A PROJECT REPORT ON

“PERSONALIZED FOOD DELIVERY APP”

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF
BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)
SUBMITTED BY

Exam Seat Number	Name Of Student	Email Id
B150394371	HARSHAL SOLANKI	harshalsolanki97@gmail.com
B150394370	AJAYBIR SINGH	ajaybirsingh21@gmail.com
B150394391	UDIT WARIKOO	udit.warikoo.1997@gmail.com
B150394210	ASHISH ARORA	arora2403ashish@gmail.com

DEPARTMENT OF COMPUTER ENGINEERING
BRACT’S
VISHWAKARMA INSTITUTE OF INFORMATION TECHNOLOGY
SURVEY NO. 3/4, KONDHWA (BUDRUK), PUNE – 411048, MAHARASHTRA (INDIA)

SAVITRIBAI PHULE PUNE UNIVERSITY 2018 -2019

CERTIFICATE

This is to certify that the project report entitled
“ **PERSONALIZED FOOD DELIVERY APP**”

Submitted by

Exam Seat Number	Name Of Student	Email Id
B150394371	HARSHAL SOLANKI	harshalsolanki97@gmail.com
B150394370	AJAYBIR SINGH	ajaybirsingh21@gmail.com
B150394391	UDIT WARIKOO	udit.warikoo.1997@gmail.com
B150394210	ASHISH ARORA	arora2403ashish@gmail.com

Are bonafide student of this institute and the work has been carried out by him/her under the supervision of **Prof. S Rathi** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

(Prof. S rathi -Guide)
Department of computer engineering

(Dr. S. R. Sakhare - HOD)
Department of computer engineering

(Dr. B. S. Karkare)
Director,
Bract’s vishwakarma institute of information technology, pune-48

Place : Pune Date : 01/06/2019

ACKNOWLEDGMENT

I would like to express my deepest appreciation to all those who provided me the possibility to complete this report. A special gratitude I give to our final year project guide Ms Snehal Rathi, whose contribution in stimulating suggestions and encouragement, helped me to coordinate my project especially in writing this report.

Furthermore I would also like to acknowledge with much appreciation the crucial role of the experts Ms Pallavi Rege & Ms Vidya Gaikwad, who gave the permission to use all required resources & references. And lastly I would like to thank my Family and Friends for constant support and encouragement they have shown throughout the project.

ABSTRACT

Use of food ordering apps is increasing day by day. With a variety of offers available people are getting used to order food online on a daily basis. The food ordering apps currently available mostly provide hotel food, consumption of such food on a daily basis is not at all healthy. Therefore we intend to introduce an app which not only provides healthy food but also gives recommendations about what the user should and should not eat. The app would process the feedback given by the user and the data about the user taken during registration to give a recommendation about what package best suits the user. This app will include packages of various cuisine on a subscription basis (of seven days). This would also reduce the time that the user spends on the app to choose the food. Apart from this, we also propose to integrate payment gateway so that the user can make payments. Payments would be done integrating UPI payment gateways.

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

NPCI	National Payment Corporation of India
POS	Point Of Sale
VPA	Virtual Payment Address
PSPs	Payment Service Providers

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

1.1 MOTIVATION

The growing use of food ordering apps has made people used to eating hotel food. This has led to consumption of hotel food on a daily basis which is not good for the health of users. Also due to increase availability of jobs has led to increased rate of working class which is a good thing but because of this people have become busy and barely get any time to cook. So homemade online food service is the need of the hour. Also people need to be more cautious about their dietary needs. Seeing this we intend to propose an app which not only provides healthy food but also guides the user about the dietary needs.

HYGIENIC FOOD **UNHYGIENIC FOOD**

The users need to be aware about the kind food they need to eat. Many times people are unaware about the ingredients and this may cause problems. Guiding the user about what food is best suited for them is the main goal here. Apart from this, the way the delivery is managed by the apps in the market is not so cost efficient. Therefore creating a route in such a way so that multiple deliveries could be managed is one of the things these apps lack. Creating such a route is also
taken into consideration here.

1.2 PROBLEM DEFINITION

Nowadays many people order food online because their lives are getting busier plus its more easy because, who would want to cook if the food was just few fingertips away ? There are various apps which provide online food delivery but most of these apps provide hotel food. Consumption of such food on a daily basis can be harmful therefore a food app that delivers homemade healthy food is the need of the hour. Also people need to take care about what they are eating not all healthy food can always be good for everyone. Therefore an app which not only provides healthy food online but also gives recommendations to the user about what is good and what is not good for them could be a real savior. Apart from this, various payment gateway available in the market charge for the transaction, therefore using a payment gateway which does not charge any extra fee would be real winner.

2. LITERATURE SURVEY

2.1 ONLINE FOOD ORDERING SYSTEM

Improving customer satisfaction can increase the customers' loyalty to a product or service provider. One way to improve it is by having a food ordering system which enables customers to purchase the products without physically visiting the shop, namely by phone or by website, and then have the product delivered to the customer's address safely and in good condition. Some food franchises in Indonesia have implemented this type of system, such as Kentucky Fried Chicken, McDonald's, and Pizza Hut.In delivery service, one of the main problems is to find the shortest path between customers' addresses in order to deliver the product in reasonably short time, to save fuel usage and to optimize the utilization of the vehicles and delivery personnel.The optimization requirement is usually represented on the delivery staff job vacancy announcement that often requires the applicants to know the streets or shortcuts in particular area or city. The routing problem that is related to the condition of food delivery service is called Traveling Salesman Problem (TSP). In TSP, the seller starts moving from his/her hometown and is required to visit several cities exactly one time before going back to his/her hometown with minimum total distance. All cities are connected to each other. In this research, the author develops a system that can optimize the delivery routing process by implementing one of the solutions to TSP, which is heuristics algorithm. In addition, the system also utilizes the Global Positioning System (GPS) technology and mapping solution software, Google Maps. **[1]**

2.2 VARIOUS ONLINE FOOD DELIVERY APPS

1. A Food Ordering System with Delivery Routing Optimization Using Global Positioning System

(GPS) Technology and Google Maps by Roy Deddy Hasiholan Tobing **[1]** based on :

- Android based Application
- Web based App
- Heuristic Routing Algorithm

2. Android Application for Local Food Ordering System by

Android Application for Local Food Ordering System by Shubham Takalkar, Devendra Phatak, Kumar Abhinav, Salman Hadi, R. H. Borhade **[2]** based on :

- Area Recognition (GPS)
- Android based app

3. Automated Food Ordering System with Real-Time Customer Feedback **[3]** based on:

- The android application on android mobiles of customers to make orders.
- The server and web applications on the restaurant-owner's laptop to customize menu and keep track of customer records.
- The central database for restaurant-owner to store updated menu information and order details.
- Wireless infrastructure to support networked communication.

4. Design and Implementation of an Android Application using WiFi-enabled Devices for the Food Servicing Industry **[4]** by Alberto Bañacia, Marc Dindo Fernando, Arnel Requillo Jr., and Nelson Rubi Jr. EE/ECE Department, University of San Carlos Cebu City, Philippines

2.3 OBSERVATIONS :

While doing the literature survey, the following observations were made:

- The delivery service business process of several food franchises in Pune. These observed business processes are the based to develop the proposed food ordering system in this research.
- Study on Android and GPS
- Study on Payment Gateway Integration through UPI
- Study on Machine Learning and Heuristic search algorithms

2.4 METHODOLOGY :

While doing the research it was observed that most of the existing apps possess the required features that an online food delivery system should consist but they lack a few advanced features like recommendation. The key aspects of our proposed system are as follow :

- Use of Machine Learning for Recommendation of the package (Menu)
- Use of UPI for

2.5 MACHINE LEARNING MODEL :

For system should to give recommendation to the user about what food they should eat, the system should consist of a model that can be trained on some data and then give recommendation based on the knowledge it has learned. We gathered information about the dietary requirements and food preferences of people from different age groups. Around 200 entries were collected and we intend to gather more data. This data consists of various attributes which well define the food preferences of people from various walks of life. As observed, most of the data is categorical. Therefore one of the classifiers only would be best suited for training the Model. By studying various classification algorithms, CART (Classification And Regression) was chosen for training the model. The major reason for choosing CART was that it does pruning while training the model, i.e it continuously keeps pruning the tree inorder to reduce the size of the tree. Also it handles continuous data as well.

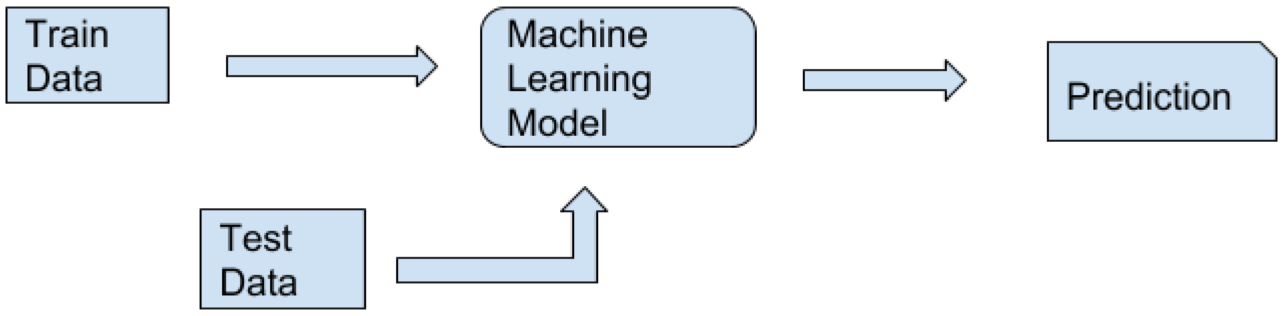


Fig 1 : Machine learning Model

Random Forest:

Random Forest is a flexible, easy to use machine learning algorithm that produces, even without hyper-parameter tuning, a great result most of the time. It is also one of the most used algorithms, because it's simplicity and the fact that it can be used for both classification and regression tasks. In this post, you are going to learn, how the random forest algorithm works and several other important things about it.
To say it in simple words: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction.
One big advantage of random forest is, that it can be used for both classification and regression problems, which form the majority of current machine learning systems. I will talk about random forest in classification, since classification is sometimes considered the building block of machine learning. Below you can see how a random forest would look like with two trees:
Random Forest has nearly the same hyper parameters as a decision tree or a bagging classifier. Fortunately, you don't have to combine a decision tree with a bagging classifier and can just easily use the classifier-class of Random Forest. Like I already said, with Random Forest, you can also deal with Regression tasks by using the Random Forest regressor.Random Forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model.Therefore, in Random Forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node. You can even make trees more random, by additionally using random thresholds for each feature rather than searching for the best possible thresholds (like a normal decision tree does).Another great quality of the random forest algorithm is that it is very easy to measure the relative importance of each feature on the prediction. Sklearn provides a great tool for this, that measures a features importance by looking at how much the tree nodes, which use that feature, reduce impurity across all

trees in the forest. It computes this score automatically for each feature after training and scales the results, so that the sum of all importance is equal to 1.

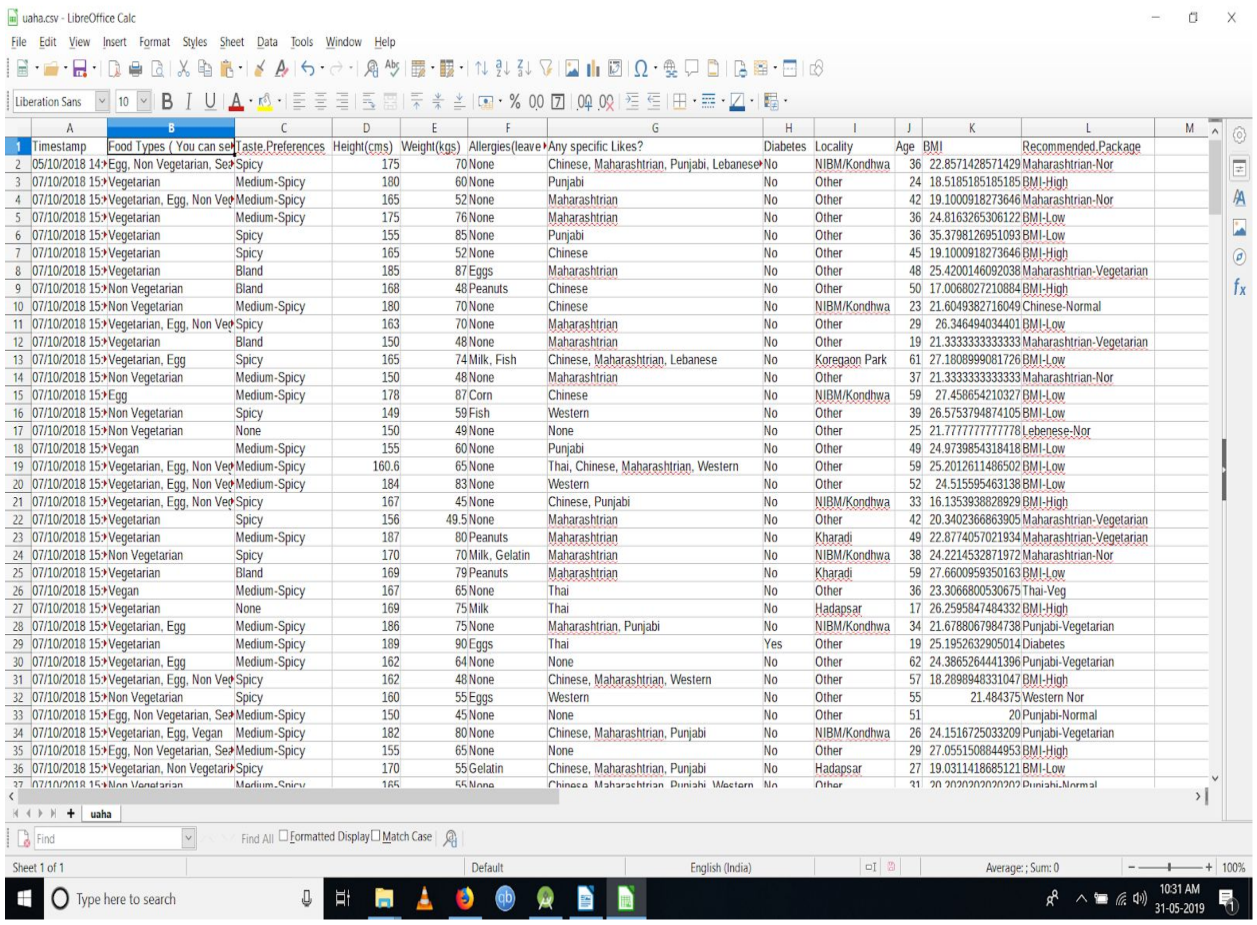
If you don’t know how a decision tree works and if you don’t know what a leaf or node is, here is a good description from Wikipedia: In a decision tree each internal node represents a “test” on an attribute (e.g. whether a coin flip comes up heads or tails), each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). A node that has no children is a leaf.

