

CS 631 Section - 007

Group - 6

Saravana Prabhu Ramasamy (Email: sr2484@njit.edu)

Koushik Chandrasekaran (Email: kc664@njit.edu)

**Material Management System
Final Report**

Retail Warehouse Product Inventory Management

Instructor

Canan Eren

Senior University Lecturer, Computer Science



Business Description and Requirements

- The warehouse is the main entity in this database design. The warehouse is responsible for storage and maintenance of products that are either manufactured or by companies or sold by some third party vendors. The warehouse will have an inventory management e-system that monitors its products, purchases, current employees, the associated product suppliers and distributors (for shipping via transportation).
- The warehouse has limited storage and processing power. Each item shipped to the warehouse is stored and recorded into the database until the capacity is reached.
- Although the company has many warehouses, this system will focus on a particular warehouse that is uniquely identified by its warehouse number (or ID). Further its location and contact number should also be recorded. To improve efficiency, the warehouse shall be compartmentalized into different sections and sub-sections based on the type of products, eg, Electronics, Home furnitures, Personal Care, Groceries (Frozen, Dairy, Deli, Fresh produce, etc.), Clothing and Accessories, etc.
- Every product has a unique barcode or an ID, its location within the warehouse (which is determined by the type of product), listed price, its quantity within the warehouse (increase when brought to the warehouse from supplier and decrease when purchased by customers).
- The associated order should contain the Order ID, the product purchased (along with its ID), quantity purchased (which will be deducted in the product table), shipping address and the distributor.
- The warehouse is run by its employees. The employees are further categorized by their designation - Manager, supervisor and worker.
- Each warehouse has one manager under whom there are many supervisors, and each supervisor has a set of workers that are responsible for the daily maintenance of the warehouse.
- The warehouse should also accommodate returns and cancellations that will be notified to the distributor and the product will return to the warehouse, and the quantity shall be restored.

Database Rules/Requirements

Warehouse

1. A warehouse has a capacity to hold different kinds of materials.
2. A warehouse is segregated into aisles.
3. Each aisle has a storage capacity and type of material stored.
4. A warehouse is identified by its ID, and has location, contact number and an assigned manager.

Warehouse_Sections

1. Section is identified by the section_name.
2. Each section has different sub sections.

Product

1. A product is any item stored in the warehouse.
2. Each product would occupy some capacity in the warehouse.
3. Each product is provided by its manufacturer or supplier.
4. Each product is classified into a type of product/material.
5. Every product has a Product_ID, Product_Type, its location within the warehouse, quantity purchased, and list_price.

Product_Type

1. A product_type has a unique ID.
2. A product_type is the higher level classification of the product.
3. Every product_type has a Name, Warehouse_Section where it is stored.

Supplier

1. A supplier provides products in bulk to the warehouse.
2. A supplier has a unique ID, Name, location from which the product arrives.

Order

1. The order may be an incoming order(Brought into the warehouse) or outgoing order(Purchased from the warehouse).

2. Every order contains a unique Order_ID, the product purchased (identified by Product_ID), quantity purchased, Status of the order, Distributor_name and the Shipping_Address.

Distributor

1. The distributor ships the products associated with an order.
2. The distributor has information about the warehouse through its location, the orders placed and the shipping address of the respective parties.
3. The distributor also has information of the Order_ID that users can use for tracking.
4. Each distributor has ID, Name, Location, Contact, Distributing_Locations.

Employee

1. An employee works in the warehouse.
2. An employee has a unique ID, name(First, Last), Address, Age, Contact, DOB, Salary, Supervisor.

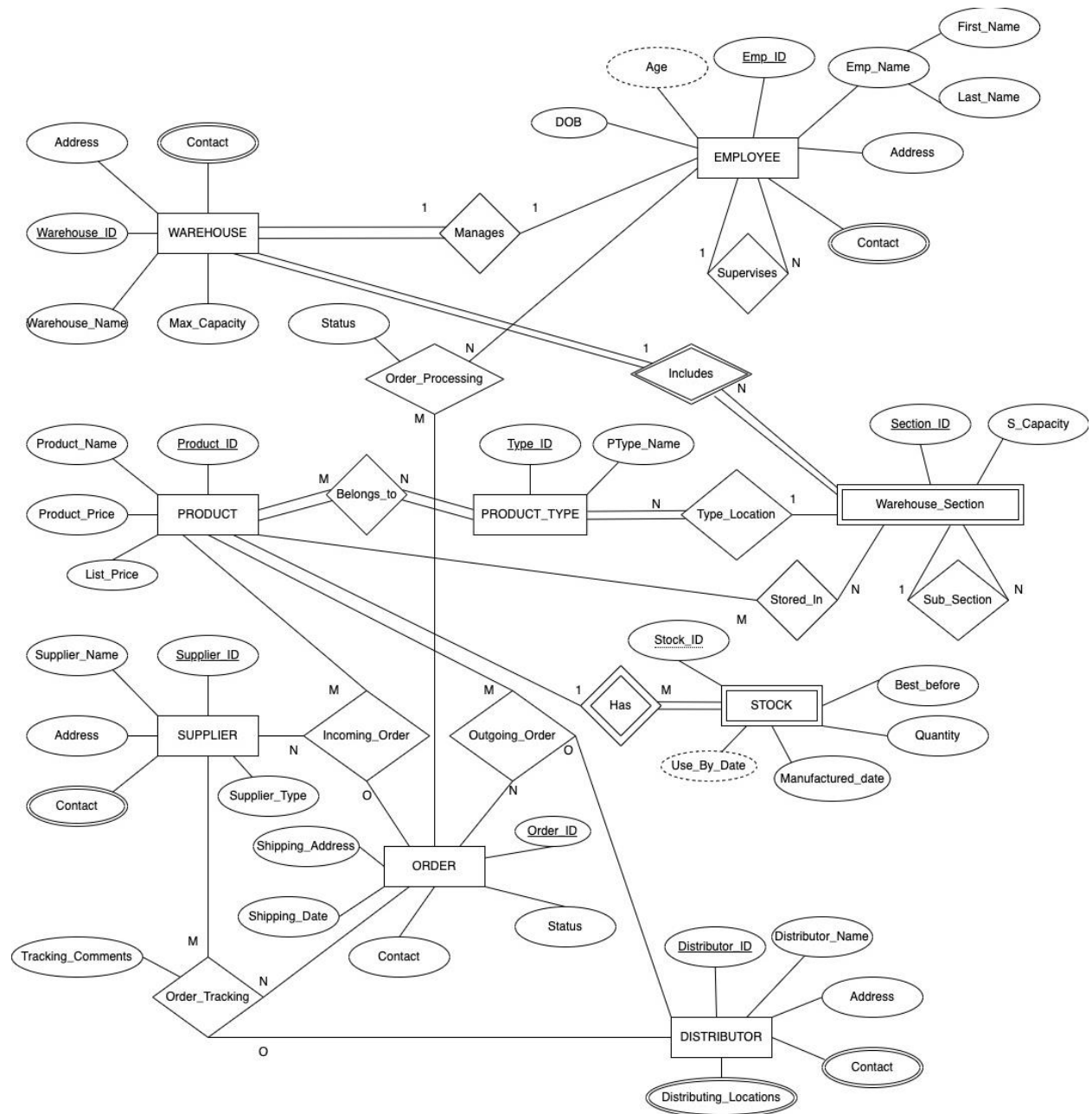
Unspecified Requirements/Assumptions

1. Sub_Section is a self-referencing relation for the entity Warehouse_Sections to pin-point the exact location of a product.
2. A product may belong to different types, for example, a gaming chair is under both gaming section, and furniture.
3. Stored_In a relation between Product and Warehouse_Sections entities, that keeps track of the section(s) that a product belongs to.
4. Supervises is another self-referencing relation between employees, based on their designation. For example, in this scenario, a manager manages some supervisors and each supervisor manages multiple workers.
5. Supervisor keeps track of the orders who is also responsible for incoming_orders, and status is an attribute defined on the Order_Processing relation.
6. Stock is a weak entity - Stock has an ID, Whenever a new stock of a product comes, Its price, quantity, manufacture date and Use by date is noted.
7. Tracking comments are also added to keep continual track of the order.

