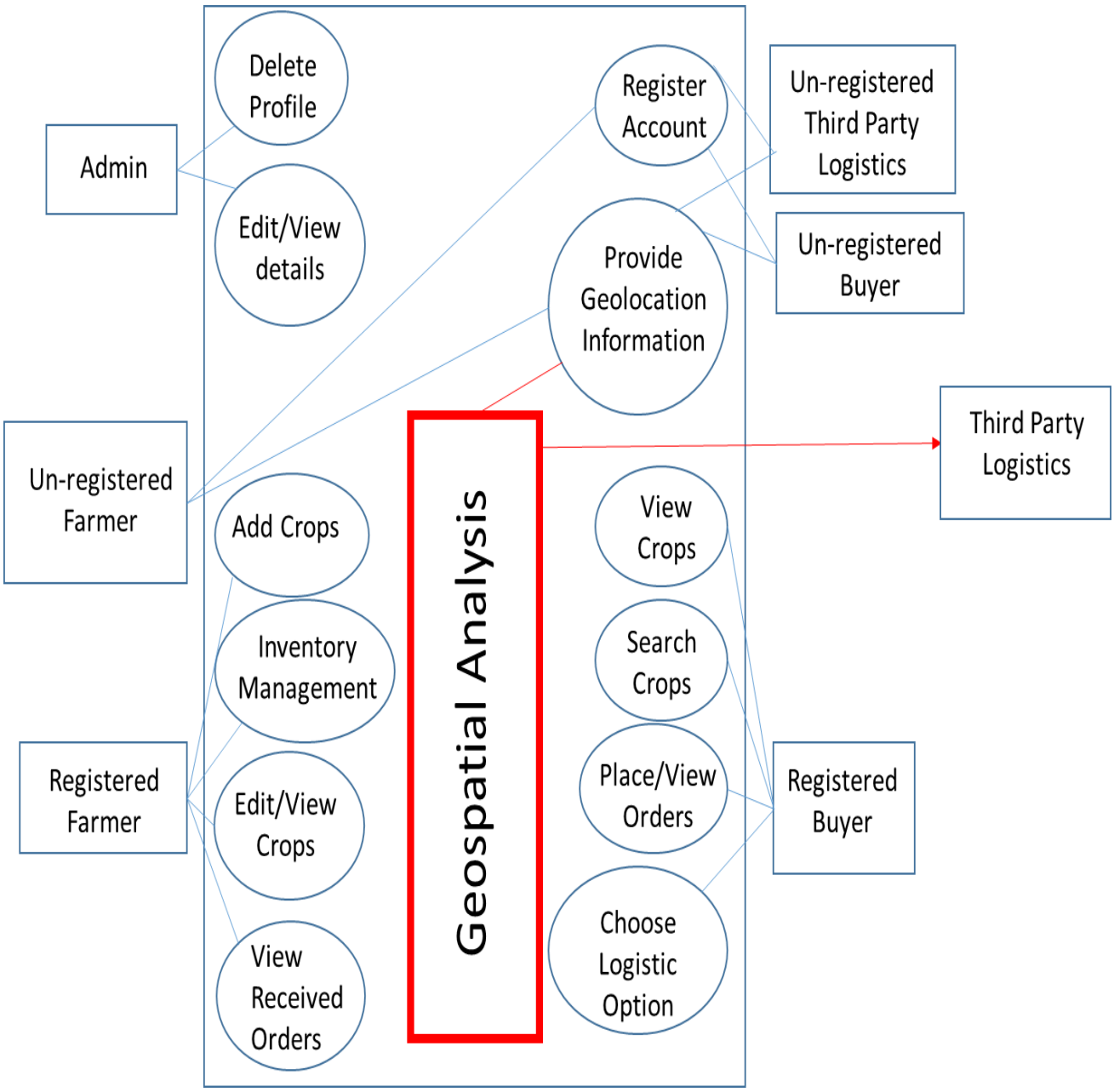

Fig. 11 Use Case Diagram

Actors:

The Actors of the system are Admin, Unregistered Farmer, Registered Farmer, Unregistered Buyer and Registered Buyer, Unregistered Third Party Logistics and Registered Third Party Logistics.

These set of use cases have been identified based on the functionalities and goals of the application.

- **Register Account**- This use case denotes a set of actions required for farmer, buyer and third party logistic to register with the application.
- **Login**- This use case denotes a set of actions required for farmer, buyer and third party logistic to login into the application.
- **Activate/De-activate Account**- This use case denotes a set of actions required for admin to activate or de-activate the Farmers.
- **View Farmers**- This use case denotes a set of actions required for admin to view the Farmers that are registered with the application.
- **Add Crops**- This use case denotes a set of actions required for Farmer to add crops.
- **Edit/View Crops**- This use case denotes a set of actions required forFarmer to change the amount of crop he has for sale at a moment.
- **View Received Orders**- This use case denotes a set of actions required for Farmer to view the list of orders he has received.
- **Inventory Management**- This use case denotes a set of actions required for manage the inventory of the farmer based on placed order.
- **View Crops**- This use case denotes a set of actions required for Buyer to view all the crops listed by the Farmers.
- **Search Crops**- This use case denotes a set of actions required for Buyer to search available crops for trade.
- **Place/View Order**- This use case denotes a set of actions required for Buyer to place and view his orders.

- **Choose Logistic Option**- This use case denotes a set of actions required for Buyer to choose from a list of logistic operator for the final delivery of his order.

- **Geospatial Analysis** – This use case denotes the all the geospatial analysis that the users will get from this system which includes network analysis, distance based information, location based information and geographically referenced dynamic map.

4.4.4 Workflow Architecture

System Workflow

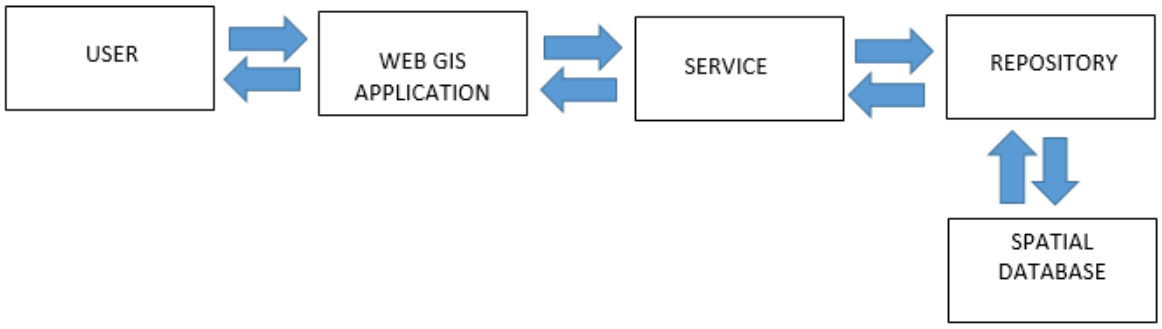

Fig. 12 System Workflow

Farmer Workflow

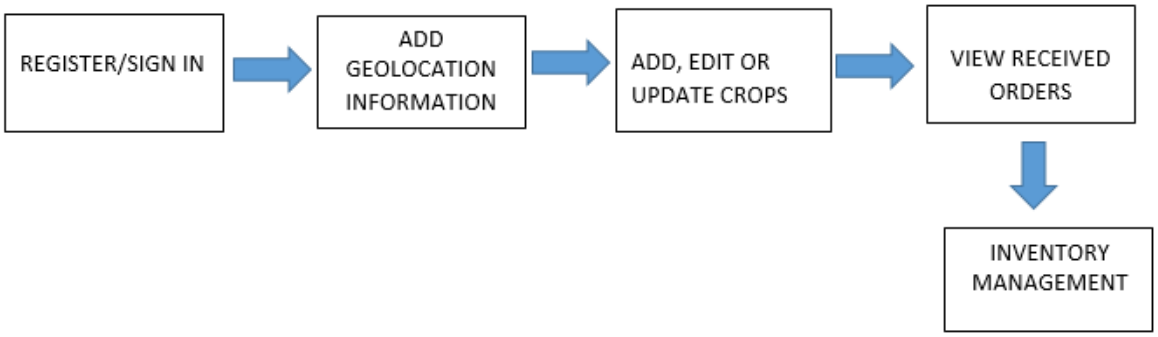

Fig. 13 Farmer Workflow

Workflow of the farmer module:

1. First, register or sign in into the Web GIS application.
2. Then, add Geolocation information to the system.
3. Add, view, edit or update the Crops as per the need.
4. View received orders.
5. Inventory management.

Buyer Workflow

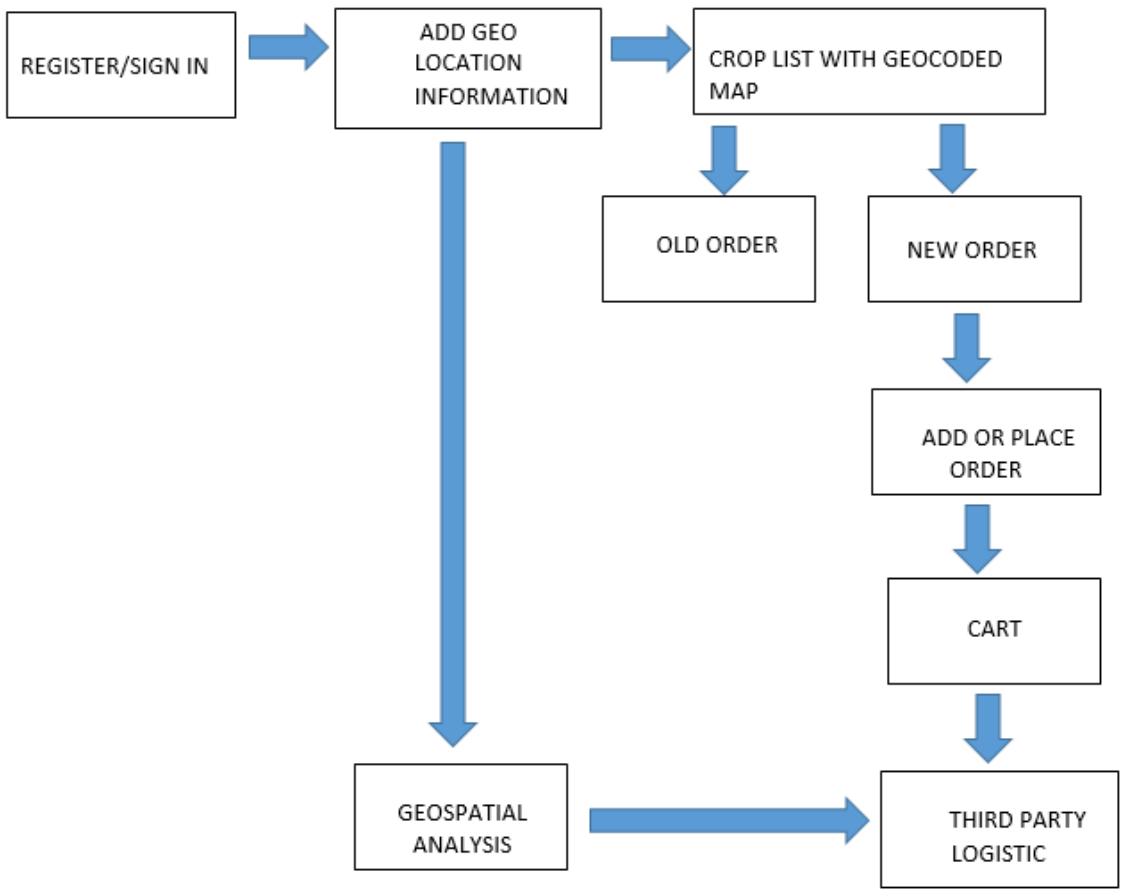

Fig. 14 Buyer Workflow

Workflow of the Buyer module:

1. First, register and sign in into the Web GIS Application.
2. Then, Add Geolocation information to the system.
3. View the Crops listed for trade by the farmer along with Geospatial analysis based Map.
4. Execute trade by placing the desired order.
5. Choose for the third party logistic based on the geospatial analysis provided by the system for final delivery.

CHAPTER 5

IMPLEMENTATION AND RESULTS

5.1 INTRODUCTION

This project aimed at developing a platform for agricultural trade between farmers and buyers with geospatial analysis component and the developed platform follows a Model-View-Controller design pattern, this along with all other technologies used and the methodology followed for the development of this Web GIS Application was outlined in the previous chapter. This chapter outlines the implementation part of the proposed Web GIS Application and the results produced. This is discussed under the following head:

A. MVC System Design

B. User Interface

C. Implementation and Result

5.2 MVC SYSTEM DESIGN

The Model View Controller design pattern is a system design pattern which provides a general solution to the problems which occurs commonly during the development of a web application and is also reusable according to the requirements of the individual application. This design pattern is used in creating this Geospatial Analysis Based Trade Application for Agriculture Business. This design pattern organizes and divides the logic of the program into three individual components. The three individual components are:

A. Model

B. View

C. Controller

This design pattern is depicted visually in Fig. 14

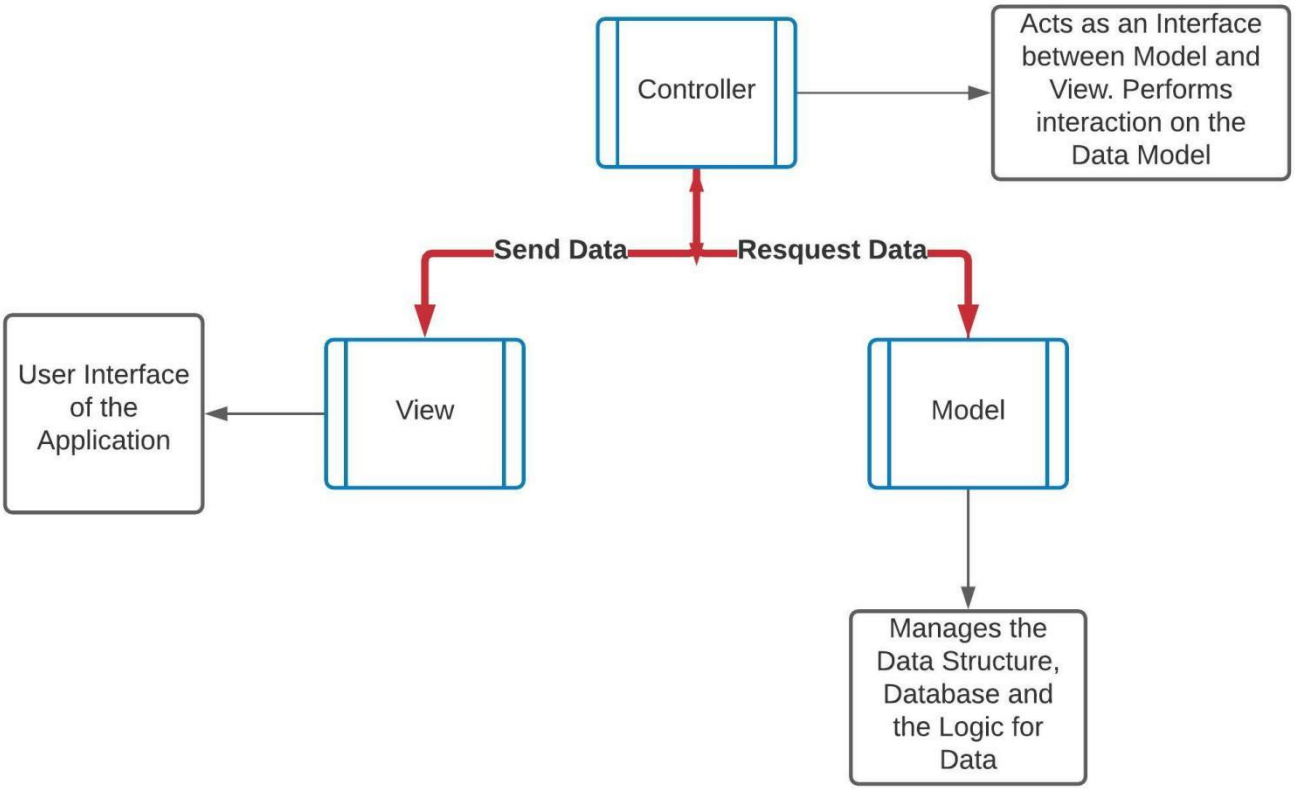

Fig. 15 MCV Design Pattern

5.2.1 Model

The role of the Model component in this Web GIS Application' system design is to create a dynamic data structure and geodatabase for storing all the geospatial and non-geospatial data that will come into the system along with establishing a connection with the created geodatabase to retrieve the required geospatial and non-geospatial data whenever requested by the application. The code associated with the implementation of this program logic is attached in the Appendix 2.

5.2.2 View

The View component of this system design deals with all the visual based interaction that the user performs with this Web GIS Application which includes all the geospatial and non-geospatial data that the user enters into the system, all the processed data that the user gets to see, the geospatial analysis based information like the shortest distance between the users geolocation, all the geographical referenced maps, all the network analysis based map and many more which are individually explained and shown further in this

chapter. The code associated with the implementation of this program logic is attached in the Appendix 2.

5.2.3 Controller

The Controller component of this design pattern here acts as an interface between the Model and View component that is to say, it connects the Model and the View component by taking the input from the View component and passing the same to the Model component after optional validation. It basically responds to the user input/request and performs interaction on the data model, i.e., it locates the source of appropriate response and gets the value and handle all the errors if raised in the process. In this Web GIS application all the User's queries like the type of crops, its quantity, its quality etc. and all the geospatial analysis on the geolocation data of the users is performed by this component. The code associated with the implementation of this program logic is attached in the Appendix 2.

5.2.4 USER INTERFACE

This Section will outline the user interface or the web pages of the developed Web GIS Application. The user interface generally abbreviated as UI acts as the contact point between the user and the Web GIS Application. The user will input the required data via the user interface and gets the appropriate response on this user interface. The processing of the request and response of the data is handled by the MVC design pattern as discussed above. The User interface of this Web GIS Application will be discussed under the following heads.

A. Common Registration Page
B. Login Page
C. Complete Registration page for Buyers, Farmers, and 3rd Party Logistic
D. Farmer's Homepage
E. Add Crop Page for Farmer
F. Order Received Page for Farmer
G. Buyer's Homepage
H. Placed Order Page for Buyer
I. Third Party Logistic Homepage

5.2.5 Common Registration page

The registration page of this Web GIS Application registers the users upon first entering the application and let them choose their category which is namely:

A. Buyer

B. Seller

C. Logistic

It takes the basic information required for the user to register initially which are

A. Username

B. Email id

C. Password

D. User Type:

   a. Buyer

   b. Logistic

   c. Seller

Fig. 12 shows the User Interface for Registration Page

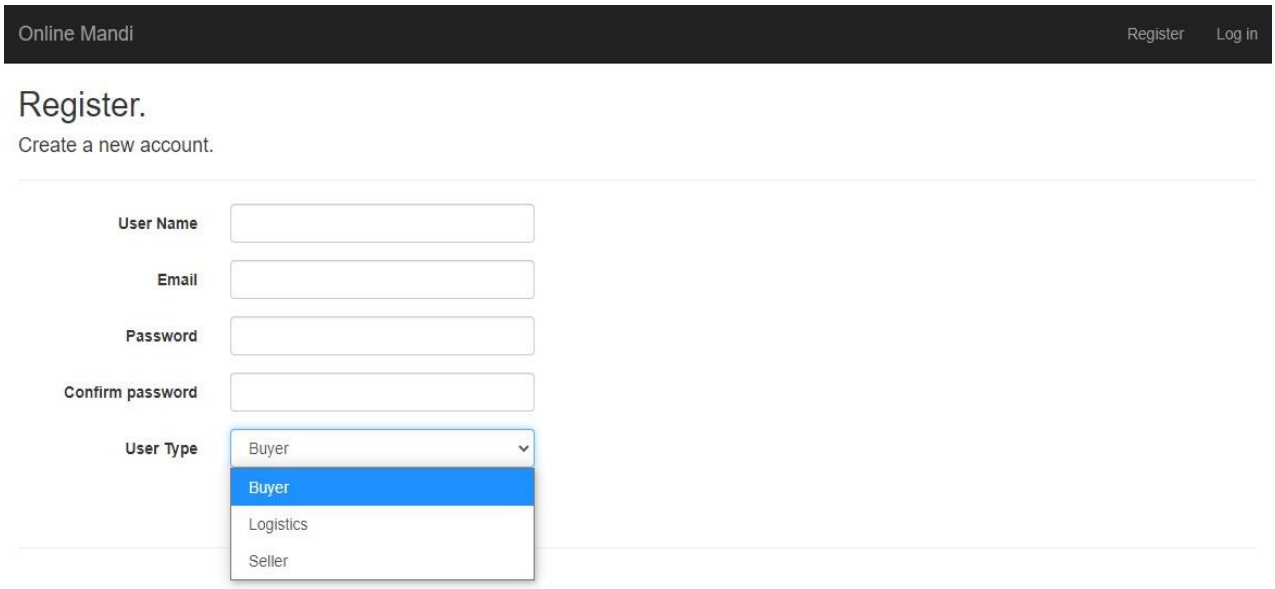

Fig. 16 Common Registration Page for the Users

5.2.6 Login Page

After the user has initially registered on the registration page, then he has to login into this Web GIS Application for further process and registration to be carried out. The login page takes the following information from the user to let them enter the system:

A. Email Id
B. Password

Fig. 13 shows the User Interface for the Login Page

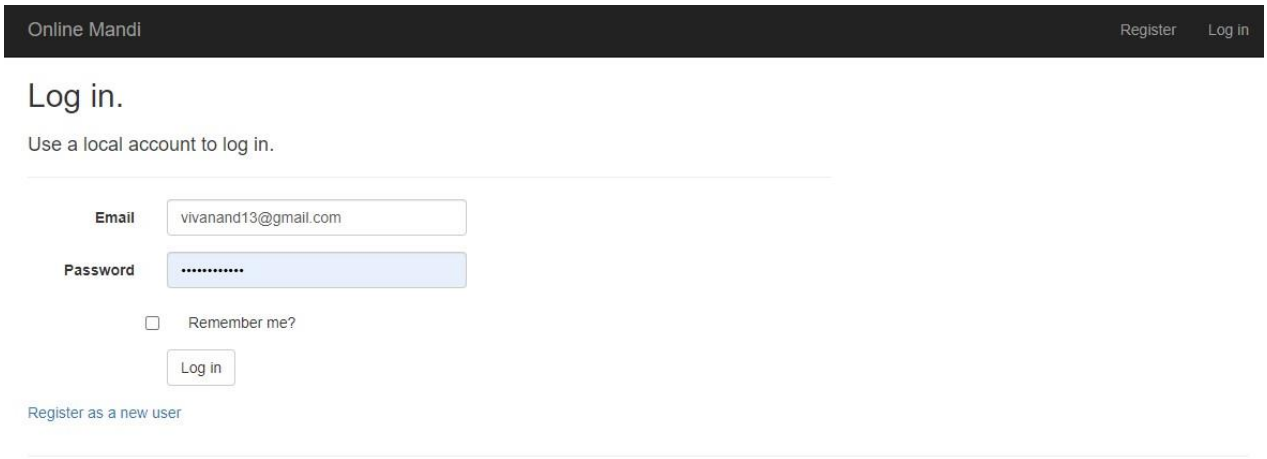

Fig. 17 Login Page for Users

5.2.7 Complete Registration Page for Buyers, Farmers, and Third Party Logistic

After completing the initial registration and then logging into the system the user must complete the final registration where they have to fill in all the necessary information for this system to function according to its intended purpose. The registration page for farmers and buyers will take the following information from them:

A. Registration type
    a. Buyer
    b. Farmer
B. Name
C. Contact No.
D. Aadhaar No.
E. PAN card No.
F. Gender

G. Date of Birth

H. Geolocation – Input taken via Map

The Fig 14. shows the user interface for the registration page for Farmers and Buyers

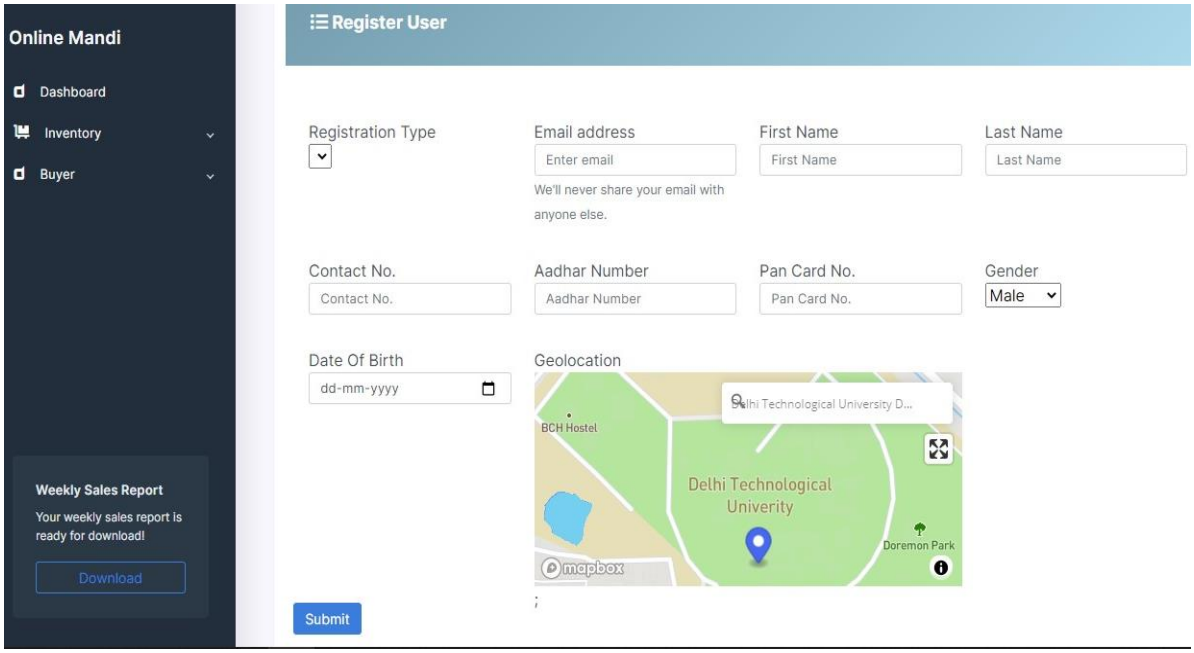

Fig. 18 Registration Page for Farmers and Buyers

The registration page for Third Party Logistic will take the following information from them:

A. Name

B. Email Id

C. Phone No.

D. Address

E. Geo Location – Taken Via Map

F. Transportation Type –

   a. Refrigerated Transportation

   b. Non-Refrigerated Transportation

   c. Both

G. Shipping Method

   a. Roadways

   b. Railways

c. Airways

ii. Rate Catalogue

H. Rate per Kilogram (Rs)

I. Rate per Kilometers (Rs)

The Fig 15. shows the user interface for the registration page for third party logistic

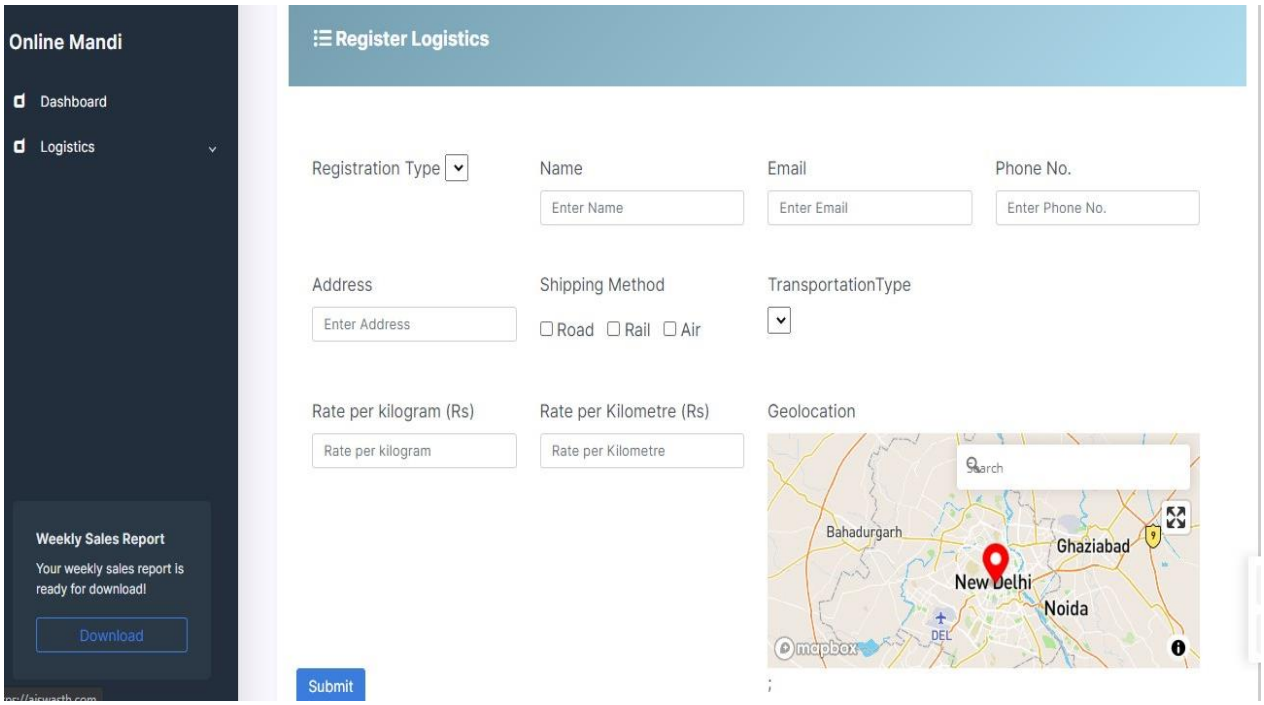

Fig. 19 Registration Page for Third Party Logistics

5.2.8 Farmer's Homepage

After the farmer has completed the complete registration process, he will then be taken to the farmer's homepage. On this page the farmer will be able to see all the crops that he has added along with the option to navigate to 'Add Crop Page' or 'My Orders Page'. The Fig. 16 shows the user interface for farmer's home page.

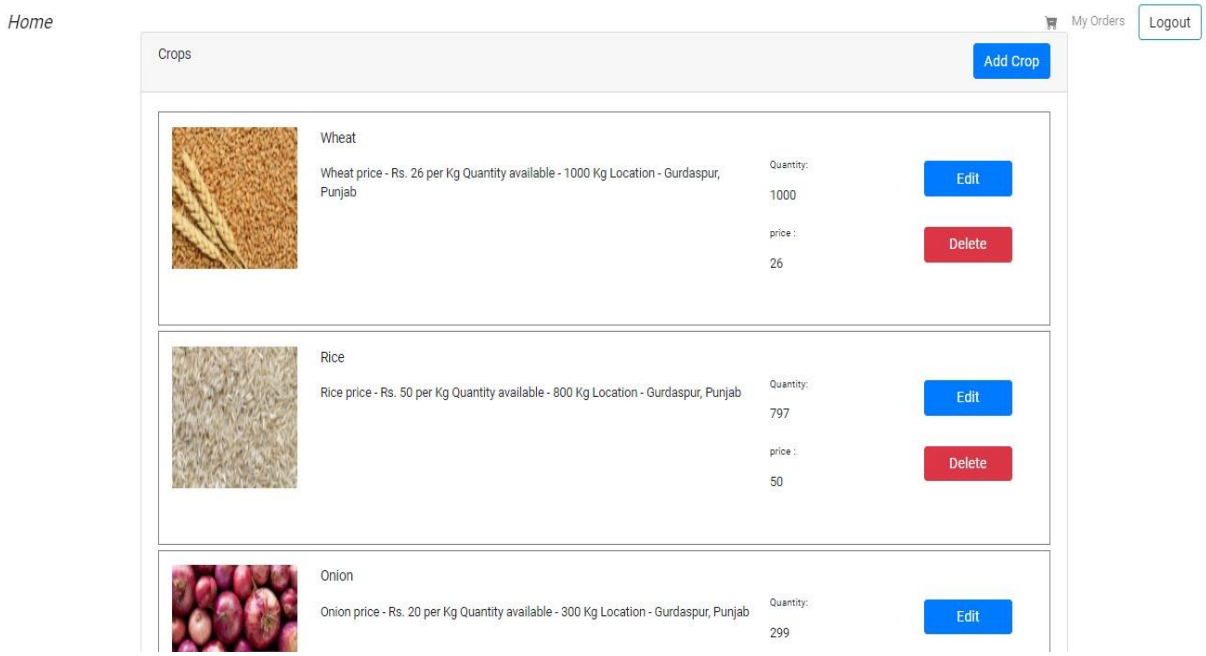

Fig. 20 Farmer's Homepage

5.2.9 Crop Page for Farmer

This page provides the farmer with the option of adding the crops he has for trade. The farmer can navigate to this page by selecting the option available on the farmer's homepage. This page will take the following information about the crop from the farmer in order to list out the crop for trade:

A. Crop Name
B. Trade Price (per Kg)
C. Quality Type
D. Maximum Trade Quantity
E. Image of the Crop
F. Crop Description

The Fig. 17. shows the user interface for farmer's add crop page

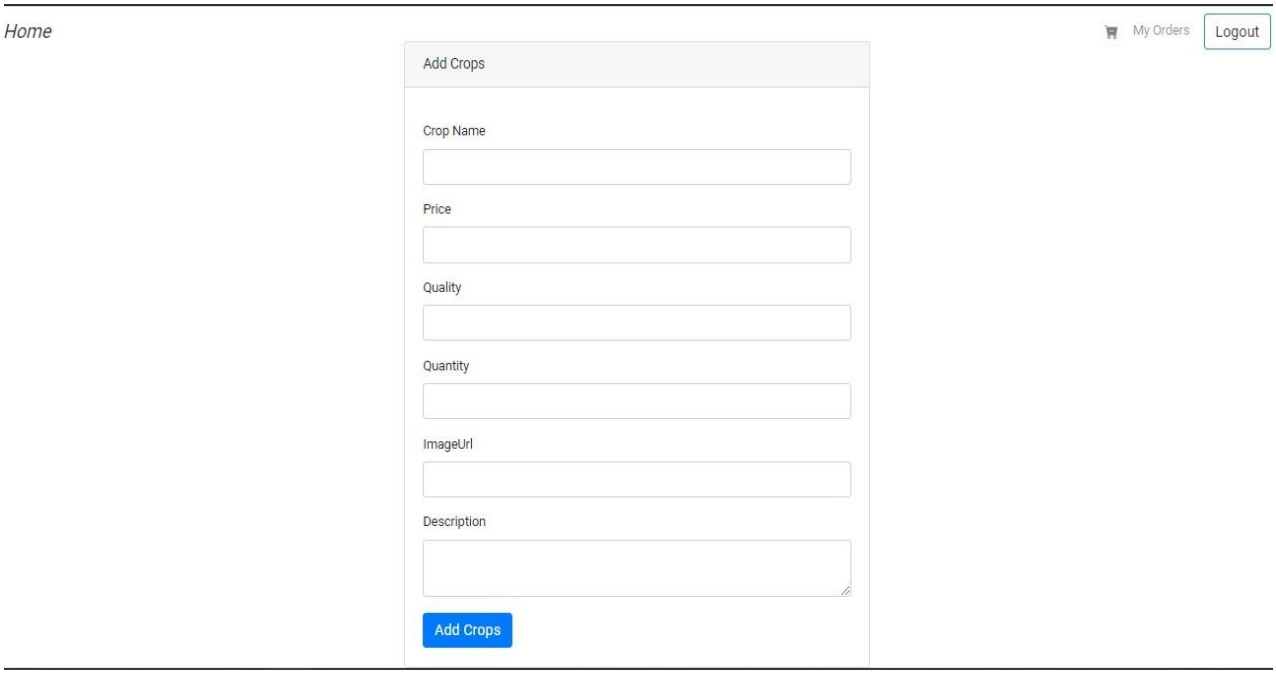

Fig. 21 Farmer's Add Crop Page

5.2.10 Order Received page For Farmer

The order received page for farmer will list out the orders that the farmer has received for trade from the buyer. It can be navigated to by selecting the 'My Orders' option in the farmer's homepage. The Fig 18. shows the user interface for the order received page for farmers.

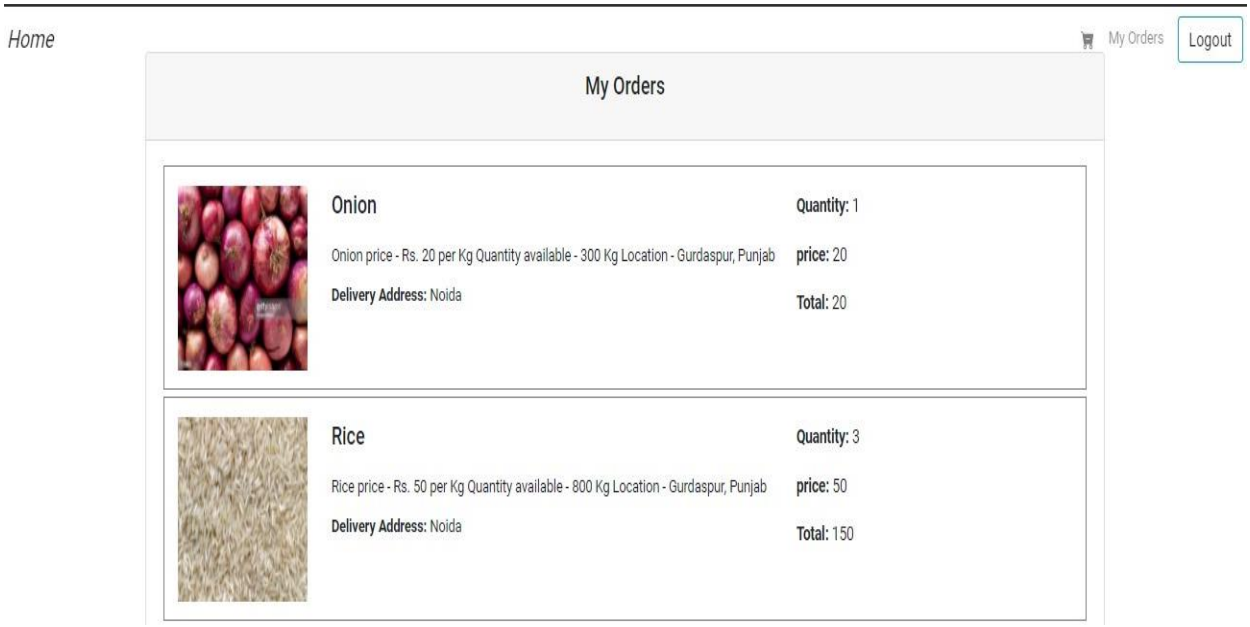

Fig. 22 Farmer's My Orders Page

5.2.11 Buyer's Homepage

After the buyer has completed the full registration process, he will then be taken to the buyer's homepage. On this page the buyer will see the broad category of the crops that are available for trade which will be put up by the farmers, on clicking a particular crop category he will be then redirected to the page that will show him all the farmers that are available for trade of that particular crop. The highlighting feature of this page are:

I. The page will show the list of all the farmers that are dealing in that particular crop of choice along with the crop's image, crop's price the quantity available for trade and its quality.

II. The page will have a geographically referenced map that will pinpoint the location of all the farmers that are available for the trade.

III. The page will highlight the address of the farmer and after doing geospatial analysis on the geolocation data of the farmers and the buyer it will highlight the shortest distance between them. This feature will allow the buyer to make a calculated decision while executing his side of the trade.

IV. The page will have the option of placing the order for the crop after making the decision based on the geospatial analysis information provided.

