Proposal done

**Normalization**

1. Identifiying Entities initially :

* User ( user\_id , email, password, firstname, lastname, address, phone number)
* Orders ( order\_id , date, user\_id , food\_id , quantity)
* Food Items ( Food\_id, name,price,description)
* Payment ( Order\_id , Payment\_id , amount)

1NF

In User table, one user can have multiple phone numbers, But we will only allow one phone number per user which will be atomic.

In all other tables, 1NF is achived.

2NF

3NF:

As we can see in Orders table, attribute quantity relies on the food id then the orders primary key which is order id   
To achive 2NF, we will construct another table called deliveries:

OrderDetails( Order\_id,food\_id,quantity)

And orders table will be: (order\_id,user\_id,date)

Every other table is 3NF.

BCNF

BCNF is achived.

Final entities:

1. User ( user\_id , email, password, firstname, lastname, address, phone number)
2. Orders ( order\_id , date , userid )
3. Food Items ( Food\_id, name,price,description)
4. Category(category\_id , category\_name, food\_id)
5. OrderDetails(slip\_id , order\_id,food\_id,quantity)
6. Payment ( Order\_id , Payment\_id , amount)
7. Delivery(deliveryid, order\_id,user\_id)

Links

[Employee-management-System Tutorial using Spring-boot, react.js and mysql | Full-Stack - YouTube](https://www.youtube.com/watch?v=jJJ07Tni81A)