राष्ट्रीय प्रौद्योगिकी संस्थान दिल्ली
National Institute of Technology Delhi
(An autonomous Institute under the aegis of Ministry of HRD, Govt. of India)

DBMS PROJECT

BIZ MANAGER

GROUP MEMBERS:
ADITYA SINGH(201210004)
HIMANSHU JAIN(201210022)
ORUGANTI BHARGHAV(201210032)

ACKNOWLEDGEMENT

It is our privilege to express our sincerest regards to our course instructor for the course of **DATABASE MANAGEMENT SYSTEM, Dr. Shelly Sachdeva,** for their valuable inputs, able guidance, encouragement, whole-hearted cooperation and constructive criticism throughout the duration of our project.

We deeply express our sincere thanks to our teaching assistants, **Ms. Kanika Soni and Mr. Anil kumar Patel**, for helping and guiding us throughout till we present our project on the topic " **Biz Manager** ". Last but not the least we express our thanks to our friends for their cooperation and support.

CONTENTS

CASE STUDY

As we all know, all types of shopkeepers always have a list of tasks to do in the shop. For each different task, shopkeeper needs different variety of objects, people and sometimes even have to go to different places.

Bill Generation, keeping track of products, sales, sales record, and many more are such a tedious tasks that is not even imaginable to be done by a single person.

And for small scale businessman, it is even difficult to appoint a person for these works. As a result, shopkeeper has to bear great loss. Usually, the growth of small-scale businesses is also not done due to these problems. And finally, these business got shut down.

Hence, we are trying to form a one-stop solution in the form of an application and website for all types of shopkeepers where they can manage their businesses easily, effectively and in less time without any charges.

INTRODUCTION

In “Biz Manager”, we have tried to make a system which contains all the information about different variety of product’s details, supplier’s details, customer details, bill records, sales record, profit and growth report by using mainly the concepts of Database Management System.

There has been many different products in a shop which requires different product id, name, supplier details, prices, quantity, expiry date, etc. So, to make it simple, the app came with a personalised and user-friendly feature of filling these details only for the first time. Then, system will automatically do the respective changes

It also comes with an automated profit and growth generation by adding data only in sales section.

We have tried to make user able to view the business growth, sales and profit on weekly as well as monthly basis as per their choice.

The very unique feature, we tried to create is to make user able to see the top 3 products selling in the shop on monthly basis, to make user able to invest properly and grow effectively.

This case study aims to design and develop an efficient database for the business owners to maintain the records of sales, products, profit, growth and bills. Along with this, it also highlights the top selling products to seller. Hence getting rid of manual system and data redundancy which we face in the manual record.

REQUIREMENTS

SOFTWARE REQUIREMENTS

- Visual Studio Code
- MySQL Workbench 8.0
- Web Browser (Chrome/Edge/Firefox etc.)
- Node.js