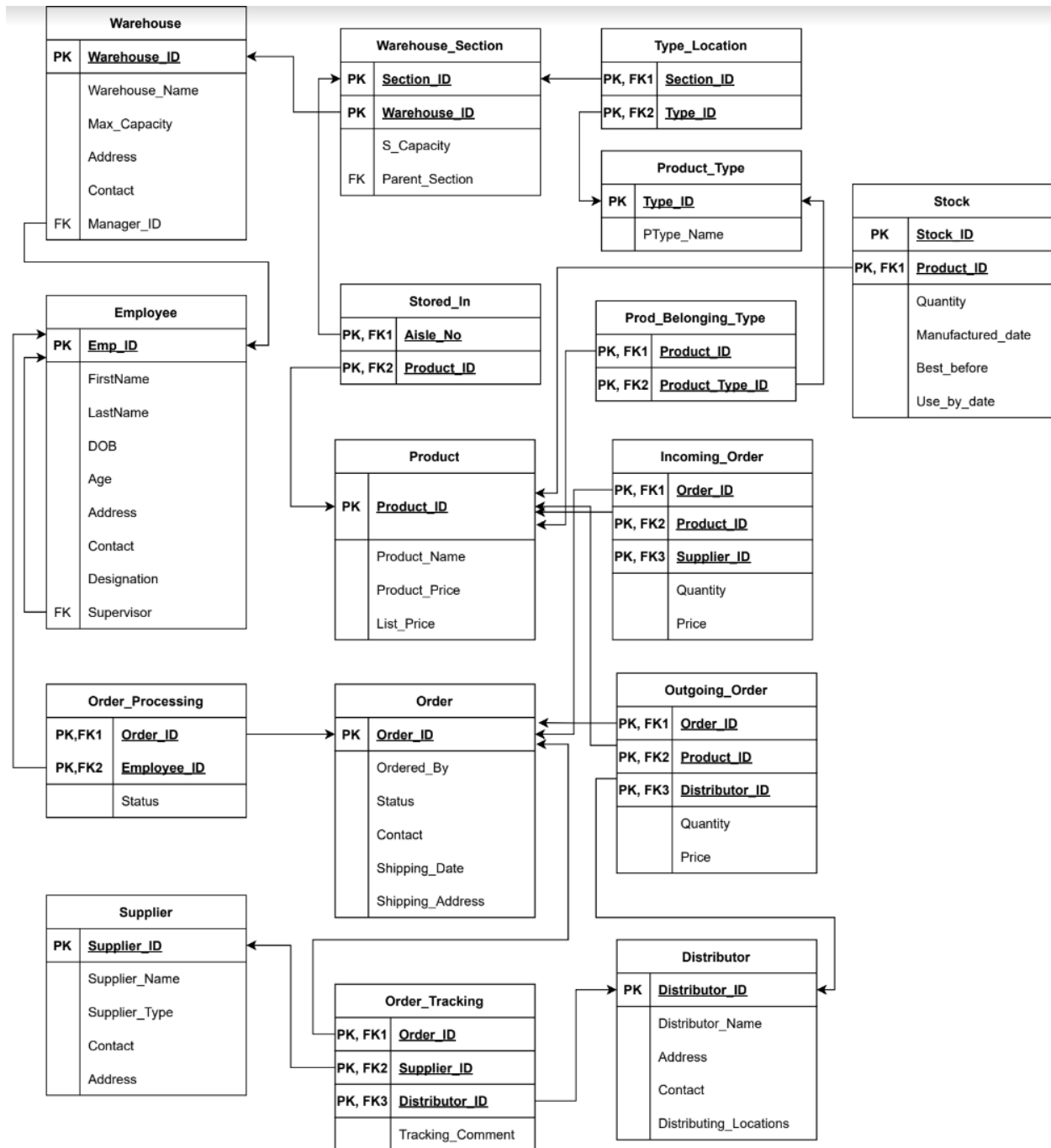
Relations Identified

1. Manages - Between Employee and Warehouse
2. Supervises - Between Employee and Employee (self)
3. Includes - Between Warehouse and Warehouse_Section
4. Belongs_to - Between Product and Product_Type
5. Stored_In - Between Product and Warehouse
6. Type_Location - Between Product_Type and Warehouse_Section
7. Sub_Section - Between Warehouse_Section and Warehouse_Section (self)
8. Has - Between Product and Stock
9. Order_Processing - Between Employee and Order
10. Incoming_Order - Between Order, Product & Supplier
11. Outgoing_Order - Between Order, Product & Distributor
12. Order_Tracking - Between Order, Supplier & Distributor

ER Diagram



Relational Schema

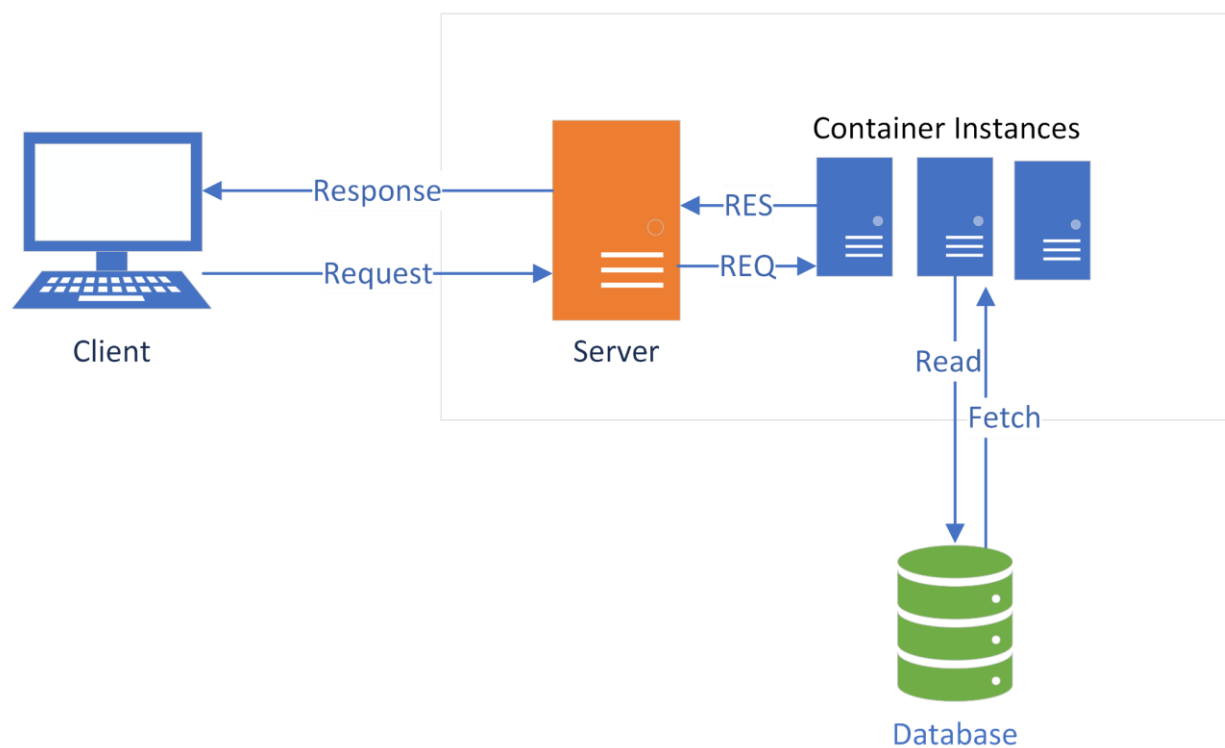


Application Program Design

We have developed the application using django framework for backend and various web technologies for the frontend namely HTML, CSS, Javascript and Bootstrap framework.

As it is a group project and the OS didn't match with each of us, we decided to develop a project in a container instance. So irrespective of the OS that the machine runs, the application can be developed and deployed.

Below is the architecture of the application.