The Fig 19,20,21,22 and 23 shows the user interface for the buyer's homepage

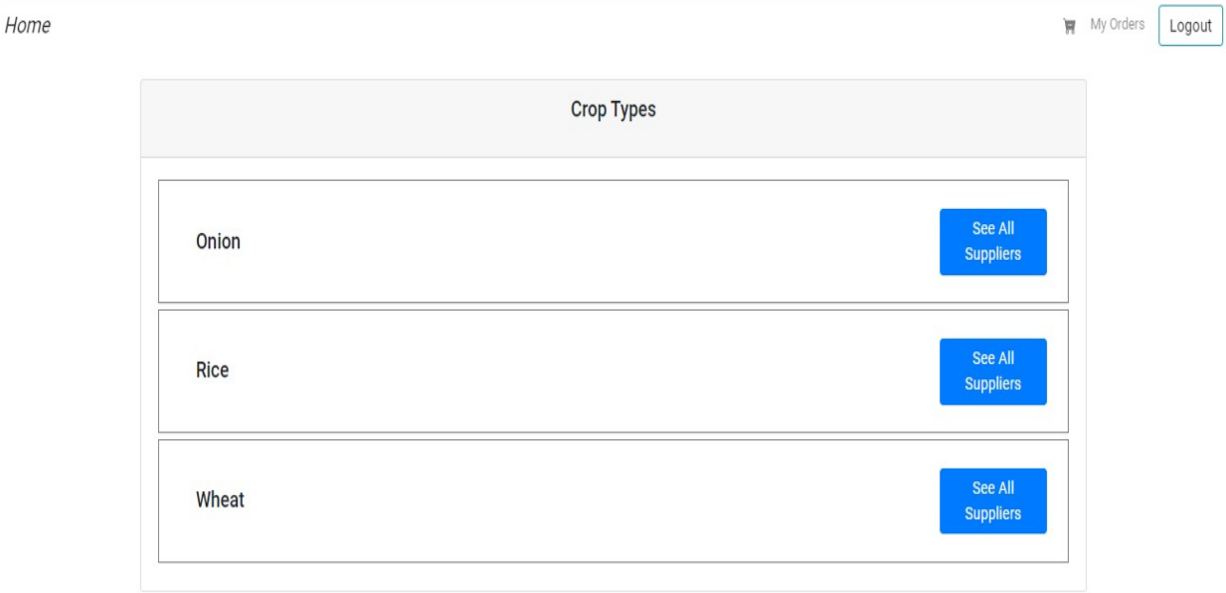

Fig. 23 Buyer's Home Page

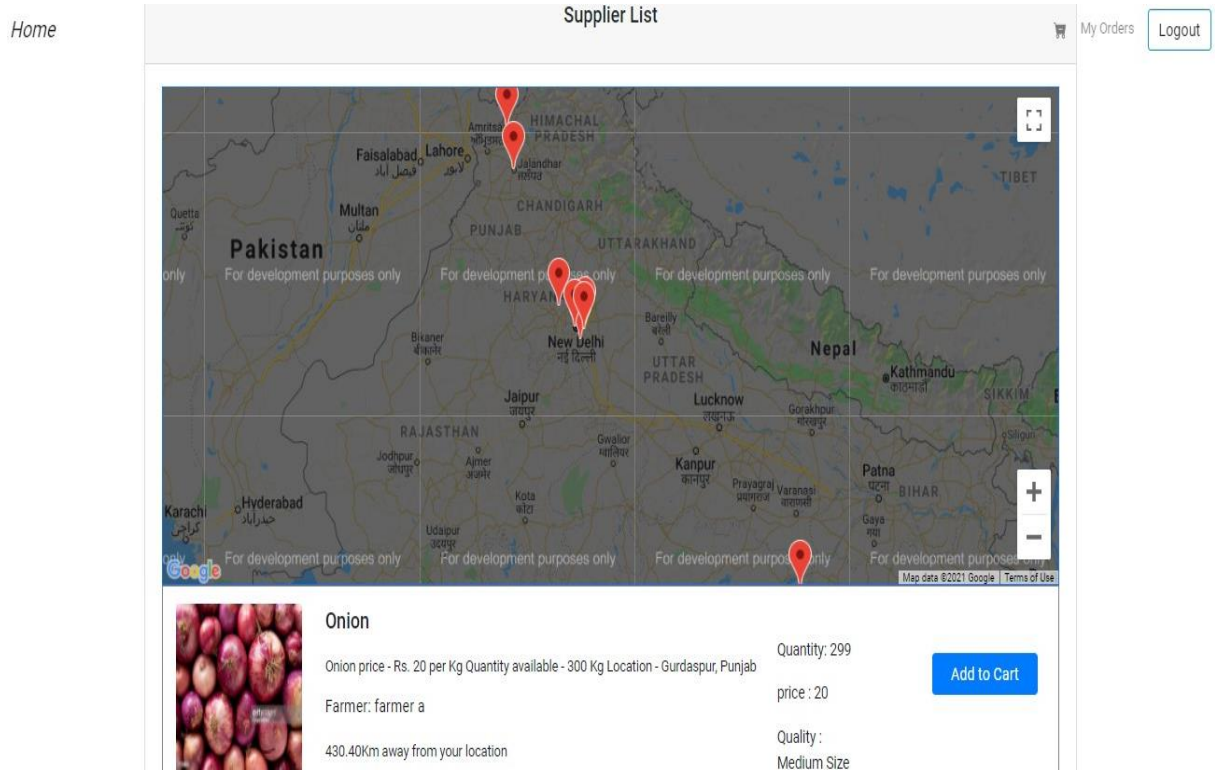

Fig. 24 List of Farmers with their Crop Description

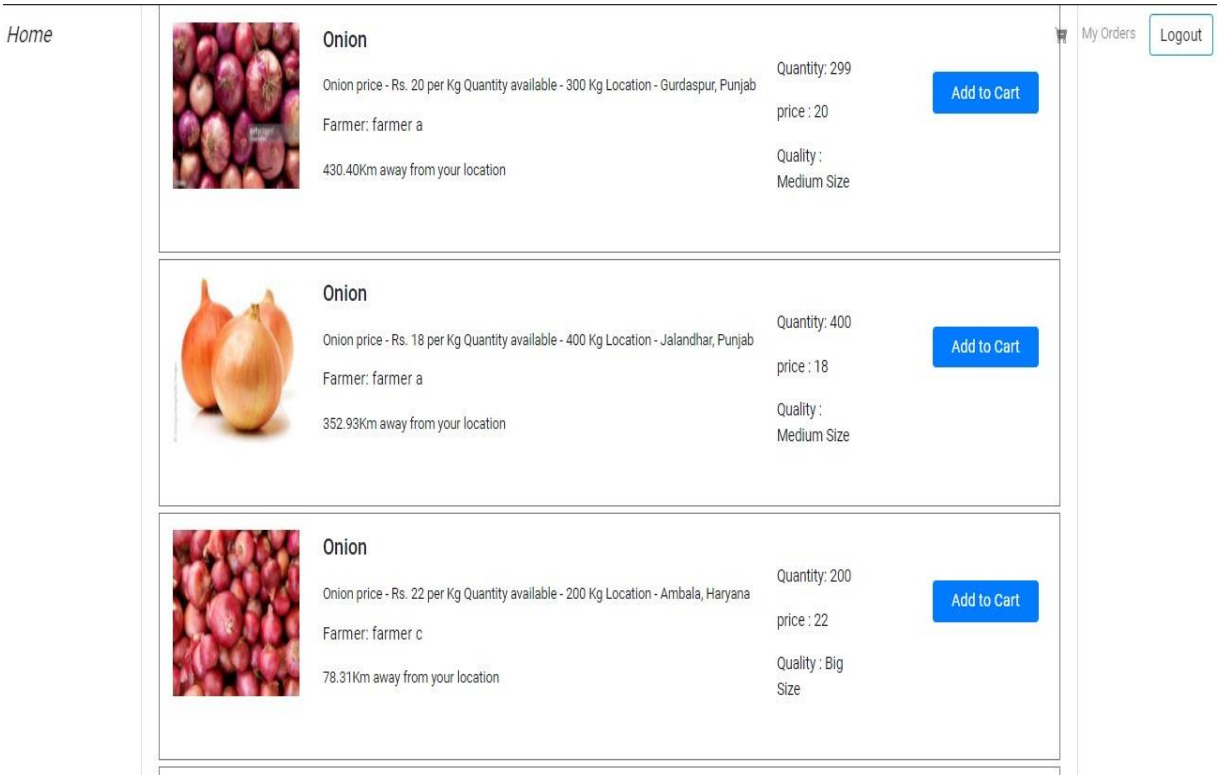

Fig. 25 List of Farmers with their Crop Description

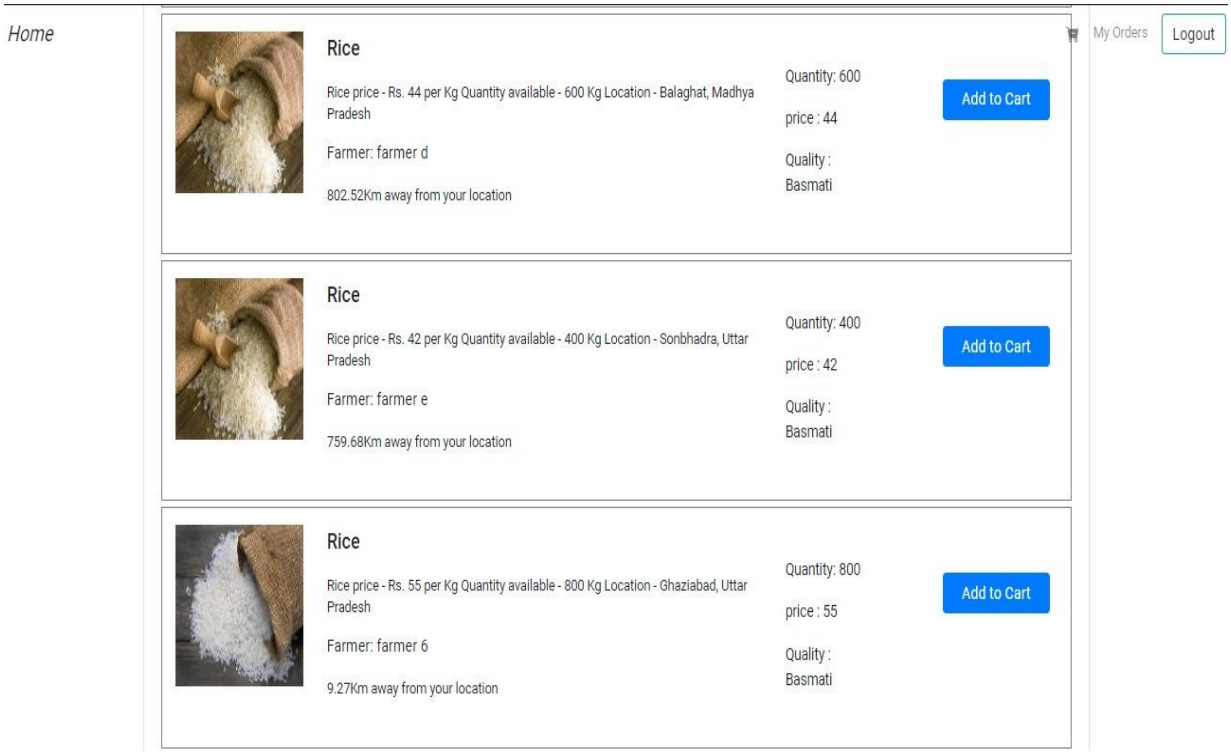

Fig. 26 List of Farmers with their Crop Description

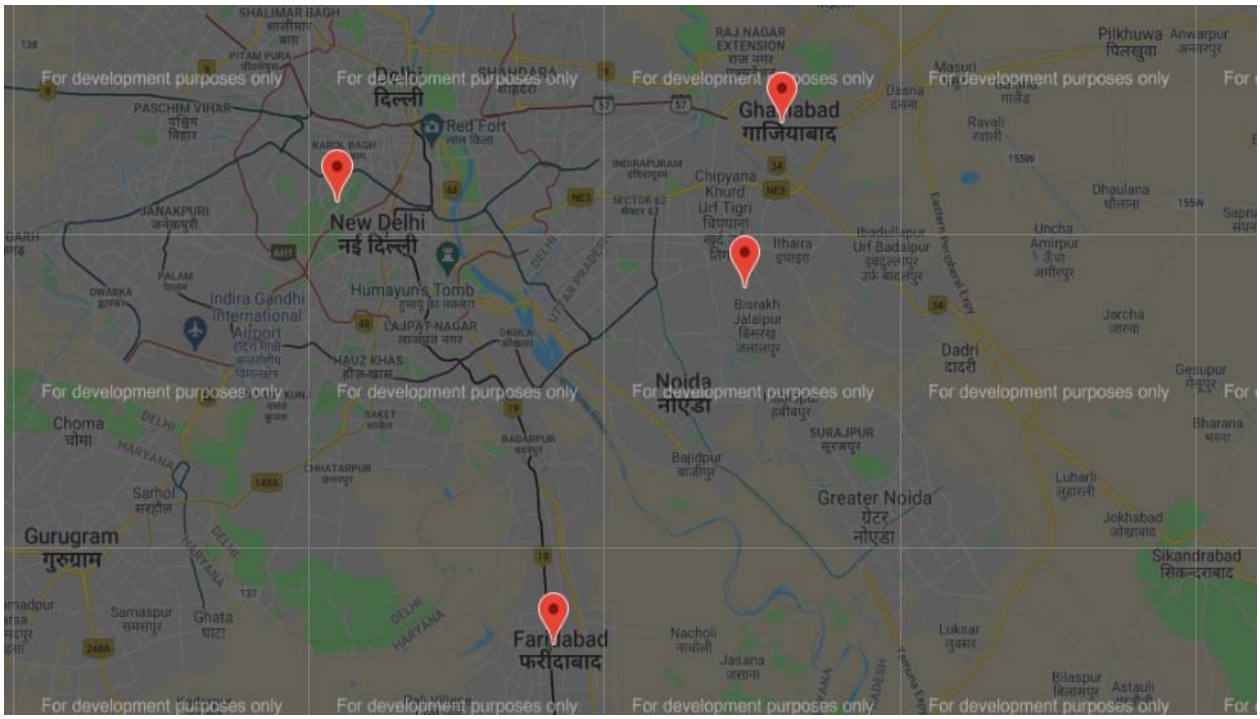

Fig. 27 Geographically Referenced Map Pointing the Location of Farmers

5.2.12 Placed Order Page for Buyer

Once the buyer has placed the order on the previous page, he can then go to his placed order page to see the orders he has placed. He can edit the orders if required. The Fig X shows the user interface of placed order page of the buyer.

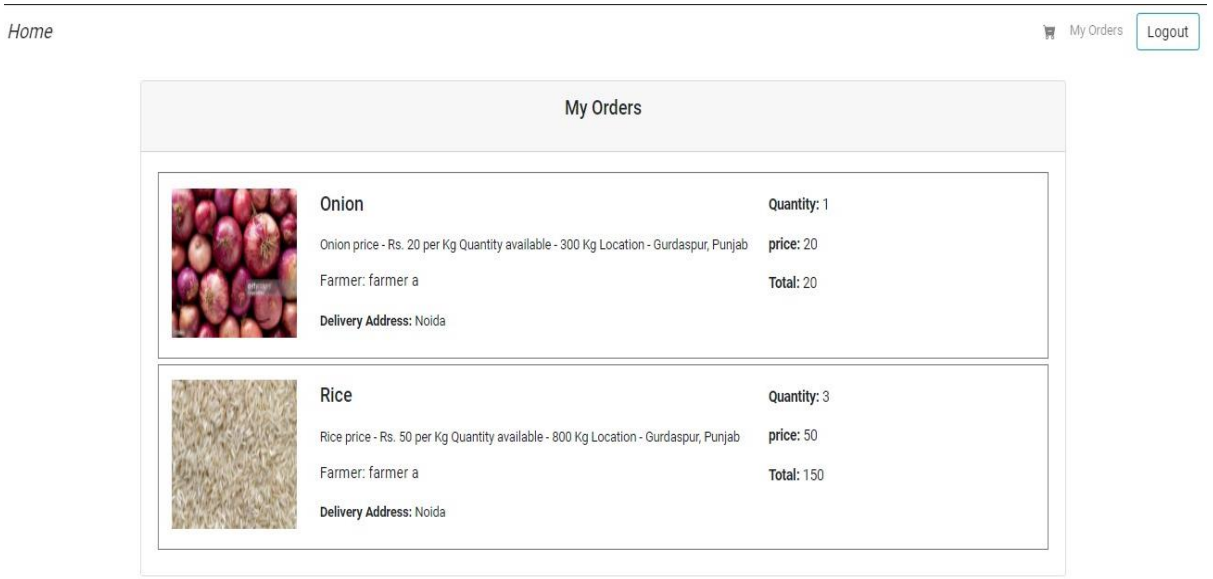

Fig. 28 My Orders Page for the Buyer

5.2.13 Third Party Logistic Homepage

The third party logistic after completing the full registration process will be taken to third party logistics' homepage. On this page the third party logistic will see all the list of orders he has received of whose logistic part they have to handle. The highlighting features of this page are:

I. This page will show the list of orders that the third-party logistic side has received along with all the necessary information.
II. Based on geospatial network analysis, the application provides point-to-point analysis and gives the most optimal route information to the third party logistic to carry out the transportation of the goods.
III. This optimal route information will be shown on a geographically referenced map along with all the navigation options.

The Fig 25 shows the user interface for third party logistics' homepage and Fig. 26 and 27 shows the Optimal route map.

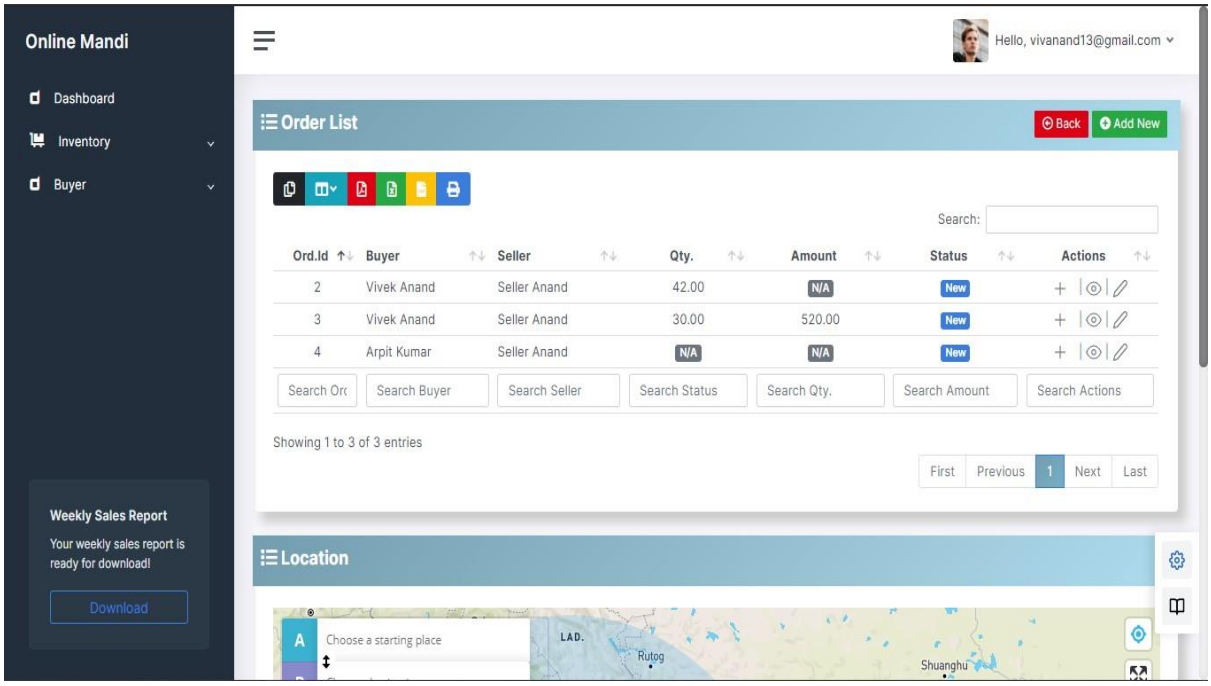

Fig. 29 Third Party Logistics' Homepage

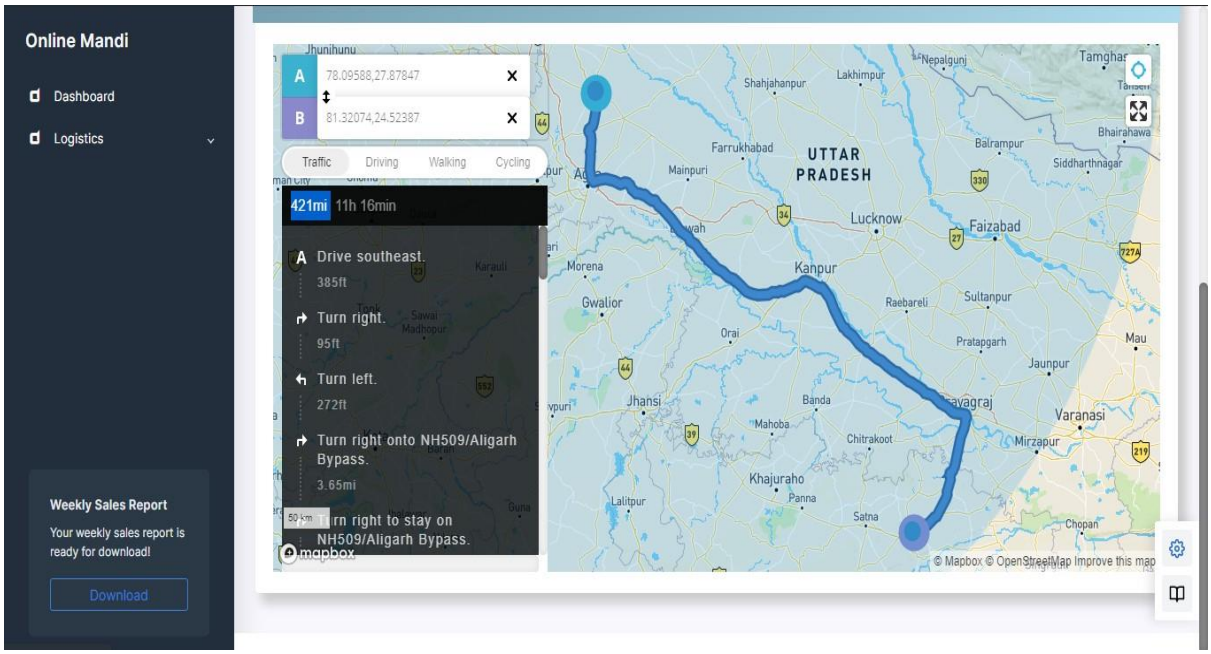

Fig. 30 Optimal Route Map

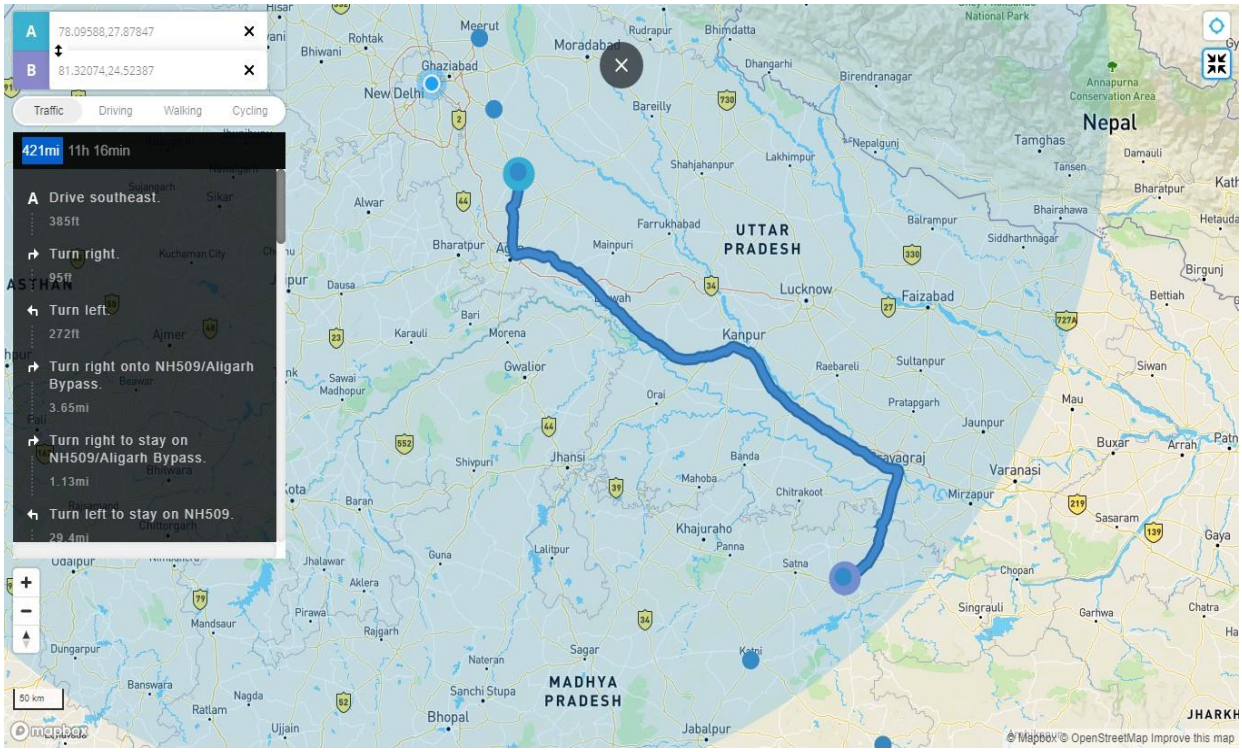

Fig. 31 Optimal Route Map

5.3 IMPLEMENTATION AND RESULT

In the previous sections of this chapter, the user interface, and the system design of this WebGIS works was outlined. In this section, the working/implementation of this WebGIS application with respect to all the end users of the system will be demonstrated. The end users of this WebGIS application are categorized as:

A. Farmer

B. Buyer

C. Third Party Logistic

Here, the step-by-step process of the working of this WebGIS application for every individual end user listed above will be shown.

