DBMS
Assignment 3

Online Restaurant Aggregator and Food Order System

Report

Group members :

1.Apurv Kumar	- 14CS10006
2.Rameshwar Bhaskaran	- 14CS30027
3.Asket Agarwal	- 14CS30006
4.Shubham Sharma	- 14CS30034
5.Aniket Choudhary	- 14CS30004

Objective :

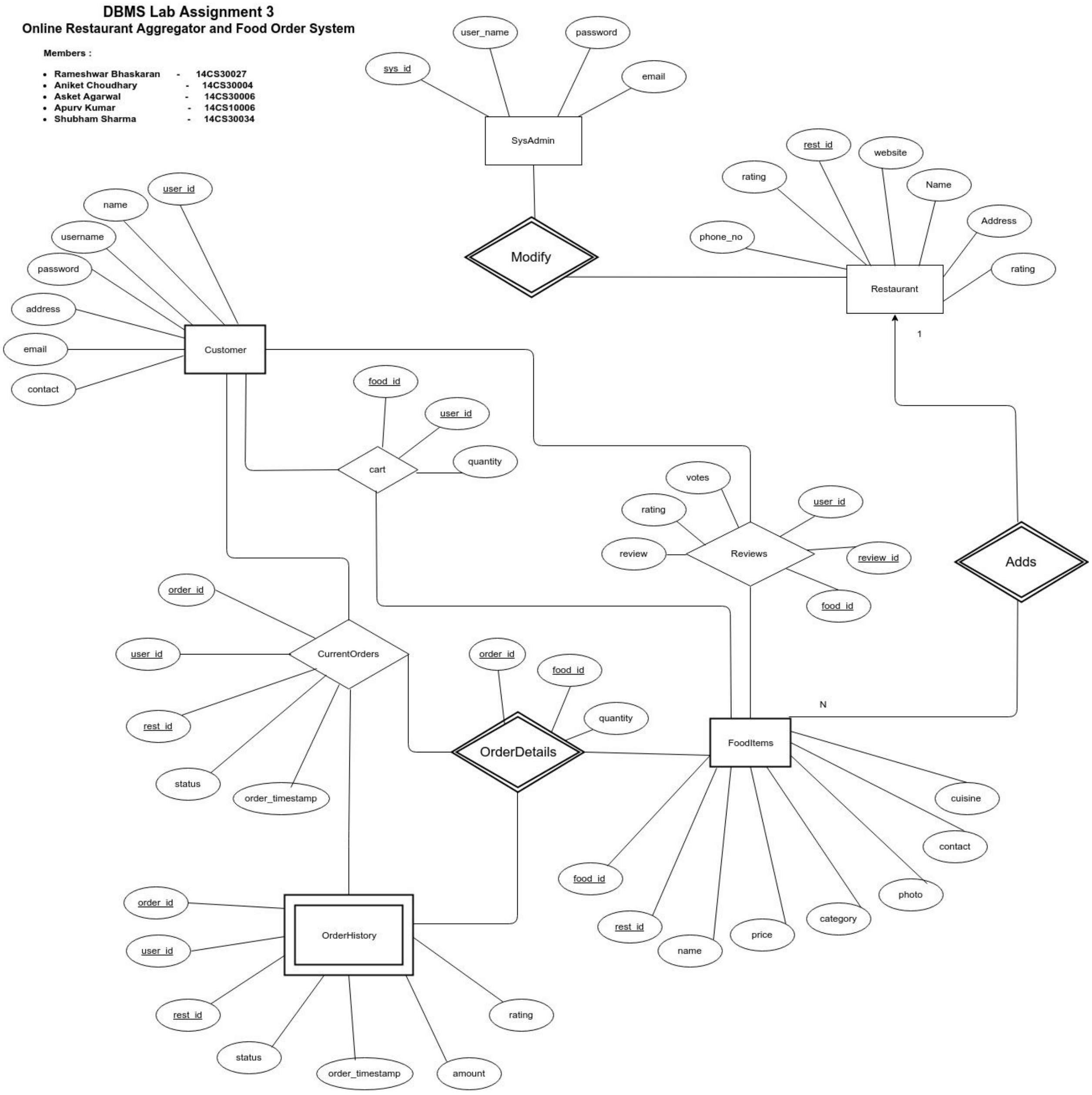
i. Web based application for online browsing and search of menu aggregated from multiple restaurants e.g., the services provided by commercial websites like www.zomato.com, www.just-eat.com .
ii. Customers can order food delivery online. Orders are to be forwarded to respective restaurants, who will deliver the food and bill the customers.
Users of the System: A. System Administrator B. Restaurants C. Customers

Functional Requirements:

i. The system must provide an entry page which will be the home page after login. The entry page should display available menu, restaurant information, search and browse by category interface and a customer purchase cart.
Optionally, the database of restaurants may be populated using the API offered by Zomato for a particular city say Kharagpur. URL: https://developers.zomato.com/api

ii. The system should provide a restaurant/customer registration module.
iii. iv. The system should allow customers to place orders and track their orders through different stages of delivery.
iv. The system should offer billing facility.
v. The system should track user history and suggest favorites (bonus points).