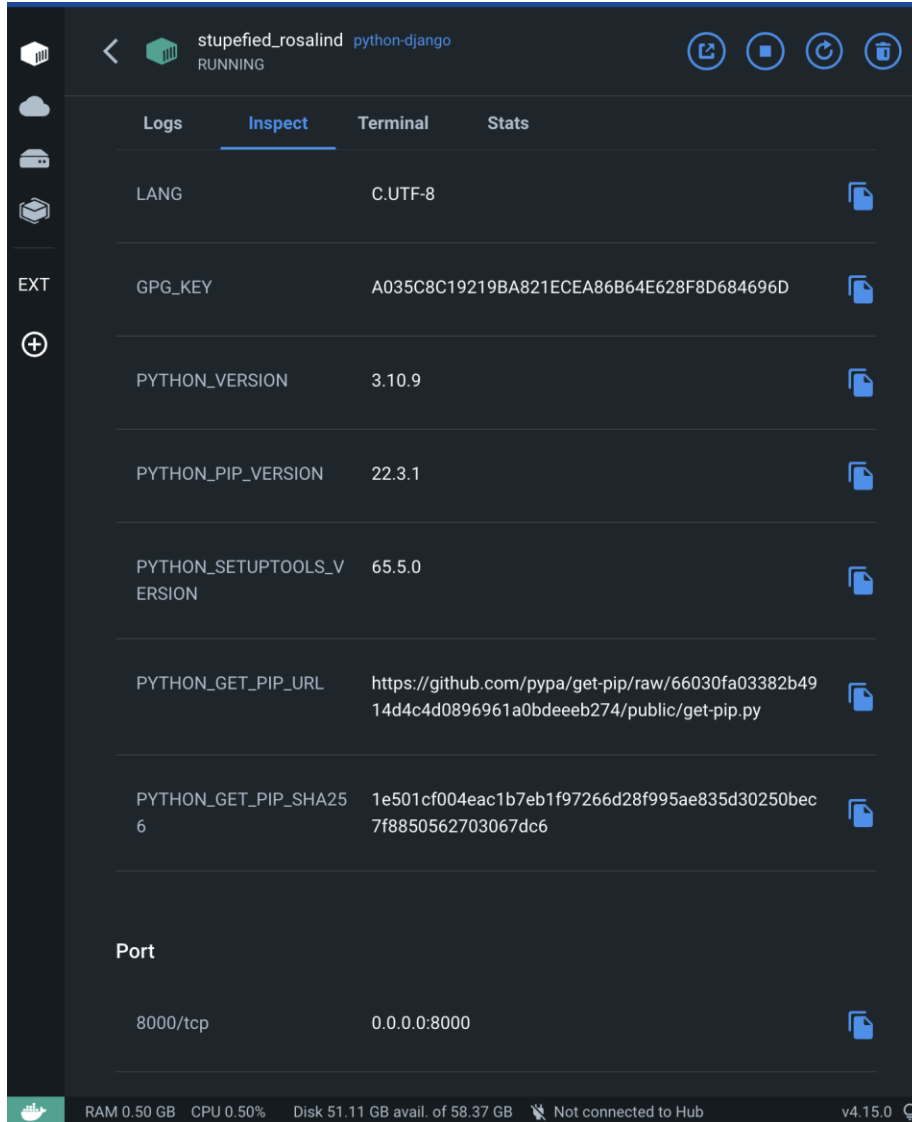
We have created a docker image with a linux environment. Installed all the required libraries and packages to run the application. When the image is deployed as a container, the application is available in the port 8000.

The advantage of this approach:

- As pointed out earlier, it facilitates cross platform development and ease in a team project without worrying about the environment.
- The deployment can be done from anywhere with any machine with docker and database connection.

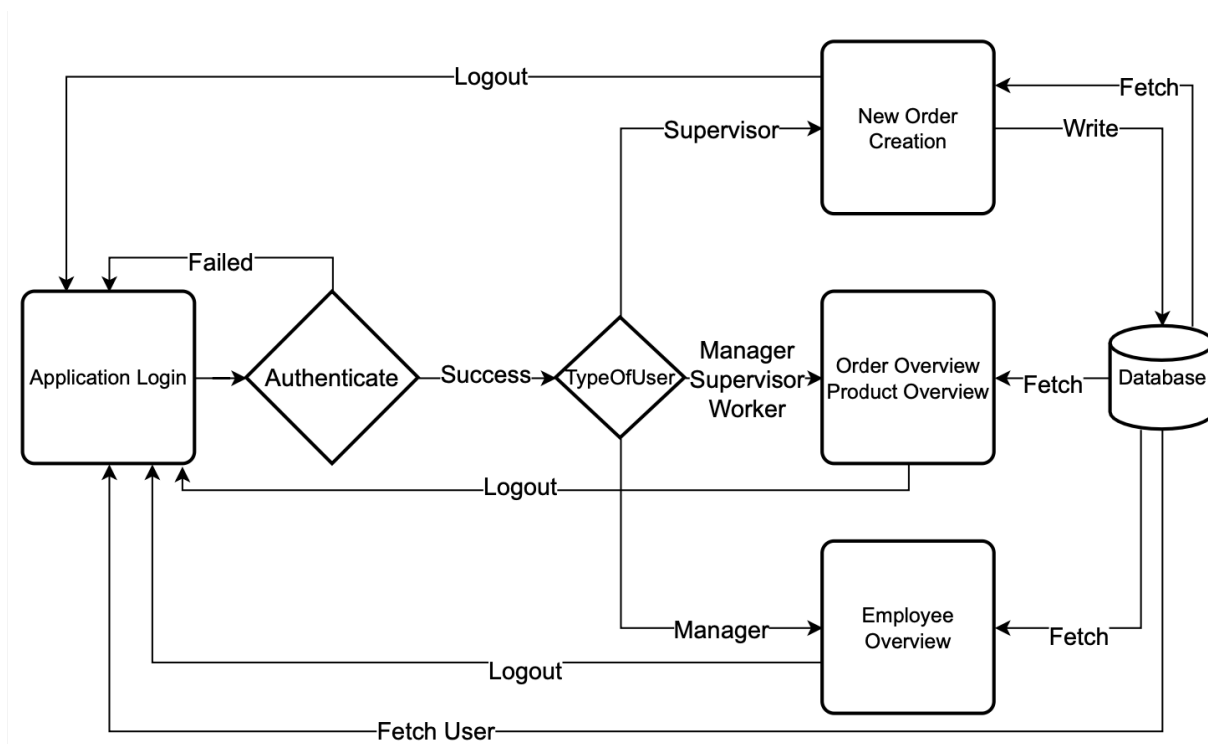
- No requirement of source code to deploy and test the application. Just the container image is sufficient which makes it readily available.

A screenshot of the container instance:

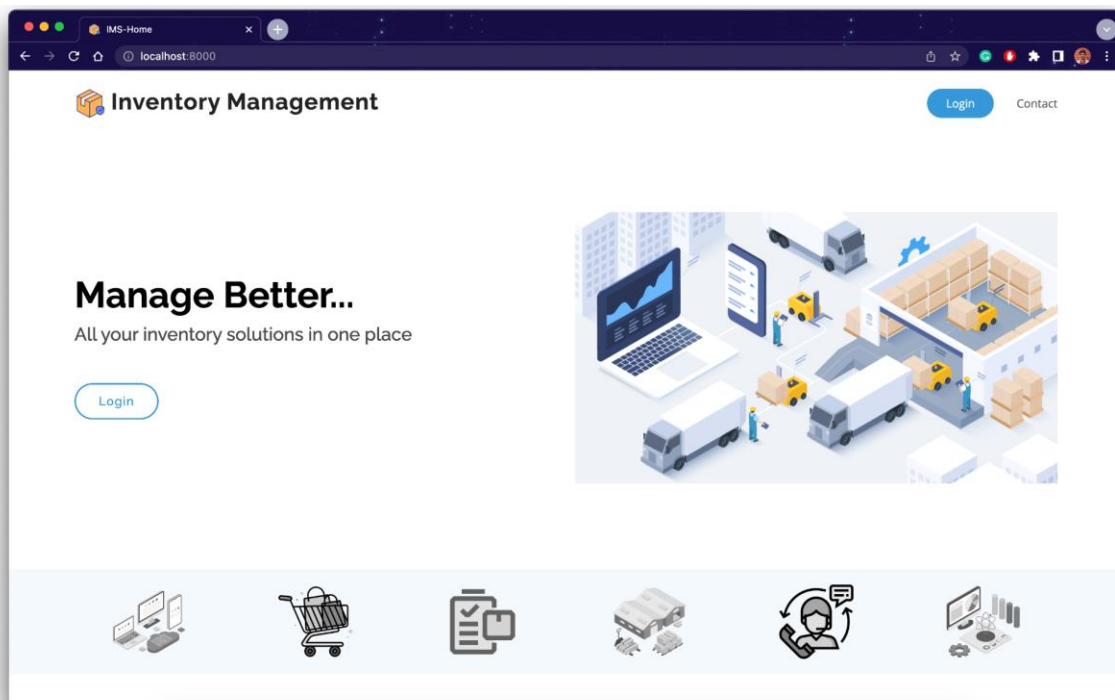


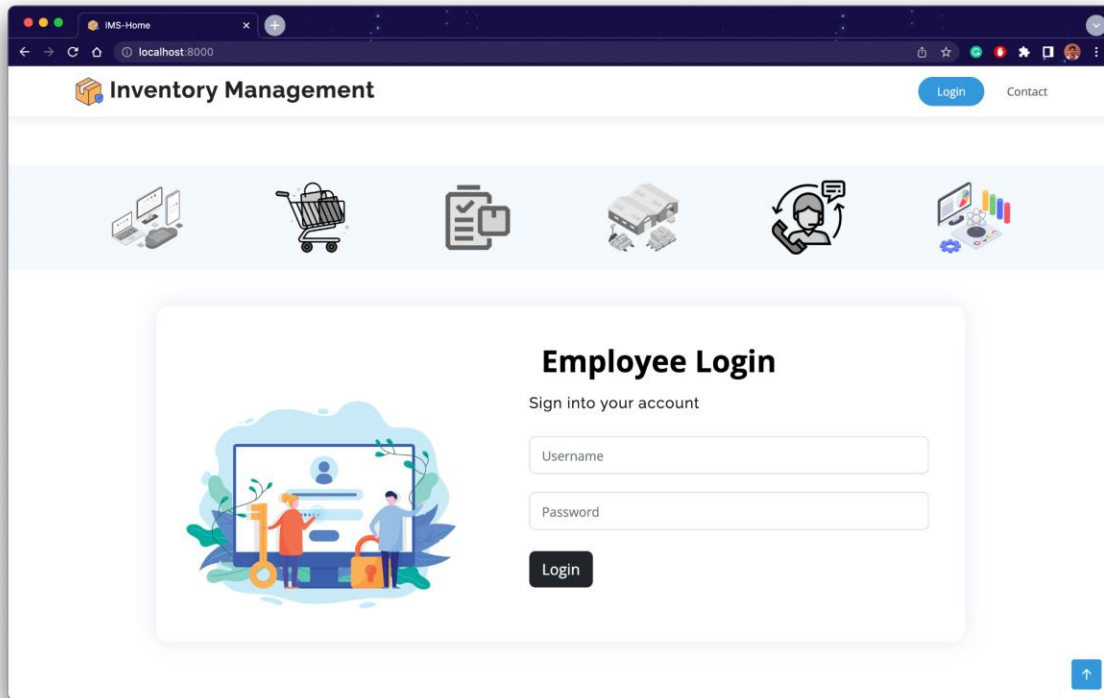
It was challenging to understand the concepts of containers and its development. But the pros surpasses the cons as it facilitated rapid team development and deployment.

Flow diagram of the application:

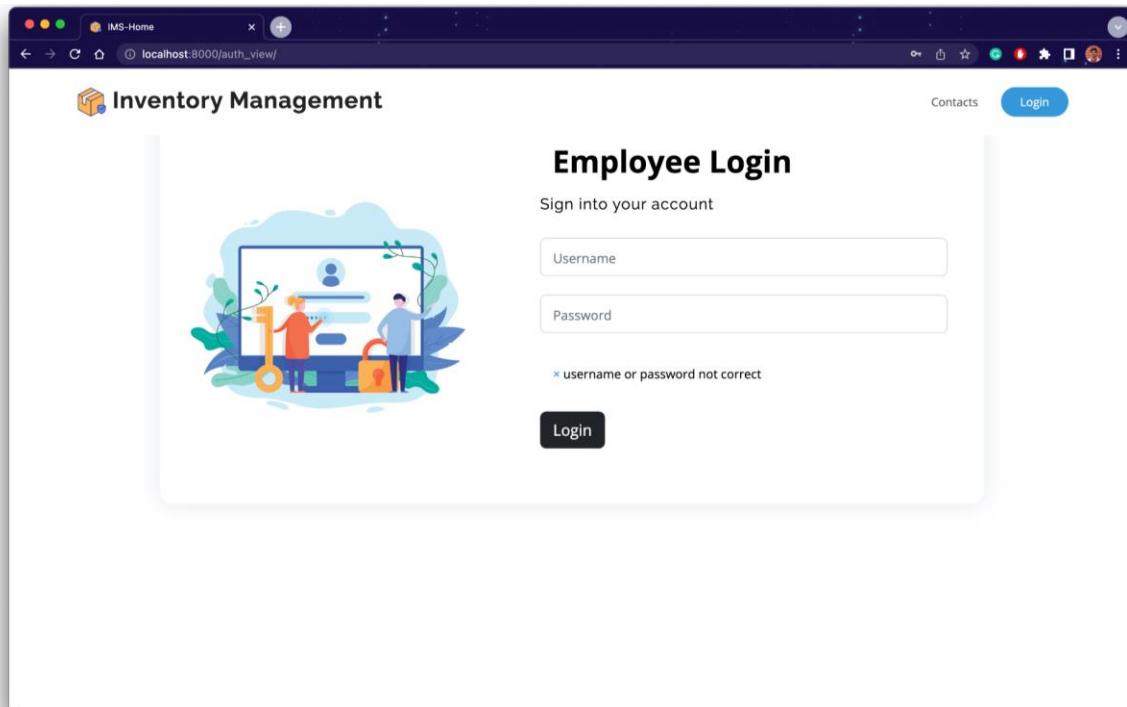


Login Page





Invalid Login



Order Overview Page

Inventory Management | Products | Employees | Contacts | Saravana Ramasamy | Logout

Orders Overview

Orders

Order ID	By	Status	Shipping City	Shipping State	Assigned To	Comments
1	Sam	Created	Harrison	NJ	None	None
2	Aron	Processing	Harrison	NJ	Kate	Order confirmed, waiting for packaging
3	Jordan	Created	Harrison	NJ	None	None
4	Kumar	Processing	Harrison	NJ	Frank	Order confirmed, waiting for packaging
5	Linda	Packing	Harrison	NJ	Frank	Order packaged, waiting for shipping
6	Aron	Processing	Harrison	NJ	Madison	Order confirmed, waiting for packaging
7	Bryce	Processing	Harrison	NJ	Madison	Order confirmed, waiting for packaging
8	Scarlett	Processing	Harrison	NJ	Frank	Order confirmed,

Incoming Orders

ID	Product	Quantity	Price	Supplier
9	Retractable Ballpoint Pens	15	37.50\$	Inkjoy
9	Premium 32 A4 Sheets 32 lb 500 sheets	10	150.00\$	HP
9	Wet/Dry Vacuum	5	350.00\$	Craftsman
9	Cordless Nailer 20V	15	2550.00\$	Dewalt
9	LED Strip Lights	20	519.80\$	Tenmiro
16	Waterproof Mattress Pad	2	200.00\$	Dewalt

Outgoing Orders

ID	Product	Quantity	Price	Distributor
1	Cordless Nailer 20V	1	249.99\$	Green Line Move Corp.
2	128 fl oz soybean vegetable oil	2	20.96\$	Kristal Cargo LP
3	Wet/Dry Vacuum	1	139.99\$	Mail Boxes Services

Product Overview Page

Inventory Management | Orders | Employees | Contacts | Saravana Ramasamy | Logout

Products Overview

Products with Stock Info

ID	Name	Manufacturer	Price	Sale Price	Quantity	Mfd date
10000	Retractable Ballpoint Pens	Inkjoy	2.50	4.44	30	Nov. 7, 2022, midnight
10001	Premium 32 A4 Sheets 32 lb 500 sheets	HP	15.00	20.99	15	Oct. 12, 2022, midnight
10002	Wet/Dry Vacuum	Craftsman	70.00	139.99	10	Nov. 5, 2022, midnight
10003	Cordless Nailer 20V	Dewalt	170.00	249.99	20	Sept. 23, 2022, midnight
10004	LED Strip Lights	Tenmiro	12.00	25.99	30	Nov. 5, 2022, midnight
10005	6ft Floor Lamp	Mainstays	8.00	14.98	None	None
10006	Waterproof Mattress Pad	Meritlife	22.22	49.99	None	None
10009	Liquid Hand Soap, Fresh Breeze 7.5 fl oz	Softsoap	0.75	1.21	None	None
10010	Pump Deep Body Wash and Moisturizer 34 fl oz	Dove	4.50	8.98	None	None
10007	128 fl oz soybean vegetable oil	Wesson	6.00	10.48	None	None
10008	Chicken Breasts 1.5lb	Perdue	2.50	5.21	None	None

Employee Overview Page

The screenshot shows the 'Employees Overview' page of the 'Inventory Management' application. The browser address bar indicates the URL is `localhost:8000/employees/`. The page header includes the application logo, navigation links for 'Orders', 'Employees', and 'Contacts', the current user 'Saravana Ramasamy', and a 'Logout' button. The main content area is titled 'Employees Overview' and features a table with employee data.