5.3.1 Implementation for Farmer

The step-by-step implementation of the farmer's side of this WebGIS Application will be demonstrated.

I. Initial Registration

The farmer must initially register by providing the necessary information. Fig. 28 shows the implementation of initial registration process for farmers:

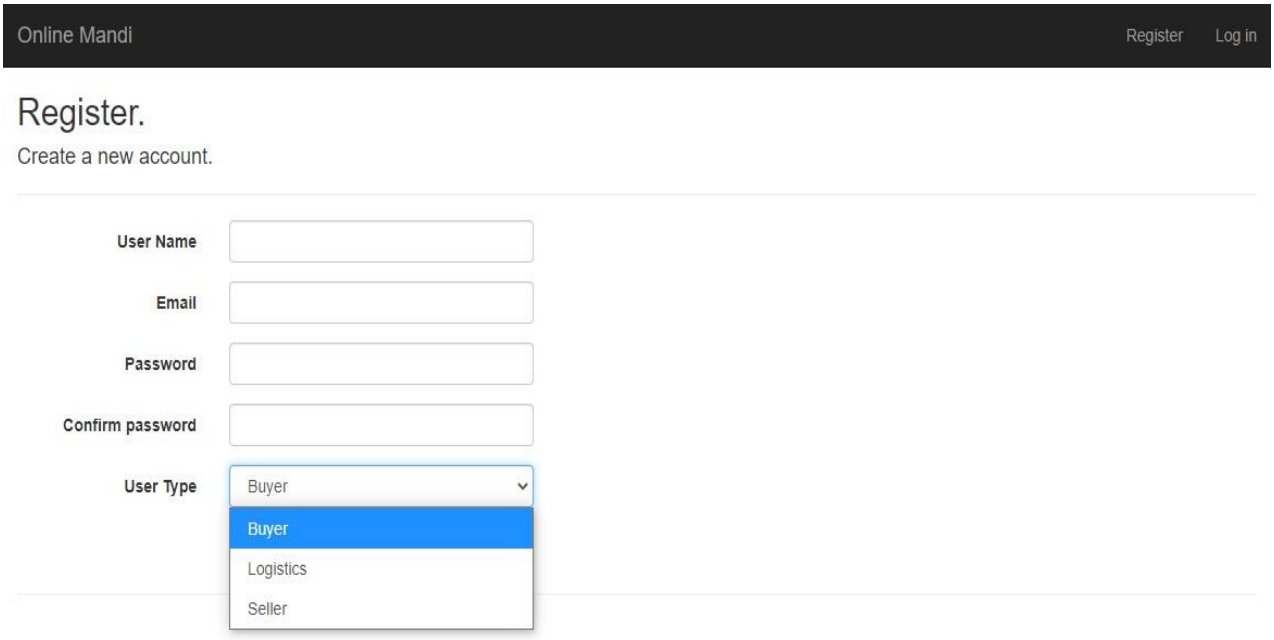

Fig. 32 Initial Registration for Farmer

II. Login

After initial registration, farmer will login into the application. Fig. 29 shows this implementation.

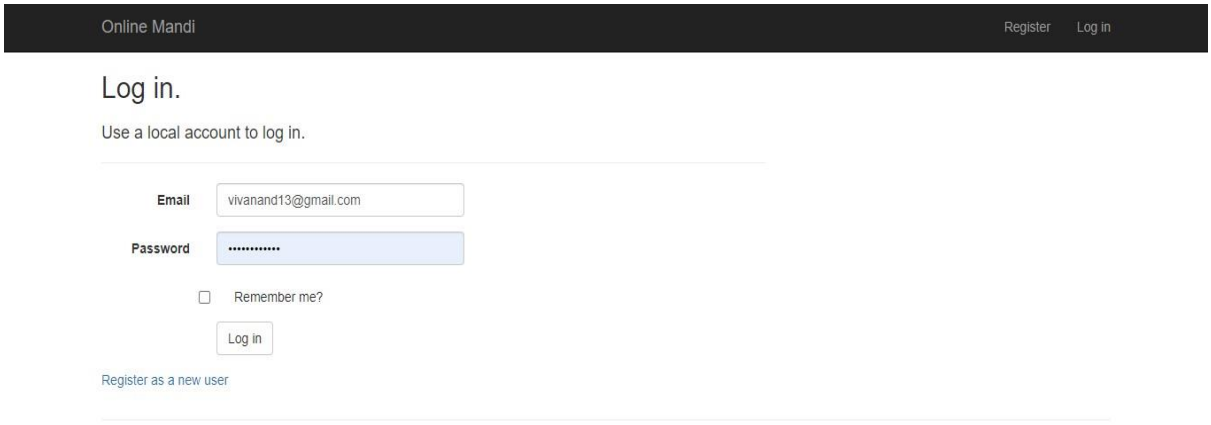

Fig. 33 Login page for Farmer

III. Complete Registration

After logging in to the application, the farmer will complete the full registration process. Fig. 30 shows this implementation.

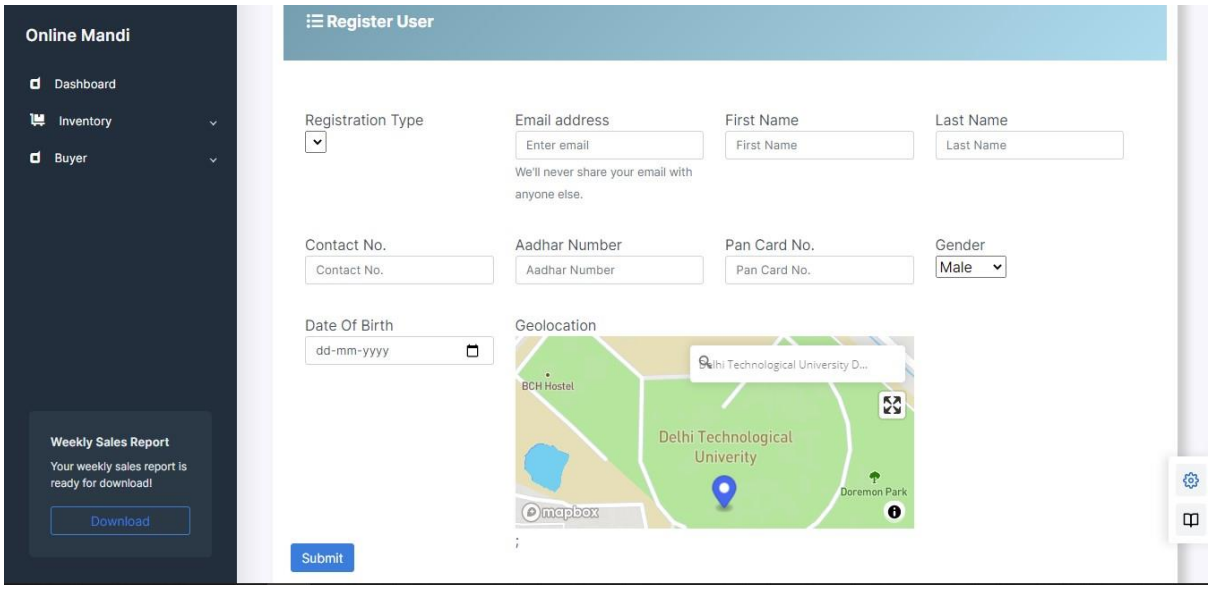

Fig. 34 Complete Registration for Farmer

IV. Adding Crops

After completing the whole registration process the farmer can add the crops he has for sale in the application. Fig. 31 shows this implementation.

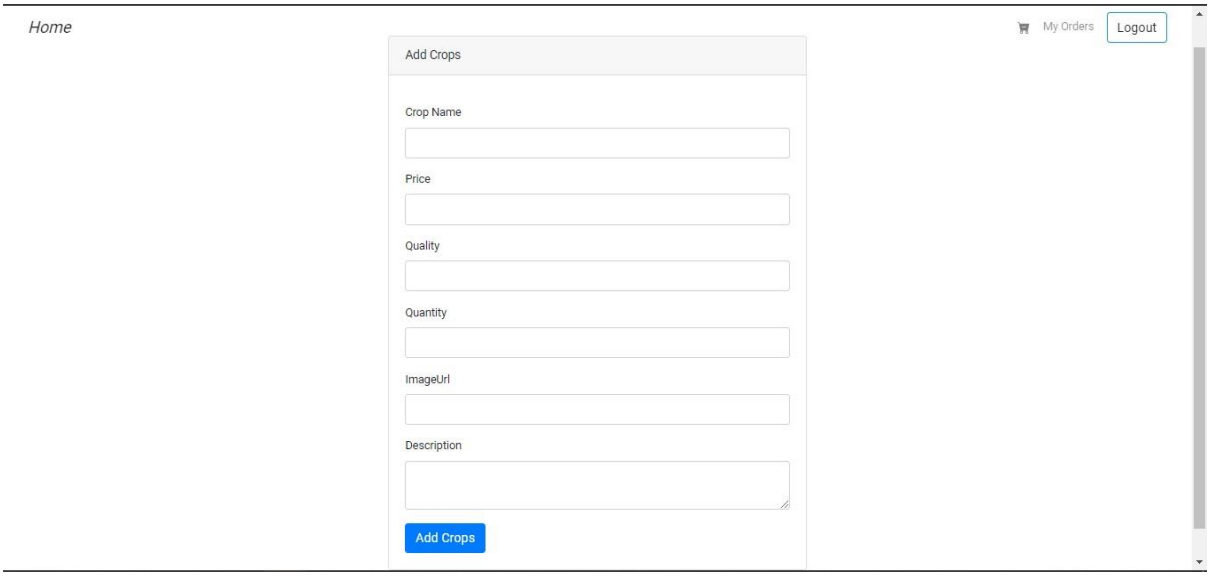

Fig. 35 Adding Crop by Farmer

V. Managing the Orders

The farmer can view and manage the orders he has received using ‘My Orders’ option provided on this homepage. Fig. 31 shows this implementation.

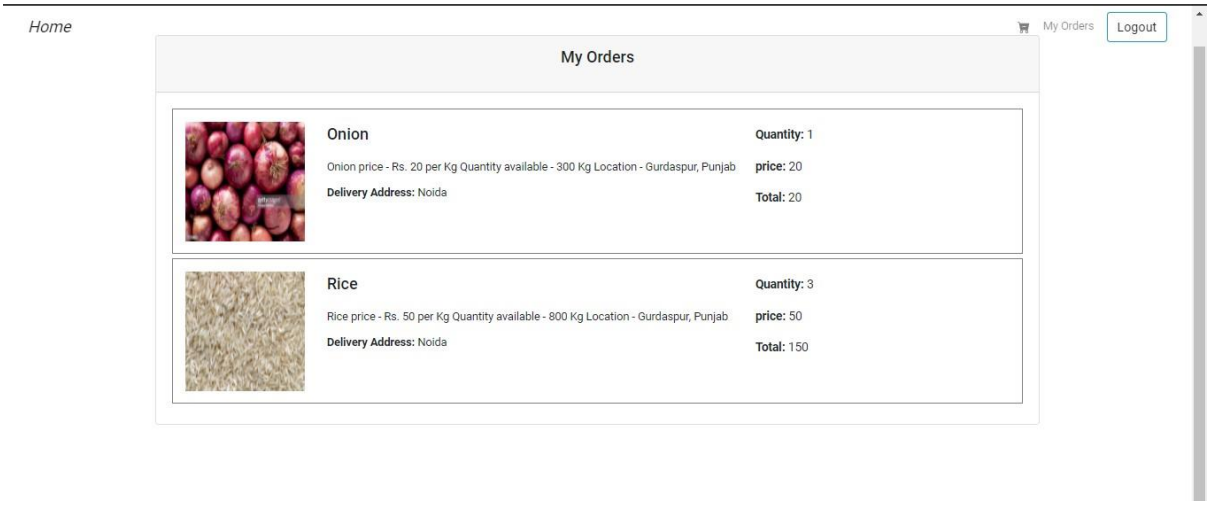

Fig. 36 Managing Orders by Farmer

VI. Viewing Farmer’s listed Crop

Farmer can view and edit the crops he has for sale using the option of ‘Add Crop’ and ‘Edit Crop’ and ‘Delete Crop’ available on his homepage. Fig. 32 and 33 shows this implementation.

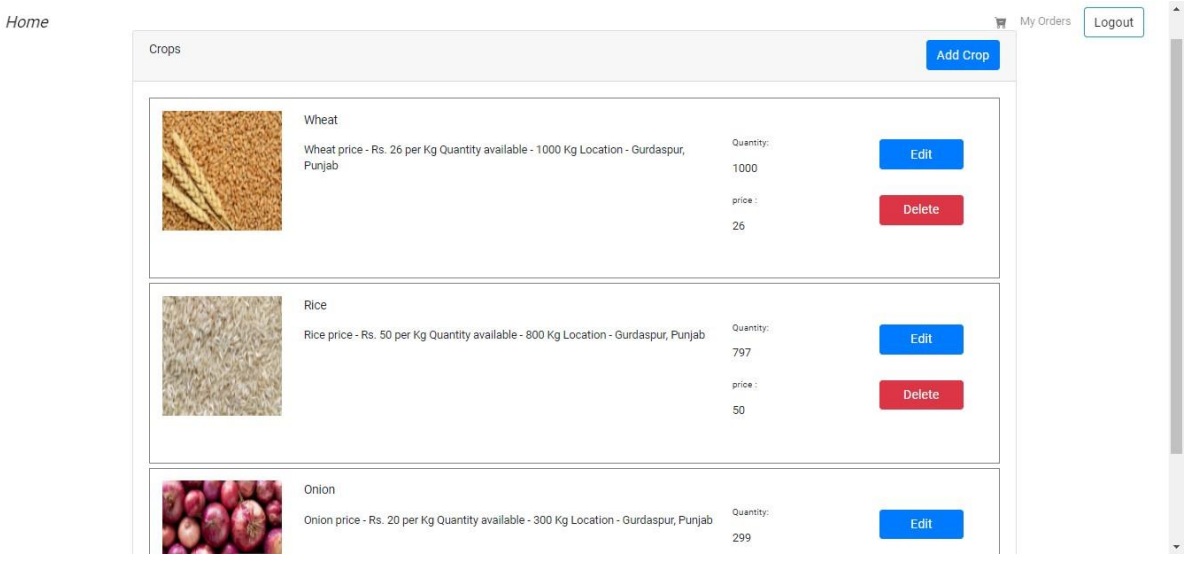

Fig. 37 Viewing Listed Crop by Farmer

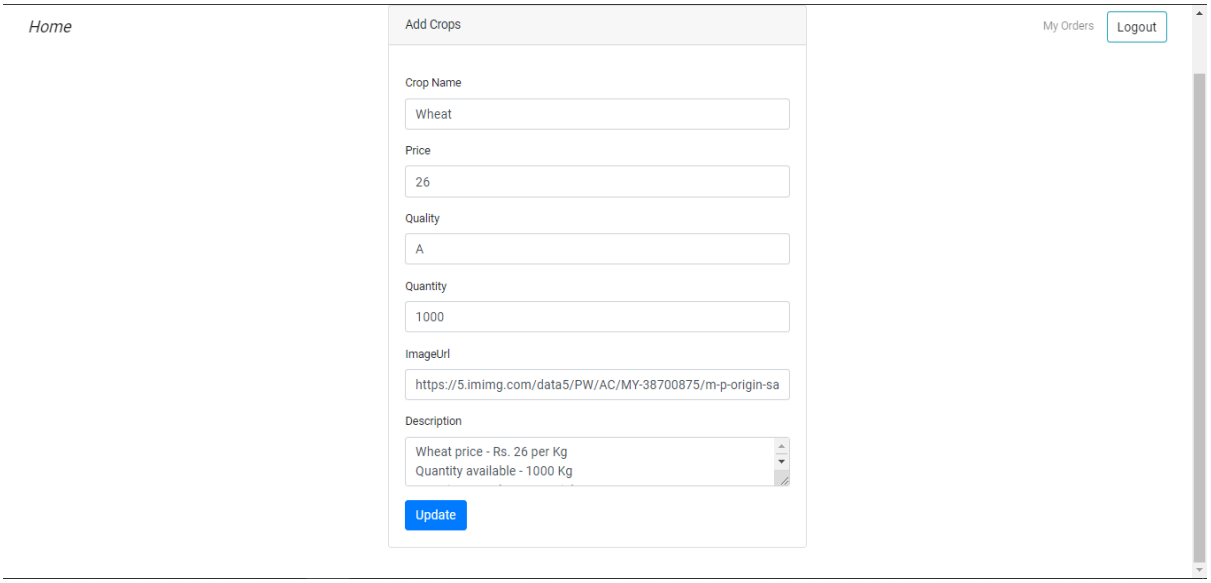

Fig. 38 Editing the crops listed by Farmer

5.3.2 Implementation for Buyer

The step-by-step implementation of the buyer's side of this WebGIS Application will be demonstrated.

1. Initial Registration

The buyer must initially register by providing the necessary information. Fig. 34 shows the implementation of initial registration process for buyers:

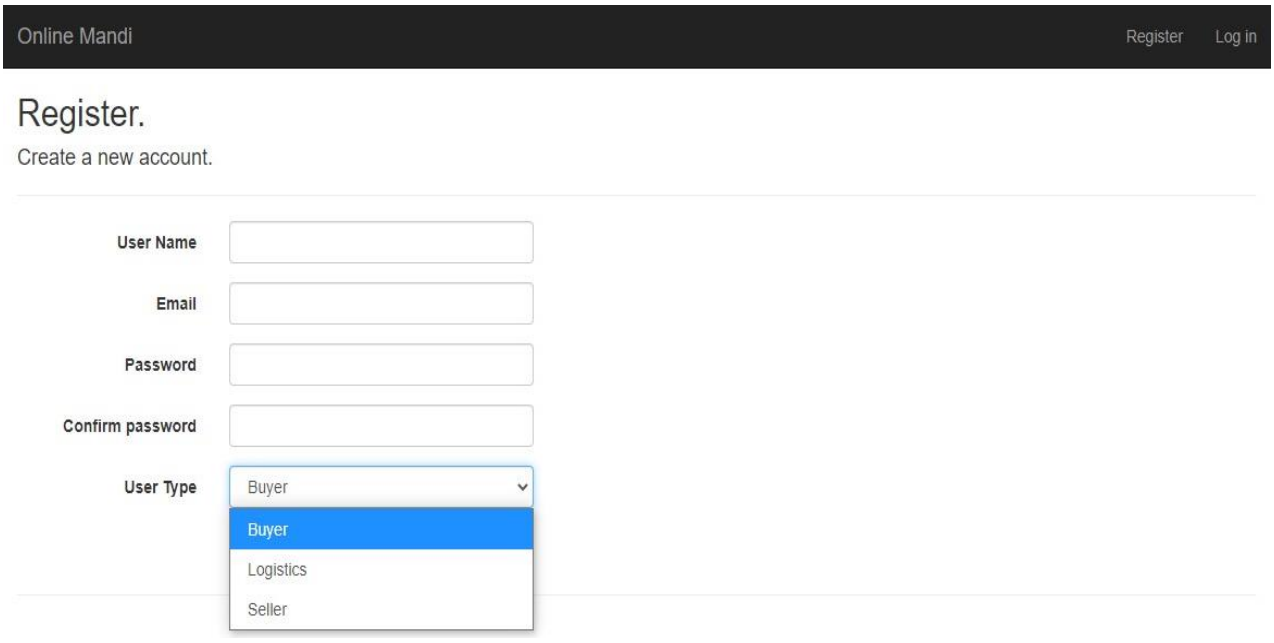

Fig. 39 Initial Registration for Buyer

2. Login

After initial registration, buyer will login into the application. Fig. 35 shows this implementation.

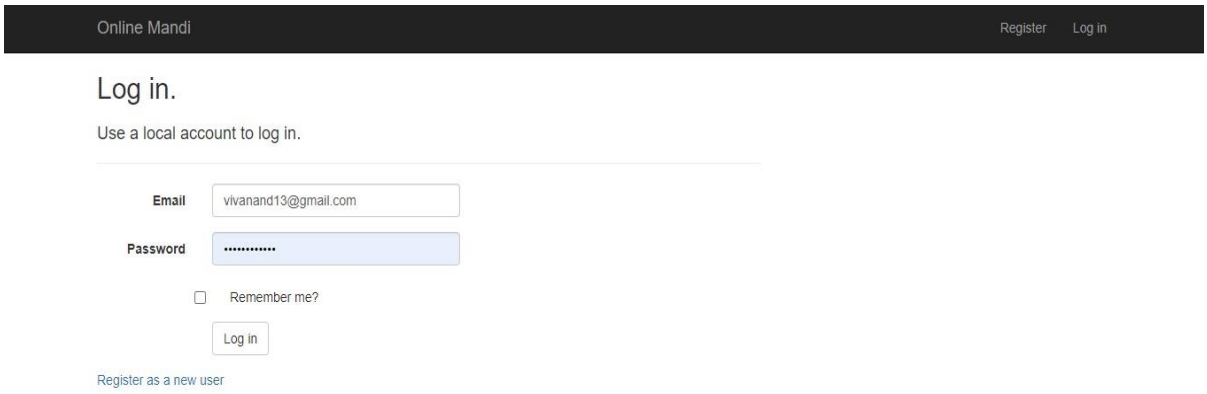

Fig. 40 Login page for Buyer

3. Complete Registration

After logging in to the application, the buyer will complete the full registration process. Fig. 36 shows this implementation.