ER Diagram :



DBMS Lab Assignment 3
Online Restaurant Aggregator and Food Order System

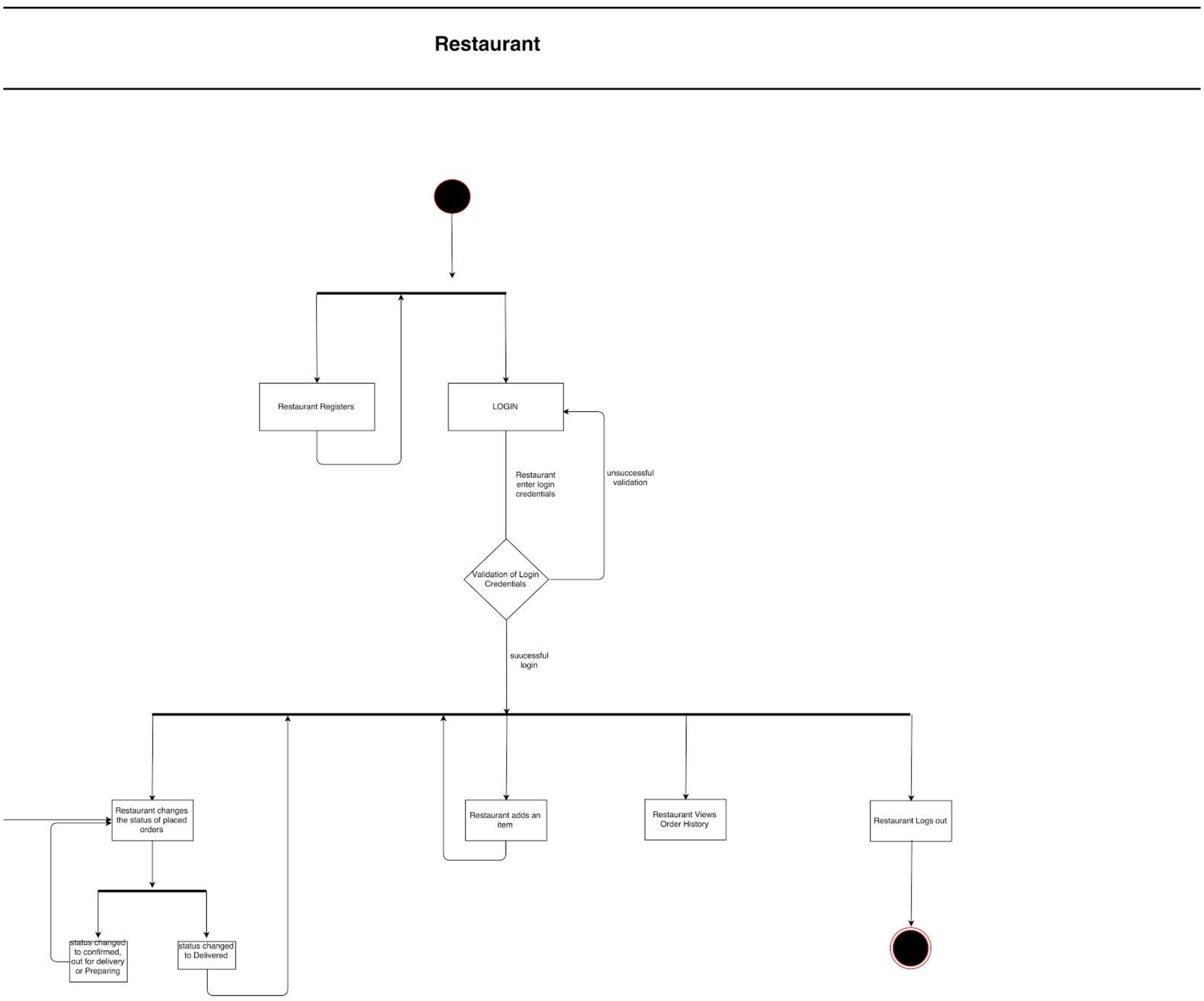
Members :

- **Rameshwar Bhaskaran - 14CS30027**
- **Aniket Choudhary - 14CS30004**
- **Asket Agarwal - 14CS30006**
- **Apurv Kumar - 14CS10006**
- **Shubham Sharma - 14CS30034**

