



2023

# Bike Store Management System- TerraBikes



Team: SQL Weavers

Course: DATA 225

12/12/2023

## Contents

<b>1. Introduction</b>	<b>2</b>
<b>2. Data Sources:</b>	<b>2</b>
<b>3. Application Design</b>	<b>2</b>
3.1. Operation Module	3
3.2. Analytical Module:	3
<b>4. Database Design</b>	<b>4</b>
<b>5. Working of the Operational module</b>	<b>5</b>
<b>6. Specifications and Usability of Operational Module</b>	<b>6</b>
6.1. Login page:	6
6.2. Forgot Password:	6
6.3. Customer Portal:	7
6.3.1. Track Orders:	8
6.3.2. Orders:	8
6.3.3. Feedback:	9
6.3.4. Greviances:	9
6.4. Employee Portal:	10
6.4.1. Orders:	11
6.4.2. Inventory:	12
6.4.3. Greviances:	14
6.4.4. Customers:	15
6.4.5. New Order (Walk-In):	15
6.5. Manager Portal:	16
<b>7. Summary for Operational Module</b>	<b>18</b>
<b>8. Working of Analytical Module</b>	<b>19</b>
<b>9. Specifications and Usability of Analytical Module</b>	<b>19</b>
9.1. Sales Dashboard	19
9.2. Employee Dashboard	20
<b>10. Summary of Analytical Module</b>	<b>21</b>
<b>11. Technical Aspects</b>	<b>22</b>
<b>12. Database Technical Details</b>	<b>22</b>
12.1. DB Objects	22
12.2. ETL	23
12.3. Queries (Operation DB)	24
12.4. Queries (Analytical DB)	26

## 1. Introduction

The Bike Store Management System is a comprehensive solution designed to meticulously catalog and manage day-to-day operations and sales activities within a dynamic cycling retail environment. In response to the growing demand for efficient store management and the need to analyze sales trends across various outlets, we have crafted a dual-functional system that seamlessly integrates operational and analytical databases.

The operational database serves as the backbone of the system, dedicated to recording and tracking the intricacies of daily transactions, inventory updates, and customer interactions. This functionality ensures a streamlined process for managing orders, stock levels, and overall store operations. With a focus on real-time data entry and retrieval, the operational database empowers store staff to efficiently navigate the daily challenges of running a bike store.

On the other hand, our analytical database plays a pivotal role in unraveling the broader narrative of the business. By harnessing the power of historical sales data, this component of the system enables stakeholders, particularly managers, to gain profound insights into sales trends over the years and across different store locations. The analytical database becomes a strategic tool for decision-making, providing a platform for managers to assess employee performance, identify top-selling products, and formulate informed strategies for business growth.

In essence, the Bike Store Management System is a sophisticated tandem of operational efficiency and analytical prowess, designed to elevate the management and performance evaluation processes within the dynamic realm of bicycle retail.

## 2. Data Sources:

<b>Mockaroo</b>	<a href="https://www.mockaroo.com">https://www.mockaroo.com</a>
<b>US Cities Name</b>	<a href="https://simplemaps.com/data/us-cities">https://simplemaps.com/data/us-cities</a>
<b>Kaggle</b>	<a href="https://www.kaggle.com/datasets/dillonmyrick/bike-store-sample-database">https://www.kaggle.com/datasets/dillonmyrick/bike-store-sample-database</a>

## 3. Application Design

The tools used during development includes Python, MySQL Workbench, PyQt Designer, ERD Plus and VSCode. It requires additional packages to be installed:

- [Pandas](#)
- [Numpy](#)
- [Matplotlib](#)
- [PyQt5 and PyQt5-Tools](#)
- [Seaborn](#)
- [MySQL Connector for Python](#)
- [SMTPLIB](#)

The application consists of two main modules: Operation Module and Analytical Module.

### 3.1. Operation Module

Customers engaging with the Bike Store Operational Database can seamlessly register, log in, and initiate various transactions to enhance their shopping experience. Within this segment of the system, customers have the capability to place new orders, submit feedback, and raise any concerns through a streamlined complaints process. When submitting a complaint, customers are prompted to provide essential details such as the nature of the issue, the product involved, and the relevant order information. To facilitate efficient tracking, the system meticulously validates these details, generating a unique order\_ID to distinctly identify and monitor each transaction and complaint.

Simultaneously, employees harness the power of the operational database to streamline their responsibilities. They can easily access information on the number of orders they've handled, assist new customer registrations, collect, and analyze customer feedback, and provide resolutions to any raised complaints. Furthermore, employees are empowered to initiate the process of ordering new inventory for the store, ensuring that the product lineup remains current and meets customer demands.

In the broader managerial context, the operational database serves as a comprehensive tool for overseeing the store's functionality. Managers gain insights into the onboarding process for new employees, monitoring their performance into the system. Additionally, the database offers a panoramic view of employee performance metrics, enabling managers to make informed decisions and implement strategies for ongoing improvement.

In essence, the Bike Store Operational Database stands as a user-centric hub, where customers, employees, and managers converge to optimize the entire spectrum of operations, from order placement to resolution management and strategic decision-making.

### 3.2. Analytical Module:

Within the Bike Store Analytical Database module, our focus revolves around leveraging the wealth of data accumulated from our operational database alongside external sources. This comprehensive analysis yields an insightful dashboard that delves into the intricacies of sales and operational performance. The dashboard provides a panoramic view of the store's landscape, encompassing diverse metrics such as sales performance, employee efficiency, and regional product dynamics.

Users with managerial roles have exclusive access to this analytical dashboard, where they can explore nuanced details of the store's performance. The system showcases comprehensive sales performance metrics, offering insights into top-selling products for each region. Users can drill down into historical data for all years, specific years, six-month intervals, quarters, or the last month, enabling a granular examination of sales trends over time.

Furthermore, the dashboard facilitates an in-depth analysis of order-related metrics, including order counts by status, the year-wise spike of bike quantities by category, and overall order trends by year and month. For a regional perspective, users can assess total orders by region, identifying patterns and optimizing strategies based on regional demand.

The managerial privilege extends to a specialized employee performance dashboard, providing key metrics for assessing workforce effectiveness. Managers can discern the top-performing regions in terms of sales, identify the highest sales contributors by region, and recognize the top 10 employees who have made significant sales contributions. This analytical capability empowers managers to make data-driven decisions, strategize for growth, and optimize operational efficiency within the dynamic landscape of the bike store.

## 4. Database Design

Throughout the conceptualization and design phases of the Bike Store Management System, a meticulous focus was placed on crafting a database structure comprised of normalized tables, upholding the highest standards of data integrity. The system's architecture was strategically tailored to ensure that each table adhered to stringent normalization principles, promoting efficient data storage and the elimination of redundancies. This deliberate approach resulted in a system characterized by its simplicity and effectiveness, facilitating a seamless process for creating and managing products within the store.

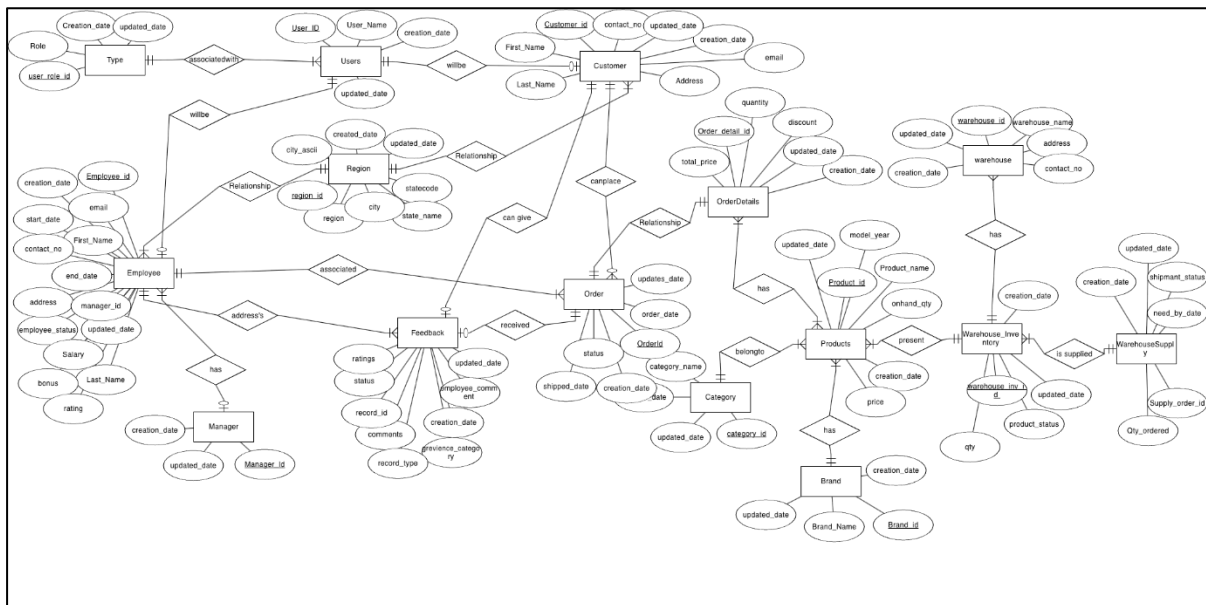


FIGURE 1: ER DIAGRAM

Normalization played a pivotal role in enhancing the efficiency of the database, optimizing the storage of information while systematically reducing redundancy. By eliminating unnecessary duplications and organizing data logically, the design achieved a lightweight profile that not only streamlined day-to-day operations but also facilitated ease of maintenance and updates. This commitment to normalization has contributed to the creation of a robust and reliable product, providing the Bike Store Management System with a solid foundation for scalability and adaptability as the store evolves over time.