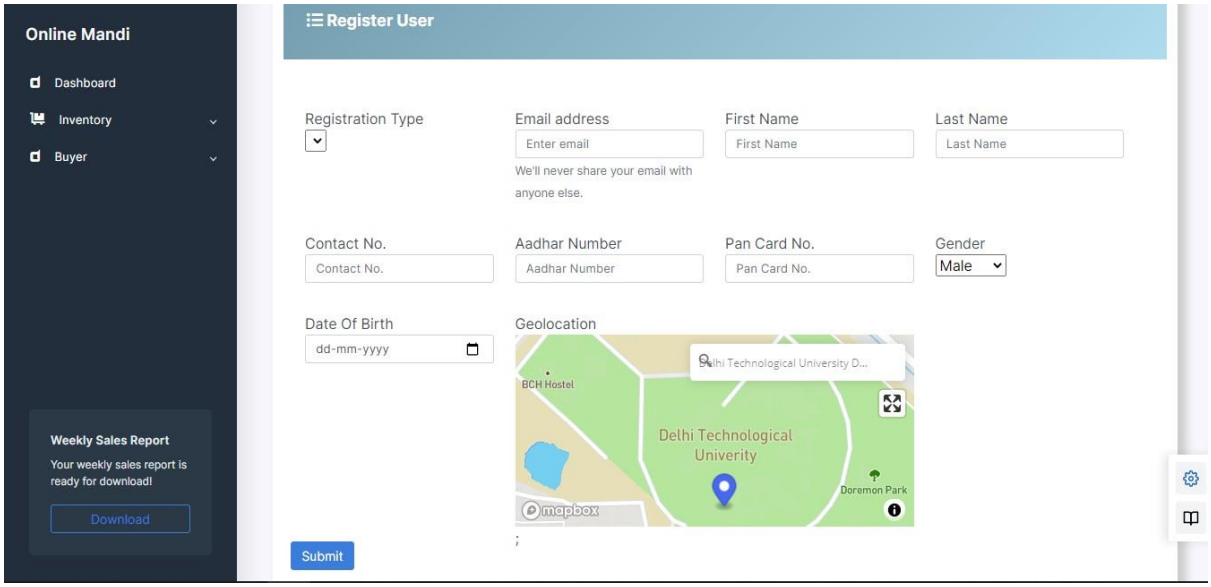

Fig. 41 Complete Registration for Buyer

4. Placing Orders for Crops

After buyer has completed the whole registration process, he can now place orders based on his requirements and on the basis of geospatial analysis provided by the application. Fig. 37 and 38 shows this implementation.

Fig. 42 Placing Order by Buyer

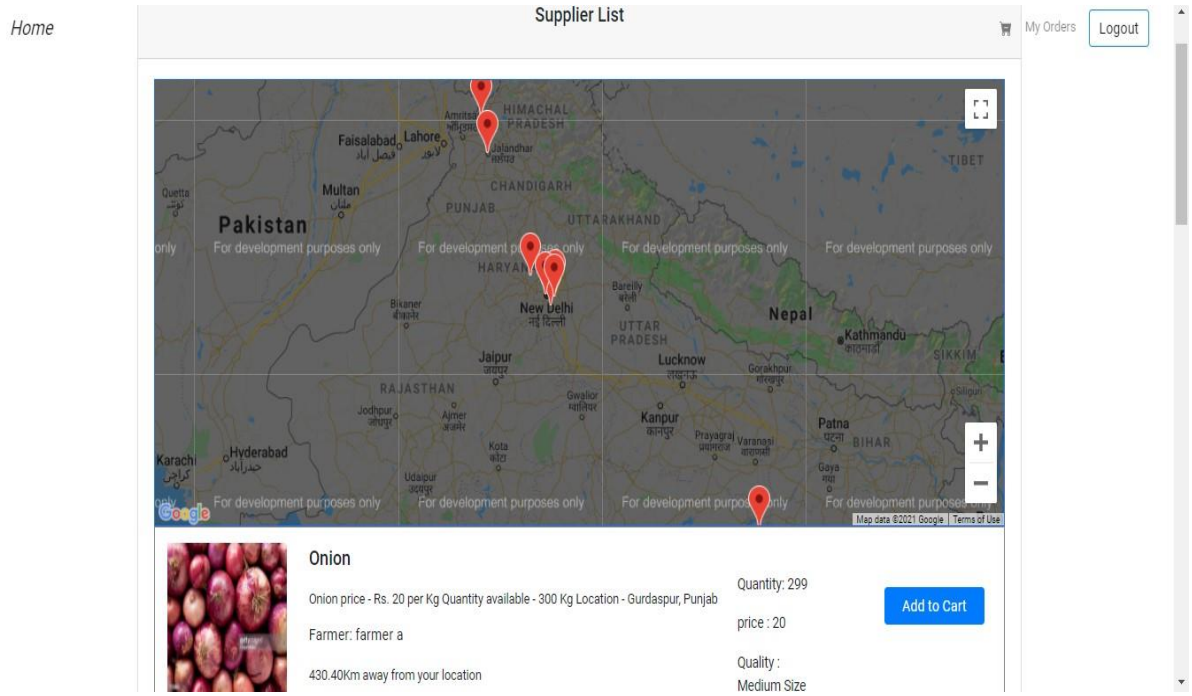

Fig. 43 Placing Order by Buyer

5. Managing Orders

After Placing the order, the buyer can view and manage the orders he has placed. Fig 43 shows this implementation.

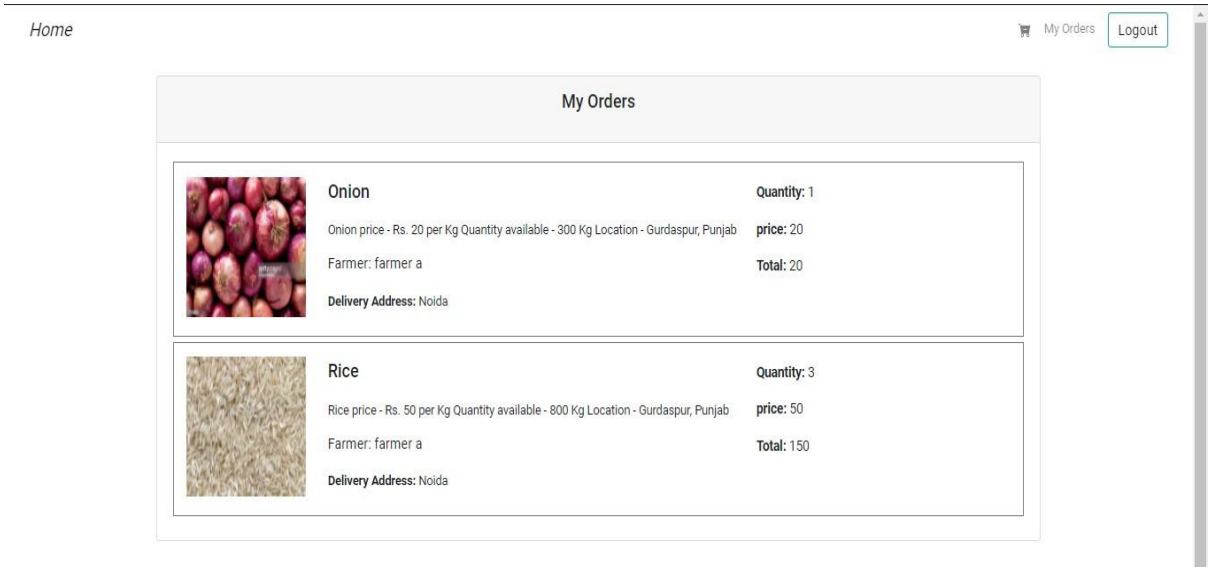

Fig. 44 Placing Order by Buyer

5.3.3 Implementation for Third Party Logistic

The step-by-step implementation of the third-party logistics' side of this WebGIS Application will be demonstrated.

1. Initial Registration

The third party logistic must initially register by providing the necessary information. Fig. 44 shows the implementation of initial registration process for third party logistic:

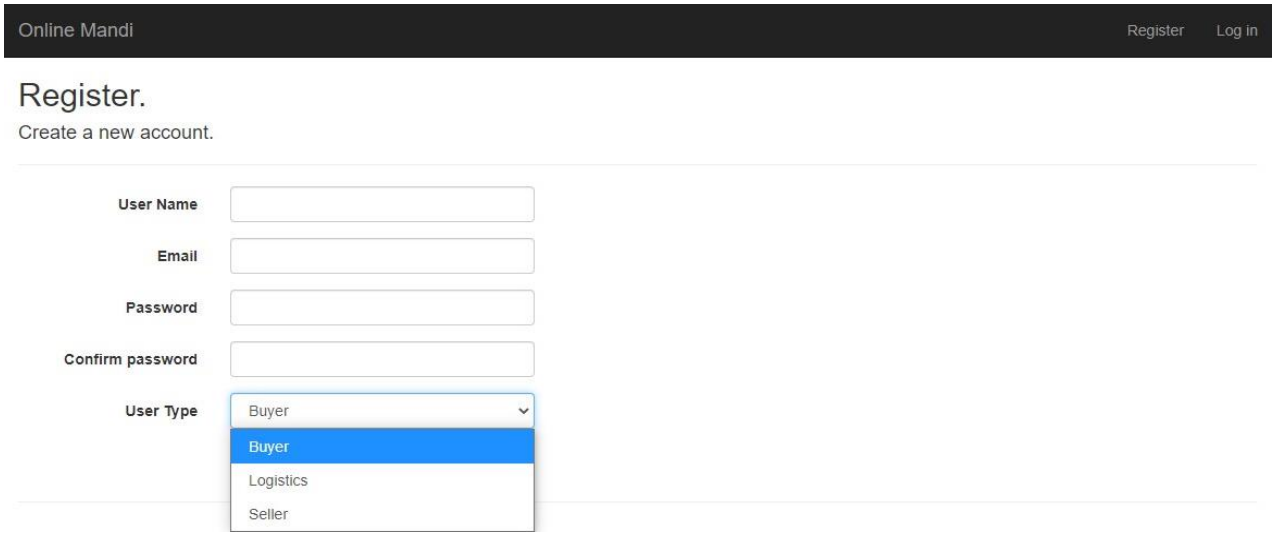

Fig. 45 Initial Registration for Third party Logistic

2. Login

After initial registration, third party logistic will login into the application. Fig. 41 shows this implementation.

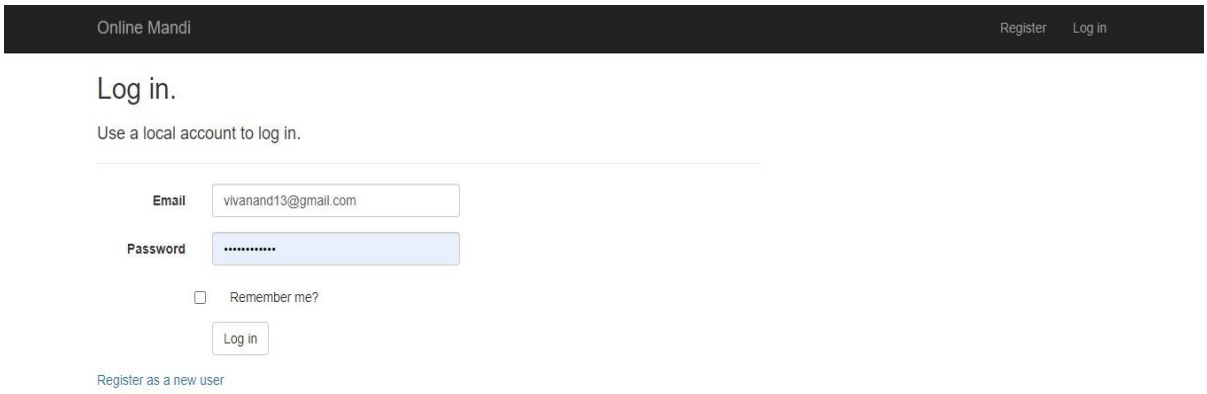

Fig. 46 Login page for Buyer

3. Complete Registration

After logging in to the application, the third party logistic will complete the full registration process. Fig. 42 shows this implementation.

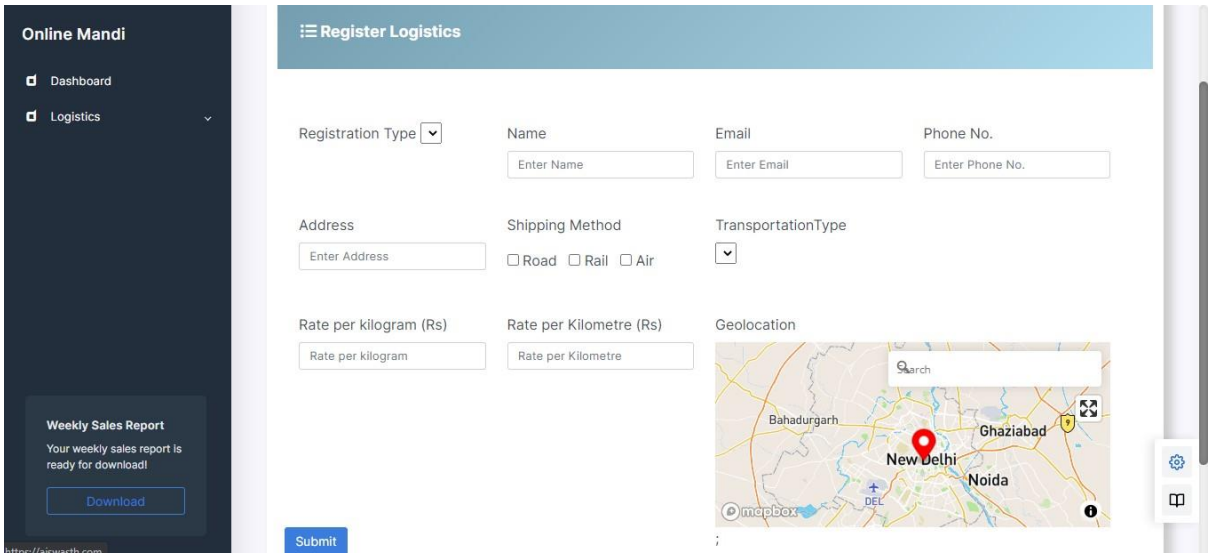

Fig. 47 Complete Registration for Third Party Logistic

4. Implementing the Logistic of Received Orders

The third party logistic can view and implement the logistic part of the trade between farmer and buyer on the basis of geospatial analysis and network analysis information provided by this WebGIS application. Fig. 43,44 and 45 shows this implementation

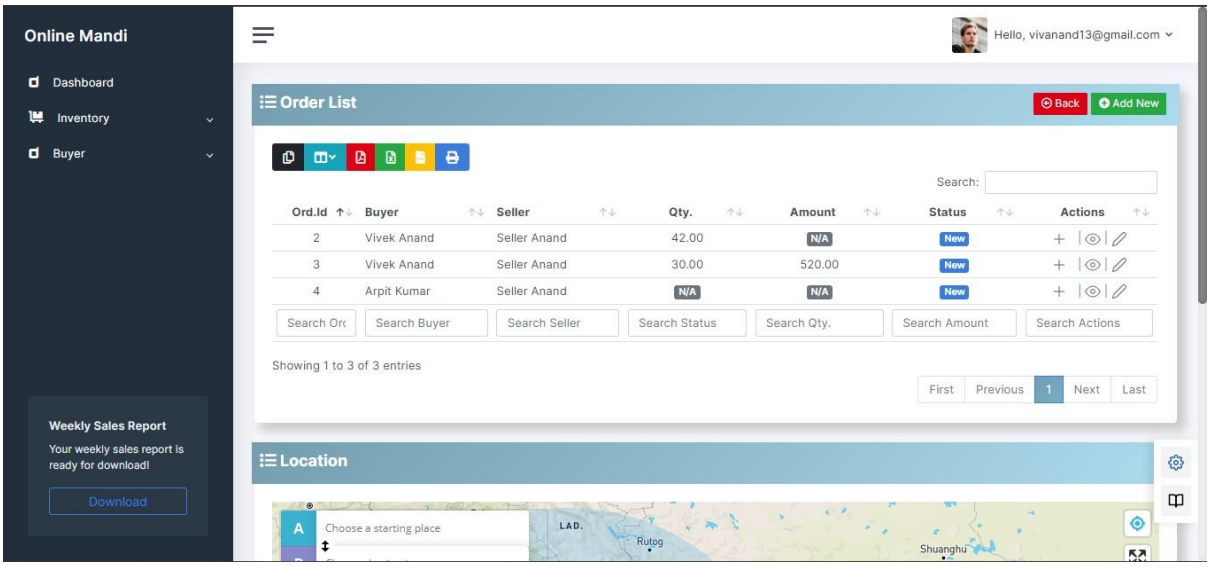

Fig. 48 Implementing the Logistic of Received Orders

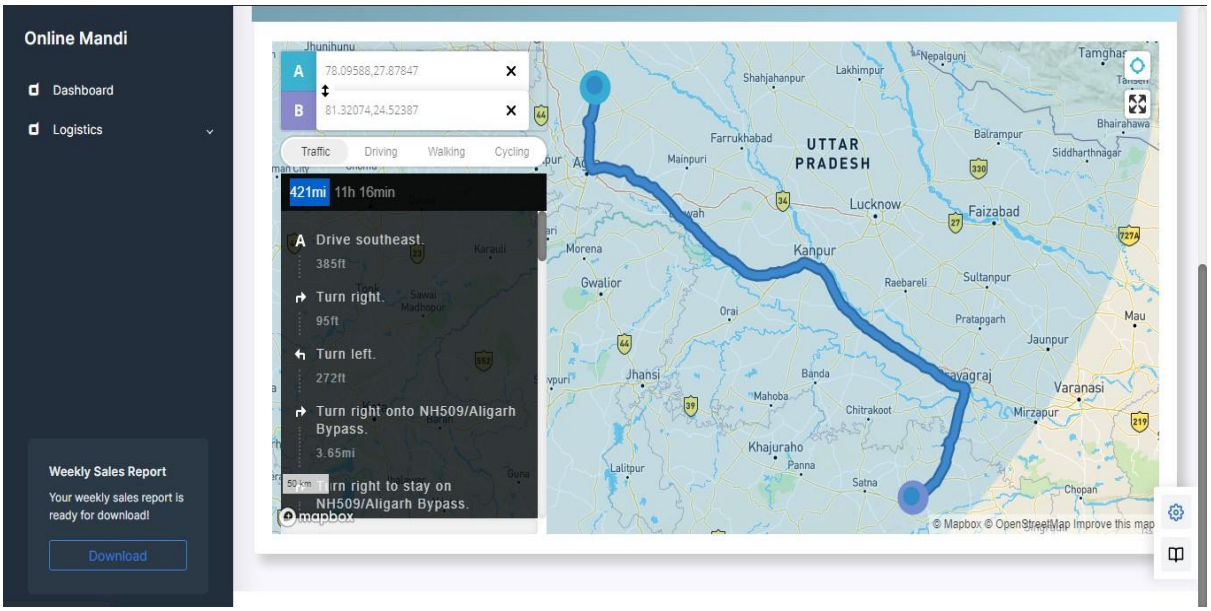

Fig. 49 Implementing the Logistic of Received Orders

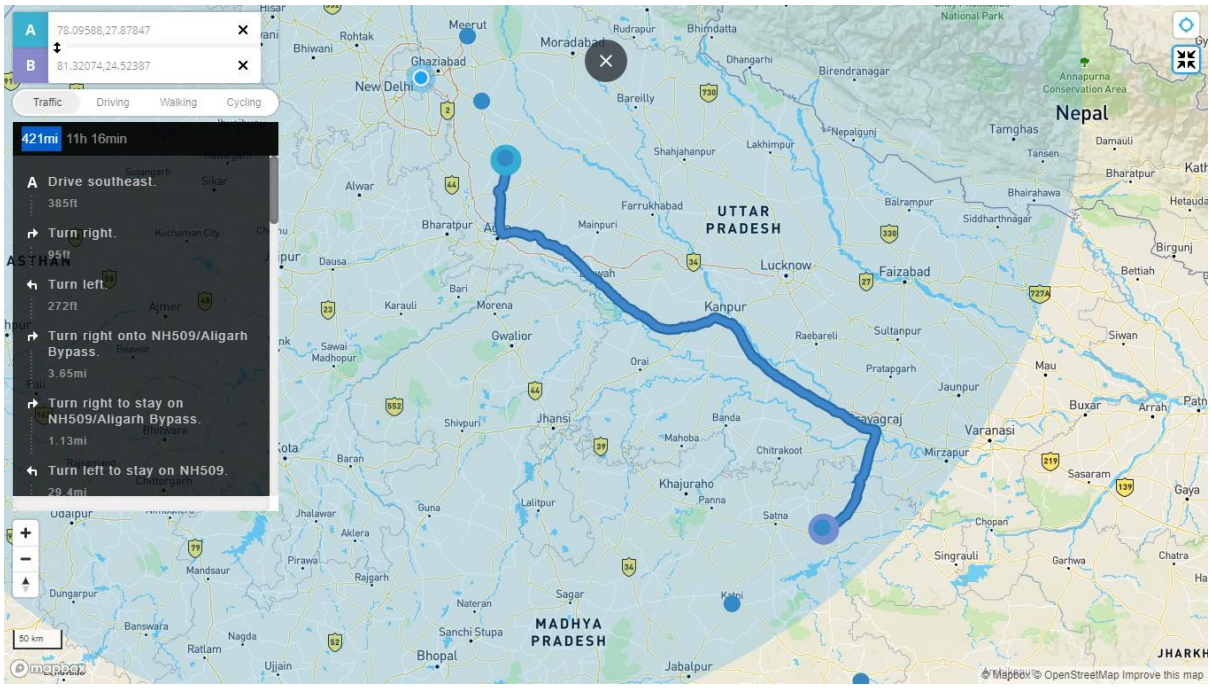

Fig. 50 Implementing the Logistic of Received

CHAPTER 6
CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

The Government of India passed the three new farm laws in the country in September 2020, which have been discussed thoroughly in *Chapter 2.* This research incorporated the implementation of the first farm law namely *The Farmers' Produce Trade and Commerce (Promotion and Facilitation) Act, 2020*. This law aims to revolutionize the current scenario of agricultural business in the country by allowing the farmers to sell their crops outside the predetermined APMC Mandis. This research worked in accordance with the first farm law and tried to create a marketplace for agricultural business, it implemented a traditional mandi type model on an online platform which used the power of geospatial data to enhance the experience for agricultural trade. This will allow the farmers to sell their produce outside the designated APMC Mandis to any part of the country which they were previously unable to do due to the restrictions imposed by the APMC committee and the lack of a geospatial analysis-based infrastructure for agricultural trade.