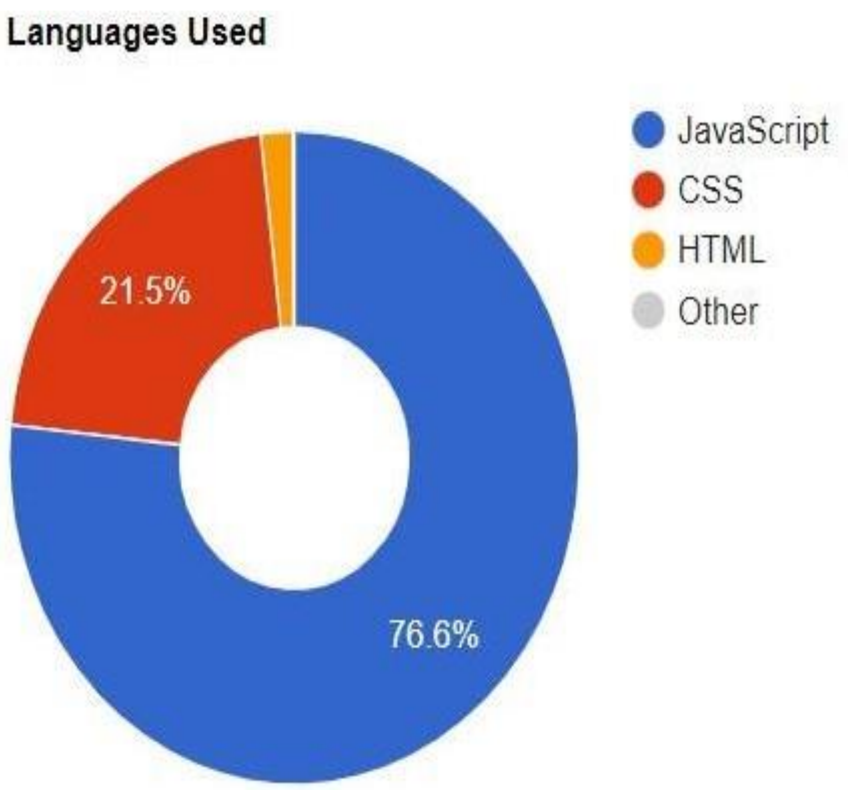
LANGUAGE REQUIREMENTS

- React Js
- Microsoft SQL
- CSS
- HTML
- Javascript

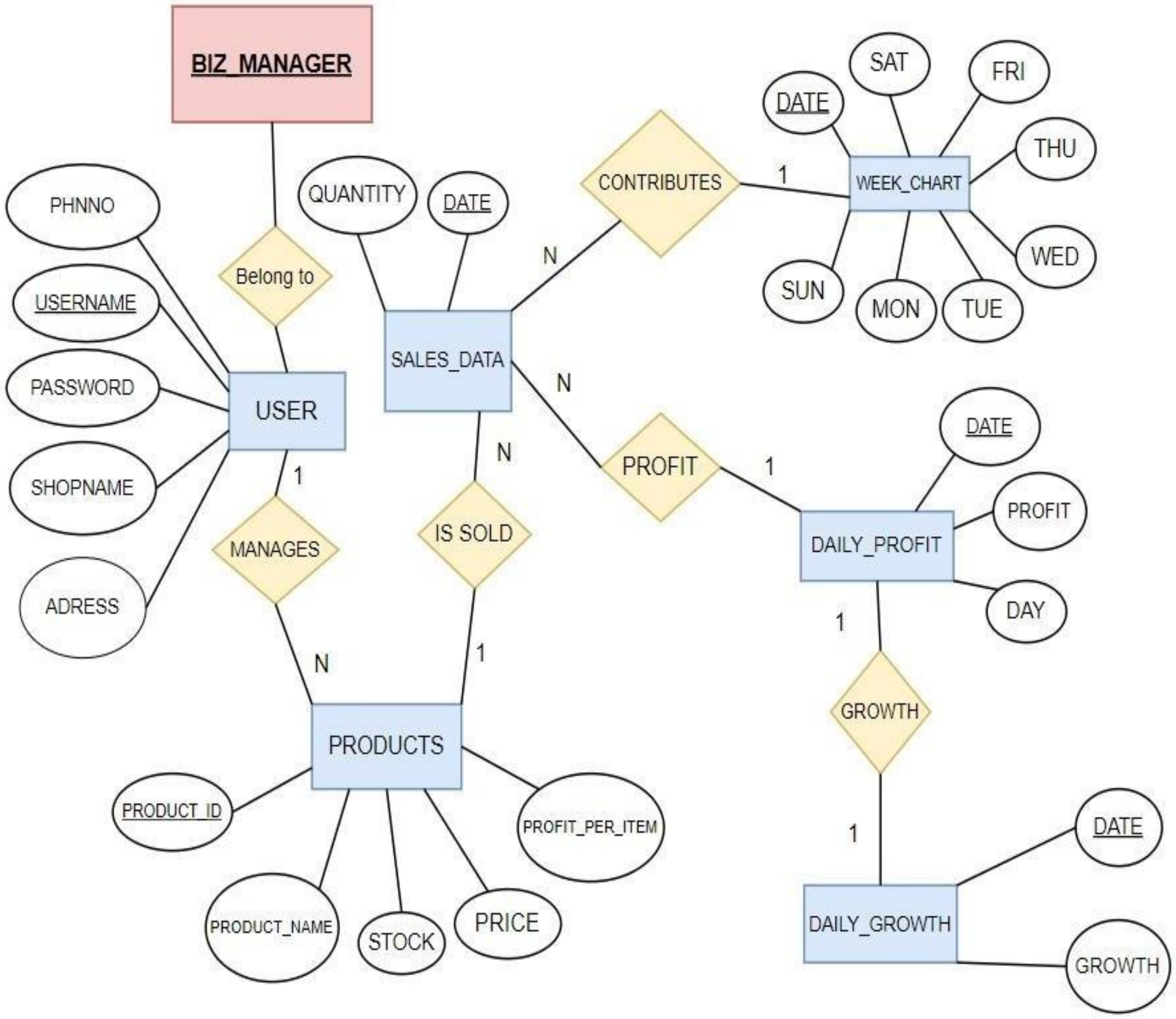
HARDWARE REQUIREMENTS

- Operating System: Windows 7/8/8.1/10.
- Memory (RAM): 1 GB of RAM required.
- Hard Disk Space: 100 MB of free space required.
- Processor: Intel Pentium 4 or later.
- Cache: 512 KB