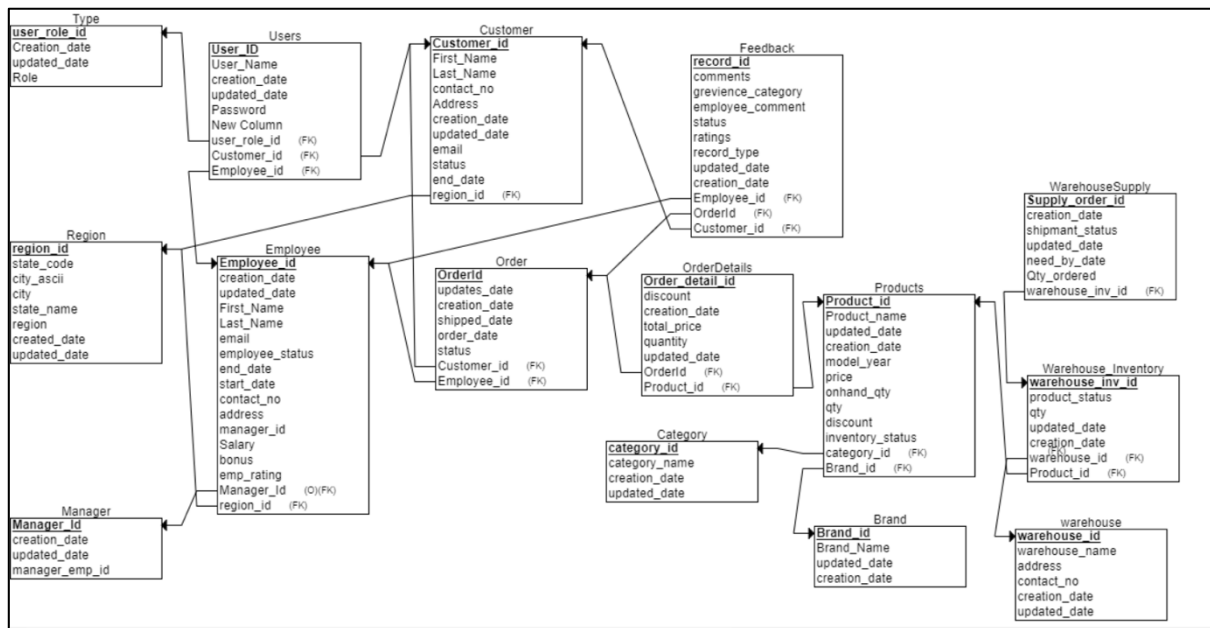


FIGURE 2: RELATIONAL SCHEMA

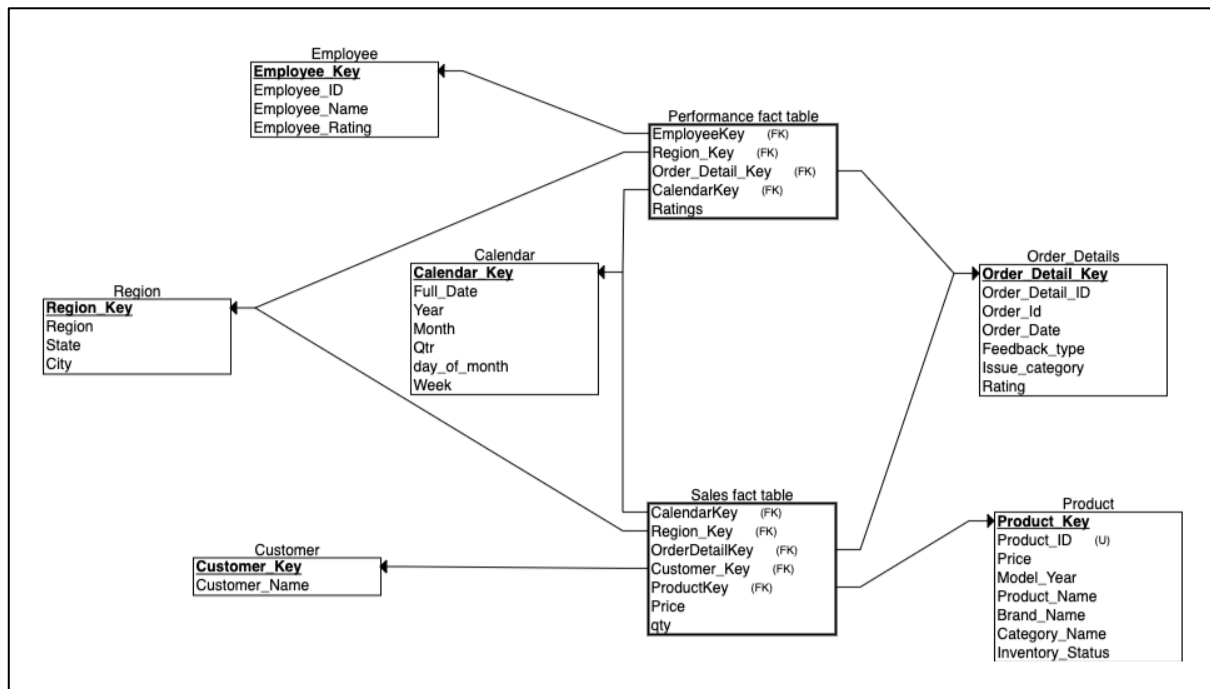


FIGURE 3: STAR SCHEMA

## 5. Working of the Operational module

The operational structure of our bike store centers around three primary roles: the Customer, Employee, and Manager. Each role carries distinct responsibilities and functionalities. For instance, the Customer role encompasses the ability to place orders, log in for personalized services, and submit grievances.

On the other hand, the Employee role involves multifaceted tasks, including logging in, assisting in the registration of new walk-in customers, managing, and tracking orders while adjusting their status, providing discounts, overseeing inventory management by monitoring in-store stock levels, and

initiating orders for replenishment from the warehouse. Employees are also responsible for addressing and tracking customer grievances, with the authority to update their status as 'rejected' or 'resolved.'

The Manager, as a key figure in our store's operation, has the capability to monitor new employee onboarding and assess employee performance. Additionally, the manager plays a crucial role in analyzing and keeping track of sales trends over the years across all regions, contributing to strategic decision-making and business growth. This comprehensive structure ensures efficient and seamless operations within our bike store, catering to the diverse needs of customers, empowering employees with essential tools, and enabling managerial oversight for informed decision-making.

## 6. Specifications and Usability of Operational Module

### 6.1. Login page:

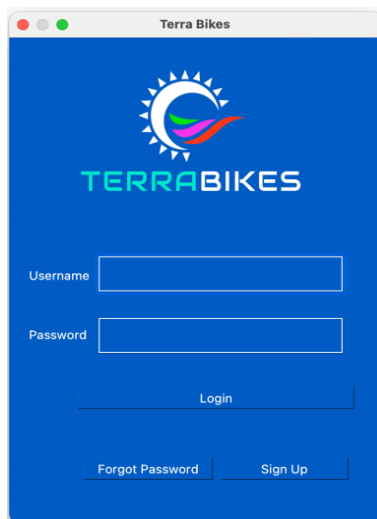


FIGURE 4: APP LANDING PAGE

- The application has login page from where we can login with the username and password. We can create a user account if you are new user.
- There are three roles:
  - **Customer** - Customer can generate new orders, raise grievances, and give product feedback.
  - **Employee** - assisting in the registration of new walk-in customers, managing, and tracking orders while adjusting their status, providing discounts, overseeing inventory management by monitoring in-store stock levels, and initiating orders for replenishment from the warehouse.
  - **Manager** - Manager is also employee, so he/she is registered within employee.

Files:

- BikeStoreMainWin.py
- BikeStoreMainWin\_ui.py
- BikeStoreMainWin.ui

### 6.2. Forgot Password:

This screen supports the verification email functionality for user verification.

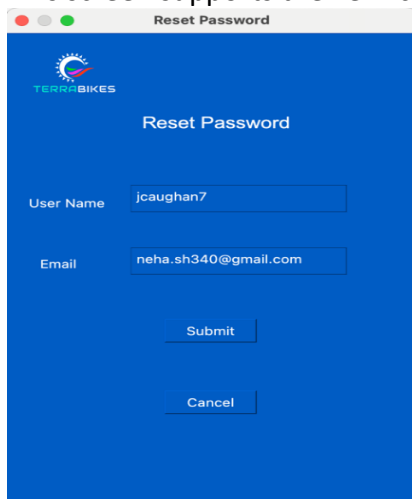


FIGURE 5: RESET PASSWORD PAGE

- For any user three options appear on the portal. I.e. login, forget password and sign up.
- Out of which one of them is “Forgot Password”.
- This option allows the user to retrieve the forgotten password.
- The user is required to enter both username and email id where they can receive a validation code.
- After receiving a validation code, they can reset their password.

Files:

- BikeStoreResetPwdDialog.py
- BikeStoreResetPwdDialog\_ui.py
- BikeStoreResetPwdDialog.ui

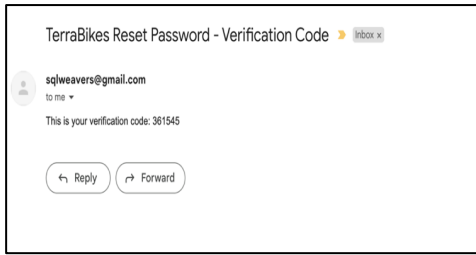


FIGURE 6: EMAIL NOTIFICATION

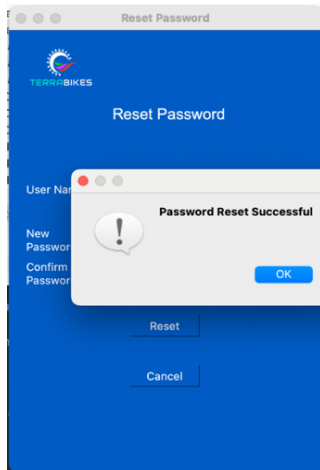


FIGURE 6: SUCCESSFUL VERIFICATION POP-UP

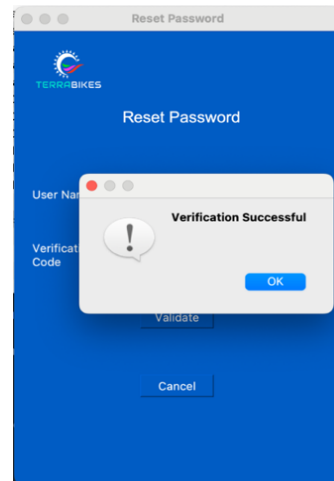
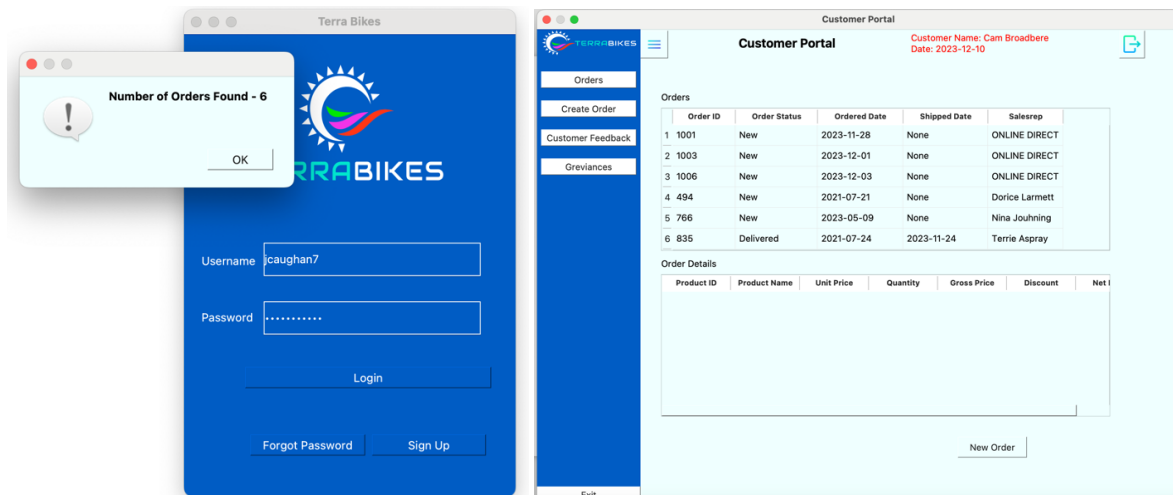


FIGURE 7: PASSWORD RESET CONFIRMATION

### 6.3. Customer Portal:



(use username as jcaughan7 and password as welcome1234)

- The customer can perform four operations:
- Track the orders that customer have placed in past and check new orders
- Create new order
- Give feedback for orders they have placed
- Submit complaint/Grievances for order they have placed



### 6.3.1. Track Orders:

**Track the orders:** From orders page customer can track the orders placed in past and new orders. From orders page and create order page customer can place new order.