The key feature of this research is to introduce the idea of a geospatial analysis based agricultural trade platform which will use the geolocation data of the users involved in the process and with the help of location intelligence and geospatial analysis it will provide additional information to the users not only to expand the number of location a farmer and buyer gets to sell or buy the agricultural based produce but it will also provide information of the optimal route and shortest distance between them to make a calculated decision. The use of geospatial analysis is on the rise and every sector is realizing the power of geographically referenced data. So, it's time to utilize the advantages that the geolocation data has to offer in the area of agricultural trade in our country to allow the farmers to have a wider range of locations to sell their produce and this research has outlined the procedure to achieve this idea.

6.2 FUTURE SCOPE

There is a great need for revolutionizing the process in which the agricultural trade is being carried out in our country and this research proposed one way to enhance the said process but needless to say every study has certain limitations and this one is no exception. This research is in new and in its inceptive period as agricultural trade part of the Indian agriculture system has never been thought of with a geospatial analysis-based perspective.

This research can be extended to incorporate the second farm law namely *The Farmers (Empowerment and Protection) Agreement on Price Assurance and Farm Services Act, 2020, which* tries to give farmers the option to go into an agreement with agribusiness firms, processors, wholesalers, exporters, or huge retailers for the offer of future cultivating produce at a pre-agreed price. This platform can be extended to let farmers and agricultural businessmen from different locations come and meet and get into an agreement for the agriculture produce hence safeguarding the farmers of the country from price volatility.

In this research, use of Data Science and Artificial Intelligence on the geographically referenced data can enhance its capability even more and should be given a serious thought.

The following are some recommendations that can be incorporated in the future once this WebGIS application starts to function and acquires sufficient data from farmers and buyers:

1. The data acquired from the trade between farmers and buyer can be used for analysis and find the trends like which crop is more in demand for a particular region and a season based on multicriteria decision making and this information can be provided to the farmers using various visualization techniques so that from next harvest they can plan their crops accordingly.
2. Analysis on data can also provide the insights that which region has surplus of a particular crop and which region is in deficit so that the crop for surplus region can be bought in the deficit region to control the crop's price volatility that happens frequently in our country.
3. To incorporate the second farm law mentioned above, the agribusiness firms can enter the quantity of crop required by them beforehand and based on the land and other necessary data provided by the farmer and by using geospatial data science and satellite imagery analysis it can be used to estimate the area of land required to

grow that quantity of crop. Based on this finding a particular farmer or list of farmers can be provided to the agribusiness firms who can cater to their requirement and based on this information they can easily facilitate the process of contract farming.

4. This application can be extended to cover all the stages of agriculture supply chain right from procuring of seeds, fertilizers, and pesticides by the farmers, to the final act of trading the crop. The application can be further developed to include the suppliers and dealers of seeds, fertilizers, inclusion of heavy agriculture machinery contractors and all the people that have a hand in agriculture supply chain from different locations on a national level. Hence, this can be used as a complete marketplace for all the phases of agricultural cycle empowered by Geospatial Data Science.
5. Big data tracking solutions, smart meters, and GPS-oriented analytics can be used to improve routing, cutting transportation costs, and offering advanced mapping of the locations of agriculture produce and vehicles.
6. An AI powered information-based system can be provided to the farmers about the weather conditions, field characteristics, climatology insights and astronomy insights. It can be used to send farmers regular updates using every communication channel which can in turn be utilized by the farmers in planning for their crop.

These are some of the recommendations that can be utilized to take this WebGIS Application a step further however with the emerging use of Data Science and AI limitless application are possible.

REFERENCES

[1] Dr. Ashok Sahu, "Report Of The Working Group On Employment,Planning & Policy For The Twelfth Five Year Plan (2012-2017)," *Rep. Work. Gr. Employment,Planning Policy Twelfth Five Year Plan*, vol. 4, no. December 2011, pp. 9–15, 2017.

[2] C. S. OFFICE, "FIRST REVISED ESTIMATES OF NATIONAL INCOME, CONSUMPTION EXPENDITURE, SAVING AND CAPITAL FORMATION 2015-16," *Minist. Stat. Program. Implement.*, 2014.

[3] A. A. Cariappa, K. K. Acharya, C. A. Adhav, R. Sendhil, and P. Ramasundaram, "Impact of COVID-19 on the Indian agricultural system: A 10-point strategy for post-pandemic recovery," *Outlook Agric.*, vol. 50, no. 1, pp. 26–33, 2021, doi: 10.1177/0030727021989060.

[4] et al. Gyanendra Singh, Sudheer Kumar, RK Sharma, Sewa Ram, "Annual Report 2019," *ICAR-Indian Inst. Wheat Barley Res.*, 2019.

[5] A. Mishra, E. Bruno, and D. Zilberman, "Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID- 19 . The COVID-19 resource centre is hosted on Elsevier Connect , the company ' s public news and information," *Sci. Total Environ.*, no. January, 2020.

[6] S. M. DEV, "Addressing COVID-19 impacts on agriculture, food security, and livelihoods in India | IFPRI : International Food Policy Research Institute," *Addressing COVID-19 impacts on agriculture, food security, and livelihoods in India*. https://www.ifpri.org/blog/addressing-covid-19-impacts-agriculture-food-security-and-livelihoods-india (accessed Aug. 11, 2021).

[7] C. P. and P. AK, "Containing COVID19 impacts on Indian Agriculture – ICRISAT," *Containing COVID19 impacts on Indian Agriculture – ICRISAT*, 2020. https://www.icrisat.org/containing-covid19-impacts-on-indian-agriculture/ (accessed Aug. 11, 2021).

[8] J. Harris, L. Depenbusch, A. A. Pal, R. M. Nair, and S. Ramasamy, "Food system disruption: initial livelihood and dietary effects of COVID-19 on vegetable producers in India," *Food Secur.*, vol. 12, no. 4, pp. 841–851, 2020, doi: 10.1007/s12571-020-01064-5.

[9] Arabinda K Padhee and Prabhu Pingali, "Lessons from a pandemic to repurpose India's agricultural policy - Nature India," *Lessons from a pandemic to repurpose India's*

agricultural policy, 2020. https://www.natureasia.com/en/nindia/article/10.1038/nindia.2020.83 (accessed Aug. 11, 2021).

[10] K. Vikram, "1,550 tonnes food grains wasted at FCI godowns during lockdown, says government data- The New Indian Express," *The New Indian Express*, 2020. https://www.newindianexpress.com/nation/2020/oct/05/1550-tonnes-food-grains-wasted-at-fcigodowns-during-lockdown-says-government-data-2205893.html (accessed Aug. 11, 2021).

[11] V. Umarji, "Covid-19: Amul faces glitches in inter-state transport with 21-day lockdown | Business Standard News," *Bussiness Standard*, 2020. https://www.business-standard.com/article/companies/covid-19-amul-faces-glitches-in-inter-state-transport-with-21-day-lockdown-120032500390_1.html (accessed Aug. 11, 2021).

[12] P. M. Swaminathan, "From Green to Ever-Green Revolution," *From Green to Ever-Green Revolution, The Financial Express*, 2009. https://www.financialexpress.com/archive/from-green-to-ever-green-revolution/499699/ (accessed Aug. 11, 2021).

[13] S. Kumar, "Reorienting the Role and Restructuring of Food Corporation of India," *Shanta Kumar Comm. Rep.*, p. 78, 2015, [Online]. Available: http://fci.gov.in/report-publication.php.

[14] P. M. S. Swaminathan, "What is Swaminathan Report? - Farmers' Issues and Recommendations | Business Standard," *2006*. https://www.business-standard.com/about/what-is-swaminathan-report#collapse (accessed Aug. 11, 2021).

[15] G. Subramanian, Arvind Chief Economic Adviser, Ministry of Finance, "A National Market for Agricultural Commodities-Some Issues and the," *Econ. Surv. 2014-15*, vol. 1, p. Chapter 8, 2015.

[16] N. Jain, "National agriculture market: a bitter harvest - Telegraph India," *Telegraph India*, 2021. https://www.telegraphindia.com/opinion/national-agriculture-market-a-bitter-harvest/cid/1685544 (accessed Aug. 11, 2021).

[17] Agricultural Census Division, "Agriculture Census 2010-11: Phase II," *Dep. Agric. Coop. Minist. Agric. NEW DELHI*, vol. 11, pp. 1–65, 2015.

[18] ArcGIS, "Distance analysis—ArcGIS Pro | Documentation," *Esri*. https://pro.arcgis.com/en/pro-app/latest/tool-reference/spatial-analyst/distance-analysis.htm (accessed Aug. 11, 2021).

[19] R. J. T. Bell, "An elementary treatise on coordinate geometry of three dimensions : Bell,

Robert J. T. (Robert John Tainsh), 1877-1963 : Free Download, Borrow, and Streaming : Internet Archive." https://archive.org/details/elementarytreati00bell/page/56/mode/2up (accessed Aug. 11, 2021).

[20] M. of Defence, "Admiralty Manual of Navigation: BR 45(1) - Great Britain. Navy Department - Google Books," *TSO*. https://books.google.co.in/books?id=xcy4K5BPyg4C&pg=PA10&redir_esc=y#v=onepage&q&f=false (accessed Aug. 11, 2021).

[21] Eugene F.Krause, "Taxicab-Geometry_-an-adventure-in-non-Euclidean-geometry.pdf." .

[22] M. Sniedovich, "Dijkstra's algorithm revisited: the dynamic programming connexion," *Control and Cybernetics*, 2006. https://www.infona.pl/resource/bwmeta1.element.baztech-article-BAT5-0013-0005/tab/summary (accessed Aug. 11, 2021).

[23] P. S. Singh, D. Chutia, and S. Sudhakar, "Development of a Web Based GIS Application for Spatial Natural Resources Information System Using Effective Open Source Software and Standards," *J. Geogr. Inf. Syst.*, vol. 04, no. 03, pp. 261–266, 2012, doi: 10.4236/JGIS.2012.43031.

[24] K. CHAUHAN, "Asp . Net," vol. 2, no. 6, pp. 261–264, 2015.

[25] S. Cvetković and D. Janković, "A comparative study of the features and performance of ORM tools in a .NET environment," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 6348 LNCS, pp. 147–158, 2010, doi: 10.1007/978-3-642-16092-9_14.

[26] Microsoft, "Microsoft SQL Server 2019 preview 2 Copyright," 2018.

[27] J. Jokar Arsanjani, A. Zipf, P. Mooney, and M. Helbich, "An Introduction to OpenStreetMap in Geographic Information Science: Experiences, Research, and Applications," *Lect. Notes Geoinf. Cartogr.*, vol. 0, no. 9783319142791, pp. 1–15, 2015, doi: 10.1007/978-3-319-14280-7_1.

[28] W. Dictionary, "Crop | Definition of Crop by Merriam-Webster," *Webster Dictionary*. https://www.merriam-webster.com/dictionary/crop (accessed Aug. 12, 2021).

District Wise Ministry of Agriculture's Authorities Contact List

No.	District	Name	Contact	Eamil id
1.	Amritsar	Gurdial Singh Bal,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	98157-46461	caoamritsar@yahoo.com
2.	Barnsala	Baldev Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9888674820	caobarnala@gmail.com
3.	Bhatinda	Bahadur Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	8360494711	pdatmabathinda@gmail.com
4.	Faridakot	Harnek Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9815975824	caofaridkot@yahoo.com
5.	Fatehgarh Sahib	Inderpal Singh Sandhu,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9872012130	caofgs.agri@punjab.gov.in
6.	Firozpur	Gurmail ingh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9872104115	caofzr@gmail.com
7.	Gurdaspur	Raminder Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9914446456 127107	caogurdaspur@gmail.com
8.	Hoshiarpur	Vinnay Kumar, Chief Agriculture Officer,Department Of	9781126253 160969	Cao.hoshiarpur@yahoo.com

		Agriculture and Farmer Welfare		
9.	Jalandhar	Surinder Singh Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9781126259 139295	pdatmajalandhar@gmail.com
10	Kapurthala	Najar Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	8872007874	atmakapurthala@yahoo.in
11	Ludhiana	Narinder Singh Banipal,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	7508081704	caoludhiana2016@gmail.com
12	Mansa	Ram Sarup,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9878903224	chiefagricultureofficer.mansa@gmail.com
13	Moga	Balwinder Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	7986843908	caomoga@yahoo.com
14	Sri Muktsar S.	Jalor Singh,Chief Agriculture Office,Department Of Agriculture and Farmer Welfarer	8360189566	caomuktsar@gmail.com
15	Pathankot	Hartarnpal Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9888848683	caopathankot@gmail.com
16	Patiala	Surjit Singh Walia,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9416089063	caopatiala@gmail.com
17	Rupnagar	Avtar Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9915579840	caoropar@gmail.com
18	SAS Nagar	Ranjit Singh Bains,Chief Agriculture Officer,Department Of	9814371020	caomohali@yahoo.in

		Agriculture and Farmer Welfare		
19	Sangrur	Jaswinderpal Singh Gerwal,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9779213097	caosangrur@yahoo.com
20	SBS Nagar	Surinder Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	8872026510	cao.nawansahar@punjabmail.gov.in
21	Taran Tarn	Kuljit Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9517126991	caotarntaran@yahoo.in
22	Fazilka	Manjit Singh,Chief Agriculture Officer,Department Of Agriculture and Farmer Welfare	9814193502	dpdatmafazilka@gmail.com

Indian Farmer's Unions and eNAM Contact List

No.	Name of the Union	Phone No.	Email
1	Bhartiya Kisan Union	092195 71111	info@bku.co.in
2	All Indian Kisan Sabha	8181080000	info@kisansabha.in
3	All India Kisan Mazdoor Sabha (AIKMS)	-	aikms.cec@gmail.com
4	Karnataka Rajya Raitha Sangha (KRRS)	-	kodihallichandrashekar@gmail.com
5	eNAM	+91-11-26862367	nam@sfac.in enam.helpdesk@gmail.com

District Wise District Magistrate Contact List

Source:

1. http://districts.nic.in/districts.php?sid=BR
2. https://ceobihar.nic.in/DEOs_Cum_DMs_List.html
3. http://districts.nic.in/districts.php?sid=UP
4. http://uphome.gov.in/DM-UP-Contact.htm
5. http://districts.nic.in/districts.php?sid=PB
6. http://revenue.delhi.gov.in/wps/wcm/connect/doit_revenue/Revenue/Home/Telephone+Directory/List+of+DC+in+Districts
7. http://districts.nic.in/districts.php?sid=DL

BIHAR DISTRICT MAGISTRATE LIST				
Sl no:	**District name**	**District Magistrate Name**	**Email Address**	**Contact No:**
1	Araria	Shri Prashanth Kumar C.H. (IAS)	dm-araria.bih@nic.in	06453-222001
2	Arwal	Ms. J Priyadharshini,IAS,District Magistrate	dm-arwal.bih@nic.in	06337-228985
3	Aurangabad	Shri Saurabh Jorawal	dm-aurangabad.bih@nic.in	06186-223167
4	Banka	**Suharsha Bhagat, (I.A.S.)**	dm-banka.bih@nic.in	06424-222304
5	Begusarai	**Shri Arvind Kumar Verma, IAS**	dm-begusarai.bih@nic.in	9473191412
6	Bhagalpur	Subrat Kumar Sen	dm-bhagalpur.bih@nic.in	+91-641-2402200
7	Bhojpur	Sri Roshan Kushwaha	dm-bhojpur.bih@nic.in	06182-221312
8	Buxar	Shri Aman Samir	dm-buxar.bih@nic.in	06183-222336
9	Darbhanga	**Shri Thiyagrajan SM**	dm-darbhanga.bih@nic.in	06272240200
10	East Champaran	Shri Shirsat Kapil Ashok	dm-motihari.bih@nic.in	06252-242700
11	Gaya	**Sri Abhishek Singh, IAS**	dm-gaya.bih@nic.]in	0631-2222900
12	Gopalganj	Dr. Nawal Kishor Choudhary	dm-gopalganj.bih@nic.in	06156-226001
13	Jamui	Avanish Kumar Singh	dm-jamui.bih@nic.in	06345-222002
14	Jehanabad	Himanshu Kumar Rai, I.A.S	dm-jehanabad.bih@nic.in	06114-223072
15	Kaimur	Navdeep Shukla, IAS	dm-bhabhua.bih@nic.in	06189-223241
16	Katihar	Shri Udayan Mishra, IAS	dm-katihar.bih@nic.in	919473191375
17	Khagaria	Alok Ranjan Ghosh, I.A.S.	dm-khagaria.bih@nic.in	06244-222135
18	Kishanganj	*Aditya Prakash, IAS*	dm-kishanganj.bih@nic.in	06456-222530
19	Lakhisarai	Sri Sanjaya Kumar Singh	dm-lakhisarai.bih@nic.in	9473191397
20	Madhepura	SHYAM BIHARI MEENA	dm-madhepura.bih@nic.in	9473191353
21	Madhubani	Amit Kumar, I.A.S.	dm-madhubani.bih@nic.in	9473191324
22	Munger	Sri Navin Kumar I.A.S.	dm-munger.bih@nic.in	9473191391
23	Muzaffarpur	Mr. Pranav Kumar, IAS	dm-muzaffarpur.bih@nic.in	9473191283
24	Nalanda	Mr. Yogendra Singh	dm-nalanda.bih@nic.in	9473191214
25	Nawada	Mr. Yashpal Meena, IAS	dm-nawadah.bih@nic.in	9473191256
26	Patna	Dr.Chandrashekhar Singh, IAS	dm-patna.bih@nic.in	9473191198
27	Purnea	Rahul Kumar IAS	dm-purnea.bih@nic.in	9473191358
28	Rohtas	Dharmendra Kumar, IAS	dm-rohtas.bih@nic.in	9473191221

29	Saharsa	Kaushal Kumar, IAS	dm-saharsa.bih@nic.in	9473191340
30	Samastipur	Shri Shashank Subhankar	dm-samastipur.bih@nic.in	9473191332
31	Saran	Dr. Nilesh Ramchandra Deore, IAS	dm-saran.bih@nic.in	9473191267
32	Sheikhpura	Inayat Khan, IAS	dm-sheikhpura.bih@nic.in	9473191400
33	Sheohar	Shree Sajjan R,IAS	dm-sheohar.bih@nic.in	9473191468
34	Sitamarhi	Sunil Kumar Yadav, IAS	dm-sitamarhi.bih@nic.in	9473191288
35	Siwan	Shri Amit Kumar Pandey, IAS	dm-siwan.bih@nic.in	9473191273
36	Supaul	Shri Mahendra Kumar,I.A.S	dm-supaul.bih@nic.in	9473191345
37	Vaishali	Smt. Udita Singh, I.A.S	dm-vaishali.bih@nic.in	9473191310
38	West Champaran	Kundan Kumar	dm-bettiah.bih@nic.in	06254 – 232534

PUNJAB DISTRICT MAGISTRATE LIST				
Sl no:	**District name**	**District Magistrate Name**	**Email Address**	**Contact No:**
1	Amritsar	Sh. Gurpreet Singh Khaira, IAS	dc.asr@punjab.gov.in	0183-2223991
2	Barnala	Shri Tej Partap Singh Phoolka, I.A.S	dc.brn@punjab.gov.in	01679-244360
3	Bathinda	Sh. Praneet, IAS	dc.btd@punjab.gov.in	91-164-2210042
4	Faridkot	Sh. Vimal Kumar Setia, I.A.S.	dc.frd@punjab.gov.in	98881-52333
5	Fatehgarh Sahib	Ms. Surabhi Malik, IAS	dc.fgs@punjabmail.gov.in	01763-232215
6	Fazilka	Sh. Arvind Pal Singh Sandhu, IAS	dc.fzk@punjab.gov.in	01638-260555
7	Ferozepur	Sr. Gurpal Singh Chahal, IAS	dc.frz@punjab.gov.in	01632-244008
8	Gurdaspur	Sh. Mohammad Ishfaq, IAS	dc.grd@punjab.gov.in	01874-247500,
9	Hoshiarpur	Ms Apneet Riyait, IAS	dc.hsr@punjab.gov.in	01882220301
10	Jalandhar	Sh. Ghanshyam Thori, IAS	dc.jal@punjab.gov.in	2224783
11	Kapurthala	Ms. Deepti Uppal IAS	dc.kpr@punjab.gov.in	01822-297220
12	Ludhiana	Sh. Varinder Kumar Sharma, IAS	dc.ldh@punjab.gov.in	0161-2403100
13	Mansa	Mr. Mohinder Pal, IAS	dc.man@punjabmail.gov.in.	01652-227700
14	Moga	Sh. Sandeep Hans, IAS	dc.mog@punjabmail.gov.in	01636-234400
15	Muktsar	M K Aravind kumar ,IAS	dc.mks@punjab.gov.in	01633-263643
16	Nawanshahr	Dr. Shena Aggarwal, I.A.S	dc.nsr@punjab.gov.in	98727-51154
17	Pathankot	Sh. Sanyam Aggarwal , IAS	dc.pathankot@gmail.com	0186-2346464
18	Patiala	Kumar Amit IAS	dc.ptl@punjab.gov.in	0175-2311300
19	Rupnagar	Ms. Sonali Giri IAS	dc.rpr@punjab.gov.in	01881 – 221150
20	Sahibzada Ajit Sindh Nagar	Sh. Girish Dayalan	dc.sasnagar.punjab@gmai.com	01722219500
21	Sangrur	Sh. Ramvir, IAS	dc.sgr@punjab.gov.in	01672-234004
22	Tarn Taran	Sh. Kulwant Singh IAS	dc.ttn@punjabmail.gov.in	01852-224101