ID	First Name	Last Name	Designation	Supervisor ID	Contact	City	State
1000	Saravana	Ramasamy	Manager	None	7892573154	Harrison	NJ
1001	Koushik	Chandrasekaran	Manager	None	8622303524	Harrison	NJ
1002	Ken	Adams	Manager	None	5712573152	Morris Plains	NJ
1003	Sydney	Smith	Manager	None	9912573154	Jersey City	NJ
1004	Haley	Dunphy	Manager	None	2512573157	Orange	NJ
1005	Kate	Upton	Supervisor	1000	8712573134	Harrison	NJ
1006	Frank	Lewis	Worker	1005	7892573612	Harrison	NJ
1007	Madison	Keath	Worker	1005	8623543184	Harrison	NJ

New Order Creation Page

The screenshot shows the 'New Order' page of the 'Inventory Management' application. The browser address bar indicates the URL is `localhost:8000/new_order/`. The page header includes the application logo, navigation links for 'New Order', 'Orders', 'Products', and 'Contacts', the current user 'Kate Upton', and a 'Logout' button. The main content area is titled 'New Order' and contains a form for creating a new order.

Provide Order Details

Order Type: Incoming Order | Product: LED Strip Lights | Quantity: | Price \$: |

Supplier: Dewalt | Distributor: The UPS Store

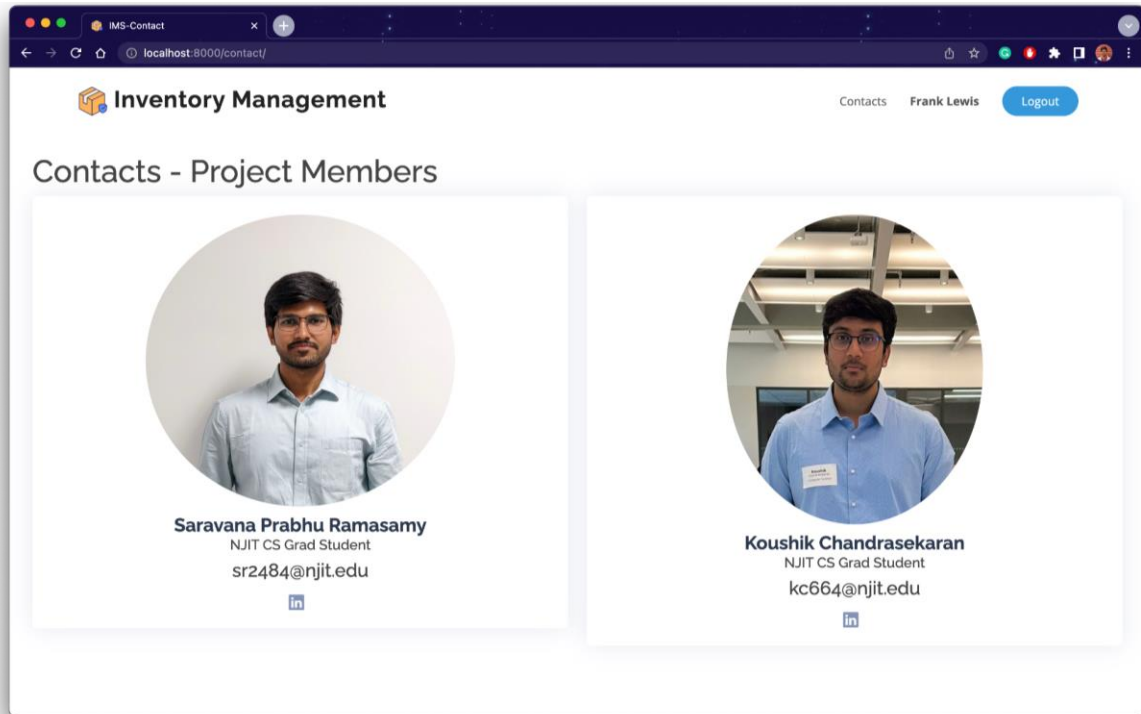
Name: Kate | Mobile: 8712573134

Address: 6th Cross Street

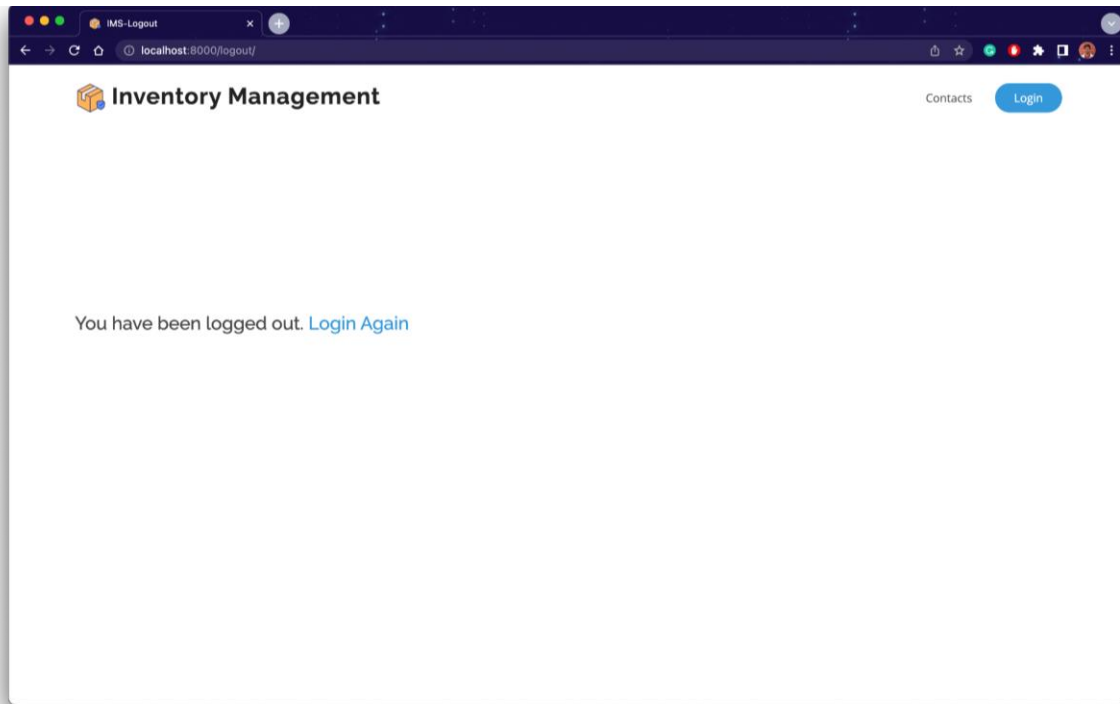
City: Harrison | State: NJ | Zip: 07029

[Submit](#) [Reset](#)

Contact Page - Project Members Overview



Logout Page



Normalization

Warehouse

Functional Dependencies:

{Warehouse_ID} → {Warehouse_Name, Max_capacity, Address, city, state, zipcode, country, Contact, Manager_ID}

Script Output x Query Result x

All Rows Fetched: 5 in 0.011 seconds

	WAREHOUSE_ID	WAREHOUSE_NAME	MANAGER_ID	MAX_CAPACITY	CONTACT	ADDRESS	CITY	SSTATE	ZIPCODE	COUNTRY
1	100	NJIT_Central	(null)	20000	5513243901	NJIT Central Building	Newark	NJ	07102	USA
2	200	NJIT_Pantry	(null)	10000	5513243902	NJIT Campus Center	Newark	NJ	07102	USA
3	300	NJIT_TechCenter	(null)	1000	5513243903	University Heights	Newark	NJ	07102	USA
4	400	NJIT_Merch	(null)	5000	5513243904	NJIT Mall	Newark	NJ	07102	USA
5	500	NJIT_Sports	(null)	3000	5513243905	NJIT Fitness Center	Newark	NJ	07102	USA

The relation is in 3NF

Warehouse	
PK	<u>Warehouse_ID</u>
	Warehouse_Name
	Max_Capacity
	Address
	Contact
FK	Manager_ID

Warehouse_Section

Functional Dependencies:

{Warehouse_ID, Section_ID} → {S_Capacity, Parent_Section}

Script Output x Query Result x

All Rows Fetched: 27 in 0.01 seconds

	WAREHOUSE_ID	SECTION_ID	S_CAPACITY	PARENT_SECTION
1	100	A1	2000	(null)
2	100	A2	1000	(null)
16	500	E3	1500	(null)
17	100	A1-1	200	A1
18	100	A1-2	100	A1

The relation is in 3NF

Warehouse_Section	
PK	<u>Section_ID</u>
PK, FK1	<u>Warehouse_ID</u>
	S_Capacity
FK	Parent_Section

Employee

Functional Dependencies:

{Emp_ID} → {FirstName, LastName, DOB, Age, Address, city, state, zipcode, country, Contact, Designation, Supervisor}

SSN is not used as it will be a problem when the solution is implemented globally.