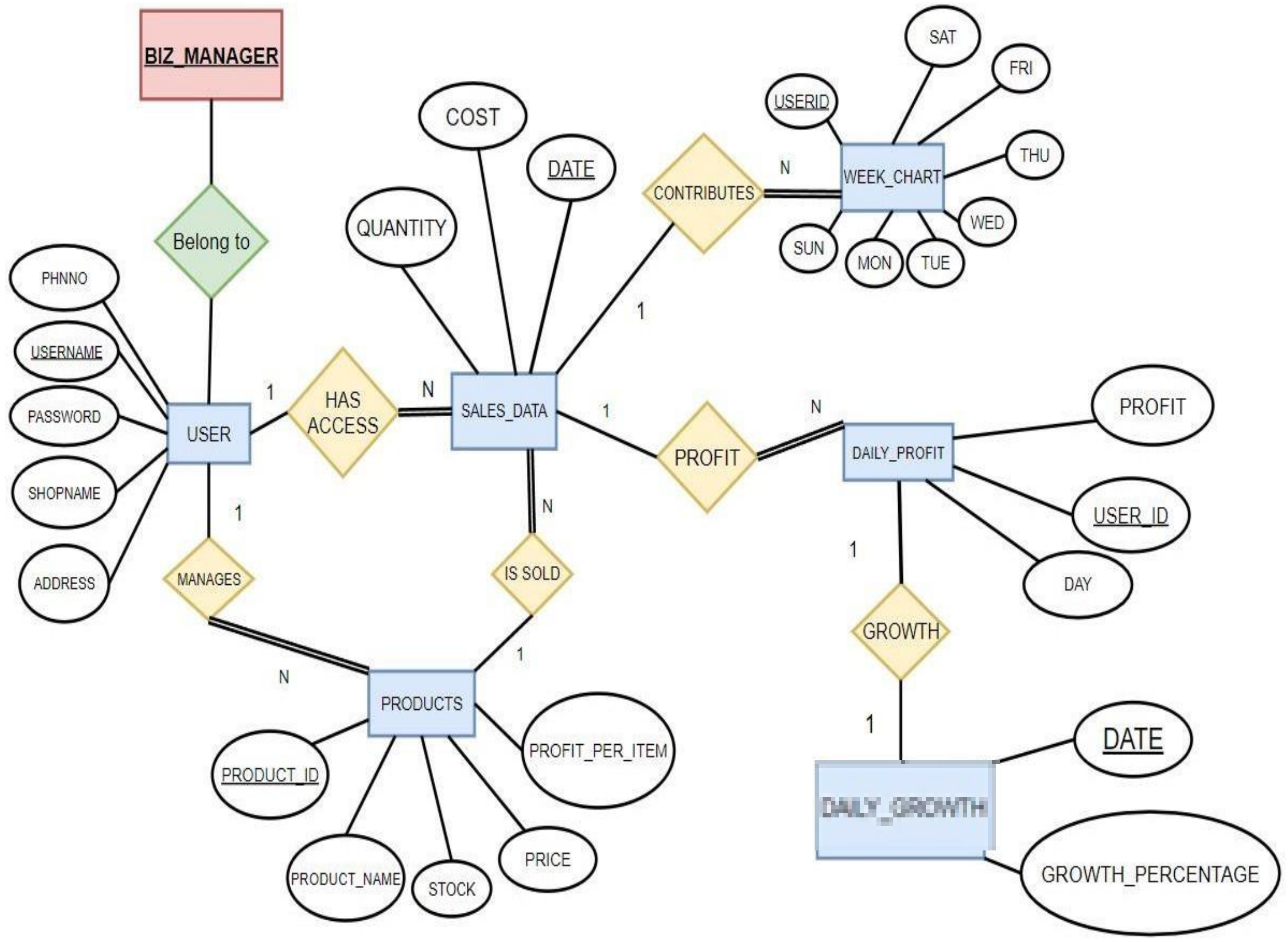
CONTINUED..



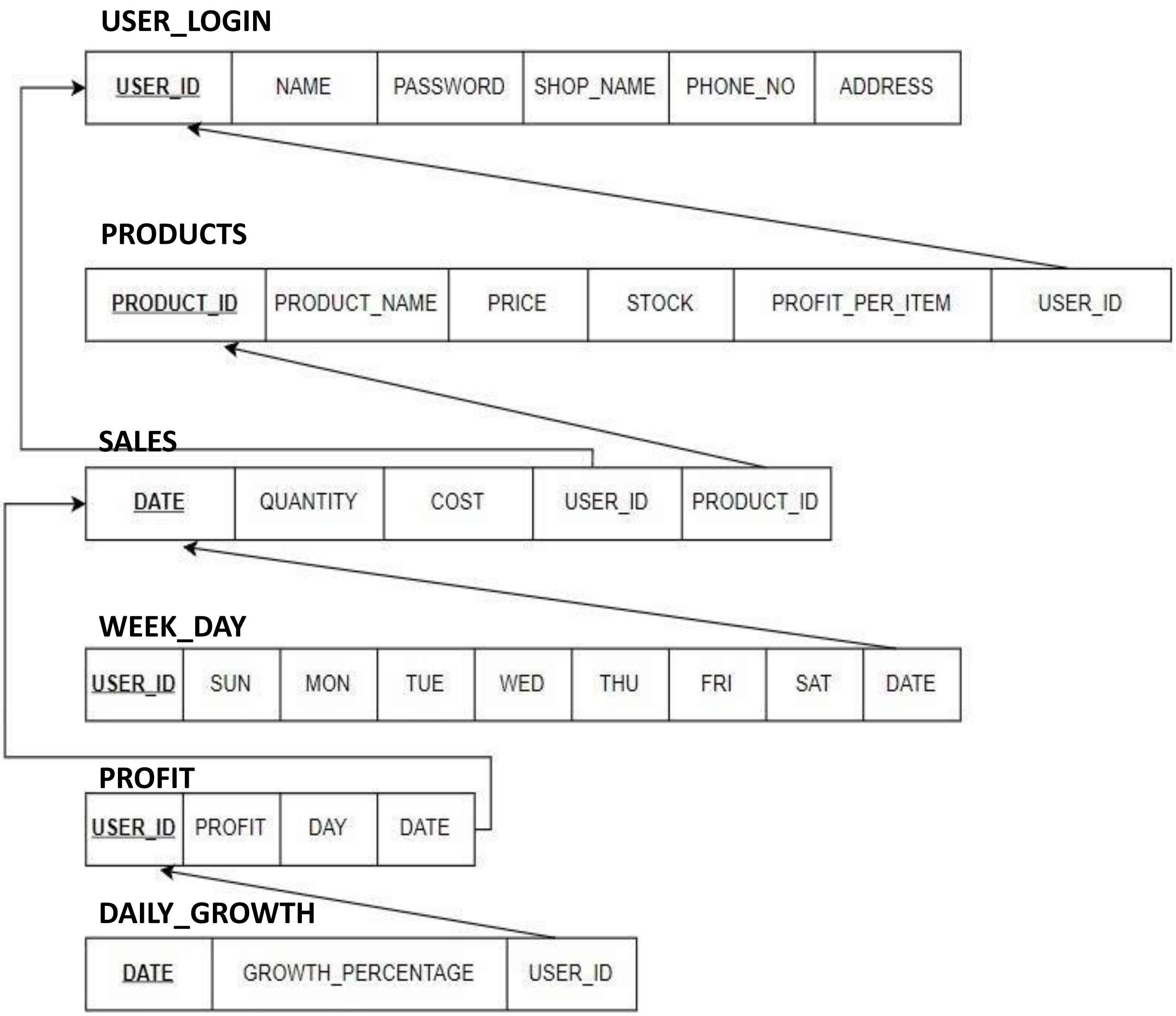
ER DIAGRAM (PREVIOUS)