Through looking at the feature importance, you can decide which features you may want to drop, because they don’t contribute enough or nothing to the prediction process. This is important, because a general rule in machine learning is that the more features you have, the more likely your model will suffer from overfitting and vice versa.

Prepossessing :

Data for the project has been collected using google forms shared over social media

The google form csv data set generated looks like this :



1	Timestamp	Food Types (You can se	Taste.Preferences	Height(cms)	Weight(kgs)	Allergies(leave	Any specific Likes?	Diabetes	Locality	Age	BMI	Recommended.Package
2	05/10/2018 14:	Egg, Non Vegetarian, Sea	Spicy	175	70	None	Chinese, Maharashtrian, Punjabi, Lebanese	No	NIBM/Kondhwa	36	22.8571428571429	Maharashtrian-Nor
3	07/10/2018 15:	Vegetarian	Medium-Spicy	180	60	None	Punjabi	No	Other	24	18.5185185185185	BMI-High
4	07/10/2018 15:	Vegetarian, Egg, Non Veg	Medium-Spicy	165	52	None	Maharashtrian	No	Other	42	19.1000918273646	Maharashtrian-Nor
5	07/10/2018 15:	Vegetarian	Medium-Spicy	175	76	None	Maharashtrian	No	Other	36	24.8163265306122	BMI-Low
6	07/10/2018 15:	Vegetarian	Spicy	155	85	None	Punjabi	No	Other	36	35.3798126951093	BMI-Low
7	07/10/2018 15:	Vegetarian	Spicy	165	52	None	Chinese	No	Other	45	19.1000918273646	BMI-High
8	07/10/2018 15:	Vegetarian	Bland	185	87	Eggs	Maharashtrian	No	Other	48	25.4200146092038	Maharashtrian-Vegetarian
9	07/10/2018 15:	Non Vegetarian	Bland	168	48	Peanuts	Chinese	No	Other	50	17.0068027210884	BMI-High
10	07/10/2018 15:	Non Vegetarian	Medium-Spicy	180	70	None	Chinese	No	NIBM/Kondhwa	23	21.6049382716049	Chinese-Normal
11	07/10/2018 15:	Vegetarian, Egg, Non Veg	Spicy	163	70	None	Maharashtrian	No	Other	29	26.346494034401	BMI-Low
12	07/10/2018 15:	Vegetarian	Bland	150	48	None	Maharashtrian	No	Other	19	21.3333333333333	Maharashtrian-Vegetarian
13	07/10/2018 15:	Vegetarian, Egg	Spicy	165	74	Milk, Fish	Chinese, Maharashtrian, Lebanese	No	Koregaon Park	61	27.1808999081726	BMI-Low
14	07/10/2018 15:	Non Vegetarian	Medium-Spicy	150	48	None	Maharashtrian	No	Other	37	21.3333333333333	Maharashtrian-Nor
15	07/10/2018 15:	Egg	Medium-Spicy	178	87	Corn	Chinese	No	NIBM/Kondhwa	59	27.458654210327	BMI-Low
16	07/10/2018 15:	Non Vegetarian	Spicy	149	59	Fish	Western	No	Other	39	26.5753794874105	BMI-Low
17	07/10/2018 15:	Non Vegetarian	None	150	49	None	None	No	Other	25	21.7777777777778	Lebenese-Nor
18	07/10/2018 15:	Vegan	Medium-Spicy	155	60	None	Punjabi	No	Other	49	24.9739854318418	BMI-Low
19	07/10/2018 15:	Vegetarian, Egg, Non Veg	Medium-Spicy	160.6	65	None	Thai, Chinese, Maharashtrian, Western	No	Other	59	25.2012611486502	BMI-Low
20	07/10/2018 15:	Vegetarian, Egg, Non Veg	Medium-Spicy	184	83	None	Western	No	Other	52	24.515595463138	BMI-Low
21	07/10/2018 15:	Vegetarian, Egg, Non Veg	Spicy	167	45	None	Chinese, Punjabi	No	NIBM/Kondhwa	33	16.1353938828929	BMI-High
22	07/10/2018 15:	Vegetarian	Spicy	156	49.5	None	Maharashtrian	No	Other	42	20.3402366863905	Maharashtrian-Vegetarian
23	07/10/2018 15:	Vegetarian	Medium-Spicy	187	80	Peanuts	Maharashtrian	No	Kharadi	49	22.8774057021934	Maharashtrian-Vegetarian
24	07/10/2018 15:	Non Vegetarian	Spicy	170	70	Milk, Gelatin	Maharashtrian	No	NIBM/Kondhwa	38	24.2214532871972	Maharashtrian-Nor
25	07/10/2018 15:	Vegetarian	Bland	169	79	Peanuts	Maharashtrian	No	Kharadi	59	27.6600959350163	BMI-Low
26	07/10/2018 15:	Vegan	Medium-Spicy	167	65	None	Thai	No	Other	36	23.3066800530675	Thai-Veg
27	07/10/2018 15:	Vegetarian	None	169	75	Milk	Thai	No	Hadapsar	17	26.2595847484332	BMI-High
28	07/10/2018 15:	Vegetarian, Egg	Medium-Spicy	186	75	None	Maharashtrian, Punjabi	No	NIBM/Kondhwa	34	21.6788067984738	Punjabi-Vegetarian
29	07/10/2018 15:	Vegetarian	Medium-Spicy	189	90	Eggs	Thai	Yes	Other	19	25.1952632905014	Diabetes
30	07/10/2018 15:	Vegetarian, Egg	Medium-Spicy	162	64	None	None	No	Other	62	24.3865264441396	Punjabi-Vegetarian
31	07/10/2018 15:	Vegetarian, Egg, Non Veg	Spicy	162	48	None	Chinese, Maharashtrian, Western	No	Other	57	18.2898948331047	BMI-High
32	07/10/2018 15:	Non Vegetarian	Spicy	160	55	Eggs	Western	No	Other	55	21.484375	Western Nor
33	07/10/2018 15:	Egg, Non Vegetarian, Sea	Medium-Spicy	150	45	None	None	No	Other	51	20	Punjabi-Normal
34	07/10/2018 15:	Vegetarian, Egg, Vegan	Medium-Spicy	182	80	None	Chinese, Maharashtrian, Punjabi	No	NIBM/Kondhwa	26	24.1516725033209	Punjabi-Vegetarian
35	07/10/2018 15:	Egg, Non Vegetarian, Sea	Medium-Spicy	155	65	None	None	No	Other	29	27.0551508844953	BMI-High
36	07/10/2018 15:	Vegetarian, Non Vegetari	Spicy	170	55	Gelatin	Chinese, Maharashtrian, Punjabi	No	Hadapsar	27	19.0311418685121	BMI-Low
37	07/10/2018 15:	Non Vegetarian	Medium-Spicy	165	55	None	Chinese, Maharashtrian, Punjabi, Western	No	Other	31	20.2020202020202	Punjabi-Normal

BMI is calculated from height and weight values using BMI calculation formulae.
From this data set all the unnecessary columns determined unimportant for model training has been dropped completely : Timestamp, Height , Weight, locality

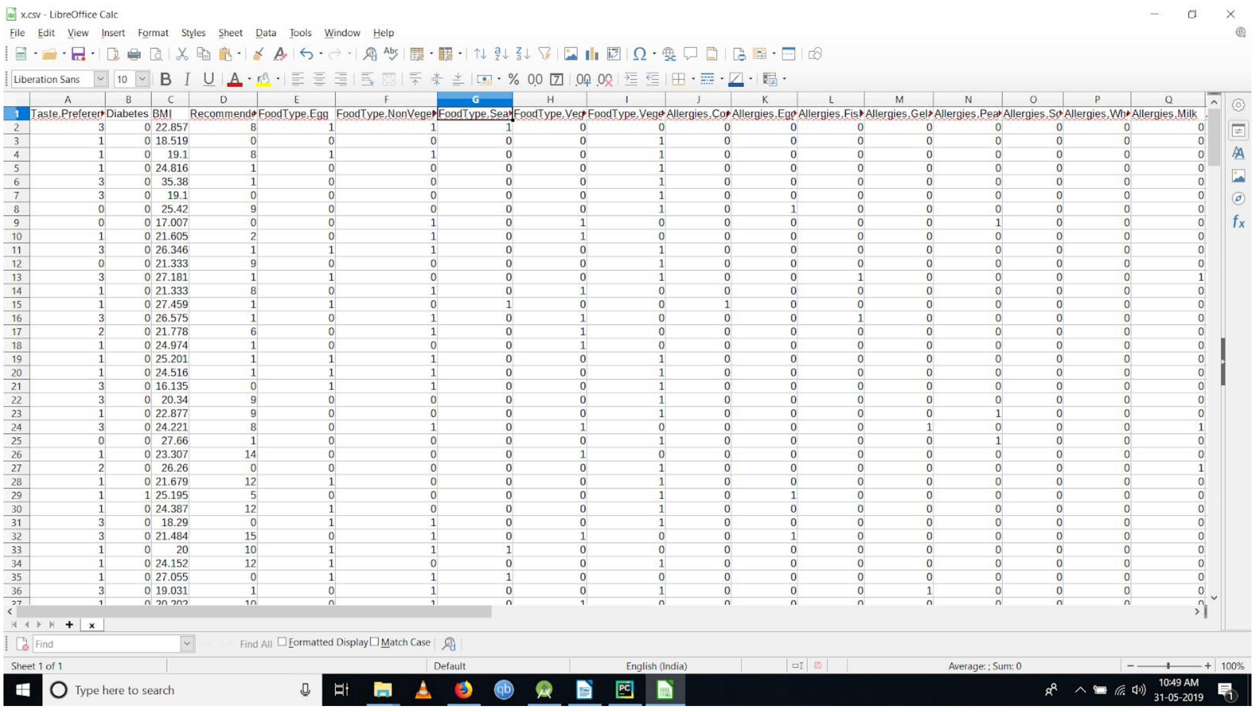
After this, multi valued columns are split up using python commands for processing and new columns are generated for existing attributes distinctly appearing in the respective multi valued column. Eg:
For instance: ‘Any specific Likes?’ is converted into columns with distinct values as “Likes.*”
where * is Chinese,Lebanese,Maharashtrian etc..

```
datalikes=dftrain['Any specific Likes?'].str.get_dummies(sep=',')

df=pd.concat([datalikes], axis=1) //join original dataset and newly generated columns

df=df.drop(['Any specific Likes?'],axis=1) //to drop all the converted column parents
```

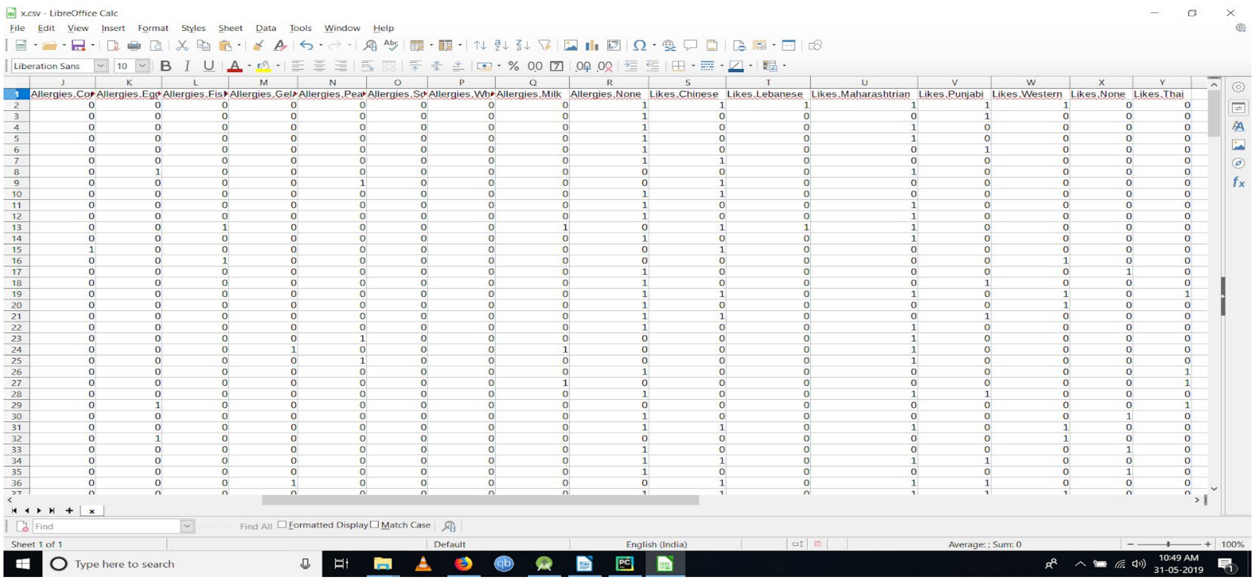
All categorical columns are prepossessed using label encoder from sklearn.
Eg:
Taste preference ranges from spicy,medium-spicy,bland and none.
Label encoders assigns values from 0,1,2,3

```
leT = preprocessing.LabelEncoder()
df['Taste.Preferences'] = leT.fit_transform(df['Taste.Preferences'])
```