Script Output x Query Result x

All Rows Fetched: 8 in 0.007 seconds

	EMP_ID	FIRSTNAME	LASTNAME	DESIGNATION	SUPERVISOR_ID	DOB	CONTACT	ADDRESS	CITY	SSTATE	ZIPCODE	COUNTRY	EMP_OF
1	1000	Saravana	Ramasamy	Manager	(null)	09-07-90	7892573154	Frank E Rodgers	Harrison	NJ	7029	USA	100
2	1001	Roushik	Chandrasekaran	Manager	(null)	28-05-92	8622303524	7th Street	Harrison	NJ	7029	USA	200
3	1002	Ken	Adams	Manager	(null)	19-03-91	5712573152	Stockton Ct	Morris Plains	NJ	7950	USA	300
4	1003	Sydney	Smith	Manager	(null)	01-01-93	9912573154	Central Ave	Jersey City	NJ	7302	USA	400
5	1004	Haley	Dunphy	Manager	(null)	17-04-89	2512573157	Cleveland Ave	Orange	NJ	7050	USA	500
6	1005	Kate	Upton	Supervisor	1000	25-02-94	8712573134	6th Cross Street	Harrison	NJ	7029	USA	100
7	1006	Frank	Lewis	Worker	1005	22-08-92	7892573612	Warren Street	Harrison	NJ	7029	USA	100
8	1007	Madison	Keath	Worker	1005	12-09-91	8623543184	Bergen Street	Harrison	NJ	7029	USA	100

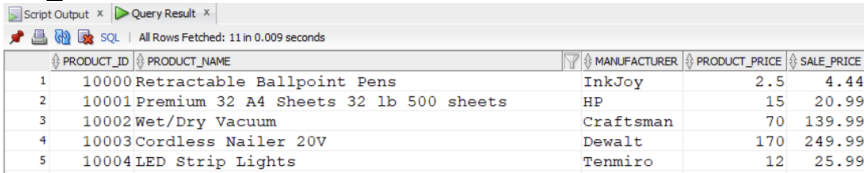
The relation is in 3NF

Employee	
PK	<u>Emp_ID</u>
	FirstName
	LastName
	DOB
	Age
	Address
	Contact
	Designation
FK	Supervisor

Product

Functional Dependencies:

{Product_ID} → {Product_Name, manufacturer, Product_Price, List_Price}



Script Output x Query Result x

All Rows Fetched: 11 in 0.009 seconds

PRODUCT_ID	PRODUCT_NAME	MANUFACTURER	PRODUCT_PRICE	SALE_PRICE
1	10000 Retractable Ballpoint Pens	InkJoy	2.5	4.44
2	10001 Premium 32 A4 Sheets 32 lb 500 sheets	HP	15	20.99
3	10002 Wet/Dry Vacuum	Craftsman	70	139.99
4	10003 Cordless Nailer 20V	Dewalt	170	249.99
5	10004 LED Strip Lights	Tenmiro	12	25.99

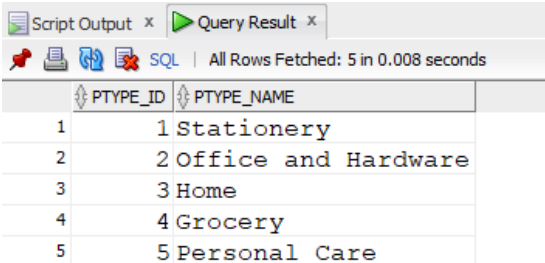
The relation is in 3NF

Product	
PK	<u>Product_ID</u>
	Product_Name
	Product_Price
	List_Price

Product_Type

Functional Dependencies:

{PType_ID} → {PType_Name}



Script Output x Query Result x

All Rows Fetched: 5 in 0.008 seconds

PType_ID	PType_Name
1	1 Stationery
2	2 Office and Hardware
3	3 Home
4	4 Grocery
5	5 Personal Care

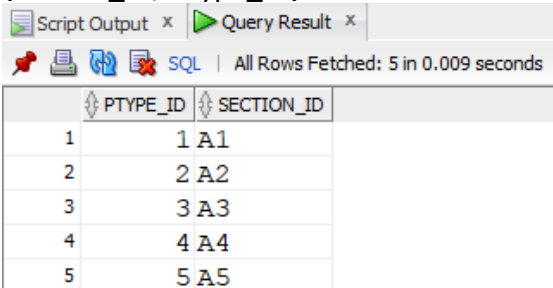
The relation is in 3NF

Product_Type	
PK	<u>Type_ID</u>
	PType_Name

Type_Location

Functional Dependencies:

{Section_ID, PType_ID}



Script Output x Query Result x

All Rows Fetched: 5 in 0.009 seconds

PType_ID	SECTION_ID
1	1 A1
2	2 A2
3	3 A3
4	4 A4
5	5 A5

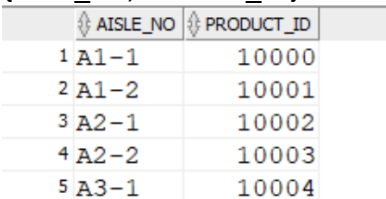
The relation is in 3NF

Type_Location	
PK, FK1	<u>Section_ID</u>
PK, FK2	<u>Type_ID</u>

Stored_IN

Functional Dependencies:

{Aisle_No, Product_ID}



AISLE_NO	PRODUCT_ID
1 A1-1	10000
2 A1-2	10001
3 A2-1	10002
4 A2-2	10003
5 A3-1	10004

The relation is in 3NF

Stored_In	
PK, FK1	<u>Aisle_No</u>
PK, FK2	<u>Product_ID</u>

Prod_Belong_Type

Functional Dependencies:

{Product_ID, PType_ID}

	PRODUCT_ID	PTYPE_ID
1	10000	1
2	10001	1
3	10002	2
4	10003	2
5	10004	3

The relation is in 3NF

Prod_Belonging_Type	
PK, FK1	<u>Product_ID</u>
PK, FK2	<u>Product_Type_ID</u>

Stock

Functional Dependencies:

{Product_ID, Stock_ID} → {Quantity,
Manufactured_date,Best_before, Use_by_date}

Script Output x Query Result x

All Rows Fetched: 5 in 0.009 seconds

	STOCK_ID	PRODUCT_ID	QUANTITY	MFD_DATE	BEST_BEFORE	USE_BY_DATE
1	1	10000	30	07-11-22	(null)	(null)
2	2	10001	15	12-10-22	(null)	(null)
3	3	10002	10	05-11-22	(null)	(null)
4	4	10003	20	23-09-22	(null)	(null)
5	5	10004	30	05-11-22	(null)	(null)

The relation is in 3NF

Stock	
PK	<u>Stock_ID</u>
PK, FK1	<u>Product_ID</u>
	Quantity
	Manufactured_date
	Best_before
	Use_by_date

Orders

Functional Dependencies:

{Order_ID} → {Ordered_by, status,contact,Shp_date, shp_Address,
shp_city, shp_state, shp_zipcode, country}

Script Output x Query Result x

All Rows Fetched: 9 in 0.009 seconds

	ORDER_ID	ORDER...	STATUS	CONTACT	SHIP_DATE	SHIP_ADDRESS	SHIP_CITY	SHIP_STATE	SHIP_ZIPCODE	COUNTRY
1	1	Sam	Created	5524378919	20-11-22	Cleveland Ave	Harrison NJ	7029	USA	
2	2	Aron	Processing	8624377822	18-11-22	Central Ave	Harrison NJ	7029	USA	
3	3	Jordan	Created	8622458634	21-11-22	7th Street	Harrison NJ	7029	USA	
4	4	Kumar	Processing	7894372356	19-11-22	Bergen Street	Harrison NJ	7029	USA	
5	5	Linda	Packing	8622373158	17-11-22	Harrison Ave	Harrison NJ	7029	USA	
6	6	Aron	Processing	8624377822	18-11-22	Central Ave	Harrison NJ	7029	USA	
7	7	Bryce	Processing	8623126748	18-11-22	Church Square	Harrison NJ	7029	USA	
8	8	Scarlett	Processing	8629893646	17-11-22	101 Supor Blvd	Harrison NJ	7029	USA	
9	9	Frank	Processing	7892573612	19-11-22	NJIT Central Building	Newark NJ	7102	USA	

The relation is in 3NF

Order	
PK	<u>Order_ID</u>
	Ordered_By
	Status
	Contact
	Shipping_Date
	Shipping_Address

Order_Processing

Functional Dependencies:

{Order_ID, Employee_ID} → {Status}

Script Output x Query Result x

All Rows Fetched: 6 in 0.009 seconds

	ORDER_ID	EMP_ID	STATUS
1	2	1005	Order confirmed, waiting for packaging
2	4	1006	Order confirmed, waiting for packaging
3	5	1006	Order packaged, waiting for shipping
4	6	1007	Order confirmed, waiting for packaging
5	7	1007	Order confirmed, waiting for packaging
6	8	1006	Order confirmed, waiting for packaging

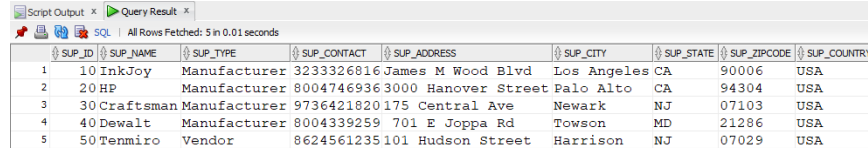
The relation is in 3NF

Order_Processing	
PK,FK1	<u>Order_ID</u>
PK,FK2	<u>Employee_ID</u>
	Status

Supplier

Functional Dependencies:

{Sup_ID} → {Sup_Name, Sup_type, Sup_Contact, Sup_city, Sup_state, sup_zipcode, sup_country }



	SUP_ID	SUP_NAME	SUP_TYPE	SUP_CONTACT	SUP_ADDRESS	SUP_CITY	SUP_STATE	SUP_ZIPCODE	SUP_COUNTRY
1	10	InkJoy	Manufacturer	3233326816	James M Wood Blvd	Los Angeles	CA	90006	USA
2	20	HP	Manufacturer	8004746936	3000 Hanover Street	Palo Alto	CA	94304	USA
3	30	Craftsman	Manufacturer	9736421820	175 Central Ave	Newark	NJ	07103	USA
4	40	Dewalt	Manufacturer	8004339259	701 E Joppa Rd	Towson	MD	21286	USA
5	50	Tenmiro	Vendor	8624561235	101 Hudson Street	Harrison	NJ	07029	USA

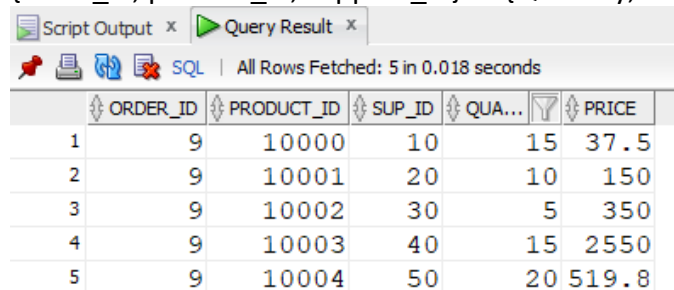
The relation is in 3NF

Supplier	
PK	<u>Supplier_ID</u>
	Supplier_Name
	Supplier_Type
	Contact
	Address

Incoming_Order

Functional Dependencies:

{order_id, product_id, supplier_id} → {Quantity, Price}



	ORDER_ID	PRODUCT_ID	SUP_ID	QUANTITY	PRICE
1	9	10000	10	15	37.5
2	9	10001	20	10	150
3	9	10002	30	5	350
4	9	10003	40	15	2550
5	9	10004	50	20	519.8

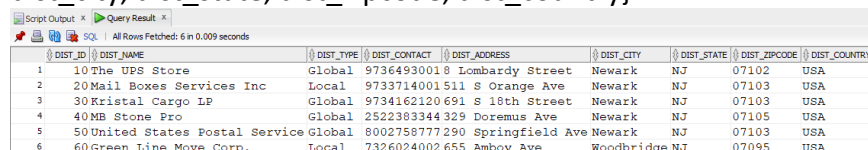
The relation is in 3NF

Incoming_Order	
PK, FK1	<u>Order_ID</u>
PK, FK2	<u>Product_ID</u>
PK, FK3	<u>Supplier_ID</u>
	Quantity
	Price

Distributor

Functional Dependencies:

{Dist_ID} → {dist_name, dist_type, dist_contact, dist_address, dist_city, dist_state, dist_zipcode, dist_country}



	DIST_ID	DIST_NAME	DIST_TYPE	DIST_CONTACT	DIST_ADDRESS	DIST_CITY	DIST_STATE	DIST_ZIPCODE	DIST_COUNTRY
1	10	The UPS Store	Global	97364930018	Lombardy Street	Newark	NJ	07102	USA
2	20	Mail Boxes Services Inc	Local	9733714001511	S Orange Ave	Newark	NJ	07103	USA
3	30	Kristal Cargo LP	Global	9734162120691	S 18th Street	Newark	NJ	07103	USA
4	40	MB Stone Pro	Global	2522383344329	Doremus Ave	Newark	NJ	07105	USA
5	50	United States Postal Service	Global	8002758777290	Springfield Ave	Newark	NJ	07103	USA
6	60	Green Line Move Corp.	Local	7326024002655	Amboy Ave	Woodbridge	NJ	07095	USA

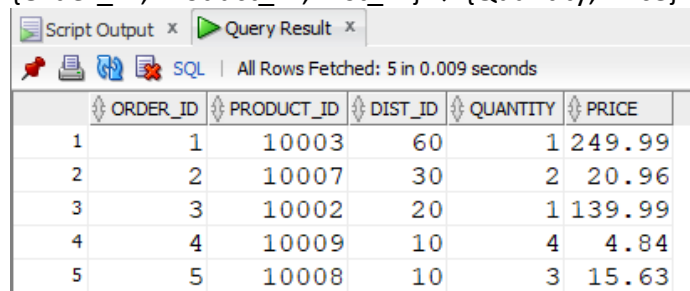
The relation is in 3NF

Distributor	
PK	<u>Distributor_ID</u>
	Distributor_Name
	Address
	Contact
	Distributing_Locations

Outgoing_Order

Functional Dependencies:

{Order_ID, Product_ID, Dist_ID} → {Quantity, Price}



	ORDER_ID	PRODUCT_ID	DIST_ID	QUANTITY	PRICE
1	1	10003	60	1	249.99
2	2	10007	30	2	20.96
3	3	10002	20	1	139.99
4	4	10009	10	4	4.84
5	5	10008	10	3	15.63

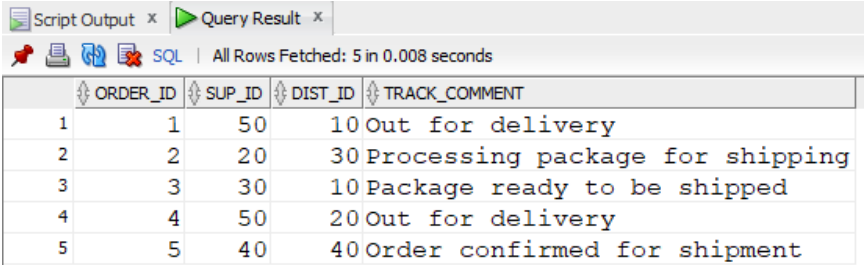
The relation is in 3NF

Outgoing_Order	
PK, FK1	<u>Order_ID</u>
PK, FK2	<u>Product_ID</u>
PK, FK3	<u>Distributor_ID</u>
	Quantity
	Price

Order_Tracking

Functional Dependencies:

{Order_ID, Sup_Id, dist_ID} → {track_comment}



The screenshot shows a database query result window with the title 'Script Output x Query Result x'. Below the title bar, there are icons for saving, printing, and refreshing, followed by the text 'SQL | All Rows Fetched: 5 in 0.008 seconds'. The main area displays a table with the following data:

	ORDER_ID	SUP_ID	DIST_ID	TRACK_COMMENT
1	1	50	10	Out for delivery
2	2	20	30	Processing package for shipping
3	3	30	10	Package ready to be shipped
4	4	50	20	Out for delivery
5	5	40	40	Order confirmed for shipment

The relation is in 3NF

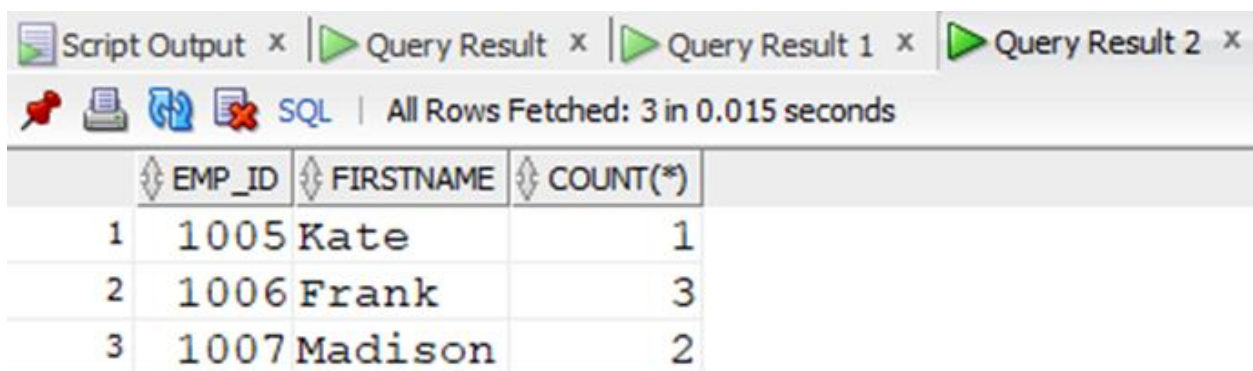
Order_Tracking	
PK, FK1	<u>Order_ID</u>
PK, FK2	<u>Supplier_ID</u>
PK, FK3	<u>Distributor_ID</u>
	Tracking_Comment