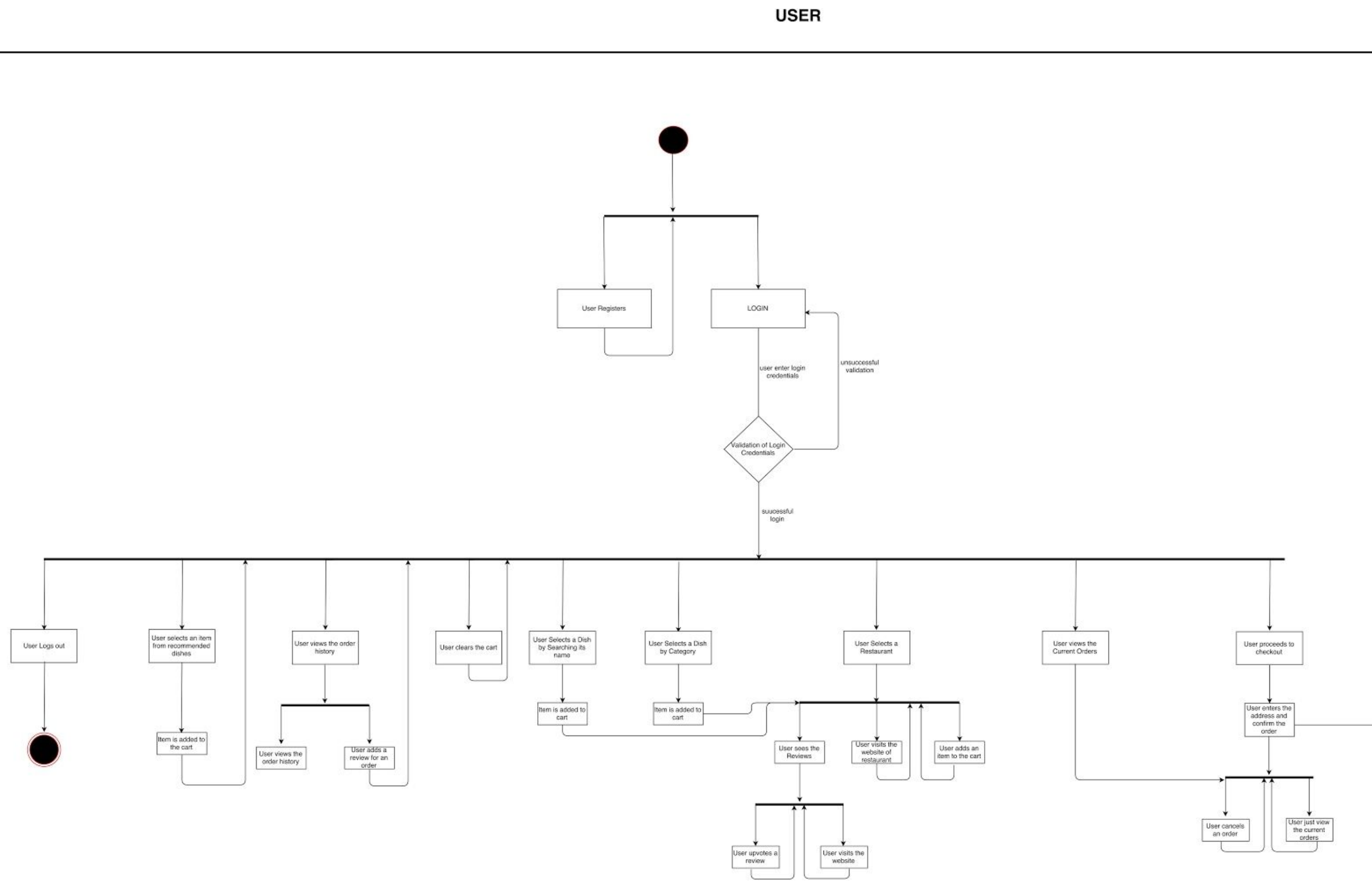
Workflow Diagram :

We used the default django admin for admin, the admin can modify any table including addition and deletion of any entries. So our project used the default django admin module and hence admin is not a part of our workflow diagram.