Outputs generated are now either nominal,binary or numeric and only necessary columns required for model are existing.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	Taste.Preferen	Diabetes	BMI	Recommende	FoodType.Egg	FoodType.NonVege	FoodType.Sea	FoodType.Veg	FoodType.Vege	Allergies.Co	Allergies.Eg	Allergies.Fis	Allergies.Gel	Allergies.Pea	Allergies.So	Allergies.Wh	Allergies.Milk
2	3	0	22.857	8	1	1	1	0	0	0	0	0	0	0	0	0	0
3	1	0	18.519	0	0	0	0	0	1	0	0	0	0	0	0	0	0
4	1	0	19.1	8	1	1	0	0	1	0	0	0	0	0	0	0	0
5	1	0	24.816	1	0	0	0	0	1	0	0	0	0	0	0	0	0
6	3	0	35.38	1	0	0	0	0	1	0	0	0	0	0	0	0	0
7	3	0	19.1	0	0	0	0	0	1	0	0	0	0	0	0	0	0
8	0	0	25.42	9	0	0	0	0	1	0	1	0	0	0	0	0	0
9	0	0	17.007	0	0	1	0	1	0	0	0	0	0	1	0	0	0
10	1	0	21.605	2	0	1	0	1	0	0	0	0	0	0	0	0	0
11	3	0	26.346	1	1	1	0	0	1	0	0	0	0	0	0	0	0
12	0	0	21.333	9	0	0	0	0	1	0	0	0	0	0	0	0	0
13	3	0	27.181	1	1	0	0	0	1	0	0	1	0	0	0	0	1
14	1	0	21.333	8	0	1	0	1	0	0	0	0	0	0	0	0	0
15	1	0	27.459	1	1	0	1	0	0	1	0	0	0	0	0	0	0
16	3	0	26.575	1	0	1	0	1	0	0	0	1	0	0	0	0	0
17	2	0	21.778	6	0	1	0	1	0	0	0	0	0	0	0	0	0
18	1	0	24.974	1	0	0	0	1	0	0	0	0	0	0	0	0	0
19	1	0	25.201	1	1	1	0	0	1	0	0	0	0	0	0	0	0
20	1	0	24.516	1	1	1	0	0	1	0	0	0	0	0	0	0	0
21	3	0	16.135	0	1	1	0	0	1	0	0	0	0	0	0	0	0
22	3	0	20.34	9	0	0	0	0	1	0	0	0	0	0	0	0	0
23	1	0	22.877	9	0	0	0	0	1	0	0	0	0	1	0	0	0
24	3	0	24.221	8	0	1	0	1	0	0	0	0	1	0	0	0	1
25	0	0	27.66	1	0	0	0	0	1	0	0	0	0	1	0	0	0
26	1	0	23.307	14	0	0	0	1	0	0	0	0	0	0	0	0	0
27	2	0	26.26	0	0	0	0	0	1	0	0	0	0	0	0	0	1
28	1	0	21.679	12	1	0	0	0	1	0	0	0	0	0	0	0	0
29	1	1	25.195	5	0	0	0	0	1	0	1	0	0	0	0	0	0
30	1	0	24.387	12	1	0	0	0	1	0	0	0	0	0	0	0	0
31	3	0	18.29	0	1	1	0	0	1	0	0	0	0	0	0	0	0
32	3	0	21.484	15	0	1	0	1	0	0	1	0	0	0	0	0	0
33	1	0	20	10	1	1	1	0	0	0	0	0	0	0	0	0	0
34	1	0	24.152	12	1	0	0	0	1	0	0	0	0	0	0	0	0
35	1	0	27.055	0	1	1	1	0	0	0	0	0	0	0	0	0	0
36	3	0	19.031	1	0	1	0	0	1	0	0	0	1	0	0	0	0



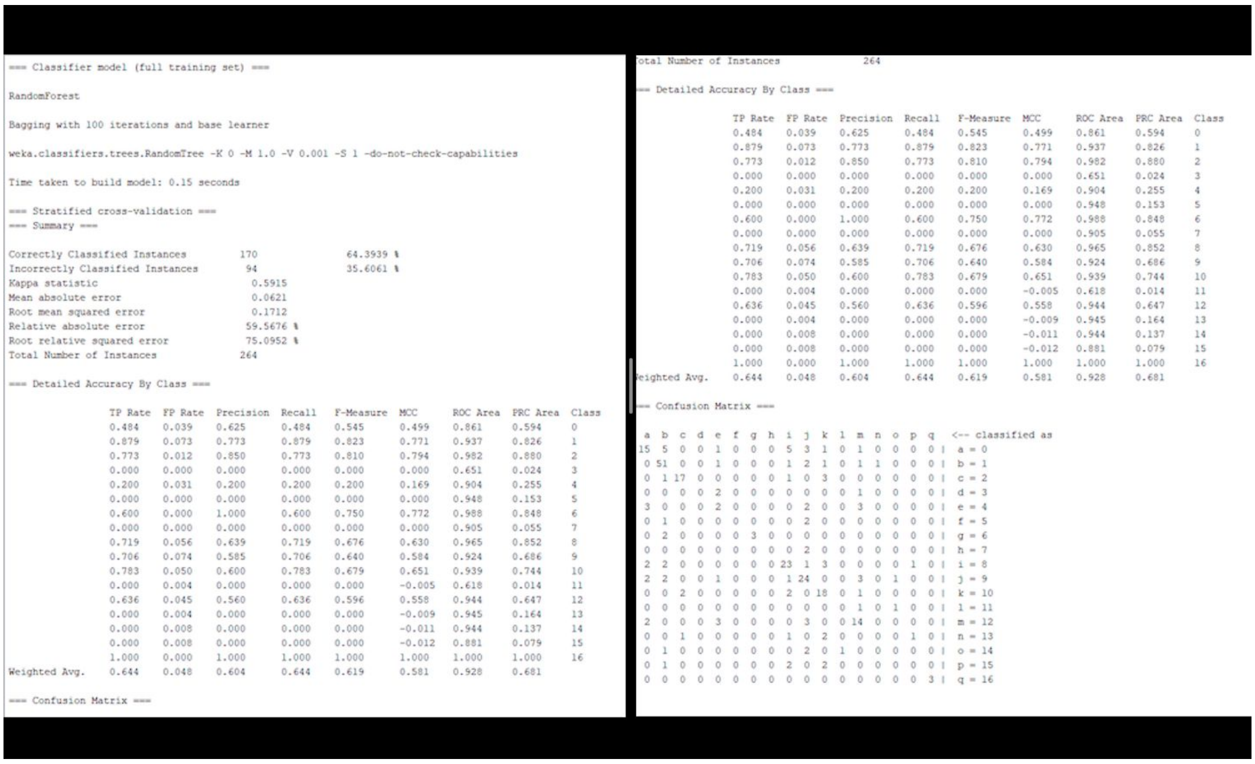
	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
1	Allergies.Co	Allergies.Eg	Allergies.Fis	Allergies.Gel	Allergies.Pea	Allergies.So	Allergies.Wh	Allergies.Milk	Allergies.None	Likes.Chinese	Likes.Lebanese	Likes.Maharashtrian	Likes.Punjabi	Likes.Western	Likes.None	Likes.Thai
2	0	0	0	0	0	0	0	0	1	1	1	1	1	1	0	0
3	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
4	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
5	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
6	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
7	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
8	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0
9	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
11	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
12	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
13	0	0	1	0	0	0	0	1	0	1	1	1	0	0	0	0
14	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
15	1	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0
16	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
17	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
18	0	0	0	0	0	0	0	0	1	0	0	0	1	0	0	0
19	0	0	0	0	0	0	0	0	1	1	0	1	0	1	0	1
20	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0
21	0	0	0	0	0	0	0	0	1	1	0	0	1	0	0	0
22	0	0	0	0	0	0	0	0	1	0	0	1	0	0	0	0
23	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
24	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	0
25	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	0
26	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1
27	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1
28	0	0	0	0	0	0	0	0	1	0	0	1	1	0	0	0
29	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1
30	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
31	0	0	0	0	0	0	0	0	1	1	0	1	0	1	0	0
32	0	1	0	0	0	0	0	0	0	0	0	0	0	1	0	0
33	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
34	0	0	0	0	0	0	0	0	1	1	0	1	1	0	0	0
35	0	0	0	0	0	0	0	0	1	0	0	0	0	0	1	0
36	0	0	0	1	0	0	0	0	0	1	0	1	1	0	0	0

We have deployed in our application using weka API in android Studio but for analysis purpose we have used the Weka tool for Windows.

For preprocessing in weka since weka reads all categorical columns from csv files as numeric conversion back to categorical is done using weka filter:
weka.attribute.unsupervised.attribute.NumerictoNominal (first-2,4-last)

This excludes BMI which is infact a numerical value

Random Forest Analysis:

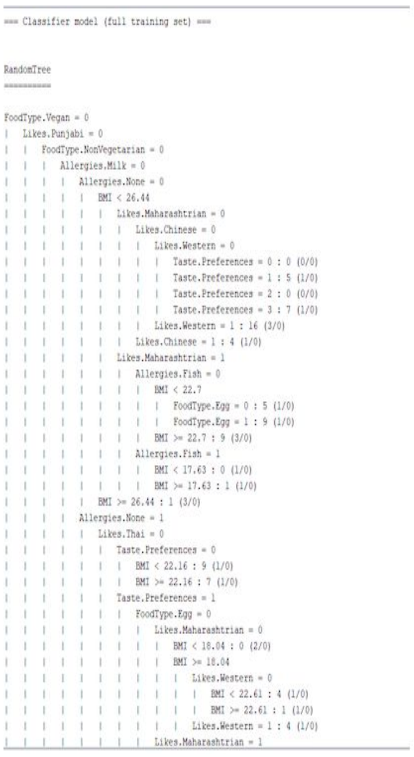


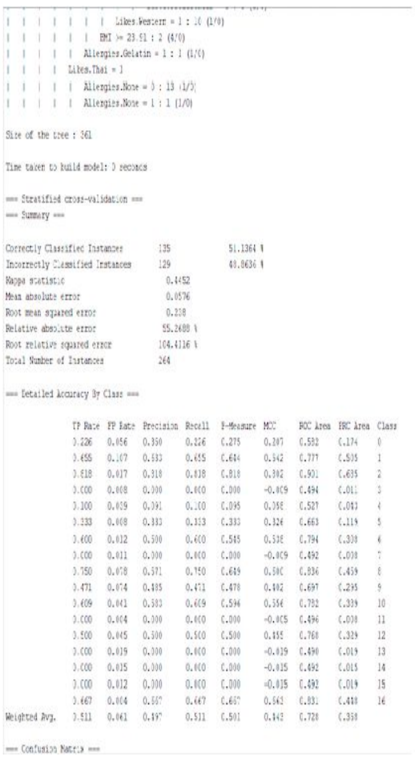
```
=== Classifier model (full training set) ===

RandomForest

Bagging with 100 iterations and base learner

weka.classifiers.trees.RandomTree -K 0 -M 1.0 -V 0.001 -S 1 -do-not-check-capabilities

Time taken to build model: 0.15 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         170               64.3939 %
Incorrectly Classified Instances        94               35.6061 %
Kappa statistic                          0.5915
Mean absolute error                      0.0621
Root mean squared error                  0.1712
Relative absolute error                 59.5676 %
Root relative squared error             75.0952 %
Total Number of Instances              264

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.484    0.039    0.625      0.484    0.545      0.499    0.861     0.594     0
                 0.879    0.073    0.773      0.879    0.823      0.771    0.937     0.826     1
                 0.773    0.012    0.850      0.773    0.810      0.794    0.982     0.880     2
                 0.000    0.000    0.000      0.000    0.000      0.000    0.651     0.024     3
                 0.200    0.031    0.200      0.200    0.200      0.169    0.904     0.255     4
                 0.000    0.000    0.000      0.000    0.000      0.000    0.948     0.153     5
                 0.600    0.000    1.000      0.600    0.750      0.772    0.988     0.848     6
                 0.000    0.000    0.000      0.000    0.000      0.000    0.905     0.055     7
                 0.719    0.056    0.639      0.719    0.676      0.630    0.965     0.852     8
                 0.706    0.074    0.585      0.706    0.640      0.584    0.924     0.686     9
                 0.783    0.050    0.600      0.783    0.679      0.651    0.939     0.744     10
                 0.000    0.004    0.000      0.000    0.000      -0.005   0.618     0.014     11
                 0.636    0.045    0.560      0.636    0.596      0.558    0.944     0.647     12
                 0.000    0.004    0.000      0.000    0.000      -0.009   0.945     0.164     13
                 0.000    0.008    0.000      0.000    0.000      -0.011   0.944     0.137     14
                 0.000    0.008    0.000      0.000    0.000      -0.012   0.881     0.079     15
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     16
Weighted Avg.    0.644    0.048    0.604      0.644    0.619      0.581    0.928     0.681

=== Confusion Matrix ===
```

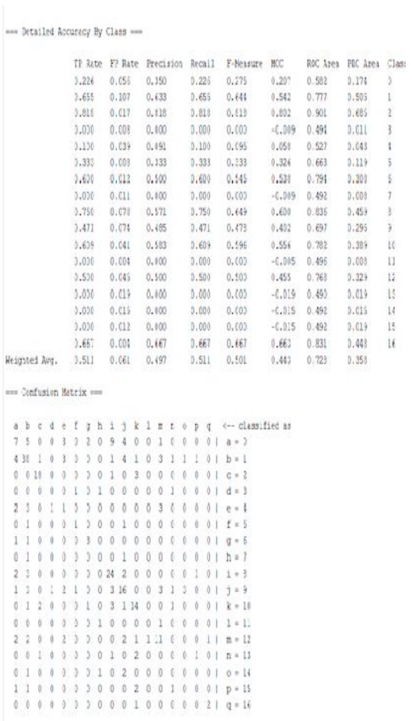
```
otal Number of Instances              264

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.484    0.039    0.625      0.484    0.545      0.499    0.861     0.594     0
                 0.879    0.073    0.773      0.879    0.823      0.771    0.937     0.826     1
                 0.773    0.012    0.850      0.773    0.810      0.794    0.982     0.880     2
                 0.000    0.000    0.000      0.000    0.000      0.000    0.651     0.024     3
                 0.200    0.031    0.200      0.200    0.200      0.169    0.904     0.255     4
                 0.000    0.000    0.000      0.000    0.000      0.000    0.948     0.153     5
                 0.600    0.000    1.000      0.600    0.750      0.772    0.988     0.848     6
                 0.000    0.000    0.000      0.000    0.000      0.000    0.905     0.055     7
                 0.719    0.056    0.639      0.719    0.676      0.630    0.965     0.852     8
                 0.706    0.074    0.585      0.706    0.640      0.584    0.924     0.686     9
                 0.783    0.050    0.600      0.783    0.679      0.651    0.939     0.744     10
                 0.000    0.004    0.000      0.000    0.000      -0.005   0.618     0.014     11
                 0.636    0.045    0.560      0.636    0.596      0.558    0.944     0.647     12
                 0.000    0.004    0.000      0.000    0.000      -0.009   0.945     0.164     13
                 0.000    0.008    0.000      0.000    0.000      -0.011   0.944     0.137     14
                 0.000    0.008    0.000      0.000    0.000      -0.012   0.881     0.079     15
                 1.000    0.000    1.000      1.000    1.000      1.000    1.000     1.000     16
eighted Avg.     0.644    0.048    0.604      0.644    0.619      0.581    0.928     0.681

=== Confusion Matrix ===
```

```
  a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q   <-- classified as
 15  5  0  0  1  0  0  0  5  3  1  0  1  0  0  0  0 |  a = 0
  0 51  0  0  1  0  0  0  1  2  1  0  1  1  0  0  0 |  b = 1
  0  1 17  0  0  0  0  0  1  0  3  0  0  0  0  0  0 |  c = 2
  0  0  0  0  2  0  0  0  0  0  0  0  1  0  0  0  0 |  d = 3
  3  0  0  0  2  0  0  0  0  2  0  0  3  0  0  0  0 |  e = 4
  0  1  0  0  0  0  0  0  0  2  0  0  0  0  0  0  0 |  f = 5
  0  2  0  0  0  0  3  0  0  0  0  0  0  0  0  0  0 |  g = 6
  0  0  0  0  0  0  0  0  0  2  0  0  0  0  0  0  0 |  h = 7
  2  2  0  0  0  0  0  0 23  1  3  0  0  0  0  1  0 |  i = 8
  2  2  0  0  1  0  0  0  1 24  0  0  3  0  1  0  0 |  j = 9
  0  0  2  0  0  0  0  0  2  0 18  0  1  0  0  0  0 |  k = 10
  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0  0 |  l = 11
  2  0  0  0  3  0  0  0  0  3  0  0 14  0  0  0  0 |  m = 12
  0  0  1  0  0  0  0  0  1  0  2  0  0  0  0  1  0 |  n = 13
  0  1  0  0  0  0  0  0  0  2  0  1  0  0  0  0  0 |  o = 14
  0  1  0  0  0  0  0  0  2  0  2  0  0  0  0  0  0 |  p = 15
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  3 |  q = 16
```