Customer Portal  
Customer Name: Cam Broadbere  
Date: 2023-12-10

Orders

Order ID	Order Status	Ordered Date	Shipped Date	Salesrep
1 494	New	2021-07-21	None	Dorice Larmett
2 766	New	2023-05-09	None	Nina Jouhning
3 835	Delivered	2021-07-24	2023-11-24	Terrie Aspray
4 1001	New	2023-11-28	None	ONLINE DIRECT
5 1003	New	2023-12-01	None	ONLINE DIRECT
6 1006	New	2023-12-03	None	ONLINE DIRECT

Order Details

Product ID	Product Name	Unit Price	Quantity	Gross Price
1 FG2023SBCCL	Cannondale CAAD13 Disc	749.99	5	3749.95
2 MTB2023SRH	Giant Anthem Advanced Pro 29	549.99	5	2749.95
3 FB2023TM8D	State Bicycle Co. Core Line	699.99	3	2099.9700000000000
4 FB2023BS6L	Giant Contend SL1	899.99	3	2699.9700000000000

New Order

### 6.3.2. Orders:

**Create New Order:** From create order page, customer can place new order, at one time customer can add 5 different products though for different products quantity can be any.

Customer Portal  
Customer Name: Cam Broadbere  
Date: 2023-12-10

Customer Name: Cam Broadbere

Product	Price	Discount	Qty.	Sub Total
EB2023MVT - Giant Escape 3	869.99	0.00	Qty. 1	0.0

Order Total: 0.0

Submit Order Cancel Order

FIGURE 8: STEP 1: ADD NEW PRODUCTS

FIGURE 10: STEP 2: ADDED 5 DIFFERENT PRODUCTS

Customer Portal  
Customer Name: Cam Broadbere  
Date: 2023-12-10

Customer Name: Cam Broadbere

Product	Price	Discount	Qty.	Sub Total
FB2023BMSD8 - Strida LT	909.99	0.00	Qty. 13	11829.87
EB2023RPBRR - Trek X-Caliber 9	649.99	0.00	Qty. 5	3249.95
FG2023QCXT - Santa Cruz Nomad	999.99	200.0	Qty. 1	799.99
HB2023S5X - Trek FX 3	789.99	0.00	Qty. 2	1579.98
HB2023GQCC3 - Rad Power Bikes RadRover	819.99	0.00	Qty. 12	9839.88

Order Total: 27299.67

Submit Order Cancel Order

FIGURE 9: STEP 3: ORDER PLACED SUCCESSFULLY

Customer Portal  
Customer Name: Cam Broadbere  
Date: 2023-12-10

Customer Name: Cam Broadbere

Order Submitted Successfully

Product	Price	Discount	Qty.	Sub Total
FB2023BMSD8 - Strida LT	909.99	0.00	Qty. 13	11829.87
EB2023RPBRR - Trek X-Caliber 9	649.99	0.00	Qty. 5	3249.95
FG2023QCXT - Santa Cruz Nomad	999.99	200.0	Qty. 1	799.99
HB2023S5X - Trek FX 3	789.99	0.00	Qty. 2	1579.98
HB2023GQCC3 - Rad Power Bikes RadRover	819.99	0.00	Qty. 12	9839.88

Order Total: 27299.67

Submit Order Cancel Order

### 6.3.3. Feedback:

**Give feedback for orders they have placed:** From Customer Feedback page customer can give rating for orders they placed. And these orders are classified as “online direct” or “in store”. When orders are placed in store the rating or feedback that you provide is reflected and updated as employee feedback as well.

**Note: From here the employee rating is updated.**

The screenshot shows the 'Customer Portal' interface for a user named 'Cam Broadbere' on '2023-12-10'. The left sidebar contains navigation links: 'Orders', 'Create Order', 'Customer Feedback' (highlighted), and 'Grievances'. The main content area has a form with the following fields: 'Customer Name' (filled with 'Cam Broadbere'), 'Order ID' (a dropdown menu showing 'Order: 1008 Status: New Ordered Date: 2023-12-10'), 'Rating' (a dropdown menu showing '5'), and 'Comments' (a large text area). At the bottom of the form are two buttons: 'Submit Feedback' and 'Cancel Feedback'. An 'Exit' button is located at the bottom left of the sidebar.

- This is the page from where customer can choose for which order they want to give feedback
- For each order we can see that order\_id is different and that helps in tracking different orders

The first screenshot shows the 'Customer Portal' interface with the 'Customer Feedback' form. The 'Comments' field now contains the text 'Process of placing order was smooth'. The second screenshot shows the same interface, but with a modal dialog box overlaid on top. The dialog box has a title bar with a red close button and contains the text 'Thank you, Feedback Submitted Successfully!!' and an 'OK' button. The 'Submit Feedback' button is visible at the bottom of the form in both screenshots.

### 6.3.4. Grievances:

**Submit complaint/Grievances for order they have placed:**

Customer Portal

Customer Name: Cam Broadbere  
Date: 2023-12-10

Order ID	Category	Status	Creation Date	The staff at the store w
1 494	Customer Service Problems	Closed	2023-11-19	The staff at the store w
2 766	Customer Service Problems	Rejected	2023-11-24	asdsadsadsad
3 835	Billing and Pricing Disputes	Closed	2023-11-24	Incorrect Bill
4 1003	Customer Service Problems	Rejected	2023-12-01	

Customer Name: Cam Broadbere

Order ID: Order: 1008 Status: New Ordered D

Issue Category: Product Quality Issues

Issue Details:

Employee Comments:

New Grievance Submit Cancel

FIGURE 10: CHOOSE ORDER FOR WHICH COMPLAINT HAS TO BE RAISED

- This is the page from where customer can raise grievances or complaints for any order
- Customer can choose a order they want to register complaint for and select issue category.
- And then they provide details for that issue and submit complaint.
- After that when a customer submits a new complaint, its status is updated as open. And its later change when employee work on it.

FIGURE 11: SELECT ISSUE CATEGORY AND GIVE DETAILS

Customer Portal

Customer Name: Cam Broadbere  
Date: 2023-12-10

Order ID	Category	Status	Creation Date	The staff at the store w
1 494	Customer Service Problems	Closed	2023-11-19	The staff at the store w
2 766	Customer Service Problems	Rejected	2023-11-24	asdsadsadsad
3 835	Billing and Pricing Disputes	Closed	2023-11-24	Incorrect Bill
4 1003	Customer Service Problems	Rejected	2023-12-01	

Customer Name: Cam Broadbere

Order ID: Order: 1008 Status: New Ordered D

Issue Category: Billing and Pricing Disputes

Issue Details: since this was a bulk order, i should have recieved some more discount

Employee Comments:

New Grievance Submit Cancel

FIGURE 12: SUBMISSION OF COMPLAINT

Customer Portal

Customer Name: Cam Broadbere  
Date: 2023-12-10

Order ID	Category	Status	Creation Date	The staff at the store w
1 494	Customer Service Problems	Closed	2023-11-19	The staff at the store w
2 766	Customer Service Problems	Rejected	2023-11-24	asdsadsadsad
3 835	Billing and Pricing Disputes	Closed	2023-11-24	Incorrect Bill
4 1003	Customer Service Problems	Rejected	2023-12-01	

Customer Name: Cam Broadbere

Order ID: Order: 1008 Status: New Ordered D

Issue Category: Billing and Pricing Disputes

Issue Details: since this was a bulk order, i should have recieved some more discount

Employee Comments:

New Grievance Submit Cancel

Grievance Submitted, We will be in touch!!

OK

FIGURE 13: STATUS OF COMPLAINT UPDATED AS OPEN

Customer Portal

Customer Name: Cam Broadbere  
Date: 2023-12-10

Order ID	Category	Status	Creation Date	The staff at the store w
1 494	Customer Service Problems	Closed	2023-11-19	The staff at the store w
2 766	Customer Service Problems	Rejected	2023-11-24	asdsadsadsad
3 835	Billing and Pricing Disputes	Closed	2023-11-24	Incorrect Bill
4 1003	Customer Service Problems	Rejected	2023-12-01	
5 1008	Billing and Pricing Disputes	Open	2023-12-10	since this was a bulk or

Customer Name: Cam Broadbere

Order ID: Order: 494 Status: New Ordered Da

Issue Category: Product Quality Issues

Issue Details: since this was a bulk order, i should have recieved some more discount

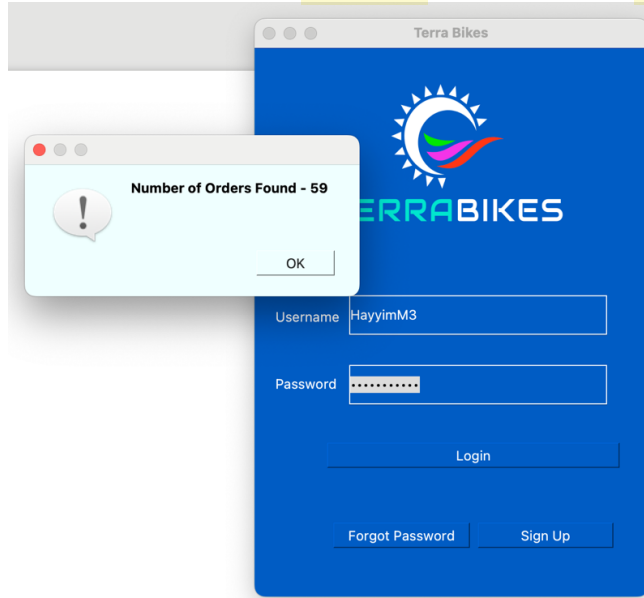
Employee Comments: None

New Grievance Submit Cancel

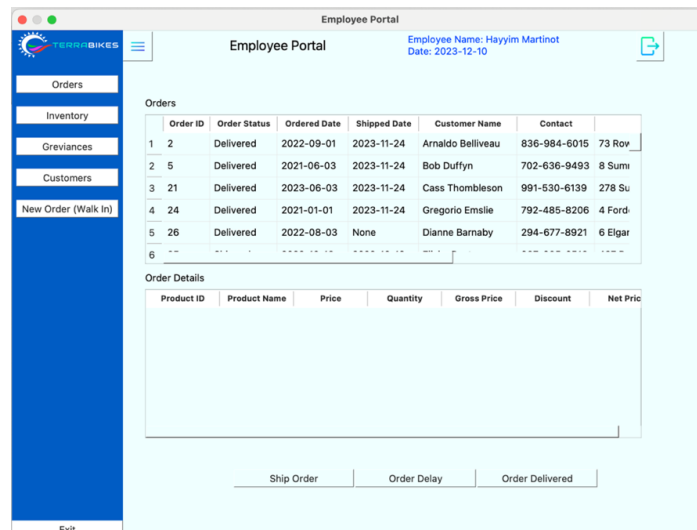
## 6.4. Employee Portal:

Successfully logged in as an employee:

(use username as HayyimM3 and password as welcome1234)



- The employee can perform five operations:
- Track the orders that customer have placed in past, check new orders, and change order status
- Inventory Management for the store
- Provide resolution of grievances raised by customers
- Can view all customers and can either disable or delete very old customers
- Registration of new walk-in customer, and help them in placing new order

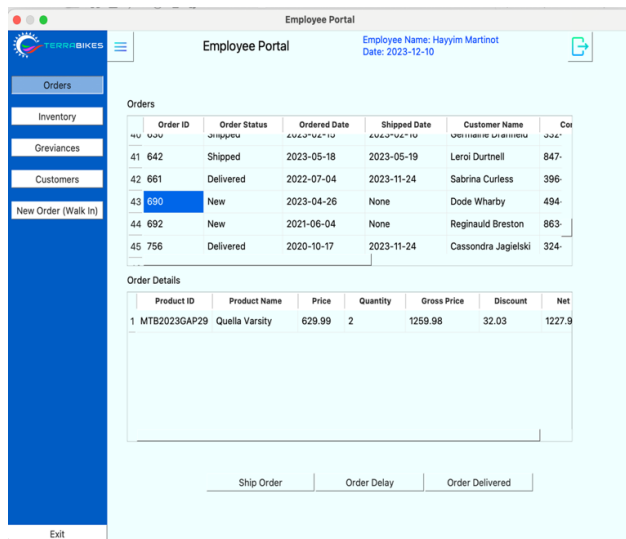


#### 6.4.1. Orders:

**Track the orders:** From orders page employee can track the orders placed in past and new orders of customer. From orders employee can change the status of order to “ship order”, “order delay”, “order delivered”. Here in following steps I have shown how we change order status from new to shipped. Likewise we can change the status to delay or delivered.

**Note:** For new order we can change status to delay and shipped. For shipped order we can change status to delivered. But we can’t mark shipped order as delayed.

FIGURE 14: HERE YOU CAN SEE THAT ORDER STATUS FOR ORDER ID 690 IS NEW



Employee Portal  
Employee Name: Hayyim Martinot  
Date: 2023-12-10

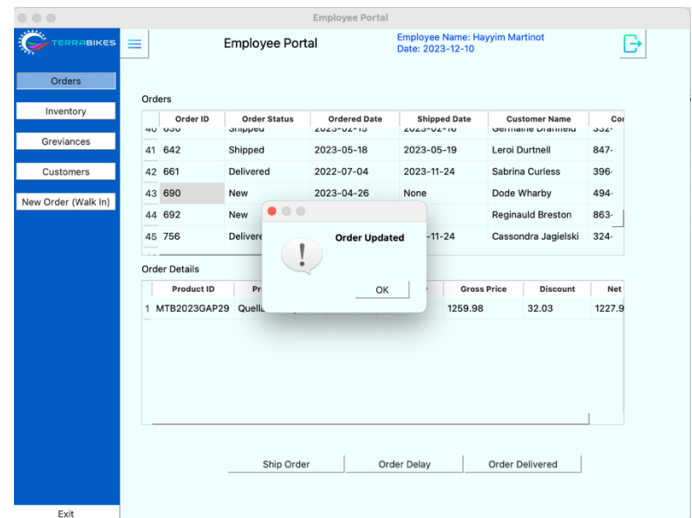
Order ID	Order Status	Ordered Date	Shipped Date	Customer Name	Contact
41 642	Shipped	2023-05-18	2023-05-19	Lerol Durnnell	847-
42 661	Delivered	2022-07-04	2023-11-24	Sabrina Curless	396-
43 690	New	2023-04-26	None	Dode Wharby	494-
44 692	New	2021-06-04	None	Reginauld Breston	863-
45 756	Delivered	2020-10-17	2023-11-24	Cassondra Jagielski	324-

Product ID	Product Name	Price	Quantity	Gross Price	Discount	Net
1 MTB2023GAP29	Quella Varsity	629.99	2	1259.98	32.03	1227.9

Ship Order   Order Delay   Order Delivered

FIGURE 15: HERE STATUS IS UPDATED



Employee Portal  
Employee Name: Hayyim Martinot  
Date: 2023-12-10

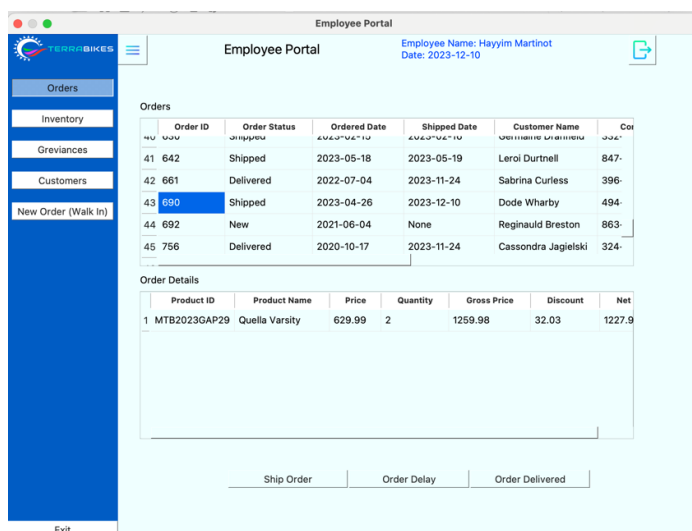
Order ID	Order Status	Ordered Date	Shipped Date	Customer Name	Contact
41 642	Shipped	2023-05-18	2023-05-19	Lerol Durnnell	847-
42 661	Delivered	2022-07-04	2023-11-24	Sabrina Curless	396-
43 690	New	2023-04-26	None	Dode Wharby	494-
44 692	New	2021-06-04	None	Reginauld Breston	863-
45 756	Delivered	2020-10-17	2023-11-24	Cassondra Jagielski	324-

Product ID	Product Name	Price	Quantity	Gross Price	Discount	Net
1 MTB2023GAP29	Quella Varsity	629.99	2	1259.98	32.03	1227.9

Ship Order   Order Delay   Order Delivered

FIGURE 16: HERE NOW ORDER STATUS IS CHANGED TO SHIPPED



Employee Portal  
Employee Name: Hayyim Martinot  
Date: 2023-12-10

Order ID	Order Status	Ordered Date	Shipped Date	Customer Name	Contact
41 642	Shipped	2023-05-18	2023-05-19	Lerol Durnnell	847-
42 661	Delivered	2022-07-04	2023-11-24	Sabrina Curless	396-
43 690	Shipped	2023-04-26	2023-12-10	Dode Wharby	494-
44 692	New	2021-06-04	None	Reginauld Breston	863-
45 756	Delivered	2020-10-17	2023-11-24	Cassondra Jagielski	324-

Product ID	Product Name	Price	Quantity	Gross Price	Discount	Net
1 MTB2023GAP29	Quella Varsity	629.99	2	1259.98	32.03	1227.9

Ship Order   Order Delay   Order Delivered

## 6.4.2. Inventory:

**Inventory Management for the store:** In inventory page employee can track stock of products available in store. And in store if stock for product is low then employee can search from all the warehouses which have that same product and can order the product.

FIGURE 17: HERE WE WANT TO KEEP STOCK FOR THIS PRODUCT ID FG2023PUCE

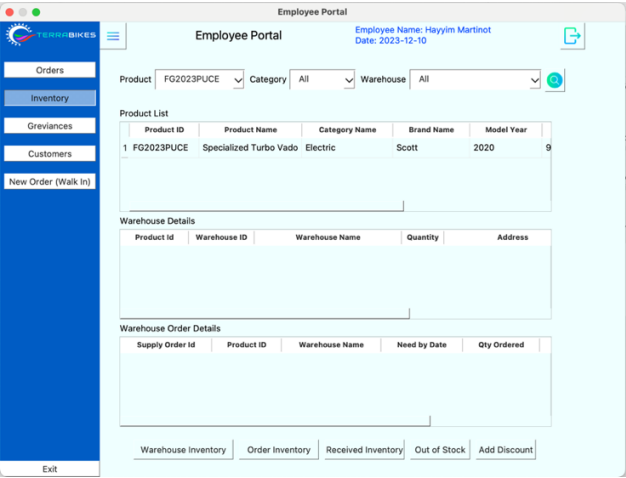


FIGURE 19: HERE WE HAVE ALL THE WAREHOUSES WHICH HOLD THE STOCK OF THIS ITEM

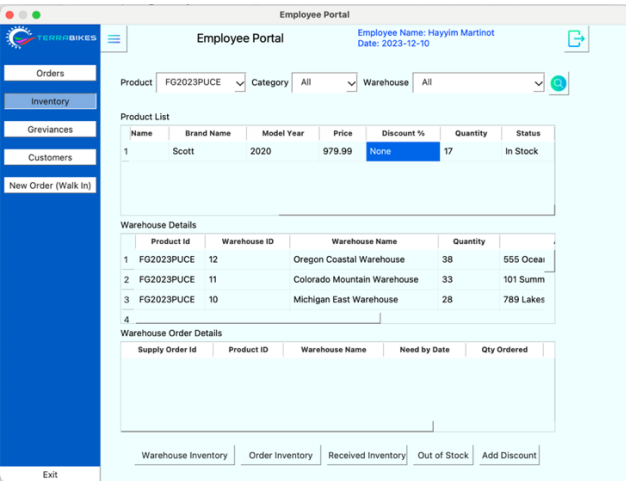


FIGURE 21: FROM HERE EMPLOYEE CAN TRACK INVENTORY ORDERED FROM A WAREHOUSE AND ALSO CHANGE THE STATUS OF ORDERED INVENTORY