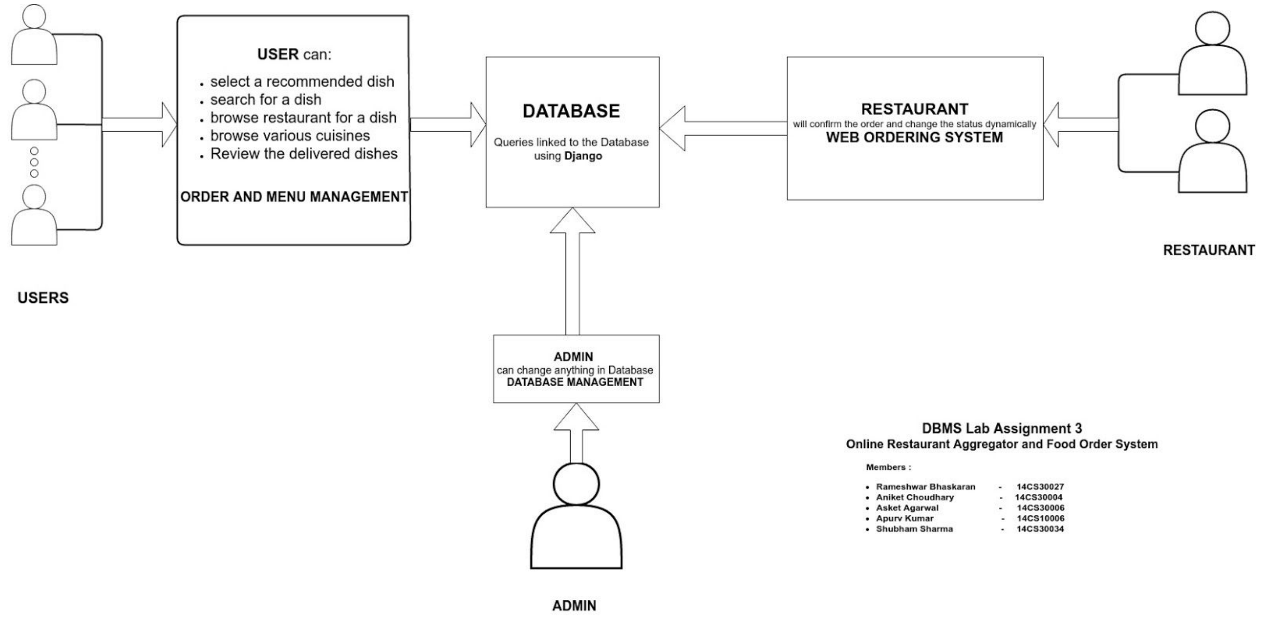
Restaurant :


Restaurant
Restaurant Registers
LOGIN
Restaurant enter login credentials
unsuccessful validation
Validation of Login Credentials
suucessful login
Restaurant changes the status of placed orders
Restaurant adds an item
Restaurant Views Order History
Restaurant Logs out
status changed to confirmed, out for delivery or Preparing
status changed to Delivered

Customer :


USER
User Registers
LOGIN
user enter login credentials
unsuccessful validation
Validation of Login Credentials
suucessful login
User Logs out
User selects an item from recommended dishes
Item is added to the cart
User views the order history
User views the order history
User adds a review for an order
User clears the cart
User Selects a Dish by Searching its name
Item is added to cart
User Selects a Dish by Category
Item is added to cart
User Selects a Restaurant
User sees the Reviews
User visits the website of restaurant
User adds an item to the cart
User upvotes a review
User visits the website
User views the Current Orders
User proceeds to checkout
User enters the address and confirm the order
User cancels an order
User just view the current orders

Module Diagram


USER can:
select a recommended dish
search for a dish
browse restaurant for a dish
browse various cuisines
Review the delivered dishes
ORDER AND MENU MANAGEMENT
USERS
DATABASE
Queries linked to the Database using Django
RESTAURANT
will confirm the order and change the status dynamically
WEB ORDERING SYSTEM
RESTAURANT
ADMIN
can change anything in Database
DATABASE MANAGEMENT
ADMIN

DBMS Lab Assignment 3
Online Restaurant Aggregator and Food Order System

Members :

• Rameshwar Bhaskaran	-	14CS30027
• Aniket Choudhary	-	14CS30004
• Asket Agarwal	-	14CS30006
• Apurv Kumar	-	14CS10006
• Shubham Sharma	-	14CS30034

Our site is currently deployed : http://fastfood.pythonanywhere.com/
Link to our project on github : https://github.com/apurv1205/fastFood

List of Tools used

1.https://www.pythonanywhere.com/ : for deploying our site and database on their web servers
2.https://github.com/ : for version control and collaborating our work
3.Zomato API - https://developers.zomato.com/api for adding restaurants and reviews information to our database.
4.http://menus.nypl.org/data - for adding food items to our databases, we had to modify the food items' rest id to synchronise it with the restaurant's information we collected from zomato API.
5.Django 1.10 - for backend and database, we used the default sqlite3 database with django.
6.python-recsys for collaborative recommendation system - https://github.com/ocelma/python-recsys
7.python-editdistance for search and browse feature to find similar words to the searched word
8.List of dependencies are as follows :

- appdirs==1.4.2
- csc-pysparse==1.1.1.4
- csc-utils==0.6.7
- decorator==4.0.11
- Divisi2==2.2.5
- Django==1.10.5
- editdistance==0.3.1
- networkx==1.11
- numpy==1.12.0
- packaging==16.8
- pyparsing==2.1.10
- python-recsys==0.2
- scipy==0.18.1
- six==1.10.0