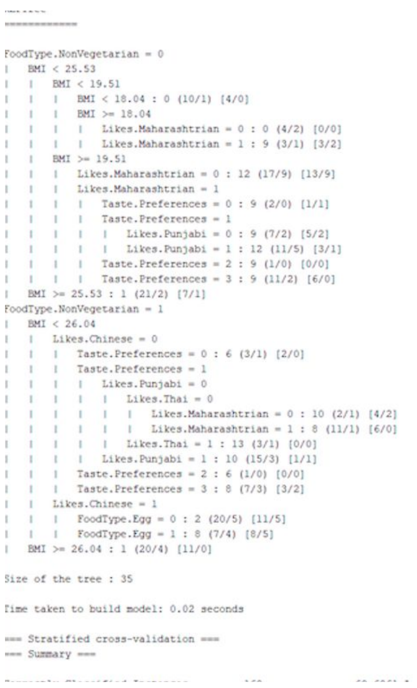
Random Tree Analysis:

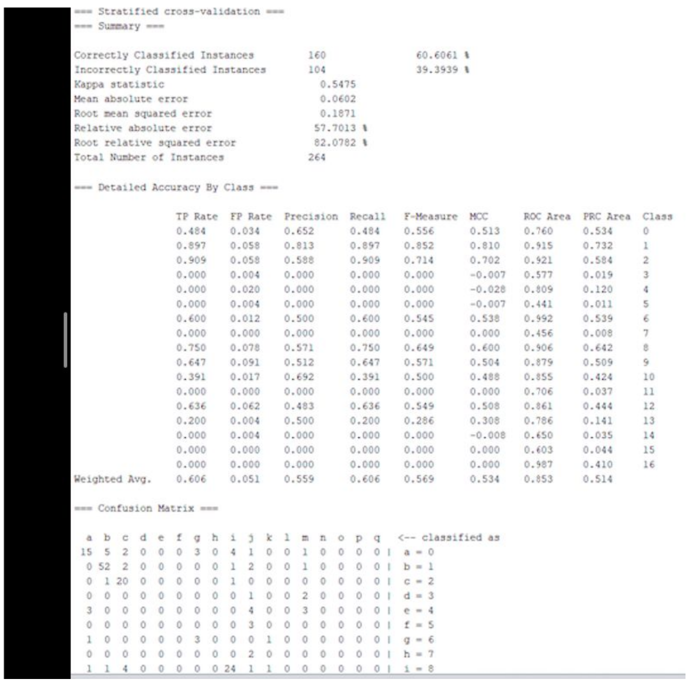

```
=== Classifier model (full training set) ===


RandomTree
==========

FoodType.Vegan = 0
|   Likes.Punjabi = 0
|   |   FoodType.NonVegetarian = 0
|   |   |   Allergies.Milk = 0
|   |   |   |   Allergies.None = 0
|   |   |   |   |   BMI < 26.44
|   |   |   |   |   |   Likes.Maharashtrian = 0
|   |   |   |   |   |   |   Likes.Chinese = 0
|   |   |   |   |   |   |   |   Likes.Western = 0
|   |   |   |   |   |   |   |   |   Taste.Preferences = 0 : 0 (0/0)
|   |   |   |   |   |   |   |   |   Taste.Preferences = 1 : 5 (1/0)
|   |   |   |   |   |   |   |   |   Taste.Preferences = 2 : 0 (0/0)
|   |   |   |   |   |   |   |   |   Taste.Preferences = 3 : 7 (1/0)
|   |   |   |   |   |   |   |   Likes.Western = 1 : 16 (3/0)
|   |   |   |   |   |   |   Likes.Chinese = 1 : 4 (1/0)
|   |   |   |   |   |   Likes.Maharashtrian = 1
|   |   |   |   |   |   |   Allergies.Fish = 0
|   |   |   |   |   |   |   |   BMI < 22.7
|   |   |   |   |   |   |   |   |   FoodType.Egg = 0 : 5 (1/0)
|   |   |   |   |   |   |   |   |   FoodType.Egg = 1 : 9 (1/0)
|   |   |   |   |   |   |   |   BMI >= 22.7 : 9 (3/0)
|   |   |   |   |   |   |   Allergies.Fish = 1
|   |   |   |   |   |   |   |   BMI < 17.63 : 0 (1/0)
|   |   |   |   |   |   |   |   BMI >= 17.63 : 1 (1/0)
|   |   |   |   |   BMI >= 26.44 : 1 (3/0)
|   |   |   |   Allergies.None = 1
|   |   |   |   |   Likes.Thai = 0
|   |   |   |   |   |   Taste.Preferences = 0
|   |   |   |   |   |   |   BMI < 22.16 : 9 (1/0)
|   |   |   |   |   |   |   BMI >= 22.16 : 7 (1/0)
|   |   |   |   |   |   Taste.Preferences = 1
|   |   |   |   |   |   |   FoodType.Egg = 0
|   |   |   |   |   |   |   |   Likes.Maharashtrian = 0
|   |   |   |   |   |   |   |   |   BMI < 18.04 : 0 (2/0)
|   |   |   |   |   |   |   |   |   BMI >= 18.04
|   |   |   |   |   |   |   |   |   |   Likes.Western = 0
|   |   |   |   |   |   |   |   |   |   |   BMI < 22.61 : 4 (1/0)
|   |   |   |   |   |   |   |   |   |   |   BMI >= 22.61 : 1 (1/0)
|   |   |   |   |   |   |   |   |   |   Likes.Western = 1 : 4 (1/0)
|   |   |   |   |   |   |   |   Likes.Maharashtrian = 1
```


```
|   |   |   |   |   |   |   Likes.Western = 1 : 10 (1/0)
|   |   |   |   |   |   BMI >= 23.91 : 2 (4/0)
|   |   |   |   |   Allergies.Gelatin = 1 : 1 (1/0)
|   |   |   |   Likes.Thai = 1
|   |   |   |   |   Allergies.None = 0 : 13 (1/0)
|   |   |   |   |   Allergies.None = 1 : 1 (1/0)

Size of the tree : 361

Time taken to build model: 0 seconds

=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         135               51.1364 %
Incorrectly Classified Instances       129               48.8636 %
Kappa statistic                          0.4452
Mean absolute error                      0.0576
Root mean squared error                  0.238
Relative absolute error                 55.2688 %
Root relative squared error            104.4116 %
Total Number of Instances              264

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.226    0.056    0.350      0.226    0.275      0.207    0.582     0.174     0
                 0.655    0.107    0.633      0.655    0.644      0.542    0.777     0.505     1
                 0.818    0.017    0.818      0.818    0.818      0.802    0.901     0.685     2
                 0.000    0.008    0.000      0.000    0.000      -0.009   0.494     0.011     3
                 0.100    0.039    0.091      0.100    0.095      0.058    0.527     0.043     4
                 0.333    0.008    0.333      0.333    0.333      0.326    0.663     0.119     5
                 0.600    0.012    0.500      0.600    0.545      0.538    0.794     0.308     6
                 0.000    0.011    0.000      0.000    0.000      -0.009   0.492     0.008     7
                 0.750    0.078    0.571      0.750    0.649      0.600    0.836     0.459     8
                 0.471    0.074    0.485      0.471    0.478      0.402    0.697     0.295     9
                 0.609    0.041    0.583      0.609    0.596      0.556    0.782     0.389     10
                 0.000    0.004    0.000      0.000    0.000      -0.005   0.496     0.008     11
                 0.500    0.045    0.500      0.500    0.500      0.455    0.768     0.329     12
                 0.000    0.019    0.000      0.000    0.000      -0.019   0.490     0.019     13
                 0.000    0.015    0.000      0.000    0.000      -0.015   0.492     0.015     14
                 0.000    0.012    0.000      0.000    0.000      -0.015   0.492     0.019     15
                 0.667    0.004    0.667      0.667    0.667      0.663    0.831     0.448     16
Weighted Avg.    0.511    0.061    0.497      0.511    0.501      0.443    0.723     0.358

=== Confusion Matrix ===
```


```
=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.226    0.056    0.350      0.226    0.275      0.207    0.582     0.174     0
                 0.655    0.107    0.633      0.655    0.644      0.542    0.777     0.505     1
                 0.818    0.017    0.818      0.818    0.818      0.802    0.901     0.685     2
                 0.000    0.008    0.000      0.000    0.000      -0.009   0.494     0.011     3
                 0.100    0.039    0.091      0.100    0.095      0.058    0.527     0.043     4
                 0.333    0.008    0.333      0.333    0.333      0.326    0.663     0.119     5
                 0.600    0.012    0.500      0.600    0.545      0.538    0.794     0.308     6
                 0.000    0.011    0.000      0.000    0.000      -0.009   0.492     0.008     7
                 0.750    0.078    0.571      0.750    0.649      0.600    0.836     0.459     8
                 0.471    0.074    0.485      0.471    0.478      0.402    0.697     0.295     9
                 0.609    0.041    0.583      0.609    0.596      0.556    0.782     0.389     10
                 0.000    0.004    0.000      0.000    0.000      -0.005   0.496     0.008     11
                 0.500    0.045    0.500      0.500    0.500      0.455    0.768     0.329     12
                 0.000    0.019    0.000      0.000    0.000      -0.019   0.490     0.019     13
                 0.000    0.015    0.000      0.000    0.000      -0.015   0.492     0.015     14
                 0.000    0.012    0.000      0.000    0.000      -0.015   0.492     0.019     15
                 0.667    0.004    0.667      0.667    0.667      0.663    0.831     0.448     16
Weighted Avg.    0.511    0.061    0.497      0.511    0.501      0.443    0.723     0.358

=== Confusion Matrix ===

  a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q   <-- classified as
  7  5  0  0  3  0  2  0  9  4  0  0  1  0  0  0  0 |  a = 0
  4 38  1  0  3  0  0  0  1  4  1  0  3  1  1  1  0 |  b = 1
  0  0 18  0  0  0  0  0  1  0  3  0  0  0  0  0  0 |  c = 2
  0  0  0  0  0  1  0  1  0  0  0  0  0  1  0  0  0 |  d = 3
  2  3  0  1  1  0  0  0  0  0  0  0  3  0  0  0  0 |  e = 4
  0  1  0  0  0  1  0  0  0  1  0  0  0  0  0  0  0 |  f = 5
  1  1  0  0  0  0  3  0  0  0  0  0  0  0  0  0  0 |  g = 6
  0  1  0  0  0  0  0  0  0  1  0  0  0  0  0  0  0 |  h = 7
  2  3  0  0  0  0  0  0 24  2  0  0  0  0  0  1  0 |  i = 8
  1  3  0  1  2  1  0  0  3 16  0  0  3  1  3  0  0 |  j = 9
  0  1  2  0  0  0  1  0  3  1 14  0  0  1  0  0  0 |  k = 10
  0  0  0  0  0  0  0  1  0  0  0  0  1  0  0  0  0 |  l = 11
  2  2  0  0  2  0  0  0  0  2  1  1 11  0  0  0  1 |  m = 12
  0  0  1  0  0  0  0  0  1  0  2  0  0  0  0  1  0 |  n = 13
  0  1  0  0  0  0  0  1  0  2  0  0  0  0  0  0  0 |  o = 14
  1  1  0  0  0  0  0  0  0  0  2  0  0  1  0  0  0 |  p = 15
  0  0  0  0  0  0  0  0  0  0  1  0  0  0  0  0  2 |  q = 16
```