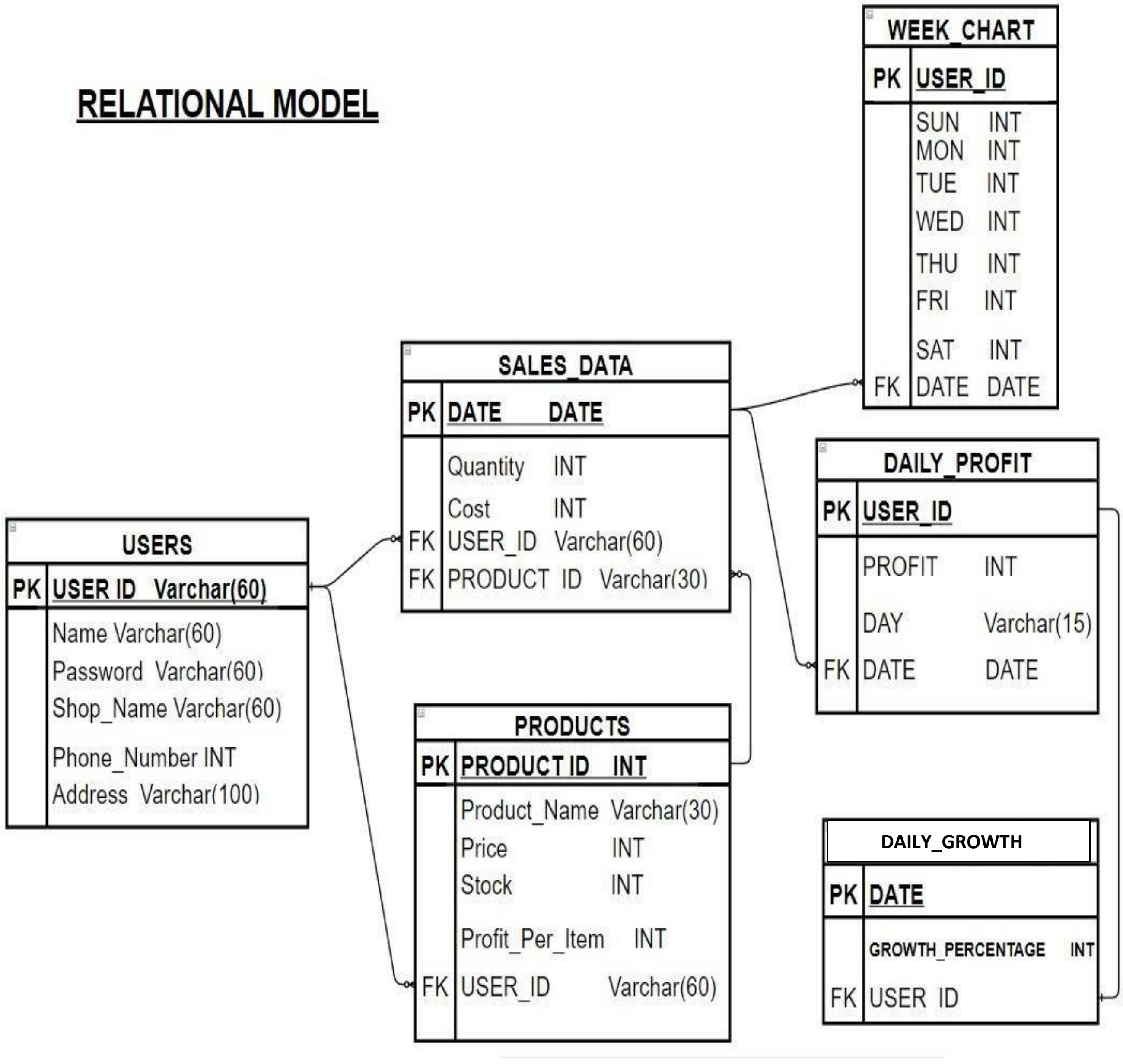
ER DIAGRAM (UPDATED)