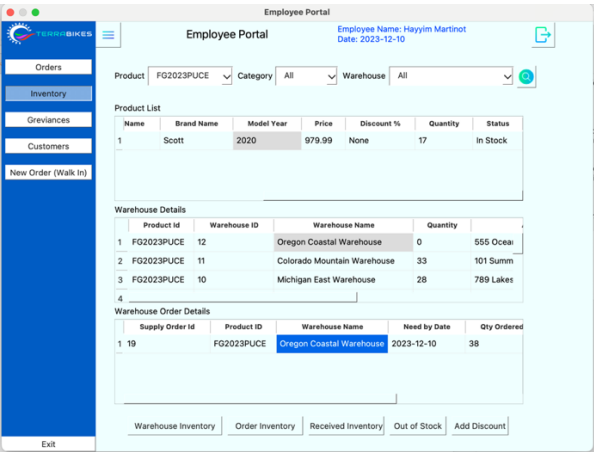


FIGURE 18: HERE WE SEE THAT QUANTITY FOR THIS PRODUCT IS 17

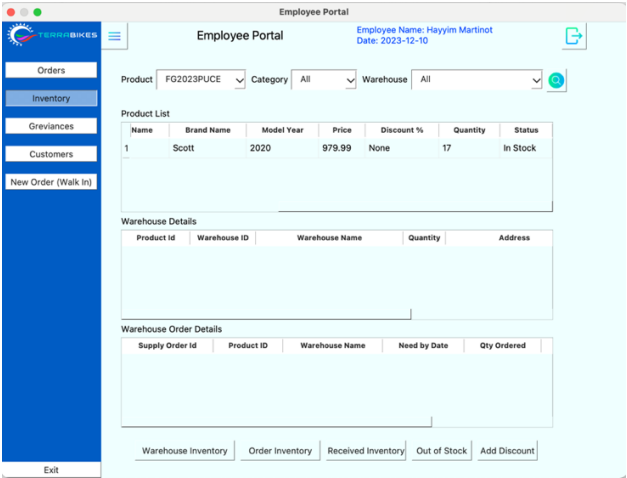


FIGURE 20: INVENTORY ORDERED

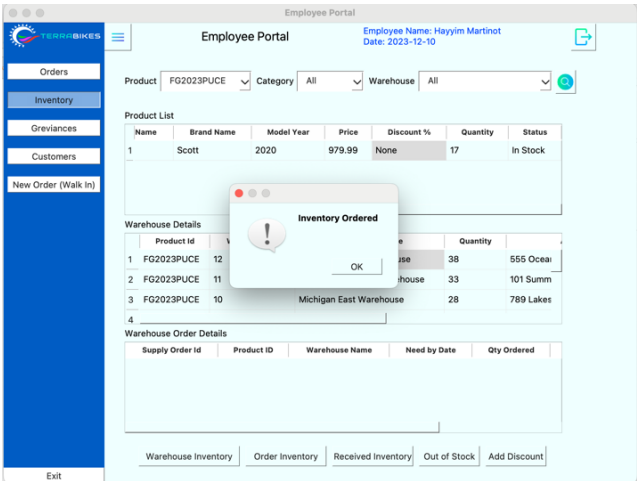


FIGURE 22: STATUS OF ORDERED INVENTORY IS CHANGED TO RECEIVED

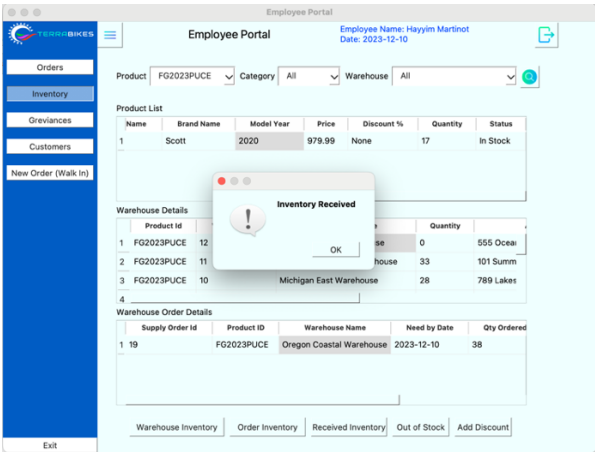
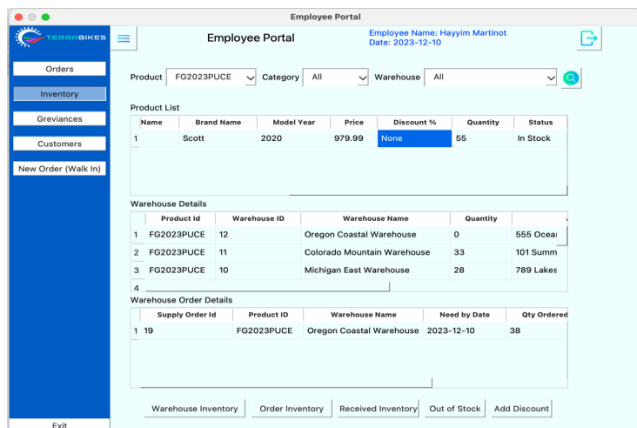


FIGURE 23: HERE WE CAN SEE THAT NOW STOCK IS UPDATED FROM 17 TO 55



### 6.4.3. Grievances:

**Provide resolution of grievances raised by customers:** In grievances page, an employee can provide resolution to the complaints which are still open.

FIGURE 24: THERE IS ONE COMPLAINT WHICH IS STILL OPEN

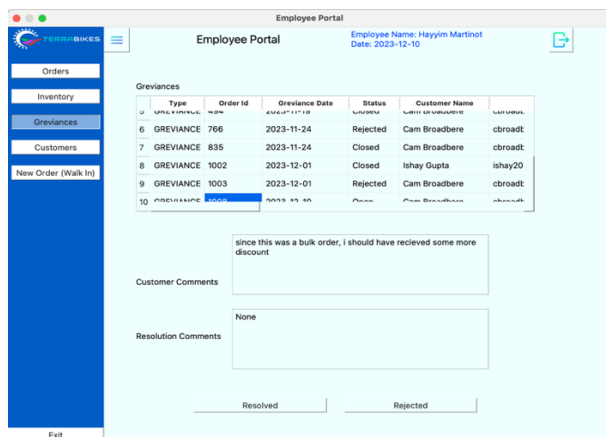


FIGURE 25: AN EMPLOYEE HERE PROVIDE RESOLUTION COMMENTS AND WILL MARK THE STATUS

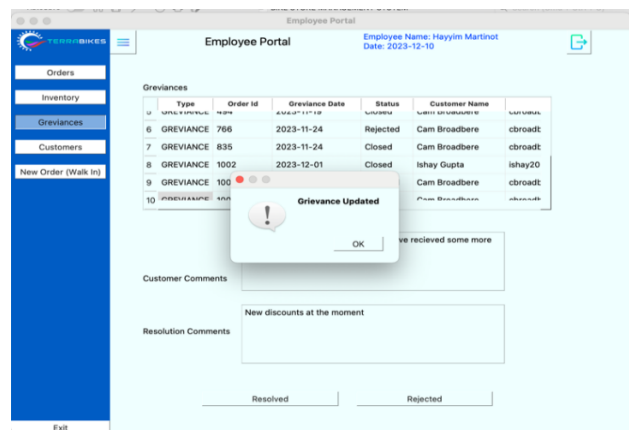
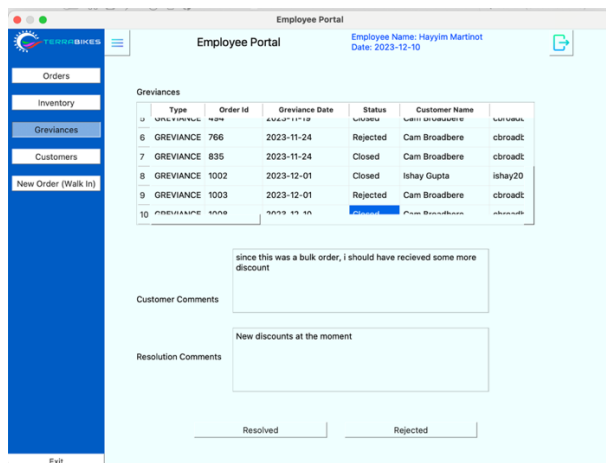


FIGURE 26: AND AFTER EMPLOYEE PROVIDES A COMMENT, STATUS IS UPDATED AS CLOSED



#### 6.4.4. Customers:

**Can view all customers and can either disable or delete very old customers:** From this page we track all the customers and see the orders they have placed over years. And if a customer has not placed order from very long time, we can either disable them or delete them.

FIGURE 27: HERE WE CAN SEARCH CUSTOMER EITHER AS CUSTOMER ID OR CUSTOMER NAME. OR JUST CLICK ON SEARCH AND SEE ALL CUSTOMERS

Customer ID	Customer Name	Status	End Date	Order Count	Contact
1	Gibb MacCumsiskey	INACTIVE	2023-11-23 00:13:20	5	214-16-
2	Deb Farrington	INACTIVE	2023-11-23 01:36:05	7	627-78
3	Retha Kirkbride	INACTIVE	2023-11-23 17:57:13	4	677-72
4	Luther Lyddiard	INACTIVE	2023-11-23 17:59:29	2	509-12
5	Hazlett Duthie	INACTIVE	2023-11-24 01:26:41	2	886-76
6	Herrick Lindenboim	ACTIVE	None	4	375-40

Order Id	Status	Ordered Date	Shipped Date	Product Count	Total Quantity
1 71	Shipped	2022-04-29	2022-04-30	2	4
2 370	Shipped	2021-07-01	2021-07-02	2	9
3 575	Delivered	2021-08-06	2023-11-24	1	3
4 839	Delivered	2022-04-06	2023-11-24	1	4

FIGURE 28: WE CAN SEE ALL THE ORDERS PLACED BY A CUSTOMER

Order Id	Status	Ordered Date	Shipped Date	Product Count	Total Quantity
1 269	New	2020-09-29	None	2	9
2 374	Delivered	2021-08-20	2023-11-24	1	3
3 477	Shipped	2020-12-15	2020-12-16	2	9
4 857	Shipped	2022-04-17	2022-04-18	2	6
5 963	Delivered	2021-08-01	2023-11-24	1	2

FIGURE 29: EMPLOYEE CAN DECIDE WHICH CUSTOMER THEY WANT TO DISABLE OR DELETE

Customer ID	Customer Name	Status	End Date	Order Count	Contact
20 20	Chadd Tante	ACTIVE	None	6	149-12-
21 21	Huey Fitzgerald	ACTIVE	None	5	191-66-
22 22	Kim Dacres	ACTIVE	None	8	578-73
23 23	Rorie Phillpotts	ACTIVE	None	3	497-9C
24 24	Cassandra Jagielski	ACTIVE	None	8	324-91
25 25	Zachariah Pettford	ACTIVE	None	4	305-73

Order Id	Status	Ordered Date	Shipped Date	Product Count	Total Quantity
1 202	Delivered	2022-04-08	2023-11-24	1	5
2 610	New	2021-12-15	None	1	1
3 933	New	2021-03-27	None	3	12

#### 6.4.5. New Order (Walk-In):

**Registration of new walk-in customer and help them in placing new order:** From this page an employee can help new customers with placing new orders and help them sign up. An employee will create an account for them, provide them with customer\_id and password which customer can change later.