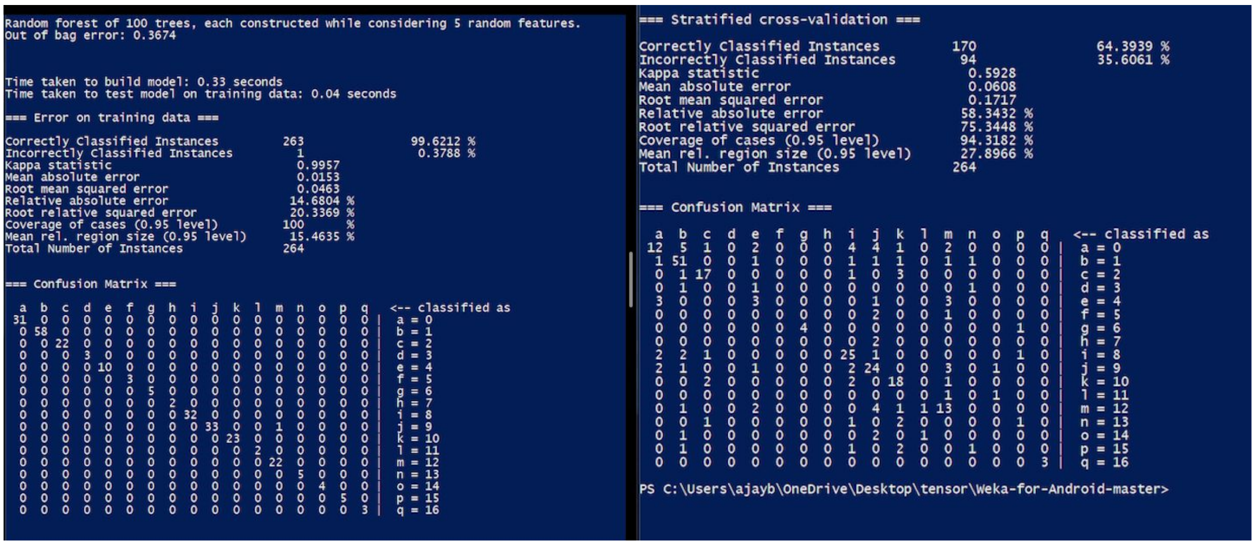
REP Tree Analysis:


```
============

FoodType.NonVegetarian = 0
|   BMI < 25.53
|   |   BMI < 19.51
|   |   |   BMI < 18.04 : 0 (10/1) [4/0]
|   |   |   BMI >= 18.04
|   |   |   |   Likes.Maharashtrian = 0 : 0 (4/2) [0/0]
|   |   |   |   Likes.Maharashtrian = 1 : 9 (3/1) [3/2]
|   |   BMI >= 19.51
|   |   |   Likes.Maharashtrian = 0 : 12 (17/9) [13/9]
|   |   |   Likes.Maharashtrian = 1
|   |   |   |   Taste.Preferences = 0 : 9 (2/0) [1/1]
|   |   |   |   Taste.Preferences = 1
|   |   |   |   |   Likes.Punjabi = 0 : 9 (7/2) [5/2]
|   |   |   |   |   Likes.Punjabi = 1 : 12 (11/5) [3/1]
|   |   |   |   Taste.Preferences = 2 : 9 (1/0) [0/0]
|   |   |   |   Taste.Preferences = 3 : 9 (11/2) [6/0]
|   BMI >= 25.53 : 1 (21/2) [7/1]
FoodType.NonVegetarian = 1
|   BMI < 26.04
|   |   Likes.Chinese = 0
|   |   |   Taste.Preferences = 0 : 6 (3/1) [2/0]
|   |   |   Taste.Preferences = 1
|   |   |   |   Likes.Punjabi = 0
|   |   |   |   |   Likes.Thai = 0
|   |   |   |   |   |   Likes.Maharashtrian = 0 : 10 (2/1) [4/2]
|   |   |   |   |   |   Likes.Maharashtrian = 1 : 8 (11/1) [6/0]
|   |   |   |   |   Likes.Thai = 1 : 13 (3/1) [0/0]
|   |   |   |   Likes.Punjabi = 1 : 10 (15/3) [1/1]
|   |   |   Taste.Preferences = 2 : 6 (1/0) [0/0]
|   |   |   Taste.Preferences = 3 : 8 (7/3) [3/2]
|   |   Likes.Chinese = 1
|   |   |   FoodType.Egg = 0 : 2 (20/5) [11/5]
|   |   |   FoodType.Egg = 1 : 8 (7/4) [8/5]
|   BMI >= 26.04 : 1 (20/4) [11/0]

Size of the tree : 35

Time taken to build model: 0.02 seconds

=== Stratified cross-validation ===
=== Summary ===
```


```
=== Stratified cross-validation ===
=== Summary ===

Correctly Classified Instances         160               60.6061 %
Incorrectly Classified Instances       104               39.3939 %
Kappa statistic                          0.5475
Mean absolute error                      0.0602
Root mean squared error                  0.1871
Relative absolute error                 57.7013 %
Root relative squared error             82.0782 %
Total Number of Instances              264

=== Detailed Accuracy By Class ===

                 TP Rate  FP Rate  Precision  Recall   F-Measure  MCC      ROC Area  PRC Area  Class
                 0.484    0.034    0.652      0.484    0.556      0.513    0.760     0.534     0
                 0.897    0.058    0.813      0.897    0.852      0.810    0.915     0.732     1
                 0.909    0.058    0.588      0.909    0.714      0.702    0.921     0.584     2
                 0.000    0.004    0.000      0.000    0.000      -0.007   0.577     0.019     3
                 0.000    0.020    0.000      0.000    0.000      -0.028   0.809     0.120     4
                 0.000    0.004    0.000      0.000    0.000      -0.007   0.441     0.011     5
                 0.600    0.012    0.500      0.600    0.545      0.538    0.992     0.539     6
                 0.000    0.000    0.000      0.000    0.000      0.000    0.456     0.008     7
                 0.750    0.078    0.571      0.750    0.649      0.600    0.906     0.642     8
                 0.647    0.091    0.512      0.647    0.571      0.504    0.879     0.509     9
                 0.391    0.017    0.692      0.391    0.500      0.488    0.855     0.424     10
                 0.000    0.000    0.000      0.000    0.000      0.000    0.706     0.037     11
                 0.636    0.062    0.483      0.636    0.549      0.508    0.861     0.444     12
                 0.200    0.004    0.500      0.200    0.286      0.308    0.786     0.141     13
                 0.000    0.004    0.000      0.000    0.000      -0.008   0.650     0.035     14
                 0.000    0.000    0.000      0.000    0.000      0.000    0.603     0.044     15
                 0.000    0.000    0.000      0.000    0.000      0.000    0.987     0.410     16
Weighted Avg.    0.606    0.051    0.559      0.606    0.569      0.534    0.853     0.514

=== Confusion Matrix ===

  a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q   <-- classified as
 15  5  2  0  0  0  3  0  4  1  0  0  1  0  0  0  0 |  a = 0
  0 52  2  0  0  0  0  0  1  2  0  0  1  0  0  0  0 |  b = 1
  0  1 20  0  0  0  0  0  1  0  0  0  0  0  0  0  0 |  c = 2
  0  0  0  0  0  0  0  0  0  1  0  0  2  0  0  0  0 |  d = 3
  3  0  0  0  0  0  0  0  0  4  0  0  3  0  0  0  0 |  e = 4
  0  0  0  0  0  0  0  0  0  3  0  0  0  0  0  0  0 |  f = 5
  1  0  0  0  0  0  3  0  0  0  1  0  0  0  0  0  0 |  g = 6
  0  0  0  0  0  0  0  0  0  2  0  0  0  0  0  0  0 |  h = 7
  1  1  4  0  0  0  0  0 24  1  1  0  0  0  0  0  0 |  i = 8
```

As can be seen from the analysis above the random forest provides highest performance metric compared to other tree methods.
A model file is generated for our dataset and placed in android app

java -cp wekaSTRIPPED.jar weka.classifiers.trees.RandomForest -I 100 -K 0 -S 1 -c 4 -t x.csv.arff -d cpu.model



```
Random forest of 100 trees, each constructed while considering 5 random features.
Out of bag error: 0.3674


Time taken to build model: 0.33 seconds
Time taken to test model on training data: 0.04 seconds

=== Error on training data ===

Correctly Classified Instances         263               99.6212 %
Incorrectly Classified Instances         1                0.3788 %
Kappa statistic                          0.9957
Mean absolute error                      0.0153
Root mean squared error                  0.0463
Relative absolute error                 14.6804 %
Root relative squared error             20.3369 %
Coverage of cases (0.95 level)         100      %
Mean rel. region size (0.95 level)      15.4635 %
Total Number of Instances              264


=== Confusion Matrix ===

  a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q   <-- classified as
 31  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 |  a = 0
  0 58  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0 |  b = 1
  0  0 22  0  0  0  0  0  0  0  0  0  0  0  0  0  0 |  c = 2
  0  0  0  3  0  0  0  0  0  0  0  0  0  0  0  0  0 |  d = 3
  0  0  0  0 10  0  0  0  0  0  0  0  0  0  0  0  0 |  e = 4
  0  0  0  0  0  3  0  0  0  0  0  0  0  0  0  0  0 |  f = 5
  0  0  0  0  0  0  5  0  0  0  0  0  0  0  0  0  0 |  g = 6
  0  0  0  0  0  0  0  2  0  0  0  0  0  0  0  0  0 |  h = 7
  0  0  0  0  0  0  0  0 32  0  0  0  0  0  0  0  0 |  i = 8
  0  0  0  0  0  0  0  0  0 33  0  0  1  0  0  0  0 |  j = 9
  0  0  0  0  0  0  0  0  0  0 23  0  0  0  0  0  0 |  k = 10
  0  0  0  0  0  0  0  0  0  0  0  2  0  0  0  0  0 |  l = 11
  0  0  0  0  0  0  0  0  0  0  0  0 22  0  0  0  0 |  m = 12
  0  0  0  0  0  0  0  0  0  0  0  0  0  5  0  0  0 |  n = 13
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  4  0  0 |  o = 14
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  5  0 |  p = 15
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  3 |  q = 16
```

```
=== Stratified cross-validation ===

Correctly Classified Instances         170               64.3939 %
Incorrectly Classified Instances        94               35.6061 %
Kappa statistic                          0.5928
Mean absolute error                      0.0608
Root mean squared error                  0.1717
Relative absolute error                 58.3432 %
Root relative squared error             75.3448 %
Coverage of cases (0.95 level)          94.3182 %
Mean rel. region size (0.95 level)      27.8966 %
Total Number of Instances              264


=== Confusion Matrix ===

  a  b  c  d  e  f  g  h  i  j  k  l  m  n  o  p  q   <-- classified as
 12  5  1  0  2  0  0  0  4  4  1  0  2  0  0  0  0 |  a = 0
  1 51  0  0  1  0  0  0  1  1  1  0  1  1  0  0  0 |  b = 1
  0  1 17  0  0  0  0  0  1  0  3  0  0  0  0  0  0 |  c = 2
  0  1  0  0  1  0  0  0  0  0  0  0  0  1  0  0  0 |  d = 3
  3  0  0  0  3  0  0  0  0  1  0  0  3  0  0  0  0 |  e = 4
  0  0  0  0  0  0  0  0  0  2  0  0  1  0  0  0  0 |  f = 5
  0  0  0  0  0  0  4  0  0  0  0  0  0  0  0  1  0 |  g = 6
  0  0  0  0  0  0  0  0  0  2  0  0  0  0  0  0  0 |  h = 7
  2  2  1  0  0  0  0  0 25  1  0  0  0  0  0  1  0 |  i = 8
  2  1  0  0  1  0  0  0  2 24  0  0  3  0  1  0  0 |  j = 9
  0  0  2  0  0  0  0  0  2  0 18  0  1  0  0  0  0 |  k = 10
  0  0  0  0  0  0  0  0  0  0  0  0  1  0  1  0  0 |  l = 11
  0  1  0  0  2  0  0  0  0  4  1  1 13  0  0  0  0 |  m = 12
  0  0  1  0  0  0  0  0  1  0  2  0  0  0  0  1  0 |  n = 13
  0  1  0  0  0  0  0  0  0  2  0  1  0  0  0  0  0 |  o = 14
  0  1  0  0  0  0  0  0  1  0  2  0  0  1  0  0  0 |  p = 15
  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  0  3 |  q = 16

PS C:\Users\ajayb\OneDrive\Desktop\tensor\Weka-for-Android-master>
```

This completes the analysis process, Further structuring and functioning of model takes place using weka api in android studio where user input generates predictions using the model for recommended package.
Note:Training and testing splits are made using cross-validation with 10 folds

Weka:

Weka is an open source Java based platform containing various machine learning algorithms. Weka is a collection of machine learning algorithms for data mining tasks. The algorithms can either be applied directly to a data set or called from your own Java code. Weka contains tools for data per-processing, classification, regression, clustering, association rules, and visualization. It is also well-suited for developing new machine learning schemes.

It can be used to detect the various hidden patterns in your data set and find the most determining factors out of many .

Cross-validation, a standard evaluation technique, is a systematic way of running repeated percentage splits. Divide a dataset into 10 pieces ("folds"), then hold out each piece in turn for testing and train on the remaining 9 together. This gives 10 evaluation results, which are averaged. In "stratified" cross-validation, when doing the initial division we ensure that each fold contains approximately the correct proportion of the class values. Having done 10-fold cross-validation and computed the evaluation results, Weka invokes the learning algorithm a final (11th) time on the entire data set to obtain the model that it prints out.

2.6 SOFTWARE DEVELOPMENT LIFE CYCLE :

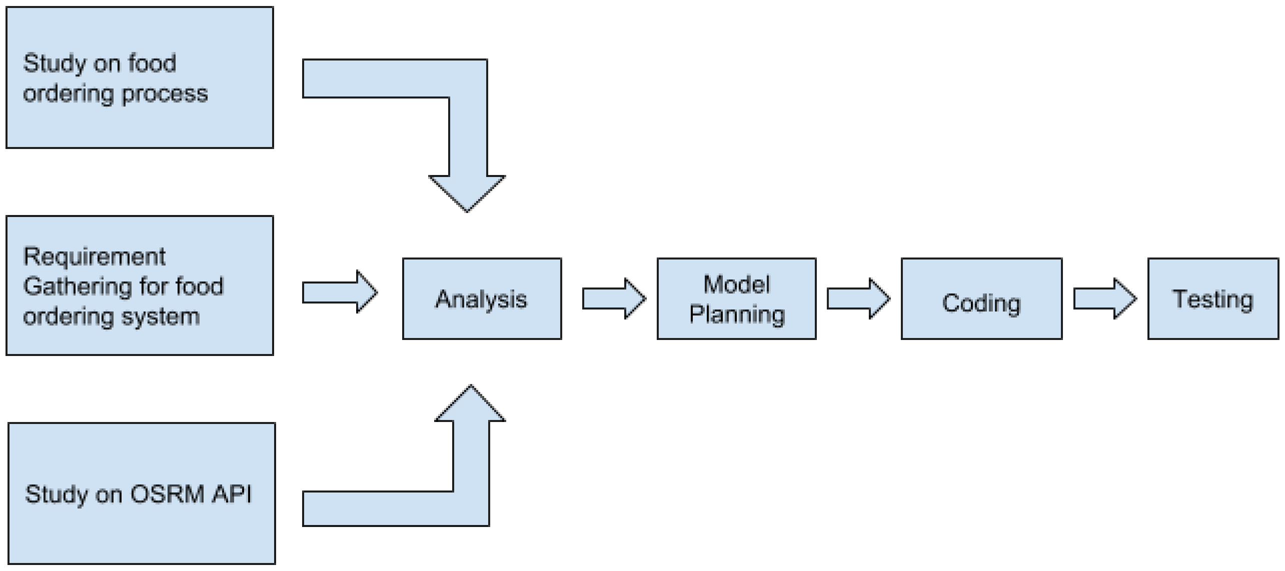

Study on food ordering process
Requirement Gathering for food ordering system
Study on OSRM API
Analysis
Model Planning
Coding
Testing

Fig 2 : SDLC

2.7 UNIFIED PAYMENT INTERFACE (UPI) :

Unified Payment Interface (UPI) allows easy and convenient medium of fund transfer between bank accounts in real-time using a virtual payment address (VPA) without entering bank details whatsoever. Customers can pay the merchant either using their VPA or by scanning the QR code at the retail stores.
As a result, more and more online retailers and apps are offering UPI payment methods to make financial transactions easier for its customers. UPI is an initiative of the National Payment Corporation of India (NPCI) which provides a common library module comprising of the UPI integration program for the websites and mobile apps.