MAPPING FROM ER MODEL TO R MODEL

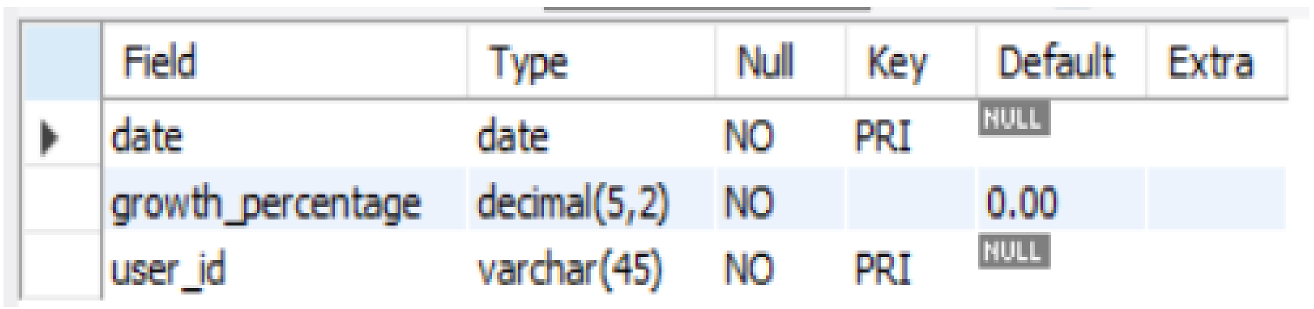

USER_LOGIN
USER_ID | NAME | PASSWORD | SHOP_NAME | PHONE_NO | ADDRESS
PRODUCTS
PRODUCT_ID | PRODUCT_NAME | PRICE | STOCK | PROFIT_PER_ITEM | USER_ID
SALES
DATE | QUANTITY | COST | USER_ID | PRODUCT_ID
WEEK_DAY
USER_ID | SUN | MON | TUE | WED | THU | FRI | SAT | DATE
PROFIT
USER_ID | PROFIT | DAY | DATE
DAILY_GROWTH
DATE | GROWTH_PERCENTAGE | USER_ID

RELATIONAL SCHEMA



DATA DICTIONARY

GROWTH TABLE



	Field	Type	Null	Key	Default	Extra
▶	date	date	NO	PRI	NULL	
	growth_percentage	decimal(5,2)	NO		0.00	
	user_id	varchar(45)	NO	PRI	NULL	

INDEXES

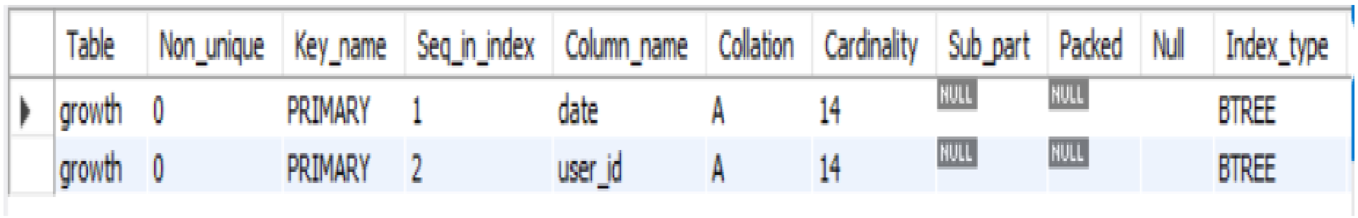
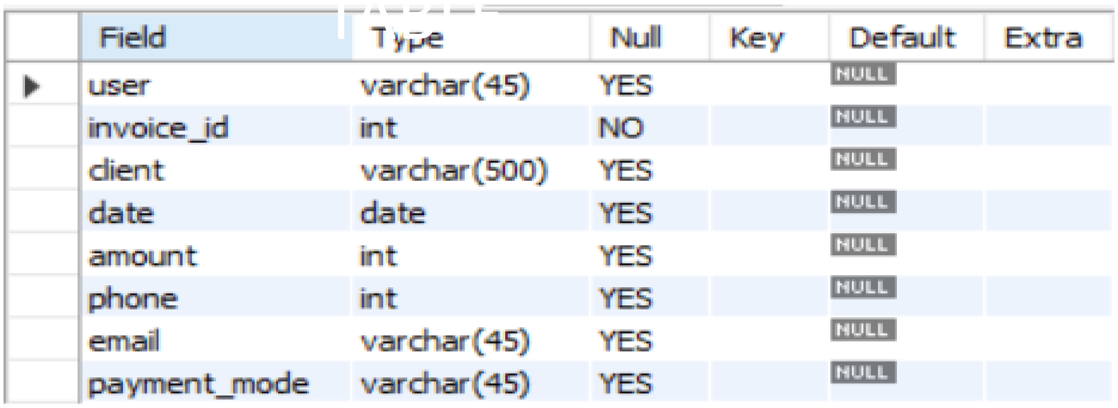


	Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
▶	growth	0	PRIMARY	1	date	A	14	NULL	NULL		BTREE
	growth	0	PRIMARY	2	user_id	A	14	NULL	NULL		BTREE

INVOICE



	Field	Type	Null	Key	Default	Extra
▶	user	varchar(45)	YES		NULL	
	invoice_id	int	NO		NULL	
	client	varchar(500)	YES		NULL	
	date	date	YES		NULL	
	amount	int	YES		NULL	
	phone	int	YES		NULL	
	email	varchar(45)	YES		NULL	
	payment_mode	varchar(45)	YES		NULL	