AN EMPLOYEE WILL ENTER ALL THE CUSTOMER DETAILS FOR NEW CUSTOMER

Employee Portal  
Employee Name: Hayyim Martinot  
Date: 2023-12-10

Customer Name: \_\_\_\_\_ Email: \_\_\_\_\_  
Street Address: \_\_\_\_\_ Phone: \_\_\_\_\_  
State: Alabama City: \_\_\_\_\_ Username: \_\_\_\_\_  
Postal Code: \_\_\_\_\_ Password: \_\_\_\_\_

Product	Price	Discount	Qty.	Sub Total
EB2023MVT - Giant Escape 3	869.99	0.00	0	0.0

Order Total: 0.0

Submit Cancel

AN EMPLOYEE WILL ENTER ALL THE CUSTOMER DETAILS FOR NEW CUSTOMER

Employee Portal  
Employee Name: Hayyim Martinot  
Date: 2023-12-10

Customer Name: Gaurav Sharma Email: sharmagaurav459@gmail.com  
Street Address: 5822 Charlotte Drive Phone: 480-302-0773  
State: California City: San Jose Username: sharmagaurav459  
Postal Code: 95123 Password: .....

Product	Price	Discount	Qty.	Sub Total
HB202355X - Trek FX 3	789.99	0.00	1	789.99
FB2023B56L - Giant Contend SL1	899.99	103.5	2	1592.98

Order Total: 2382.97

Submit Cancel

CUSTOMER CREATED SUCCESSFULLY

Employee Portal  
Employee Name: Hayyim Martinot  
Date: 2023-12-10

Customer Name: Gaurav Sharma Email: sharmagaurav459@gmail.com  
Street Address: 5822 Charlotte Drive Phone: 480-302-0773  
State: California City: San Jose Username: sharmagaurav459  
Postal Code: 95123 Password: .....

Product	Price	Discount	Qty.	Sub Total
HB202355X - Trek FX 3	789.99	0.00	1	789.99
FB2023B56L - Giant Contend SL1	899.99	103.5	2	1592.98

Order Total: 2382.97

Submit Cancel

ORDER CREATED SUCCESSFULLY

Employee Portal  
Employee Name: Hayyim Martinot  
Date: 2023-12-10

Customer Name: Gaurav Sharma Email: sharmagaurav459@gmail.com  
Street Address: 5822 Charlotte Drive Phone: 480-302-0773  
State: California City: San Jose Username: sharmagaurav459  
Postal Code: 95123 Password: .....

Product	Price	Discount	Qty.	Sub Total
HB202355X - Trek FX 3	789.99	0.00	1	789.99
FB2023B56L - Giant Contend SL1	899.99	103.5	2	1592.98

Order Total: 2382.97

Submit Cancel

## 6.5. Manager Portal:

Successfully logged in as a manager: (use username as **HyDu19** and password as **welcome1234**)

Terra Bikes

**TERRABIKEs**

Username: **HyDu19**

Password: .....

Login

Forgot Password Sign Up

- Manager can perform 3 operations:
- Can see all employee that have worked across all the stores
- Can update current employee information and can create new employee
- Have access for analytical dashboard to track sales across all regions, employee performance



These are original employee details. And here I want to update the employee end date, bonus and rating.

Also, from this page we can create a new employee, for that we need to enter all the details of employee. Here manager can create a temporary password for employee which they can change later.

## 7. Summary for Operational Module

### Summary For Operational Module

The operational module of our Bike Store Management System is designed to facilitate seamless day-to-day operations and enhance customer experience within the bike store. It encompasses three primary roles: Customer, Employee, and Manager.

- **Customer Operations:**

- Login and Dashboard: Customers can log in to access a personalized dashboard. The dashboard allows customers to track past and new orders, create new orders, provide feedback, and submit complaints.
- Order Placement: Customers can place new orders by adding up to 5 different products. The system generates a unique order\_ID for efficient tracking.
- Feedback and Complaints: Customers can give feedback and ratings for orders, influencing employee ratings. A streamlined process allows customers to submit complaints, specifying the nature of the issue, product details, and order information.

- **Employee Operations:** Employees can track orders, change order statuses (e.g., ship, delay, delivered), and manage inventory. Inventory management includes searching for products in various warehouses and placing orders to replenish stock.
  - Inventory Management: Employees can monitor and order inventory from different warehouses to maintain optimal stock levels in the store.
  - Grievance Resolution: Employees address and provide resolutions to customer complaints, updating the status as 'resolved' or 'closed.'
  - Customer Management: Employees can view all customers, track their orders over the years, and disable or delete inactive customers.
  - Customer Registration: Employees can assist new walk-in customers by registering them in the system, providing a customer ID, and helping them place new orders.
- **Manager Operations:**
  - Employee Management: Managers can view all employees across regions, track their status, end dates, and order counts. Updating employee information and creating new employees is possible from the manager's perspective.
  - In essence, the operational module ensures efficient coordination among customers, employees, and managers, optimizing order processes, inventory management, and strategic decision-making within the bike store.

## 8. Working of Analytical Module

The Bike Store Management System is a tailored solution designed to streamline operations and enhance manager oversight within our dynamic retail environment. Crafted with a focus on sales trends and employee performance, the system caters to manager, offering unique functionalities to optimize business processes.

- **Sales Trends Overview:** At the managerial level, our system provides a comprehensive "Sales Trends Overview Dashboard." This dashboard empowers managers to gain insights into sales trends across diverse regions over multiple years. Key performance indicators (KPIs) are prominently featured, enabling a quick assessment of overall sales performance. The system generates detailed analysis on regional sales, facilitating strategic decision-making.
- **Employee Performance Tracking:** The managerial privilege extends to a specialized employee performance dashboard, providing key metrics for assessing workforce effectiveness. Managers can recognize the top 10 employees who have made significant sales contributions. This analytical capability empowers managers to make data-driven decisions, strategize for growth, and optimize operational efficiency within the dynamic landscape of the bike store.

Furthermore, we can ROLL UP the results on a monthly, quarterly, and weekly basis depending on the trend manager wants to check.

In essence, our Bike Store Management System is designed to empower managers with a holistic view of sales trends and employee performance. The dual dashboards cater to the unique needs of different managerial levels, ensuring seamless and effective management within the bike store.

## 9. Specifications and Usability of Analytical Module

### 9.1. Sales Dashboard

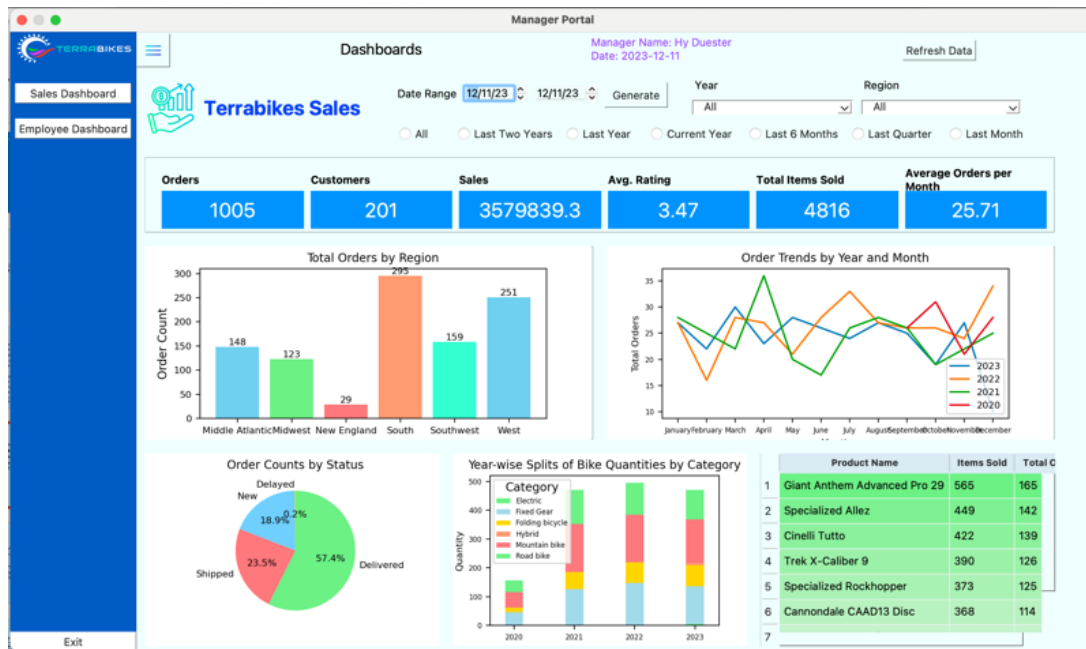


FIGURE 30: SALES DASHBOARD

- “Sales dashboard”: - This is Sales Dashboard page where manager views this page. It displays the overall data in organized and graphical ways.
- Functionalities: - Manager can select a date range from and through to see the sales by Month-Year. To display the sales data in Month-Year we need to use the concept of DRILL-UP.
- Manager can see total number of orders generated by a region and manager can also filter a particular region as well.
- We can see the top 10 products that were sold in that year and region.
- You can also see how many orders are in which status.
- Also, you can analyse the year wise split of bike quantities by category.

Through radio buttons we can drill up or drill down over years to see the sales performance. And, through region dropdown we can slice and dice for the region-specific sales trend.

## 9.2. Employee Dashboard

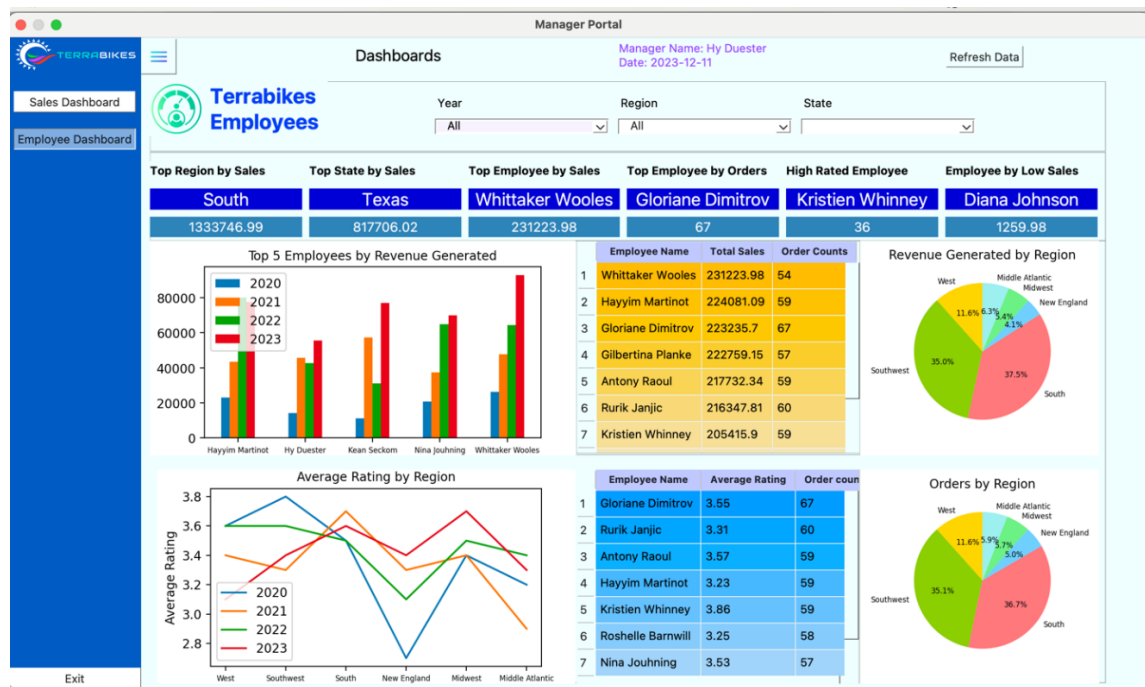


FIGURE 31: EMPLOYEE DASHBOARD

- “Employee dashboard”: - This is Employee Dashboard page where manager views can view employee performance over years for different regions.
- Functionalities: - Manager can select a date range from and through to see the employee performance by Month-Year. To display the employee data in Month-Year we need to use the concept of DRILL-UP.
- Manager can see total number of orders generated by a region and manager can also filter a particular region as well.
- You can also see orders and revenue generated by region.
- Manager can also see the average rating over years in different regions.
- We can see the top 10 employees for that region.
- manager can see over years how much revenue top 5 employees are generating.