UTTAR PRADESH DISTRICT MAGISTRATE LIST

Sl no:	District name	District Magistrate Name	Email Address	Contact No:
1	Agra	Shri Prabhu N Singh	dmagr@nic.in	9454417509
2	Aligarh	Chandra Bhushan Singh I.A.S.	dmali@nic.in	9454417513
3	Allahabad	Shri Sanjay Kumar Khatri	dmall@nic.in	9454417517
4	Ambedkar Nagar	Shri Samuel Paul N IAS	dmamb@nic.in	9454417539
5	Amethi	Sh. Arun Kumar,IAS	dmamethi-up@nic.in	9454418891
6	Amroha	Shri Bal Krishna Tripathi, IAS	dmjpn@nic.in	9454417571
7	Auraiya	Shri Sunil Kumar Verma, I.A.	dmaur@nic.in	9454417550
8	Azamgarh	Rajesh Kumar (IAS)	dmaza@nic.in	9454417521
9	Baghpat	Shri Raj Kamal Yadav (IAS)	dmbag@nic.in	9454417562
10	Bahraich	Dr. Dinesh Chandra, I.A.S.	dmbeh@nic.in	9454417535
11	Ballia	Aditi Singh, IAS	dmbal@nic.in	9454417522
12	Balrampur	Smt. Shruti	dmblr@nic.in	9454417536
13	Banda	Shri Anand Kumar Singh(IAS)	dmban@nic.in	9454417531
14	Barabanki	Dr Adarsh Singh, IAS	dmbab@nic.in	9454417540
15	Bareilly	Mr. Nitish Kumar, IAS	dmbar@nic.in	9454417524
16	Basti	Saumya Agarwal	dmbas@nic.in	9454417528
17	Bhadohi	Ms. Aryaka Akhoury, I.A.S	dmsrn@nic.in	9454417568
18	Bijnor	Shri Umesh Mishra (IAS)	dmbij@nic.in	9454417570
19	Budaun	Mrs Deepa Ranjan (I.A.S.)	dmbud@nic.in	9454417525
20	Bulandshahr	Mr. Ravindra Kumar IAS	dmbul@nic.in	9454417563
21	Chandauli	Shri Sanjeev Singh, IAS	dmchn@nic.in	9454417576
22	Chitrakoot	Shubhrant Kumar Shukla (IAS)	dmchi@nic.in	9454417532
23	Deoria	Shri Ashutosh Niranjan IAS	dmdeo@nic.in	9454417543
24	Etah	ANKIT KUMAR AGRAWAL ,I.A.S	dmeth@nic.in	9454417514
25	Etawah	Mrs. Shruti Singh, I.A.S	dmetw@nic.in	9454417551
26	Faizabad	Anuj Kumar Jha	dmfai@nic.in	9454417541
27	Farrukhabad	Sh. Manvendra Singh, I.A.S.	dmfar@nic.in	9454417552
28	Fatehpur	Apurva Dubey (I.A.S)	dmfat@nic.in	9454417518
29	Firozabad	Chandra Vijay Singh IAS	dmfir@nic.in	9454417510
30	Gautam Buddha nagar	Mr. Suhas Lalinakere Yathiraj	dmgbn@nic.in	9971020646
31	Gaziabad	Shri Rakesh Kumar Singh	dmgha@nic.in	9454417565

		I.A.S.		
32	Ghazipur	Shri Mangala Prasad Singh	dmgaz@nic.in	9454417577
33	Gonda	Mr. Markandey Shahi (I.A.S.)	dmchi@nic.in	9454417532
34	Gorakhpur	Sh. K. Vijayendra Pandian (IAS)	dmgor@nic.in	9454417544
35	Hamirpur	Shri Gyaneshwar Tripathi IAS	dmham@nic.in	9454417533
36	Hapur	ANUJ SINGH, I.A.S	dmhap@nic.in	9454449848
37	Hardoi	Mr.Avinash Kumar	dmhar@nic.in	9454417556
38	Hathras	Ramesh Ranjan (IAS)	dmhat@nic.in	05722227076
39	Jalaun	Mrs. Priyanka Niranjan	dmjal@nic.in	9454417548
40	Jaunpur	Shri Manish Kumar Verma (I.A.S.)	dmjau@nic.in	9454417578
41	Jhansi	Shri Andra Vamsi (IAS)	dmjha@nic.in	9454417547
42	Kannauj	Rakesh Kumar Mishra IAS	dmknj@nic.in	9454417555
43	Kanpur Dehat	Shri Jitendra Pratap Singh I A S	dmkan@nic.in	9454417553
44	Kanpur Nagar	Sh. Alok Tiwari, IAS	dmkap@nic.in	9454417554
45	Kashiraam Nagar (kasganj)	Shri Chandra Prakash Singh IAS	dmkr-up@nic.in	9454417516
46	Kaushambi	Mr. SUJEET KUMAR, IAS	dmkos@nic.in	9454417519
47	Kushinagar	S. Rajalingam (IAS)	dmkus@nic.in	9454417545
48	Lakhimpur-Kheri	Dr. Arvind Kumar Chaurasiya, IAS	dmlkh@nic.in	9454417558
49	Lalitpur	annavi dineshkumar	dmlal@nic.in	9454417549
50	Lucknow	Shri Abhishek Prakash (I.A.S.)	dmluc@nic.in	9415005000
51	Maharajganj	Dr. Ujjwal Kumar (IAS)	dmmah@nic.in	9454417546
52	Mahoba	Mr. Satyendra Kumar(IAS)	dmmab@nic.in	9454417534
53	Mainpuri	Mahendra Bahadur Singh	dmmai@nic.in	9454417511
54	Mathura	Shri Navneet Singh Chahal, IAS	dmmat@nic.in	9454417512
55	Mau	Shri Amit Singh Bansal, IAS	dmmau@nic.in	9454417523
56	Meerut	Sh. K. Balaji IAS	dmmee@nic.in	9454417566
57	Mirzapur	Shri Praveen kumar laxkar (I.A.S.)	dmmir@nic.in	9454417567
58	Moradabad	Shri Shailendra Kumar Singh	dmmor@nic.in	9454417572
59	Muzaffarnagar	Mrs. Selva Kumari J. , IAS	dmmuz@nic.in	9454417574
60	Pilibhit	Pulkit Khare (I.A.S.)	dmpil@nic.in	9454417526
61	Pratapgarh	Dr.Nitin Bansal	dmpra@nic.in	9454417520
62	Raebareli	Vaibhav Srivastava	dmrae@nic.in	9454417559

		(I.A.S.)		
63	Rampur	Ravindra Kumar Mander	dmram@nic.in	9454417573
64	Saharanpur	Shri Akhilesh Singh IAS	dmsah@nic.in	9454417575
65	Sambhal	Mr. Sanjeev Ranjan	dmbm-up@nic.in	9454416890
66	Sant Kabir Nagar	Ms. Divya Mittal (IAS)	dmskn@nic.in	9454417529
67	Shahjahanpur	Indra Vikram Singh	dmsha@nic.in	9454417527
68	Shamli	Ms Jasjit Kaur, IAS	dmshm@nic.in	9454416996
69	Shravasti	Mr. T.K.Shibu IAS	dmshr@nic.in	9454417538
70	Siddharth nagar	Mr. Deepak Meena, IAS	dmsid@nic.in	9454417530
71	Sitapur	Mr. Vishal Bharadwaj	dmsit@nic.in	9454417560
72	Sonbhadra	Shri Abhishek Singh I.A.S.	dmson@nic.in	9454417569
73	Sultanpur	Mr. Raveesh Gupta, IAS	dmsul@nic.in	9454417542
74	Unnao	Shri Ravindra Kumar	dmunn@nic.in	9454417561
75	Varanasi	Shri Kaushal Raj Sharma (I.A.S.)	dmvar@nic.in	9454417579

DELHI DISTRICT MAGISTRATE LIST

Sl no:	**District name**	**District Magistrate Name**	**Email Address**	**Contact No:**
1	Central Delhi	Ms Akriti Sagar , IAS	dcentral@nic.in	23282903
2	East Delhi	Ms. Sonika Singh	dceast@nic.in	011-22021656
3	New Delhi	Dr. Monica Priyadarshini, IAS	dcnd@nic.in	9899465305
4	North Delhi	Mrs.Isha Khosla	dcnorth@nic.in	9899193642
5	North East Delhi	Ms.Geetika Sharma	dcne@nic.in	22122732
6	North West Delhi	Sh.Cheshta Yadav	dcnw@nic.in	9873745563
7	Shahdara	Shri Sanjeev Kumar	dcshah.rev.delhi@nic.in	9717549903
8	South Delhi	Dr. Ankita Chakravarty	dcsouth@nic.in	8289008855
9	South East Delhi	Sh.Viswendra	dcse.rev.delhi@nic.in	9717778491
10	South West Delhi	Mr.Naveen Aggarwal	dcsw@nic.in	9873873411
11	West Delhi	Ms.Kriti Garg	dcwest@nic.in	25107118

Synthetic Data Generated

User Database:

_id	firstName	lastName	geoLocation.lat	geoLocation.lon	address	email	contactNo	aadharNu	birthDate	gender	panNo	userType
60c76827c	farmer	a	32.041943	75.405334	Gurdaspui	abc1@gm	8.89E+09	8.89E+09	########	M	8.89E+09	farmer
60c76c83c	farmer	a	31.326015	75.57618	Jalandhar	abc2@gm	8.89E+09	8.89E+09	########	M	8.89E+09	farmer
60c7713ec	farmer	c	29	76.776695	Ambala, H	abc3@gm	1.13E+08	1.13E+08	########	M	1.13E+08	farmer
60d60976c	farmer	6	28.667856	77.449791	Ghaziabac	abc6@gm	9.84E+10	9.03E+10	########	M	3.95E+09	farmer
60d971671	farmer	h	28.6294016	77.185024	DTU, Delh	def8@gm	8.69E+12	9.87E+13	########	F	9.61E+13	farmer
60b92348c	abc	def	28.5999104	77.4373376	abc	abc@gma	4.75E+09	4.75E+09	########	M	4.75E+09	farmer
60c76e02c	buyer	a	28.5868032	77.4275072	Noida	def1@gm	1.23E+09	1.23E+09	########	M	1.23E+09	buyer
60d60e0fc	farmer	g	28.4112	77.3132	Faridabad	abc7@gm	9.82E+10	9.8E+10	########	M	93193109	farmer
60c7765fc	farmer	e	24	83.07863061	Sonbhadr	abc5@gm	2.12E+10	2.12E+10	########	M	2.12E+10	farmer
60c772c1c	farmer	d	21.812876	80.18383	Balaghat,	abc4@gm	2.12E+10	2.12E+10	########	M	2.12E+10	farmer

_id	firstName	lastName	geoLocation.lat	geoLocation.lon	address	email	contactNo	aadharNu	birthDate	gender	panNo	userType
60c76827c	farmer	m	32.041943	75.405334	Gurdaspur,	abc1@gm	8.89E+09	8.89E+09	########	M	8.89E+09	farmer
60c76c83c	farmer	n	31.326015	75.57618	Jalandhar P	abc2@gm	8.89E+09	8.89E+09	########	M	8.89E+09	farmer
60c7713ec	farmer	o	29	76.776695	Ambala, Ha	abc3@gm	1.13E+08	1.13E+08	########	M	1.13E+08	farmer
60d60976c	farmer	p	28.667856	77.449791	Ghaziabad,	abc6@gm	9.84E+10	9.03E+10	########	M	3.95E+09	farmer
60d971671	farmer	q	28.6294016	77.185024	DTU, Delhi	def8@gm	8.69E+12	9.87E+13	########	F	9.61E+13	farmer
60b92348c	abc	def	28.5999104	77.4373376	abc	abc@gma	4.75E+09	4.75E+09	########	M	4.75E+09	farmer
60c76e02c	buyer	a	28.5868032	77.4275072	Noida	def1@gm	1.23E+09	1.23E+09	########	M	1.23E+09	buyer
60d60e0fc	farmer	r	28.4112	77.3132	Faridabad,	abc7@gm	9.82E+10	9.8E+10	########	M	93193109	farmer
60c7765fc	farmer	s	24	83.07863061	Sonbhadra,	abc5@gm	2.12E+10	2.12E+10	########	M	2.12E+10	farmer
60c772c1c	farmer	t	21.812876	80.18383	Balaghat, M	abc4@gm	2.12E+10	2.12E+10	########	M	2.12E+10	farmer

_id	firstName	lastName	geoLocation.lat	geoLocation.lon	address	email	contactNo	aadharNu	birthDate	gender	panNo	userType
60c76827c	farmer	1	32.041943	75.405334	Gurdaspur,	abc1@gm	8.89E+09	8.89E+09	########	M	8.89E+09	farmer
60c76c83c	farmer	2	31.326015	75.57618	Jalandhar P	abc2@gm	8.89E+09	8.89E+09	########	M	8.89E+09	farmer
60c7713ec	farmer	3	29	76.776695	Ambala, Ha	abc3@gm	1.13E+08	1.13E+08	########	M	1.13E+08	farmer
60d60976c	farmer	4	28.667856	77.449791	Ghaziabad,	abc6@gm	9.84E+10	9.03E+10	########	M	3.95E+09	farmer
60d971671	farmer	5	28.6294016	77.185024	DTU, Delhi	def8@gm	8.69E+12	9.87E+13	########	F	9.61E+13	farmer
60b92348c	abc	def	28.5999104	77.4373376	abc	abc@gma	4.75E+09	4.75E+09	########	M	4.75E+09	farmer
60c76e02c	buyer	6	28.5868032	77.4275072	Noida	def1@gm	1.23E+09	1.23E+09	########	M	1.23E+09	buyer
60d60e0fc	farmer	7	28.4112	77.3132	Faridabad,	abc7@gm	9.82E+10	9.8E+10	########	M	93193109	farmer
60c7765fc	farmer	8	24	83.07863061	Sonbhadra,	abc5@gm	2.12E+10	2.12E+10	########	M	2.12E+10	farmer
60c772c1c	farmer	9	21.812876	80.18383	Balaghat, M	abc4@gm	2.12E+10	2.12E+10	########	M	2.12E+10	farmer

Crop Database

_id	cropNam	description	farmerId	price	quality	quantity
60c776c1c	Wheat	Wheat price - Rs. 29 per Kg Quantity available - 500 Kg Location - Sonbhadra,Uttar Pradesh	60c7765fca548a304c7d04f	29	UP-2338	500
60d971f81	Onion	Onion price - Rs. 28 per Kg Quantity available - 400 Kg Location - DTU, Delhi	60d971671318ad2fa4c735	28	Small Size	400
60d61102c	Onion	Onion price - Rs. 33 per Kg Quantity available - 300 Kg Location - Faridabad, Haryana	60d60e0fde77af28143467	31	Medium S	1000
60d9723c1	Wheat	Wheat price - Rs. 26 per Kg Quantity available - 1000 Kg Location - DTU, Delhi	60d971671318ad2fa4c735	26	A 108	1000
60c77189c	Wheat	Wheat price - Rs. 24 per Kg Quantity available - 800 Kg Location - Ambala, Haryana	60c7713eca548a304c7d04	24	WH-542	800
60d9726f1	Rice	Rice price - Rs. 50 per Kg Quantity available - 800 Kg Location - DTU, Delhi	60d971671318ad2fa4c735	50	Basmati	800
60d60bb8	Onion	Onion price - Rs. 33 per Kg Quantity available - 300 Kg Location -Ghaziabad, Uttar Pradesh	60d60976de77af28143467	33	Medium S	300
60c76d8fc	Onion	Onion price - Rs. 18 per Kg Quantity available - 400 Kg Location - Jalandhar, Punjab	60c76c83ca548a304c7d04e	18	Medium S	400
60c7730bc	Wheat	Wheat price - Rs. 30 per Kg Quantity available - 900 Kg Location - Balaghat, Madhya Pradesh	60c772c1ca548a304c7d04f	30	UP-2338	900
60d61083c	Rice	Rice price - Rs. 62 per Kg Quantity available - 1800 Kg Location - Faridabad, Haryana	60d60e0fde77af28143467	62	Basmati	1800
60c777bdc	Onion	Onion price - Rs. 24 per Kg Quantity available - 500 Kg Location - Sonbhadra, Uttar Pradesh	60c7765fca548a304c7d04f	24	Big Size	500
60c76d56c	Rice	Rice price - Rs. 44 per Kg Quantity available - 600 Kg Location - Jalandhar, Punjab	60c76c83ca548a304c7d04e	44	Basmati	600
60c76d0dc	Wheat	Wheat price - Rs. 22 per Kg Quantity available - 756 Kg Location - Jalandhar, Punjab	60c76c83ca548a304c7d04e	22	VL-832	756

_id	cropNam	description	farmerId	price	quality	quantity
60c776c1c	Wheat	Wheat price - Rs. 29 per Kg Quantity available - 500 Kg Location - Sonbhadra,Uttar Pradesh	60c7765fca548a304c7d04f	29	UP-2338	500
60d971f81	Onion	Onion price - Rs. 28 per Kg Quantity available - 400 Kg Location - DTU, Delhi	60d971671318ad2fa4c735	28	Small Size	400
60d61102c	Onion	Onion price - Rs. 33 per Kg Quantity available - 300 Kg Location - Faridabad, Haryana	60d60e0fde77af28143467	31	Medium S	1000
60d9723c1	Wheat	Wheat price - Rs. 26 per Kg Quantity available - 1000 Kg Location - DTU, Delhi	60d971671318ad2fa4c735	26	A 108	1000
60c77189c	Wheat	Wheat price - Rs. 24 per Kg Quantity available - 800 Kg Location - Ambala, Haryana	60c7713eca548a304c7d04	24	WH-542	800
60d9726f1	Rice	Rice price - Rs. 50 per Kg Quantity available - 800 Kg Location - DTU, Delhi	60d971671318ad2fa4c735	50	Basmati	800
60d60bb8c	Onion	Onion price - Rs. 33 per Kg Quantity available - 300 Kg Location -Ghaziabad, Uttar Pradesh	60d60976de77af28143467	33	Medium S	300
60c76d8fc	Onion	Onion price - Rs. 18 per Kg Quantity available - 400 Kg Location - Jalandhar, Punjab	60c76c83ca548a304c7d04e	18	Medium S	400
60c7730bc	Wheat	Wheat price - Rs. 30 per Kg Quantity available - 900 Kg Location - Balaghat, Madhya Pradesh	60c772c1ca548a304c7d04f	30	UP-2338	900
60d61083c	Rice	Rice price - Rs. 62 per Kg Quantity available - 1800 Kg Location - Faridabad, Haryana	60d60e0fde77af28143467	62	Basmati	1800
60c777bdc	Onion	Onion price - Rs. 24 per Kg Quantity available - 500 Kg Location - Sonbhadra, Uttar Pradesh	60c7765fca548a304c7d04f	24	Big Size	500
60c76d56c	Rice	Rice price - Rs. 44 per Kg Quantity available - 600 Kg Location - Jalandhar, Punjab	60c76c83ca548a304c7d04e	44	Basmati	600
60c76d0dc	Wheat	Wheat price - Rs. 22 per Kg Quantity available - 756 Kg Location - Jalandhar, Punjab	60c76c83ca548a304c7d04e	22	VL-832	756

The Source Code for the developed WebGIS Application

User.js

```
const mongoose = require('mongoose');
const bcrypt = require('bcrypt');
const Schema = mongoose.Schema;

const UserSchema = new Schema({
    email: {
        type: String,
        unique: true,
        lowercase: true,
        trim: true
    },
    firstName: {
        type: String,
        required: true,
        lowercase: true,
        trim: true
    },
    lastName: {
        type: String,
        lowercase: true,
        trim: true
    },
    contactNo: {
        type: Number,
        required: true,
        unique: true
    },
    aadharNumber: {
        type: Number,
        required: true,
        unique: true
    },
    panNo: {
        type: String,
        required: true,
        unique: true
    },
    gender: {
        type: String,
        required: true,
        enum: ['M', 'F']
    },
    birthDate: {
        type: String,
        required: true
    },
    password: {
        type: String,
        minlength: 7,
        required: true
    },
```

```
    address: {
        type: String,
        required: true
    },
    userType: {
        type: String,
        required: true,
        enum: ['farmer', 'buyer'],
        default: 'buyer'
    },
    geoLocation: {
        lat: {type: Number, required: true},
        lon: {type: Number, required: true}
    }
}, {
    timestamps: { createdAt: 'createdAt', updatedAt: 'updatedAt' }
});

UserSchema.pre('save', function(next) {

    const user = this;

    if(!user.isModified('password')) return next();

    bcrypt.genSalt(4, function(err, salt) {
        bcrypt.hash(user.password, salt, (err, hash) => {
            if (err) return next(err);
            user.password = hash;
            next();

        });
    });
});

UserSchema.methods.comparePassword = function(candidatePassword, cb) {
    bcrypt.compare(candidatePassword, this.password, function(err,
isMatch) {
        if (err) return cb(err);
        cb(null, isMatch);
    });
};


module.exports = mongoose.model('User', UserSchema);
```