INDEXES

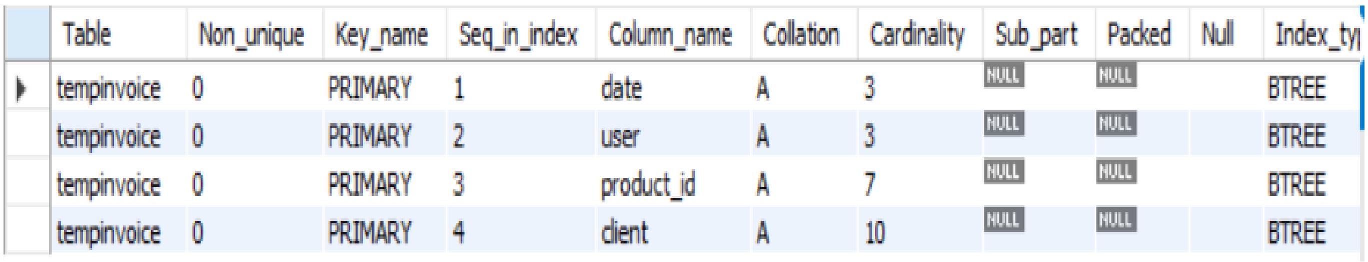
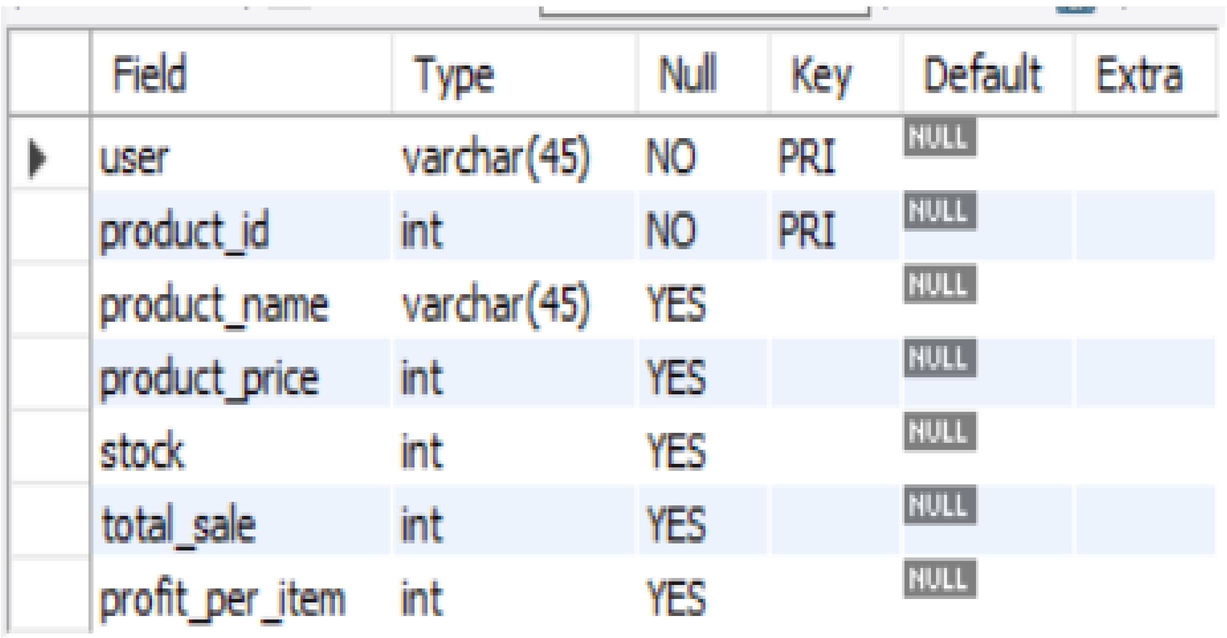


Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_ty
tempinvoice	0	PRIMARY	1	date	A	3	NULL	NULL		BTREE
tempinvoice	0	PRIMARY	2	user	A	3	NULL	NULL		BTREE
tempinvoice	0	PRIMARY	3	product_id	A	7	NULL	NULL		BTREE
tempinvoice	0	PRIMARY	4	client	A	10	NULL	NULL		BTREE

PRODUCT TABLE



Field	Type	Null	Key	Default	Extra
user	varchar(45)	NO	PRI	NULL	
product_id	int	NO	PRI	NULL	
product_name	varchar(45)	YES		NULL	
product_price	int	YES		NULL	
stock	int	YES		NULL	
total_sale	int	YES		NULL	
profit_per_item	int	YES		NULL	