## 10. Summary of Analytical Module

The Analytical Module of the Bike Store Management System offers a comprehensive and strategic approach to overseeing the store's performance. It empowers managers with a holistic view of sales trends and employee performance.

This module features a Sales Dashboard that provides insights into sales trends across different regions over multiple years. Allowing managers to quickly assess overall sales performance. The system generates detailed analyses on regional sales, facilitating strategic decision-making. Managers can also track employee performance through a employee dashboard, recognizing the top contributors and making data-driven decisions for growth and operational efficiency.

Furthermore, the system allows for rolling up results on a monthly, quarterly, and weekly basis, providing flexibility in trend analysis based on manager’s preferences. In essence, the Analytical

Module serves as a valuable tool for managers, offering dual dashboards to optimize decision-making processes and enhance overall management within the dynamic landscape of the bike store.

## 11. Technical Aspects

Following is the list of the files for this application.

S.No.	File Name	Type	Comments
1	BikeStoreCustMainDialog.py	Python	Customer Portal Dialog
2	BikeStoreCustMainDialog.ui	Qt5 UI File	Customer Portal Dialog
3	BikeStoreCustMainDialog_ui.py	Python	Customer Portal Dialog
4	BikeStoreEmplMainDialog.py	Python	Employee Portal Dialog
5	BikeStoreEmplMainDialog.ui	Qt5 UI File	Employee Portal Dialog
6	BikeStoreEmplMainDialog_ui.py	Python	Employee Portal Dialog
7	BikeStoreMainWin.py	Python	Main Window (Login Page)
8	BikeStoreMainWin.ui	Qt5 UI File	Main Window (Login Page)
9	BikeStoreMainWin_ui.py	Python	Main Window (Login Page)
10	BikeStoreManagerDashDialog.py	Python	Manager Dashboard Dialog
11	BikeStoreManagerDashDialog.ui	Qt5 UI File	Manager Dashboard Dialog
12	BikeStoreManagerDashDialog_ui.py	Python	Manager Dashboard Dialog
13	BikeStoreManagerMainDialog.py	Python	Manager Portal Dialog
14	BikeStoreManagerMainDialog.ui	Qt5 UI File	Manager Portal Dialog
15	BikeStoreManagerMainDialog_ui.py	Python	Manager Portal Dialog
16	BikeStoreResetPwdDialog.py	Python	Reset Password Dialog
17	BikeStoreResetPwdDialog.ui	Qt5 UI File	Reset Password Dialog
18	BikeStoreResetPwdDialog_ui.py	Python	Reset Password Dialog
19	BikeStoreSignUpDialog.py	Python	Sign Up Dialog
20	BikeStoreSignUpDialog.ui	Qt5 UI File	Sign Up Dialog
21	BikeStoreSignUpDialog_ui.py	Python	Sign Up Dialog
22	BikeStoreUtils.py	Python	Utility File used in Application
23	TerraBikes.ipynb	Jupyter Notebook	Notebook used to Launch the Application
24	resources.qrc	Resource File	QT Resources File
25	resources_rc.py	Python	QT Resources Python File
26	terrabikes.ini	Config File	Database Config File (Operational)
27	terrabikes_bi.ini	Config File	Database Config File (Analytical)
28	TerraBikes.py	Python	Alternate way to Launch
29	terrabikes.sql	SQL Dump	Operational DB Dump
30	terrabikes_bi.sql	SQL Dump	Analytical DB Dump

## 12. Database Technical Details

### 12.1. DB Objects

Database Type	Object Type	Name	Usage
Operational	Table	Brand	Employee Portal: Brand Details
Operational	Table	Category	Employee Portal: Category Details
Operational	Table	Customer	Customer Portal: Customer Details
Operational	Table	Employee	Employee Portal: Employee Details
Operational	Table	Feedback	Employee Portal: Feedback & Grievances Customer Portal: Feedback & Grievances

Operational	Table	Manager	Manager Portal: Manager Details
Operational	Table	Orders	Customer Portal: Order Records
Operational	Table	Order_Details	Customer Portal: Order Details
Operational	Table	Products	Customer Portal: Product Details Employee Portal: Product and Inventory Details
Operational	Table	Regions	Region Information
Operational	Table	Type	User Type
Operational	Table	Users	User Login Information
Operational	Table	Warehouse	Employee Portal: Warehouse Details
Operational	Table	Warehouse_inventory	Employee Portal: Warehouse Inventory
Operational	Table	Warehouse Supply	Employee Portal: Warehouse Supply Orders
Operational	Function	create_employee	Employee Portal: Insert Employee Records
Operational	Function	gen_rand_empid	Employee Portal: Generate Random Employee ID
Operational	Function	InsertCustomerAndUser	Employee Portal: Insert Customer and User Details
Operational	Function	InsertFeedback	Employee Portal: Insert Feedback
Operational	Function	InsertOrder	Customer Portal: Insert Order Records
Operational	Function	InsertOrderDetails	Customer Portal: Insert Order Detail Records
Operational	Function	InsertWarehouseInventory	Employee Portal: Insert Warehouse Inventory
Operational	Function	InsertWarehouseSupply	Employee Portal: Insert Warehouse Supply Orders
Operational	Function	Update_employee	Manager Portal: Update Employee Records
Operational	Function	UpdateProductInventory	Employee Portal: Update Inventory Status
Analytical	Procedure	Refresh_dwh_prc	Stored Procedure to Refresh Analytical DB from Operation DB
Analytical	Table	Calendar	Time Dimensions
Analytical	Table	Customer	Customer Dimension Table
Analytical	Table	Employee	Employee Dimension Table
Analytical	Table	Order_Details	Order Dimension Table
Analytical	Table	Performance	Employee Performance Fact Table
Analytical	Table	Product	Product Dimension Table
Analytical	Table	Region	Region Dimension Table
Analytical	Table	Sales	Sales Fact Table
Analytical	View	Employee_sales_view	View for Employee Dashboard

## 12.2. ETL

We created and generated the data from Mockaroo and combined the dataset from Kaggle and transformed the generated data such that it is in accordance with the application logic.

- Operational:** Initial Data-load was done using Python and SQL Workbench. The date columns of the database were formatted to YYYY-MM-DD to have consistency in the entire application. Next, we concatenated the fields such as street name, city, zipcode together to get the address field. Once all transformations are done, we loaded the data to SQL workbench by directly querying the workbench as below:

Table: Customer	Query: <pre>INSERT INTO Customer (customer_id, first_name, last_name, email_id, contact, address, creation_date, updated_date) VALUES (1, 'Gibb', 'MacCumiskey', 'gmaccumiskey0@sphinn.com', '214-164-9758', '008 Ilene Terrace, Corpus Christi, Texas 78426', '2022-12-09', '2022-03-03');</pre>
-----------------	--



- **Analytical:** We wrote a stored procedure to load data of operational database to analytical database. The SP pulls the data from operational and loads the analytical db.

The stored procedure is as below:

```
CREATE DEFINER='root'@'localhost' PROCEDURE `refresh_dwh_prc`(OUT o_status VARCHAR(20))
BEGIN
    SET FOREIGN_KEY_CHECKS = 0;
    TRUNCATE TABLE Sales;
    TRUNCATE TABLE Performance;
    TRUNCATE TABLE Customer;
    TRUNCATE TABLE Region;
    TRUNCATE TABLE Product;
    TRUNCATE TABLE Calendar;
    TRUNCATE TABLE Order_Details;
    TRUNCATE TABLE Employee;
    SET FOREIGN_KEY_CHECKS = 1;

    -- Load Products Table
    INSERT INTO Product( Product_Key, Product_ID, Price, discount_percent, Model_Year, Product_Name, Brand_Name,
    Category_Name, Inventory_Status)
    SELECT NULL product_key, p.product_id, p.price, p.discount_percent, p.model_year, p.product_name,
    b.brand_name, c.category_name, p.inventory_status FROM terrabikes.products p, terrabikes.brand b, terrabikes.category c
    WHERE b.brand_id = p.brand_id AND c.category_id = p.category_id;
    -- Load Calendar Table
    INSERT INTO calendar ( Calendar_Key, Full_Date, Year, Month, Qtr, day_of_month, Week)
    SELECT DISTINCT NULL, ordered_date, YEAR(ordered_date), MONTH(ordered_date), QUARTER(ordered_date),
    DAY(ordered_date), WEEK(ordered_date) FROM terrabikes.orders;
    -- Load Region Table
    INSERT INTO region(Region_Key, Region, State)
    SELECT DISTINCT NULL, region, state_name FROM terrabikes.regions;
    -- Load Employee Table
    INSERT INTO Employee( Employee_Key, Employee_ID, Employee_Name, Employee_Rating)
    SELECT NULL, employee_id, CONCAT(first_name, ' ', last_name) employee_name, emp_rating FROM
    terrabikes.employee;
    -- Load Customer Table
    INSERT INTO Customer( Customer_Key, customer_id, Customer_Name)
    SELECT NULL, customer_id, CONCAT(first_name, ' ', last_name) customer_name FROM terrabikes.customer;
    -- Load Order Details Table
    INSERT INTO Order_Details( Order_Detail_Key, Order_Id, Order_Detail_ID, Order_Date, order_status, Issue_category,
    Rating, status) SELECT NULL, o.order_id, od.order_detail_id, o.ordered_date, o.order_status, (SELECT f.grevience_category FROM
    terrabikes.feedback f WHERE order_id = o.order_id AND record_type = 'GREVIANCE' ) Issue_category, (SELECT f.rating FROM
    terrabikes.feedback f WHERE order_id = o.order_id AND record_type = 'FEEDBACK' ) rating,
    (SELECT f.status FROM terrabikes.feedback f WHERE order_id = o.order_id AND record_type = 'GREVIANCE' ) status
    FROM terrabikes.orders o JOIN terrabikes.order_details od ON od.order_id = o.order_id ORDER BY od.order_detail_id;
    -- Load Sales Table
    INSERT INTO Sales ( Product_Key, Calendar_Key, Order_Detail_Key, Region_Key, Customer_Key, Price, qty)
    SELECT p.product_key, cd.calendar_key, od.order_detail_key, r.region_key, c.customer_key, todd.price,
    todd.quantity FROM terrabikes.orders tod, terrabikes.order_details todd, terrabikes.customer tc, terrabikes.regions tr, customer c,
    region r, product p, order_details od, calendar cd WHERE tod.order_id = todd.order_id AND tc.customer_id = tod.customer_id
    AND tr.region_id = tc.region_id AND c.customer_id = tc.customer_id AND r.state = tr.state_name AND r.region = tr.region AND
    p.product_id = todd.product_id AND od.order_detail_id = todd.order_detail_id AND cd.full_date = od.order_date;
    -- Load Performance Table
    INSERT INTO Performance ( Calendar_Key, Employee_Key, Region_Key, Order_Detail_Key, employee_ratings)
    SELECT c.calendar_key, e.employee_key, r.region_key, od.order_detail_key, te.emp_rating FROM
    terrabikes.orders tod, terrabikes.order_details todd, terrabikes.employee te, terrabikes.regions tre, calendar c,
    employee e, region r, order_details od WHERE tod.order_id = todd.order_id AND te.employee_id = tod.employee_id
    AND tre.region_id = te.region_id AND c.full_date = tod.ordered_date AND e.employee_id = tod.employee_id AND r.state =
    tre.state_name AND r.region = tre.region AND od.order_detail_id = todd.order_detail_id;
    SET o_status = 'Success';
END
```