2.8 INTEGRATING UPI WITH ANY APPLICATION:

Currently, only the banks can act as Payment Service Providers (PSPs) for UPI and hence the first step to integrate UPI in your website/application is to partner or tie up with a bank as a merchant. The banks which are offering the UPI SDK include RBL, ICICI, Yes Bank, and Axis Bank.
Once you have the access to UPI SDK, you just need to add it to your website/application to enable transactions using UPI.

2.9 BENEFITS OF USING UPI:

NPCI has set UPI specifications for the Android phones, IOS and Windows based mobiles will also be able to download UPI based apps. Some of the benefits for the applications for integrating UPI include:

1. Instant transfers
Considering that UPI also works in real-time and transfers funds immediately without revealing the bank account details to the payee. Undoubtedly, it has become one of the most used payment modes other than the debit/credit cards. Integrating UPI will add another payment mode to your app which attracts users, as the more payment options an app has, the better it is for the users.
2. Negligible charges
Moreover, as compared to the other payment methods like NEFT and IMPS, UPI does not charge anything. So, integrating your app with UPI gives your users a free of charge payment method.
3. More secure
Further, as paying through UPI does not require users to enter their bank account and card details for making payments, it is definitely more secure mode of payment. So, if your app offers payments through UPI, the users can make quick and easy payments without having to load their e-wallet or enter bank details.
UPI is being considered as an incredibly well-designed payment solution as compared to all the other payment mediums. Integrating UPI in your website/application would surely attract more users as it is a convenient and a preferred payment mode.

2.10 TYPES OF UPI PAYMENT GATEWAYS/APPS

1.Web-flow UPI integration
Platform: Web, mobile web, Android, iOS
This is the most popular UPI integration mode. Let's see how it works from a customer perspective.

1. Customer selects items, fills details like shipping address, delivery instructions etc
2. Selects UPI as payment mode, enters his mobile app UPI ID, also called as VPA (Virtual Payment Address), verifies and submits it.
3. Need to check his mobile phone and do a *two-way authentication* which involves:

a)Opening the UPI app (in this case BHIM app) using app Pass code/Face-id/thumb impression as the case may be.

b)Authenticating the transfer by entering UPI PIN. (UPI-PIN is a 4-6 digit secret code you create/set when you link a bank account with your UPI mobile app.)

Once the payment has been authenticated by the customer, the transaction will be marked as successful.

Payment flow:

The customer has to first enter his VPA(or UPI ID), open his UPI mobile app, do 2-factor authentication on his mobile phone and then come back to the website.

Suitability:

Any business that wants to provide UPI as a payment mode and doesn't mind redirecting the customer to a third party site (UPI app provider such as Google Pay, PhonePe). It is a relatively economical UPI integration mode.

2. UPI GOOGLE PAY INTEGRATION

Platform: Web, mobile web, Android, iOS (Payer needs to have Google Pay on their phone)

This is similar to the UPI web flow. Hence the payer needs to enter his phone number instead of the UPI ID.

UPI Google Pay integration is a very convenient checkout flow as the payer/customer need not remember the UPI ID/VPA. Cash-free provides UPI Google Pay integration.

Payment flow:

Customer enters a mobile number instead of UPI VPA/UPI ID, do a 2-factor authentication (as explained before) and completes the payment.

Suitability:

Any small and medium business that wants to provide UPI payment option. This mode should be used when you see that Google Pay is a preferred UPI payment mode among customers.

3. Intent flow UPI integration

Platform: Android

Payment flow:

As soon as the user chooses the UPI payment app, the app installed in his mobile launches automatically, the user doesn't need to enter UPI VPA or phone number as it is auto-filled along with other payment details including the amount to be paid.

Suitability:

Intent flow is a friction less checkout experience as it automatically launches a preferred UPI mobile app during payment. It is ideal when your customers are placing an order directly on an Android app. Intent flow is offered as a part of Cashfree's Android SDK.

4. UPI SDK flow integration

Platform: Android, iOS

Using the UPI SDK, the merchant can receive the payment without the customer having to open any third party app. This form of integration works only on mobiles(Android & iPhone) and is provided by banks like RBL, ICICI, Yes Bank, and Axis Bank. For this, you need to contact the bank directly and request for NPCI UPI Android SDK to receive payments.

Payment flow:

I. In this case, the bank will create a VPA and then you can get paid by customers on the same VPA on your mobile app. Once you have the access to UPI SDK, you need to add it to your website/mobile application to enable transactions using UPI.

II. Here no separate UPI app such as BHIM, Google Pay etc is required. Since there is no redirection to any third party application, using UPI SDK integration, the conversion rate increases.

Suitability:

Typically big businesses having a high volume of daily inward transactions on its mobile app opt for this type of integration.

2.10 ANALYSIS :

The food ordering system can be divided into the following main components :

- Android App
- Payment Gateway Integration through UPI
- Machine Learning based Model (for prediction of food choice for user)

3. SOFTWARE REQUIREMENTS SPECIFICATION

3.1 INTRODUCTION

3.1.1 PROJECT SCOPE :

This system will help to manage & run the online food ordering business systematically . In this system, we will provide an App that can be used by the customers to order food. Customers can also give feedback through this app. So that vendor can evaluate the whole system.

This will ultimately lead to providing good quality food to the customers and create an opportunity to appoint more chefs to provide home-made hygienic food .Customers can also make payments through debit or credit cards using POS which will be integrated within the management system. Initially the food delivery service is limited within the Pune city (Specific Areas).

3.1.2 USER CLASSES AND CHARACTERISTICS :

The following are the main user classes with their characteristics :

1.Customers :

The customers are the registered users who will order the food. They should be able to choose a package of their own choice which will include 7-days menu. The customers should be able to create a package of their own from the available list of items. The customers should also be able to give feedback.

2. Cooks :

The cooks should be able to see how many customers have opted their weekly package so that they can keep a track on their inventory. The should be able to update and notify the users in case of
any last moment changes.

3.1.3 ASSUMPTIONS AND DEPENDENCIES :

- The system will use third party service for payment
- The system will use third party service for delivery of food
- Only the cooks who were trained by the employer can register as cooks in the system
- The system has geographical limitation up-to one city

3.2 FUNCTIONAL REQUIREMENTS

3.2.1 Payment Process

3.2.1.1 Description and Priority :

The user should be able to do payment through any UPI supported App

Priority – HIGH

Stimulus/Response Sequences :

The user can pay through any of the UPI supported apps.

Functional Requirements:

The user needs to install at-least one UPI app for doing payments.

REQ-1: User should have a bank account being set up with the UPI app

REQ-2: UPI supported apps should be installed and setup

3.2.2 Recommendation of Personalized package based on the preferences: Description and Priority:

Based on the data received from the registration form and daily feedback the customers will be recommended a personalized package based on his/her likes/dislikes and dietary needs

Priority – HIGH

Stimulus/Response Sequences :

Customer can then choose between the available packages and the recommended one

Functional Requirements:

The customer needs to give feedback and give relevant information during registration

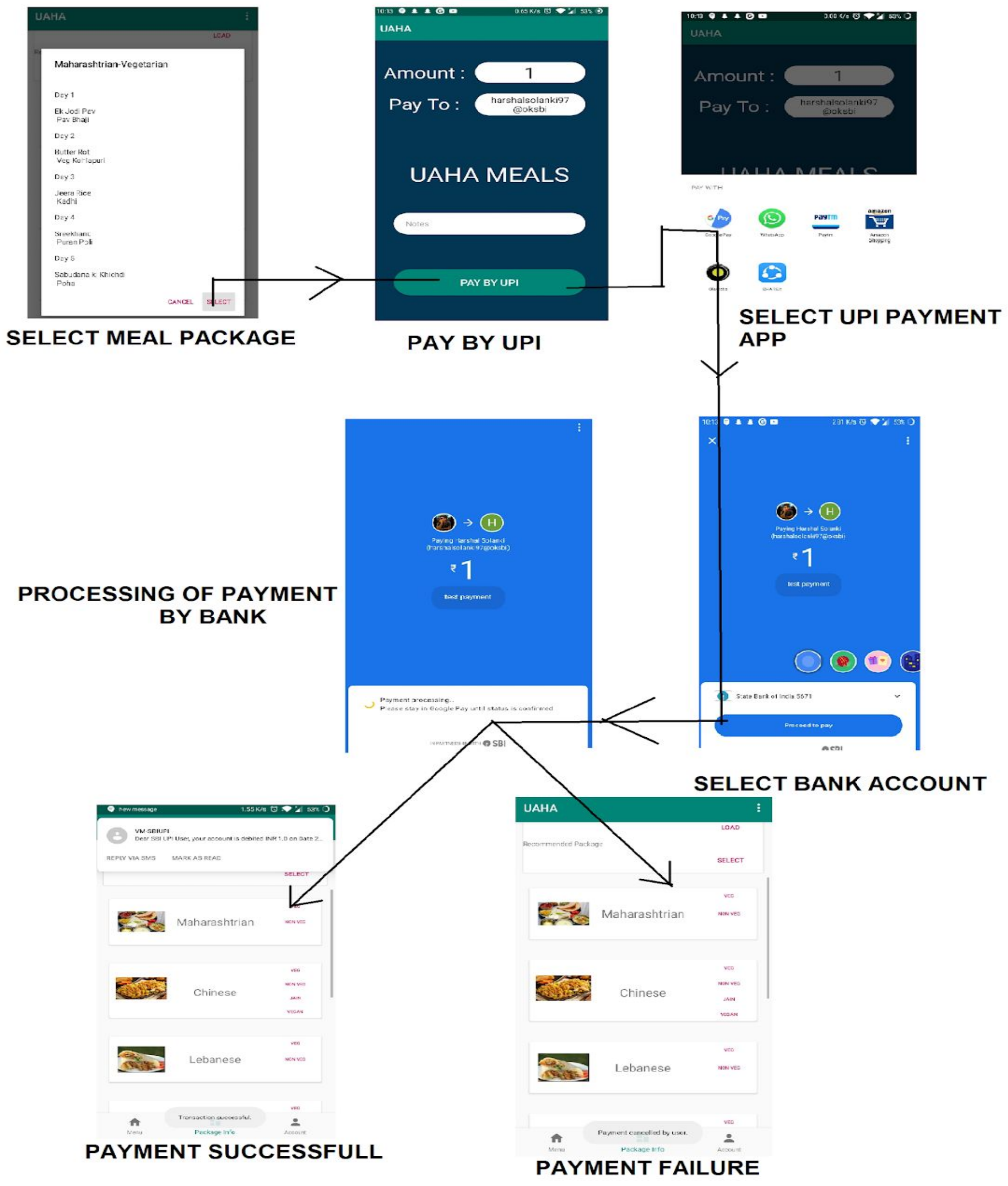
This data can be used to suggest a package according to the customers need.

REQ-1: Customers should determine personal information correctly during registration

REQ-2: Customers should give daily feedback

3.3 EXTERNAL INTERFACE REQUIREMENTS

3.3.1 User Interfaces :



3.3.2 Hardware Interface :

- Android smartphone
- Min RAM 1 GB
- Min internal memory 16 GB

3.3.3 Software Requirements :

- Android with version 7.0 and above

3.3.4 Communication Requirements :

- Internet Access
- GPS

3.4 NON-FUNCTIONAL REQUIREMENTS :

3.4.1 Performance Requirements : The system should be SMART:

S-Specific
M-Measurable
Achievable
R- Realistic
T- Timely

Response time- Should be consistent (ideally a latency of less than a second is acceptable) Workload - The system should handle many concurrent users at a time

Scalability – The system should be able to incorporate the changes made in the packages and also handle the increase in no of users.
Platform Considerations Android

3.4.2 Safety Requirements : App only works over internet

Do not use if the internet is not working properly as the app might not perform as desired in very low bandwidth Keep GPS on so that users Location could be detected

3.4.3 Security Requirements :

Email based authentication by confirming the email address.