SQL Queries

1. Containing GROUP BY

Only display employees with orders assigned to them along with the number of orders

```
SELECT OP.EMP_ID, E.FIRSTNAME, COUNT(*)  
FROM Order_Processing OP, Employee E  
WHERE OP.EMP_ID = E.EMP_ID  
Group by OP.EMP_ID, E.FIRSTNAME;
```



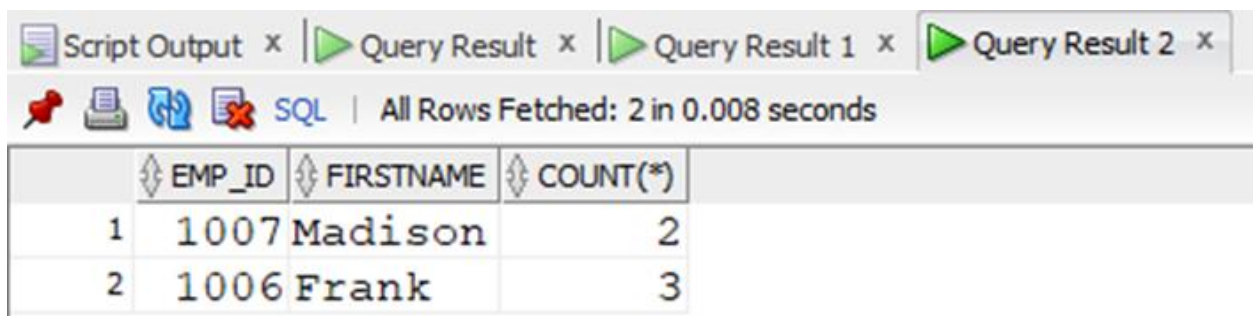
The screenshot shows a SQL query execution interface with tabs for 'Script Output', 'Query Result', 'Query Result 1', and 'Query Result 2'. Below the tabs, it indicates 'All Rows Fetched: 3 in 0.015 seconds'. The query results are displayed in a table with columns EMP_ID, FIRSTNAME, and COUNT(*).

	EMP_ID	FIRSTNAME	COUNT(*)
1	1005	Kate	1
2	1006	Frank	3
3	1007	Madison	2

2. Containing GROUP BY and HAVING

Get the employees with two or more orders assigned to them along with the number in ascending order

```
SELECT OP.EMP_ID, E.FIRSTNAME, COUNT(*)  
FROM Order_Processing OP, Employee E  
WHERE OP.EMP_ID = E.EMP_ID  
GROUP BY OP.EMP_ID, E.FIRSTNAME  
HAVING COUNT(*) >= 2  
ORDER BY COUNT(*) ASC;
```



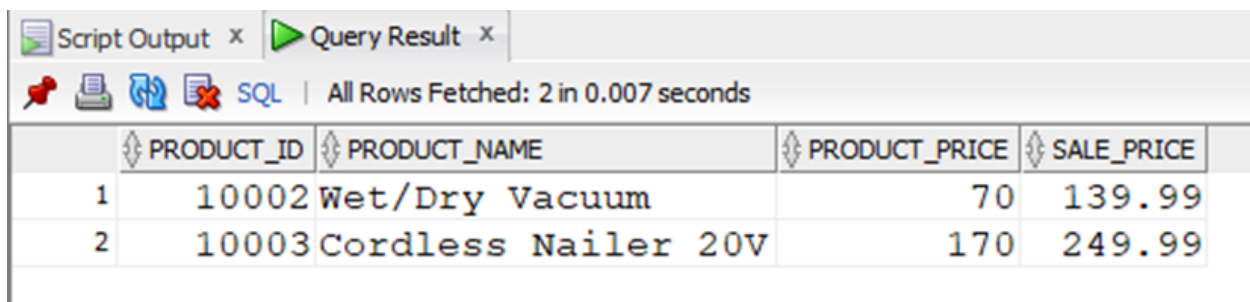
The screenshot shows a SQL query execution interface with tabs for 'Script Output', 'Query Result', 'Query Result 1', and 'Query Result 2'. Below the tabs, it indicates 'All Rows Fetched: 2 in 0.008 seconds'. The query results are displayed in a table with columns EMP_ID, FIRSTNAME, and COUNT(*).

	EMP_ID	FIRSTNAME	COUNT(*)
1	1007	Madison	2
2	1006	Frank	3

3. Containing nested query with ALL

Get all the products with cost greater than all the products in the Home Section

```
SELECT P.Product_Id, P.Product_Name, P.Product_Price,
P.Sale_Price
FROM Product P
WHERE P.Product_Price > ALL (
    SELECT PP.Product_Price
    FROM Product PP, Prod_Belong_Type PBT, Product_Type PT
    WHERE PP.Product_ID = PBT.Product_ID and PBT.PType_ID =
PT.PType_ID
    and PT.PType_Name = 'Home');
```



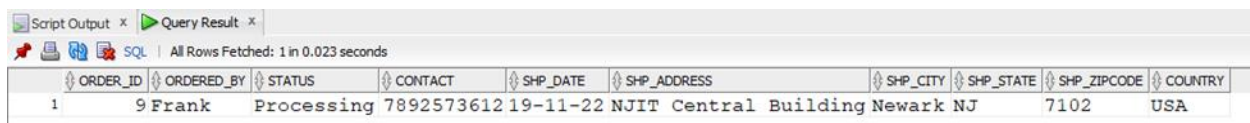
The screenshot shows a SQL query result window with two tabs: 'Script Output' and 'Query Result'. The 'Query Result' tab is active, displaying a table with 5 columns: PRODUCT_ID, PRODUCT_NAME, PRODUCT_PRICE, and SALE_PRICE. The table contains two rows of data.

	PRODUCT_ID	PRODUCT_NAME	PRODUCT_PRICE	SALE_PRICE
1	10002	Wet/Dry Vacuum	70	139.99
2	10003	Cordless Nailer 20V	170	249.99

4. Containing nested query with IN

Get all orders that are being supplied by a supplier

```
SELECT *
FROM Orders O
WHERE O.Order_ID IN (
    SELECT IO.Order_ID
    FROM Incoming_Order IO, Supplier S
    WHERE IO.Sup_ID = S.Sup_ID and S.Sup_Name = 'InkJoy');
```



The screenshot shows a SQL query result window with two tabs: 'Script Output' and 'Query Result'. The 'Query Result' tab is active, displaying a table with 10 columns: ORDER_ID, ORDERED_BY, STATUS, CONTACT, SHIP_DATE, SHIP_ADDRESS, SHIP_CITY, SHIP_STATE, SHIP_ZIPCODE, and COUNTRY. The table contains one row of data.

ORDER_ID	ORDERED_BY	STATUS	CONTACT	SHIP_DATE	SHIP_ADDRESS	SHIP_CITY	SHIP_STATE	SHIP_ZIPCODE	COUNTRY
1	9 Frank	Processing	7892573612	19-11-22	NJIT Central Building	Newark	NJ	7102	USA

Challenges Faced

- The database creation was the most time-consuming part of this project. We also had some unspecified data assumptions that were later included and modified to match the business requirements of this project.
- Earlier on, we also faced some difficulties connecting to the Oracle database through Django due to limited availability of resources, but eventually after research, we resolved this problem with the help of some supporting libraries.
- While working on the frontend and backend integration, we had some trouble receiving backend trigger updates. The server sometimes failed to respond due to improper connection, which led to websites crashing at some point. This main issue was due to improper scalability when we added more relations to match the modified requirements. This was then resolved during later iterations of the project.

Conclusion

This project gave us extensive knowledge over the implementation of database concepts. There was a lot of discussion and research, resulting in skill development over database management. Finally, the application development over a real-world scenario helped us relate the concepts with use cases that decrease redundancy and increase performance.