## Queries (Operation DB)

Usage	Query
User Information	<pre>select t.role from users u, type t</pre>

	<pre> where upper(u.username) = upper(%s) and u.pwd = md5(%s) and t.user_role_id = u.user_role_id </pre>
Fetch User Email	<pre> select username, c.email_id, e.employee_id from users u left join customer c on c.customer_id = u.customer_id left join employee e on e.employee_id = u.employee_id where upper(u.username) = upper(%s) </pre>
Update Passwords	<pre> update users set pwd = md5(%s) where upper(username) = upper(%s) </pre>
Customer Name	<pre> select concat(first_name, ' ', c.last_name) from users u, customer c where c.customer_id = u.customer_id and upper(u.username) = upper(%s) </pre>
Order Information	<pre> select o.order_id, o.order_status, o.ordered_date, o.shipped_date, concat(e.first_name, ' ', e.last_name) as "Sales Rep" from orders o, employee e, customer c, users u where upper(u.username) = upper(%s) AND u.customer_id = c.customer_id AND o.employee_id = e.employee_id AND o.customer_id = c.customer_id; </pre>
Product Information	<pre> select p.product_id, p.product_name, p.price as "unit price", od.quantity, (p.price*od.quantity) as "gross price", od.discount, CASE WHEN od.discount is not null THEN round(((p.price*od.quantity) - od.discount),2) ELSE round((p.price*od.quantity),2) END as "net price" from orders o, order_details od, products p where o.order_id = od.order_id and p.product_id = od.product_id and o.order_id = %s; </pre>
Product Discount	<pre> select price, quantity, discount_percent from products where product_id = %s </pre>
Order Drop Down	<pre> select concat('Order: ', order_id, ' Status: ', order_status, ' Ordered Date: ', ordered_date) from orders o, users u where o.customer_id = u.customer_id and upper(u.username) = upper(%s) </pre>
Grievance Details	<pre> select f.order_id, f.grievance_category, f.status, f.creation_date, f.comments, f.emp_comments from feedback f, customer c, users u where f.record_type = 'GREVIANCE' and c.customer_id = f.customer_id and c.customer_id = u.customer_id and upper(u.username) = upper(%s) </pre>
Inventory Information	<pre> select p.product_id, p.product_name, c.category_name, b.brand_name, p.model_year, p.price, p.discount_percent, p.quantity, p.inventory_status from products p, category c, brand b where p.brand_id = b.brand_id and c.category_id = p.category_id </pre>
Warehouse Information	<pre> select wi.product_id, w.warehouse_id, w.warehouse_name, wi.quantity, w.address, w.contact, wi.status, wi.warehouse_inv_id from warehouse w, warehouse_inventory wi where w.warehouse_id = wi.warehouse_id and wi.product_id = %s </pre>

Warehouse Orders	<pre> select ws.supply_order_id, wi.product_id, w.warehouse_name, ws.need_by_date, ws.qty_ordered, ws.shipment_status,        ws.creation_date from warehouse w, warehouse_inventory wi, warehouse_supply ws where w.warehouse_id = wi.warehouse_id and wi.warehouse_inv_id = ws.warehouse_inv_id and wi.product_id = %s and w.warehouse_id = %s </pre>
Customer Details	<pre> select customer_id, concat(first_name, ' ', last_name) customer_name, status, end_date,        (select count(order_id) from orders where customer_id = c.customer_id) order_count,        contact, email_id, address, city, state_name, region from customer c, regions r where c.region_id = r.region_id </pre>
Employee List	<pre> select e.employee_id, concat(e.first_name, ' ', e.last_name) as Employee_Name, e.status,        e.end_date, r.region, (select count(order_id) from orders where employee_id = e.employee_id) as order_count from users u, manager m, employee e, regions r where u.username = %s and u.employee_id = m.manager_emp_id and e.region_id = r.region_id and e.manager_id = m.manager_id; </pre>
Employee Details	<pre> select e.employee_id, e.email_id, e.contact, e.address, r.state_name, r.city, e.emp_rating,        e.bonus, e.salary from employee e, regions r where e.region_id = r.region_id and e.employee_id = %s </pre>
Employee Page	<pre> select concat(e.first_name, ' ', e.last_name) as Name , TRIM(SUBSTRING_INDEX(e.address, ' ', 1)) address, r.state_name, r.city, TRIM(SUBSTRING_INDEX(e.address, ' ', -1)) as Postal_code , e.contact, e.start_date, e.end_date, e.Salary, e.email_id, e.emp_rating, e.bonus, e.leaves_taken, u.username from employee e, regions r, users u where e.region_id = r.region_id and e.employee_id = u.employee_id </pre>

### 12.3. Queries (Analytical DB)

Usage	Query
Summary Labels (Sales Dashboard)	<pre> SELECT COUNT(DISTINCT od.order_id) order_count, COUNT(DISTINCT s.customer_key) customer_count, SUM(s.qty) total_items, ROUND(SUM(s.price), 2) total_sales, ROUND(AVG(od.rating), 2) average_rating, ROUND((COUNT(DISTINCT od.order_id) / ((DATEDIFF(MAX(order_date), MIN(order_date)) / 365) * 12)), 2) avg_orders </pre>

	<pre> FROM     sales s,     order_details od WHERE     s.order_detail_key = od.order_detail_key </pre>
Region Orders (Sales Dashboard)	<pre> select r.region, count(distinct od.order_id)     from sales s, region r, order_details od     where s.region_key = r.region_key     and od.order_detail_key = s.order_detail_key group by r.region     order by 1 </pre>
Order Status Counts (Sales Dashboard)	<pre> select count(*), order_status from order_details where 1=1 group by order_status     order by 1 </pre>
Order Counts by Year (Sales Dashboard)	<pre> select cd.year, monthname(cd.full_date), count(distinct od.order_id), cd.month     from sales s, calendar cd, order_details od     where s.calendar_key = cd.calendar_key     and od.order_detail_key = s.order_detail_key group by cd.year, monthname(cd.full_date), cd.month     order by 1 desc,4 </pre>
Quantities By Category (Sales Dashboard)	<pre> select c.year, p.category_name, count(s.qty)     from product p, sales s, order_details o, calendar c     where s.product_key = p.product_key     and o.order_detail_key = s.order_detail_key     and c.calendar_key = s.calendar_key group by p.category_name, c.year     order by 1 desc, 2 </pre>
Top Products (Sales Dashboard)	<pre> select p.product_name, sum(qty) items_sold, count(distinct o.order_id) orders     from sales s, product p, calendar cd, order_details o     where s.product_key = p.product_key     and cd.calendar_key = s.calendar_key and o.order_detail_key = s.order_detail_key group by p.product_name     order by 3 desc     limit 10 </pre>
Top Employee By Sales (Employee Dashboard)	<pre> with top_emp as (select employee_id, round(sum(price),2)     from employee_sales_view     where 1=1     and year = (select max(year) from calendar) group by employee_id     order by 2 desc limit 5) select ev.year, ev.employee_id, ev.employee_name,     round(sum(ev.price),2) as Revenue_generated, count(distinct ev.order_id) as count_of_orders     from employee_sales_view ev, top_emp te     where ev.employee_id = te.employee_id group by year, employee_id, employee_name     order by 1,4 desc; </pre>
Avg. Rating by Employee (Employee Dashboard)	<pre> select year, region, round(avg(rating),1)     from employee_sales_view     where 1=1 </pre>

	group by year, region order by 1,2 desc
Top 10 Employees (Employee Dashboard)	select employee_name, round(sum(price),2), count(distinct order_id) from employee_sales_view where 1=1 group by employee_name order by 2 desc limit 10
Employee Performance by Region (Employee Dashboard)	select region, round(sum(price),2) as Revenue_generated, COUNT(DISTINCT order_id) as count_of_orders from employee_sales_view where 1=1 group by region order by 1 desc;
Summary Labels (Employee Dashboard)	select region, count(DISTINCT order_id) as order_count, round(sum(price),2) as Revenue_generated from employee_sales_view where 1=1 group by region order by 3 desc limit 1 select state, count(DISTINCT order_id) as order_count, round(sum(price),2) as Revenue_generated from employee_sales_view where 1=1 group by state order by 3 desc limit 1 select employee_name, round(sum(price),2) as Revenue_generated from employee_sales_view where 1=1 group by employee_name order by 2 desc limit 1 select employee_name, count(distinct order_id) as count_of_orders from employee_sales_view where 1=1 group by employee_name order by 2 desc limit 1 select employee_name, count(distinct order_ID) AS TOTAL_ORDERS from employee_sales_view where rating >= 4 group by employee_name order by 2 desc LIMIT 1 select employee_name, round(sum(price),2) as Revenue_generated from employee_sales_view where 1=1 group by employee_name order by 2 asc limit 1