3.4.4 Software Quality Attributes :

Salable : The system can handle increase in no of users Portable : The system can be used through an app Robust : The system can handle any erroneous
Adaptive : Adapts to the requirements of the user and changing nature of packages

Ease of use : User-friendly design

3.5 SYSTEM REQUIREMENTS :

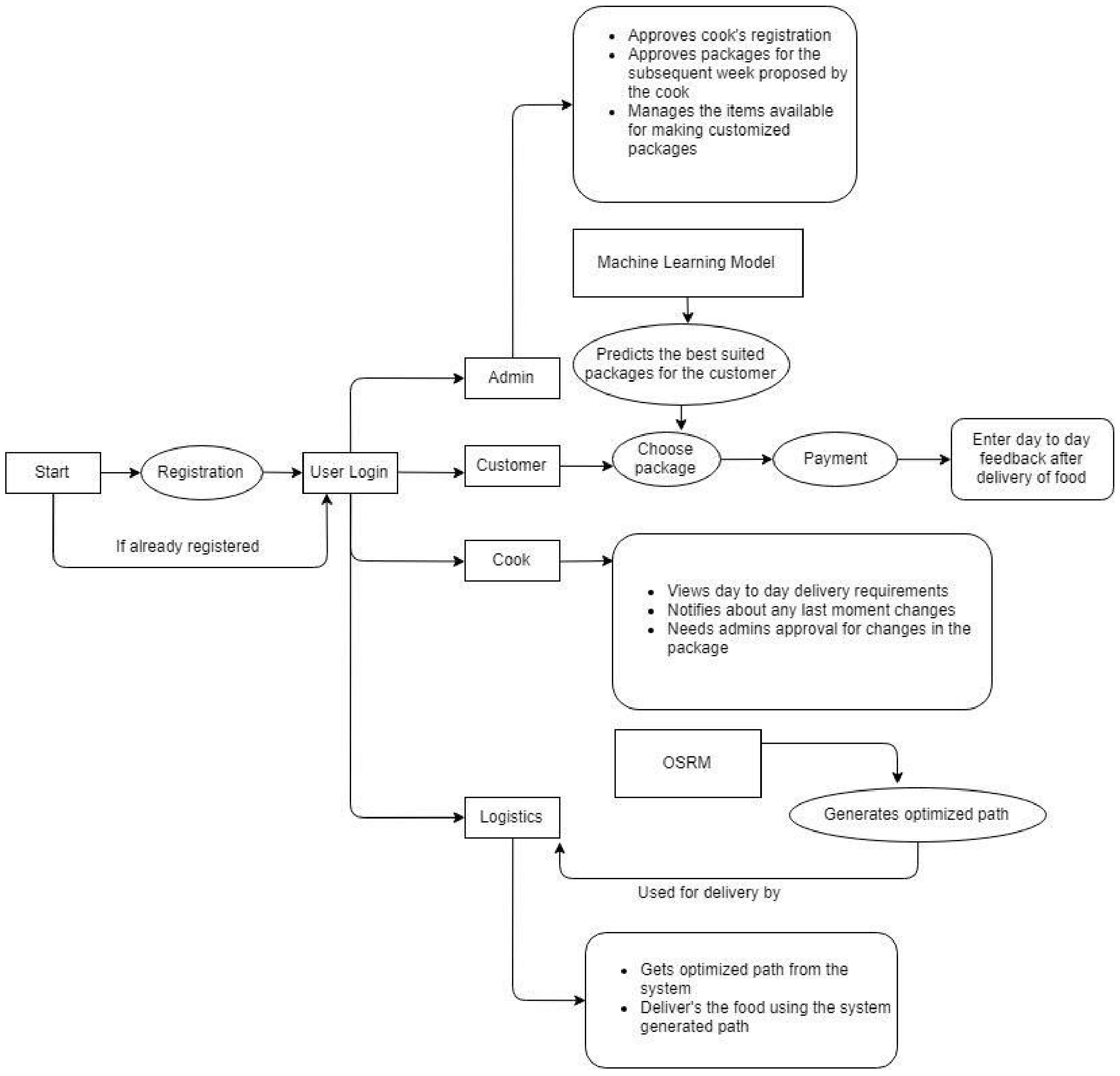
3.5.1 Database Requirements :

At backed the data would be stored on fire base. The same would be used for querying while doing database operations.

3.5.2 Software Requirements (Platform choice):

The system is to developed using android studio for front end and freebase for back-end as well as server

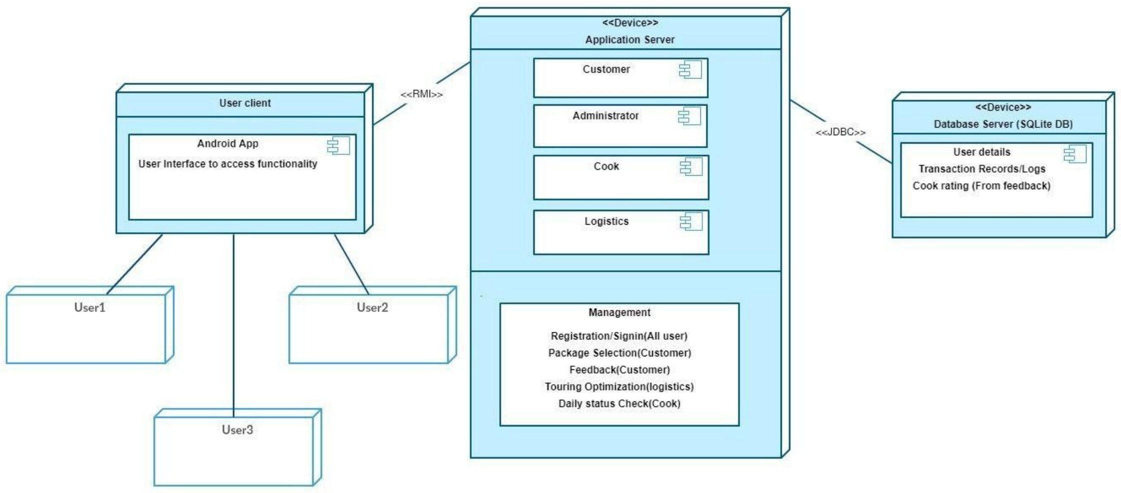
FLOW OF THE SYSTEM


Approves cook's registration
Approves packages for the subsequent week proposed by the cook
Manages the items available for making customized packages
Machine Learning Model
Predicts the best suited packages for the customer
Admin
Start
Registration
User Login
Customer
Choose package
Payment
Enter day to day feedback after delivery of food
If already registered
Cook
Views day to day delivery requirements
Notifies about any last moment changes
Needs admins approval for changes in the package
OSRM
Generates optimized path
Logistics
Used for delivery by
Gets optimized path from the system
Deliver's the food using the system generated path