Controller.js

```
const utils = require("../utility/util");
const User = require('../model/user.js')
const Crop = require('../model/crop')
const {Order} = require('../model/cart')


var farmerController = {

    addCrop: async function(req, res) {
        try {
```

```
            // const token = req.session.JWT_TOKEN;
            // if(!token) {
            //     return res.sendStatus(401)
            // }
            const { cropName, price, quantity, quality, description,
imageUrl } = req.body;
            const farmer = req.user

            const farmerExist = await User.exists({ _id: farmer.id,
userType: 'farmer' });

            if(!farmerExist) return res.status(200).json({ "status":
false});

            const crop = new Crop({ cropName, price, quality, quantity,
imageUrl,  description, farmerId: farmer.id });
            const cropProduct = await crop.save();

            if(!cropProduct) return res.status(200).json({ "status":
"DB UPDATE FAILED!!" });

            return res.status(200).json({ "status": true, cropProduct
})

        } catch(error) {
            console.log(error)
        }
    },
    updateCrop: async function(req, res) {
        try {
            // const token = req.session.JWT_TOKEN;
            // if(!token) {
            //     res.sendStatus(401)
            // }
            const { cropName, price, quantity, quality, description,
imageUrl, cropId } = req.body;
            const farmer = req.user

            const farmerExist = await User.exists({ _id: farmer.id,
userType: 'farmer' });

            if(!farmerExist) return res.status(200).json({ "status":
false});

            const cropExist = await Crop.exists({ _id: cropId,
farmerId: farmer.id });

            if(!cropExist) return res.status(200).json({ "status":
false });

            var updateCrop = await Crop.updateOne({ _id: cropId,
farmerId: farmer.id }, { cropName, price, quality, quantity, imageUrl,
description })

            res.status(200).json({"status": true, crop: {cropName,
price, quantity, quality, description, imageUrl, cropId}});

        } catch(error) {
```

```
            console.log(error)
            res.sendStatus(500)
        }

    },
    deleteCrop: async function(req, res) {
        try {

            // const token = req.session.JWT_TOKEN;
            // if(!token) {
            //     return res.sendStatus(401)
            // }
            const { cropId } = req.body;
            const farmer = req.user

            const farmerExist = await User.exists({ _id: farmer.id,
userType: 'farmer' });

            if(!farmerExist) return res.status(200).json({ "status":
false});

            const cropExist = await Crop.exists({ _id: cropId,
farmerId: farmer.id });

            if(!cropExist) return res.status(200).json({ "status":
false });

            cropDeleted = await Crop.deleteOne({ _id: cropId })

            if(cropDeleted) return res.status(200).json({"status":
true, "deleted": true})

            res.sendStatus(500)

        } catch(error){
            console.log(error)
            res.sendStatus(500)
        }

    },
    getCrop: async function(req, res) {
        try {
            // const token = req.session.JWT_TOKEN;
            // if(!token) {
            //     return res.sendStatus(401)
            // }
            const { cropId } = req.body;
            const farmer = req.user

            const farmerExist = await User.exists({ _id: farmer.id,
userType: 'farmer' });

            if(!farmerExist) return res.status(200).json({ "status":
false});
            var options = {
                farmerId: farmer.id,
            }
            if(cropId) options._id = cropId;
```

```
            const cropList = await Crop.find(options).select({
'createdAt': 0, 'updatedAt': 0, 'farmerId': 0 })
            if(!cropList) res.status(200).json({"status": true, crop:
[], size: 0})
            res.status(200).json({"status": true, crop: cropList, size:
cropList.length})
        } catch(error) {
            console.log('Error ', error)
            res.sendStatus(500)
        }
    },
    myOrders: async function(req, res) {
        try {
           /* var user = req.user;

            var orderList = await Order.find({farmerId:
user.id}).lean();

            if(!orderList) return res.status(200).json({status: false})

            res.status(200).json({status: true, orderList, length:
orderList.length})*/
            var user = req.user;

            var orderList = await Order.find({farmerId:
user.id}).populate({
                path: "crops.cropId",
                select: "cropName price farmerId imageUrl description",
                populate: {
                    path: "farmerId",
                    select: "firstName lastName address"
                }
            }).lean()

            if(!orderList) return res.status(200).json({status: false})
            orderList.forEach(order => {
                order.crops = order.crops[0]
            })
            //orderList.crops = orderList.crops[0]
            res.status(200).json({status: true, orderList, length:
orderList.length})

        } catch(error) {
            console.log(error);
            res.sendStatus(500);
        }
    }
}

module.exports = farmerController;
```

Model.js

```
const Crop = require("../model/crop");
const User = require("../model/user");
const {Cart, Order} = require("../model/cart");
const utils = require('../utility/util');
```

```
var farmerController = {

    cropList: async function(req, res) {
        try {

            var cropList = await Crop.distinct("cropName", {"quantity":
{ $gt: 0 }})

            res.status(200).json({"status": true, crops: cropList,
length: cropList.length})

        } catch (error) {
            console.log(error);
            res.sendStatus(500);
        }
    },
    getCropBuyersList: async function(req, res) {
        try {

            var user = req.user;

            var buyer = await User.findOne({_id: user.id, userType:
user.userType}).lean();

            const cropName = req.body.cropName;

            var buyerGeolocation = buyer.geoLocation

            if(!cropName) return res.status(200).json({"status":
false})

            const buyerList = await Crop.find({"cropName": cropName,
"quantity": { $gt: 0 }}).populate({
                path: "farmerId",
                select: "firstName lastName contactNo address
geoLocation"
            }).lean()

            buyerList.forEach(buyer => {
                var farmerLocation = buyer.farmerId.geoLocation;
                buyer.distanceFromFarmer =
Number(utils.distance(buyerGeolocation.lat, buyerGeolocation.lon,
farmerLocation.lat, farmerLocation.lon, 'K')).toFixed(2)+'Km'
            })

            return res.status(200).json({"status": true, cropList:
buyerList, length: buyerList.length, buyerGeolocation})

        } catch(error) {
            console.log(error);
            res.sendStatus(500);
        }
    },
    getCartItems: async function(req, res) {
        const user = req.user;

        try {
```

```
            const cart = await Cart.findOne({ userId: user.id
}).populate({
                path: "items.cropId",
                select: "cropName price farmerId imageUrl description",
                populate: {
                    path: "farmerId",
                    select: "firstName lastName address"
                }
            }).select({ "userId": 0 });

            res.status(200).json({
                cart
            });


        } catch(err) {
            console.log(err);
            res.sendStatus(500);
        }
    },
    addCartItems: async function(req, res) {
        const  cropId = req.body.cropId;
        const  quantity = Number(req.body.quantity);

        const user = req.user;
        try {

            const cropDetails = await Crop.findById(cropId).lean();

            // Product Id doesn't exist in Product Collection
            if(!cropDetails) return res.status(200).json({
                status: false
            });

            if(quantity > cropDetails.quantity) {
                return res.status(200).json({"status": true, "message":
"Quantity not available"})
            }

                // Fetch user Cart
            const cartDetails = await Cart.findOne({ userId: user.id
});

            if(cartDetails) {
                // check if Product Id already exist
                const indexFound = cartDetails.items.findIndex(item =>
item.cropId == cropId );

                // Remove item from the cart if quantity is set to zero
                if( indexFound !== -1 && quantity == 0) {
                    cartDetails.items.splice(indexFound, 1);
                    if(cartDetails.items.length == 0) {
                        cartDetails.subTotal = 0;
                    } else {
                        cartDetails.subTotal =
cartDetails.items.map(item => item.total).reduce((acc, next) => acc +
next);
                    }
```

```
                } else if(quantity == -1 &&
cartDetails.items[indexFound].quantity + quantity == 0) {
                    cartDetails.items.splice(indexFound, 1);
                    if(cartDetails.items.length == 0) {
                        cartDetails.subTotal = 0;
                    } else {
                        cartDetails.subTotal =
cartDetails.items.map(item => item.total).reduce((acc, next) => acc +
next);
                    }
                }
                else if(indexFound !== -1) {
                    cartDetails.items[indexFound].quantity =
cartDetails.items[indexFound].quantity + quantity;
                    cartDetails.items[indexFound].total =
cartDetails.items[indexFound].quantity * cropDetails.price;
                    cartDetails.items[indexFound].price =
cropDetails.price;
                    cartDetails.subTotal = cartDetails.items.map(item
=> item.total).reduce((acc, next) => acc + next);
                  //  await Crop.updateOne({_id: cropId}, {quantity:
cropDetails.quantity - quantity})
                } else if (quantity > 0) {
                    cartDetails.items.push({
                        cropId,
                        quantity,
                        price: cropDetails.price,
                        total: parseInt(cropDetails.price * quantity)
                    });
                  // await Crop.updateOne({_id: cropId}, {quantity:
cropDetails.quantity - quantity})

                    cartDetails.subTotal = cartDetails.items.map(item
=> item.total).reduce((acc, next) => acc + next);

                } else {
                    return res.status(400).json({
                        "status": false
                    })
                }

                const data = await cartDetails.save();
                res.status(200).json({
                    "status" : true,
                    data: data
                })
            } else {
                const cartData = {
                    items: [{
                        cropId: cropId,
                        quantity: quantity,
                        price: cropDetails.price,
                        total: parseInt(cropDetails.price * quantity)
                    }],
                    subTotal: parseInt(cropDetails.price * quantity),
                    userId: user.id
                }
```

```
                const cartAdd = await Cart.create(cartData);
              //  await Crop.updateOne({_id: cropId}, {quantity:
cropDetails.quantity - quantity})

                res.status(200).json({
                    "status": "Process Completed",
                    cartAdd
                })
            }
        } catch(err) {
            console.log(err);
            res.sendStatus(500);
        }
    },
    removeCartItems: async function(req, res) {
        const cropId = req.body.cropId;
        const user = req.user;

        try {

            const cart = await Cart.findOne({ userId: user.id });

            if(!cart) return res.status(200).json({ "status": true });

            const indexFound = cart.items.findIndex(item => item.cropId
== cropId );

            if(indexFound !== -1) {
                cart.items.splice(indexFound, 1);
                    if(cart.items.length == 0) {
                        cart.subTotal = 0;
                    } else {
                        cart.subTotal = cart.items.map(item =>
item.total).reduce((acc, next) => acc + next);
                    }
            }

            const cartSave = await cart.save();

            res.status(200).json({ "status": true,
                cartSave });


        } catch(error) {
            console.log(error);
            res.sendStatus(500);
        }
    },
    placeOrder: async function(req, res) {
        const user = req.user;
        try {

            const userDetails = await User.findOne({_id:
user.id}).lean();

            const cartDetails = await Cart.findOne({ userId: user.id
}).populate({
                path: "items.cropId" ,
```

```
                select: "cropName price"
            });


            if(!cartDetails || cartDetails.items.length == 0) return
res.status(200).json({"status": "Invalid request"});

            let cartItems = cartDetails.items;

            let productList = [];

            for(item of cartItems) {

                let productOrder = {
                    cropId: item.cropId._id,
                    quantity: item.quantity,
                    price: item.cropId.price,
                    total: item.quantity * item.cropId.price
                };

                var cropDetails = await Crop.findOne({_id:
item.cropId._id}).lean()

                await Crop.updateOne({_id: item.cropId._id}, {quantity:
cropDetails.quantity - item.quantity})

                let finalOrder = new Order({
                    crops: productOrder,
                    userId: user.id,
                    address: userDetails.address,
                    status: "Processing",
                    subTotal: item.quantity * item.cropId.price,
                    farmerId:  cropDetails.farmerId
                })
                productList.push(finalOrder)

            }

            var orderSave = await Order.insertMany(productList);

            if(!orderSave) return res.status(200).json({ "status":
false })

            cartDetails.items = [];
            cartDetails.subTotal = 0;

            const cartEmpty = await cartDetails.save();

            res.status(200).json({status: true, orderSave});



        } catch (err) {
            console.log(err)
            res.sendStatus(500);
        }
    },
    myOrders: async function(req, res) {
```

```
        try {

            var user = req.user;

            var orderList = await Order.find({userId:
user.id}).populate({
                path: "crops.cropId",
                select: "cropName price farmerId imageUrl description",
                populate: {
                    path: "farmerId",
                    select: "firstName lastName address"
                }
            }).lean()
            if(!orderList) return res.status(200).json({status: false})
            orderList.forEach(order => {
                order.crops = order.crops[0]
            })
            //orderList.crops = orderList.crops[0]
            res.status(200).json({status: true, orderList, length:
orderList.length})

        } catch(error) {
            console.log(error);
            res.sendStatus(500);
        }
    }

}

module.exports = farmerController;
```

distance.py

```
import os
import osgeo.ogr
import shapely.geometry
import shapely.wkt
import psycopg2
import pyproj
import pprint
from shapely import speedups

speedups.disable()

def calc_distance(point, geometry):
    return shapely.geometry.Point(point).distance(geometry)

def calc_score(route_segments):
    total = 0
    for segment in route_segments:
        total = total + sum(segment['gps_distances'])
    return total

def calc_circle_with_radius(center_point, radius):
    geod = pyproj.Geod(ellps="WGS84")
    sLong,sLat = center_point
    eLong,eLat,iHeading = geod.fwd(sLong, sLat, 0, radius)
```

```
    lat_delta = abs(sLat - eLat)
    return shapely.geometry.Point(sLong, sLat).buffer(lat_delta)

def point_at_start_of_segment(next_point, segment):
    num_points = len(segment['gps_points'])
    if num_points > 0:
        average_distance = sum(segment['gps_distances']) / num_points
        startpoint_coord = segment['linestring'].coords[0]
        startpoint = shapely.geometry.Point(startpoint_coord)
        endpoint_coord = segment['linestring'].coords[-1]
        endpoint = shapely.geometry.Point(endpoint_coord)
        distance_to_start = calc_distance(next_point, startpoint)
        distance_to_end = calc_distance(next_point, endpoint)
        if distance_to_start < 2 * average_distance:
            if distance_to_end > 2 * average_distance:
                return True
    return False

def develop_route(next_point, route, route_candidates, cursor):
    if len(route['segments']) == 1:
        if point_at_start_of_segment(next_point,route['segments'][0]):
            return []
    last_segment = route['segments'][-1]

    if point_in_route_segment(next_point, last_segment):
        next_distance =
calc_distance(next_point,last_segment['linestring'])
        last_segment['gps_points'].append(next_point)
        last_segment['gps_distances'].append(next_distance)
        route['score'] = calc_score(route['segments'])
        return [route]

    last_point = last_segment['linestring'].coords[-1]

    endpoint = shapely.geometry.Point(last_point)
    cursor.execute("SELECT id FROM endpoints " +
     "WHERE endpoint=ST_GeomFromText(%s)",
     (endpoint.wkt,))
    endpoint_id = cursor.fetchone()[0]

    possible_segment_ids = []
    cursor.execute("SELECT directed_segment_id " +
     "FROM endpoint_segments " +
     "WHERE endpoint_id=%s", (endpoint_id,))
    for row in cursor:
        possible_segment_ids.append(row[0])

    new_candidates = []
    for directed_segment_id in possible_segment_ids:
        cursor.execute("SELECT road_segment_id," +
         "ST_AsText(centerline) " +
         "FROM directed_segments " +
         "WHERE id=%s", (directed_segment_id,))
        road_segment_id,wkt = cursor.fetchone()
        linestring = shapely.wkt.loads(wkt)

        next_distance = calc_distance(next_point, linestring)
```

```
            new_segment = {}
            new_segment['directed_segment_id'] = directed_segment_id
            new_segment['linestring'] = linestring
            new_segment['gps_points'] = [next_point]
            new_segment['gps_distances'] = [next_distance]

            new_candidate = {}
            new_candidate['segments'] = []
            new_candidate['segments'].extend(route['segments'])
            new_candidate['segments'].append(new_segment)
            new_candidate['directed_segment_ids'] = []

new_candidate['directed_segment_ids'].extend(route['directed_segment_id
s'])

new_candidate['directed_segment_ids'].append(directed_segment_id)

            if not route_is_valid(new_candidate,
route_candidates,new_candidates):
                continue
                new_candidate['score'] =
calc_score(new_candidate['segments'])
                new_candidates.append(new_candidate)

    return new_candidates

def point_in_route_segment(point, segment):
    endpoint = shapely.geometry.Point(segment['linestring'].coords[-1])
    distance_to_linestring = calc_distance(point,segment['linestring'])
    distance_to_endpoint = calc_distance(point, endpoint)
    if distance_to_linestring == distance_to_endpoint:
        return False

    gps_coords = []
    gps_coords.extend(segment['gps_points'])
    gps_coords.append(point)
    gps_length = shapely.geometry.LineString(gps_coords).length
    segment_length = segment['linestring'].length

    if gps_length > segment_length:
        return False
    return True

def route_is_valid(route, route_candidates, new_candidates):
    route_roads = route['directed_segment_ids']
    for other_route in route_candidates:
        if route_roads == other_route['directed_segment_ids']:
            return False

    for other_route in new_candidates:
        if route_roads == other_route['directed_segment_ids']:
            return False

    if len(route['segments']) >= 2:
        last_segment = route['segments'][-1]
        prev_segment = route['segments'][-2]
        last_segment_end = last_segment['linestring'].coords[-1]
        prev_segment_start = prev_segment['linestring'].coords[0]
```

```
        if last_segment_end == prev_segment_start:
            return False

    directed_segment_ids = set()
    for segment in route['segments']:
        directed_segment_id = segment['directed_segment_id']
        if directed_segment_id in directed_segment_ids:
            return False
        else:
            directed_segment_ids.add(directed_segment_id)
    return True

gps_tracks = []
for fName in os.listdir("E:/Circle/2. Urban Mobility/Phase 2-Map
Matching/gps-data"):
    if fName.endswith(".gpx"):
        srcFile = osgeo.ogr.Open("E:/Circle/2. Urban Mobility/Phase 2-
Map Matching/gps-data/" + fName)
        layer = srcFile.GetLayerByName("tracks")

        for feature_num in range(layer.GetFeatureCount()):
            feature = layer.GetFeature(feature_num)
            geometry = feature.GetGeometryRef()

            if geometry.GetGeometryName() == "MULTILINESTRING":
                for geom_num in range(geometry.GetGeometryCount()):
                    wkt =
geometry.GetGeometryRef(geom_num).ExportToWkt()
                    gps_tracks.append((fName, wkt))
            elif geometry.GetGeometryName() == "LINESTRING":
                wkt = geometry.ExportToWkt()
                gps_tracks.append((fName, wkt))
connection = psycopg2.connect(database="gps_heatmap",
 user="postgres",password="password")
cursor = connection.cursor()

cursor.execute("UPDATE road_segments SET tally=0")
connection.commit()

def gpsPoints(gps_track):
    x_cord = gps_track.xy[0]
    y_cord = gps_track.xy[1]
    gps_points = []
    for i in range(len(gps_track.xy[0])):
        gps_points.append((x_cord[i],y_cord[i]))
    return(gps_points)

for fName,track_wkt in gps_tracks:
    print ("Processing " + fName)
    gps_track = shapely.wkt.loads(track_wkt)

    gps_points = gpsPoints(gps_track)
    while len(gps_points) > 0:
        circle = calc_circle_with_radius(gps_points[0], 10)
        cursor.execute("SELECT count(*) FROM road_segments " +
         "WHERE ST_Intersects(ST_GeomFromText(%s),centerline)",
(circle.wkt,))
        if cursor.fetchone()[0] == 0:
```

```
            del gps_points[0]
        else:
            break

    while len(gps_points) > 0:
        circle = calc_circle_with_radius(gps_points[-1], 10)
        cursor.execute("SELECT count(*) FROM road_segments " +
         "WHERE ST_Intersects(ST_GeomFromText(%s)," +
         "centerline)", (circle.wkt,))
        if cursor.fetchone()[0] == 0:
            del gps_points[-1]
        else:
            break


    search_distance = 750
    flag=0
    while flag==0:
        circle = calc_circle_with_radius(gps_points[0],search_distance)
        cursor.execute("SELECT id FROM endpoints " +
 "WHERE ST_Contains(ST_GeomFromText(%s)," +
 "endpoint)", (circle.wkt,))

        possible_endpoints = []
        for row in cursor:
            possible_endpoints.append(row[0])


        possible_road_segments = []
        for endpoint_id in possible_endpoints:
            cursor.execute("SELECT directed_segment_id " +
                           "FROM endpoint_segments " +
                           "WHERE endpoint_id=%s", (endpoint_id,))
            for row in cursor:
                directed_segment_id = row[0]
                possible_road_segments.append((directed_segment_id,
endpoint_id))


        route_candidates = []
        for directed_segment_id,endpoint_id in possible_road_segments:
            cursor.execute("SELECT ST_AsText(centerline) " +
                           "FROM directed_segments WHERE id=%s",
                           (directed_segment_id,))
            wkt = cursor.fetchone()[0]
            linestring = shapely.wkt.loads(wkt)
            gps_distance = calc_distance(gps_points[0],linestring)
            segment = {'directed_segment_id' :
directed_segment_id,'linestring' : linestring,'gps_points':
[gps_points[0]],'gps_distances': [gps_distance]}
            route_segments = [segment]
            candidate = {'segments':
route_segments,'directed_segment_ids' : [directed_segment_id],'score':
calc_score(route_segments)}
            route_candidates.append(candidate)

            if len(route_candidates) >= 25:
                flag=1
```

```
                break
            else:
                search_distance = search_distance + 100
                continue

    for next_point in gps_points[1:]:
        num_routes_to_process = len(route_candidates)
        for i in range(num_routes_to_process):
            route = route_candidates.pop(0)

            new_candidates = develop_route(next_point,
route,route_candidates, cursor)
            route_candidates.extend(new_candidates)
        print(route_candidates)
        #while len(route_candidates) > 0:
            #highest = None
            #for index,route in enumerate(route_candidates):
                #if highest == None:
                    #highest = index
                #elif route['score'] >
route_candidates[highest]['score']:
                    #highest = index

            #del route_candidates[highest]

    best_route = None
    for route in route_candidates:
        if len(route['segments']) >= 2:
            if best_route == None:
                best_route = route
            elif route['score'] < best_route['score']:
                best_route = route
    if best_route == None: continue
    for segment in best_route['segments']:
        cursor.execute("SELECT road_segment_id " + "FROM
directed_segments WHERE id=%s",(segment['directed_segment_id'],))
        road_segment_id = cursor.fetchone()[0]
        cursor.execute("UPDATE road_segments SET tally=tally+1" +
"WHERE id=%s", (road_segment_id,))

connection.commit()
```

index.html

```
<!DOCTYPE html>
<html>
<head>
  <base href="/" />
  <title>Ecommerce Web</title>
  <meta name="viewport" content="width=device-width, initial-scale=1">

  <!-- bootstrap css -->
  <link href="//netdna.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" rel="stylesheet" />
  <link href="https://fonts.googleapis.com/css?family=Roboto:300,400,500&display=swap" rel="stylesheet">
  <link href="https://fonts.googleapis.com/icon?family=Material+Icons" rel="stylesheet">
<link rel="stylesheet" href="styles.f30db0106a8390be65a3.css"></head>
<body class="mat-typography">
  <app>Loading...</app>
<script src="runtime.e227d1a0e31cbccbf8ec.js" defer></script><script src="polyfills.a4021de53358bb0fec14.js" defer></script><script src="main.22482dee89193febf2c5.js" defer></script></body>
</html>
```