4 SYSTEM DESIGN

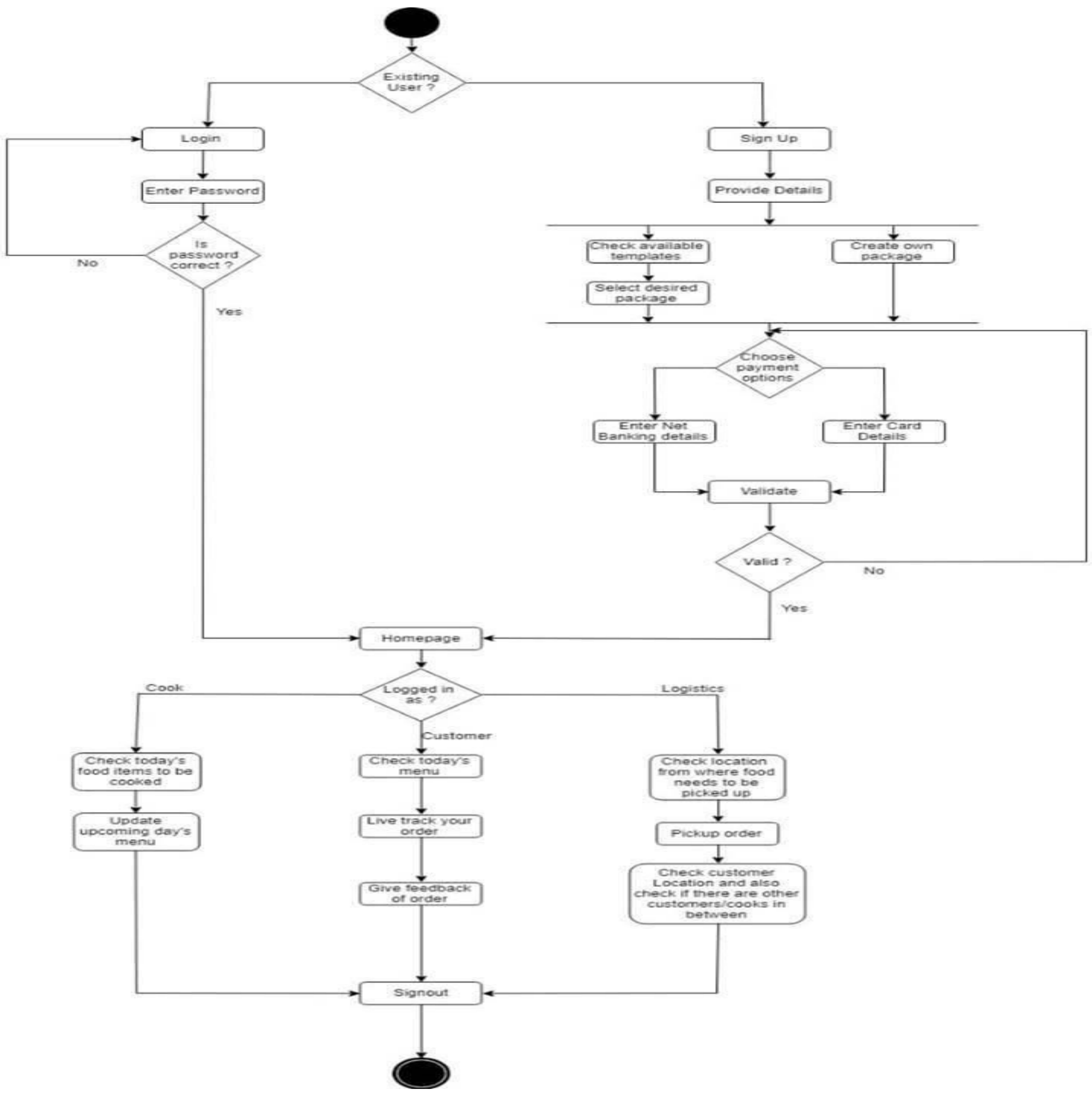
4.1 SYSTEM ARCHITECTURE

Deployment Diagram For Personalized Food App

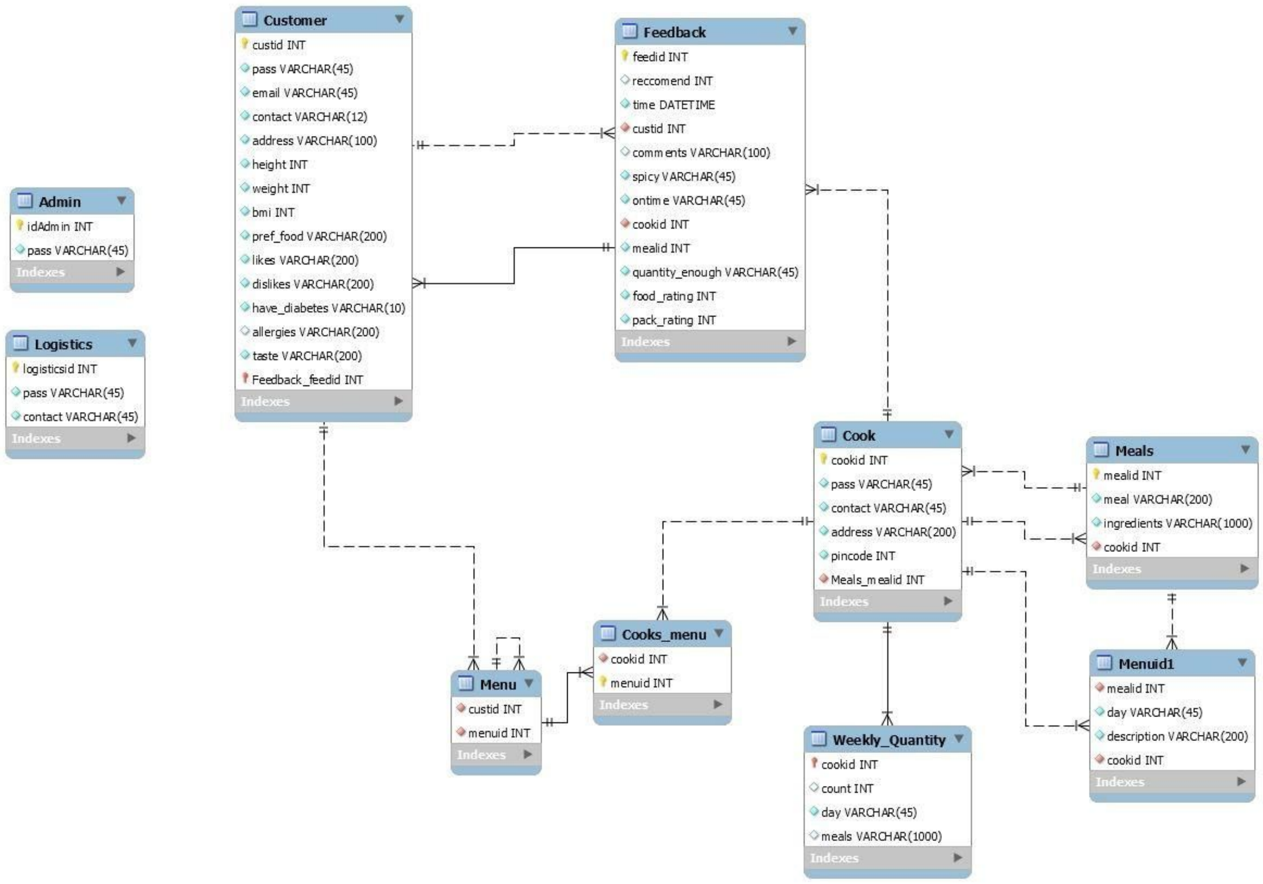


creately
www.creately.com • Online Diagramming

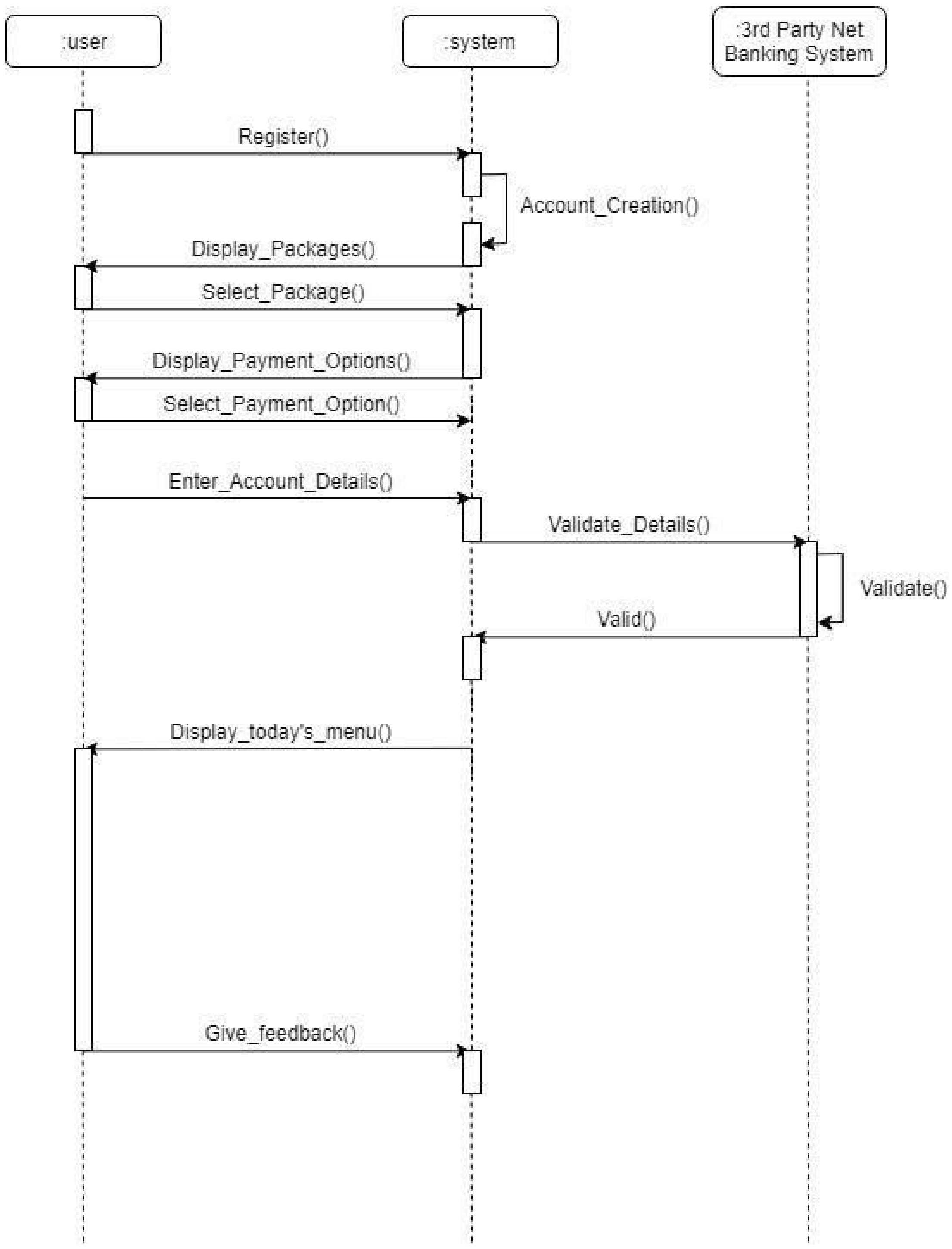
4.2 DATA FLOW DIAGRAMS



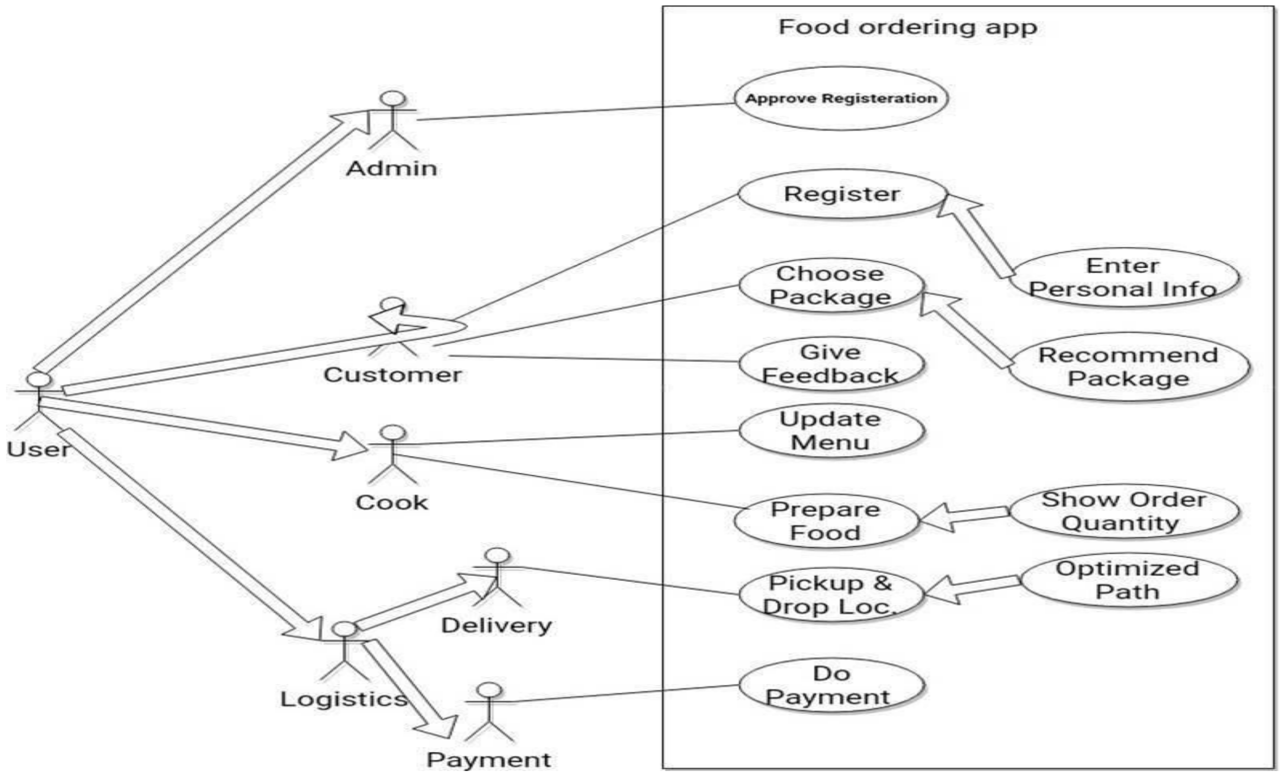
4.3 ENTITY RELATIONSHIP DIAGRAM



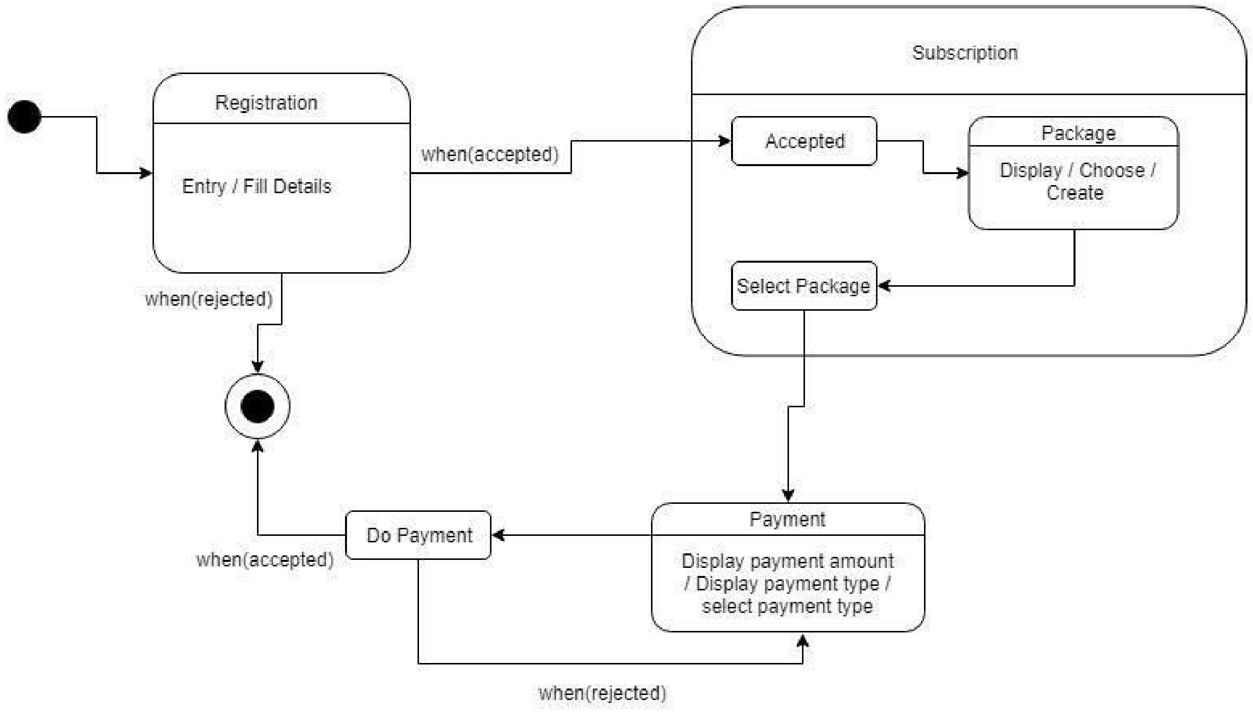
4.4 SEQUENCE DIAGRAM

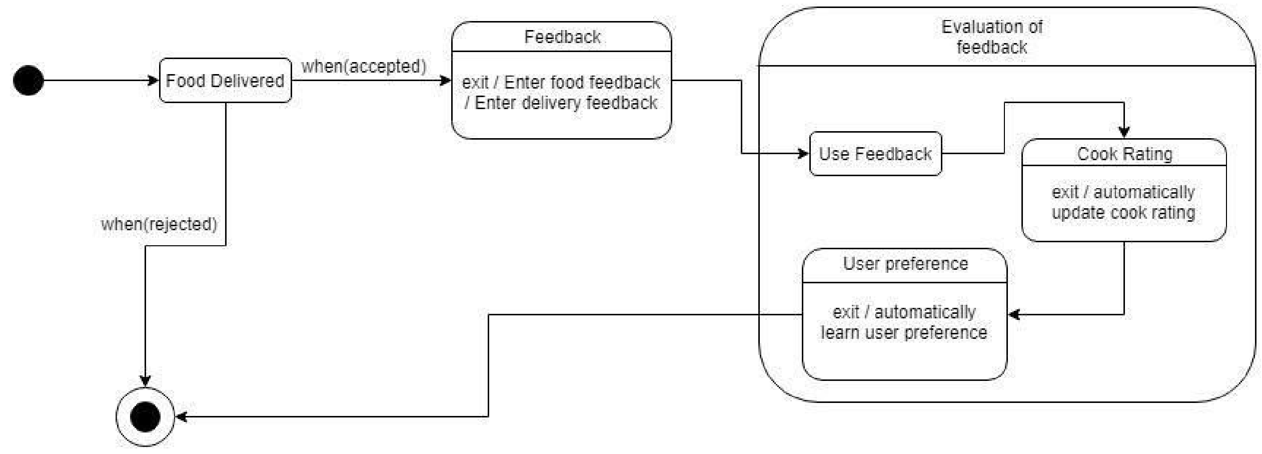


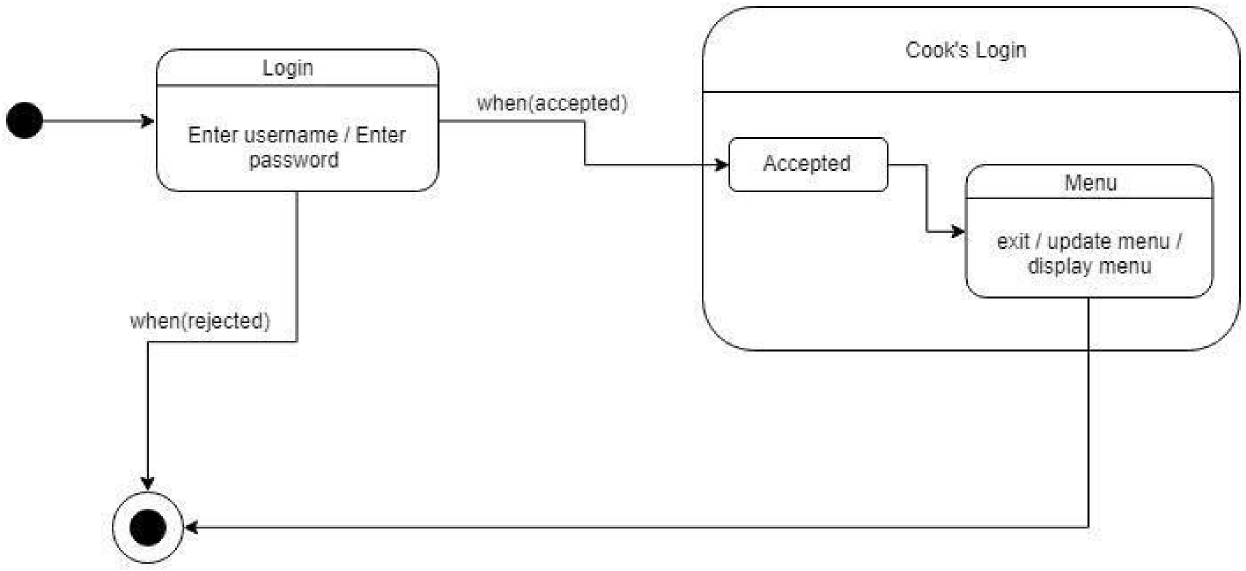
4.5 USE CASE DIAGRAM



4.6 STATE TRANSITION DIAGRAMS




Food Delivered
when(accepted)
Feedback
exit / Enter food feedback / Enter delivery feedback
Evaluation of feedback
Use Feedback
Cook Rating
exit / automatically update cook rating
User preference
exit / automatically learn user preference
when(rejected)


Login
Enter username / Enter password
when(accepted)
Cook's Login
Accepted
Menu
exit / update menu / display menu
when(rejected)

5 OTHER SPECIFICATION

5.1 ADVANTAGES:

Unlike other conventional food delivery apps this app will provide multi pickup and multiple drops for logistics team

The subscription a select and forget system where the user need not constantly browse but the feedback provided will customize food app according to the preference of the customer at the back-end

5.2 LIMITATIONS:

The system will use third party service for payment

The system will use third party service for delivery of food

Only the cooks who were trained by the employer can register as cooks in the system

The system has geographical limitation up-to one city

6 CONCLUSION AND FUTURE WORK

People who order food online on a daily basis are fed up of eating hotel food. Thus, we present a food ordering system that will provide a platform for people to order hygienic home cooked food online, as there are very less sources of getting home cooked food online. Therefore, this system would prove to be a promising one. This system also ensures good quality of service and customer satisfaction. Therefore, the proposed food ordering

system has the potential to attract customers and also adds to the efficiency of maintaining customers ordering and billing sections.

REFERENCES

A Food Ordering System with Delivery Routing Optimization Using Global Positioning System
(GPS) Technology and Google Maps- Roy Deddy Hasiholan Tobing

Android Application for Local Food Ordering System Shubham Takalkar, Devendra Phatak, Kumar Abhinav, Salman Hadi, R. H. Borhade Information Technology Engineering Department, Smt. Kashibai Navale College Of Engineering, Pune. Savitribai Phule Pune University

Automated Food Ordering System with Real-Time Customer Feedback by Shweta Shashikant
Tanpure, Priyanka R. Shidankar, Madhura M. Joshi

Design and Implementation of an Android Application using wifi-enabled Devices for the Food Servicing Industryby Alberto Bañacia, Marc Dindo Fernando, Arnel Requillo Jr., and Nelson Rubi Jr. EE/ECE Department, University of San Carlos Cebu City, Philippines