INDEXES

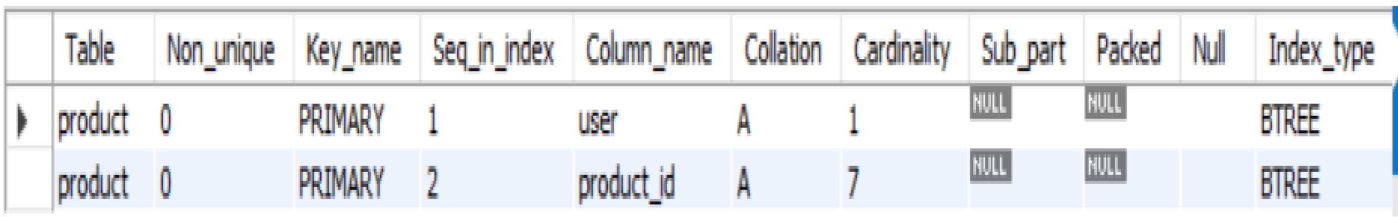
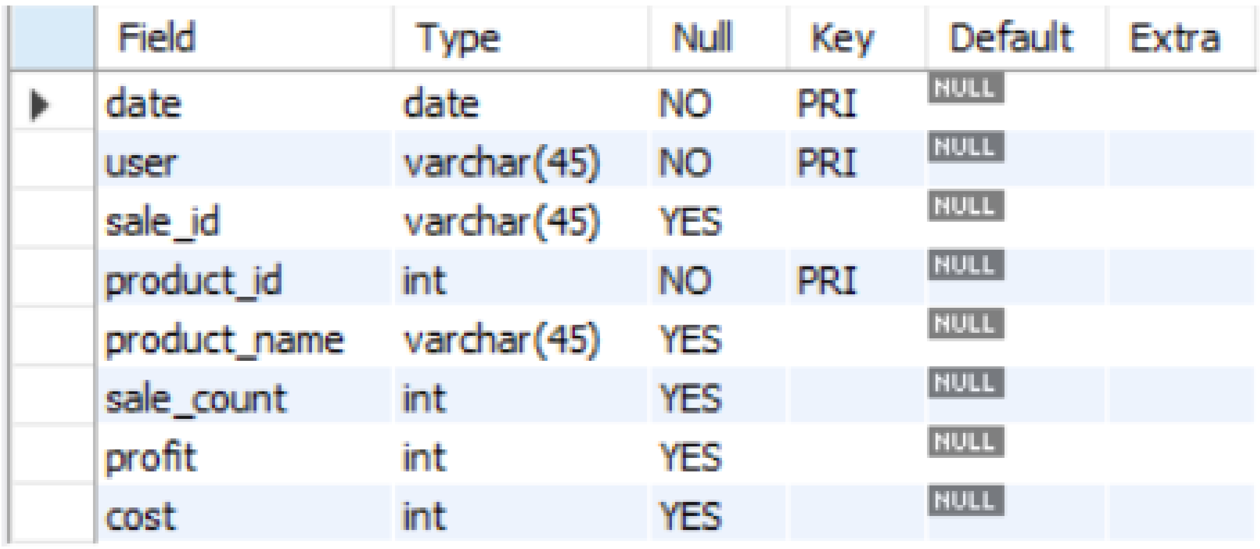


Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
product	0	PRIMARY	1	user	A	1	NULL	NULL		BTREE
product	0	PRIMARY	2	product_id	A	7	NULL	NULL		BTREE

SALES TABLE



Field	Type	Null	Key	Default	Extra
date	date	NO	PRI	NULL	
user	varchar(45)	NO	PRI	NULL	
sale_id	varchar(45)	YES		NULL	
product_id	int	NO	PRI	NULL	
product_name	varchar(45)	YES		NULL	
sale_count	int	YES		NULL	
profit	int	YES		NULL	
cost	int	YES		NULL	

INDEXES

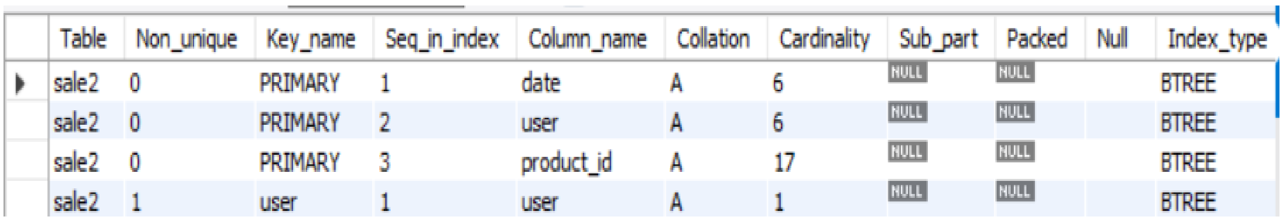
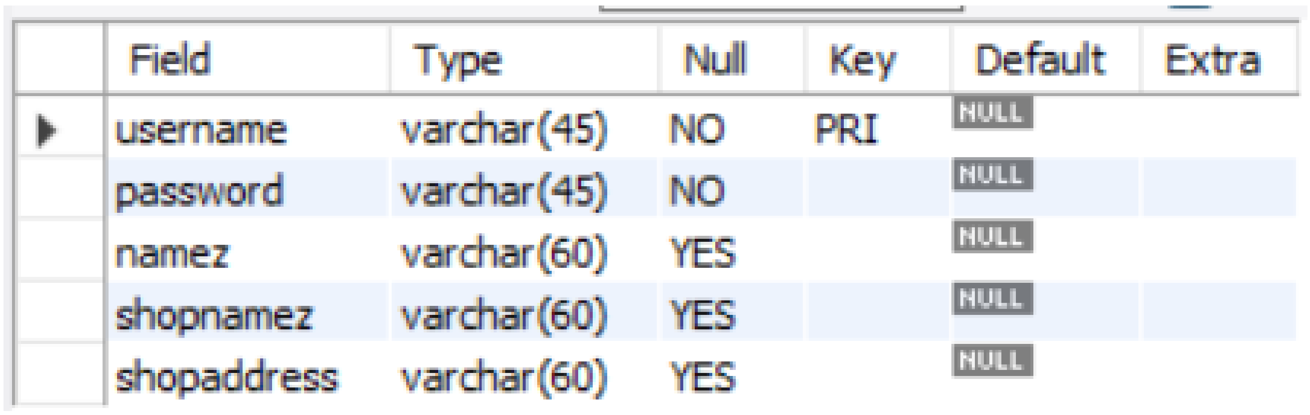


Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
sale2	0	PRIMARY	1	date	A	6	NULL	NULL		BTREE
sale2	0	PRIMARY	2	user	A	6	NULL	NULL		BTREE
sale2	0	PRIMARY	3	product_id	A	17	NULL	NULL		BTREE
sale2	1	user	1	user	A	1	NULL	NULL		BTREE

USER TABLE



Field	Type	Null	Key	Default	Extra
username	varchar(45)	NO	PRI	NULL	
password	varchar(45)	NO		NULL	
namez	varchar(60)	YES		NULL	
shopnamez	varchar(60)	YES		NULL	
shopaddress	varchar(60)	YES		NULL	

INDEXES

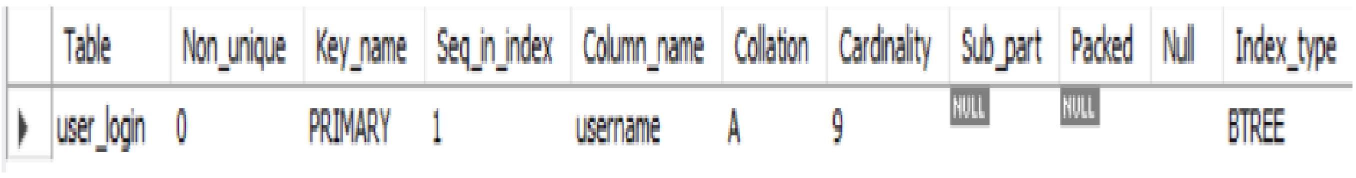
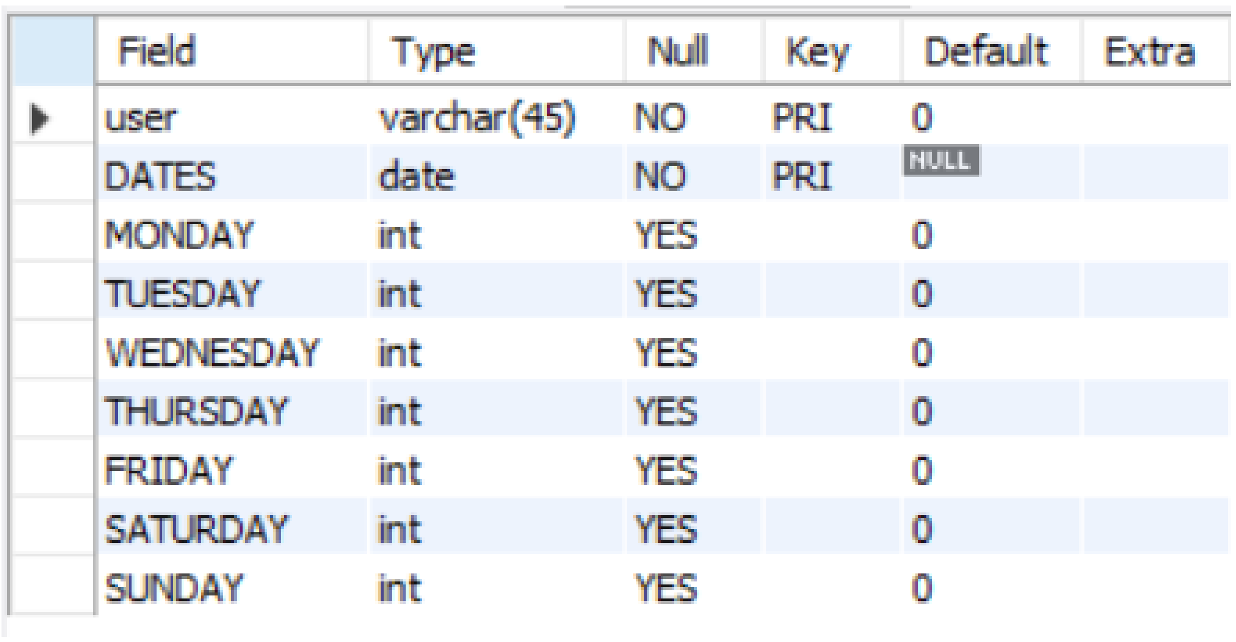


Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
user_login	0	PRIMARY	1	username	A	9	NULL	NULL		BTREE

WEEKDAY TABLE



Field	Type	Null	Key	Default	Extra
user	varchar(45)	NO	PRI	0	
DATES	date	NO	PRI	NULL	
MONDAY	int	YES		0	
TUESDAY	int	YES		0	
WEDNESDAY	int	YES		0	
THURSDAY	int	YES		0	
FRIDAY	int	YES		0	
SATURDAY	int	YES		0	
SUNDAY	int	YES		0	

INDEXES

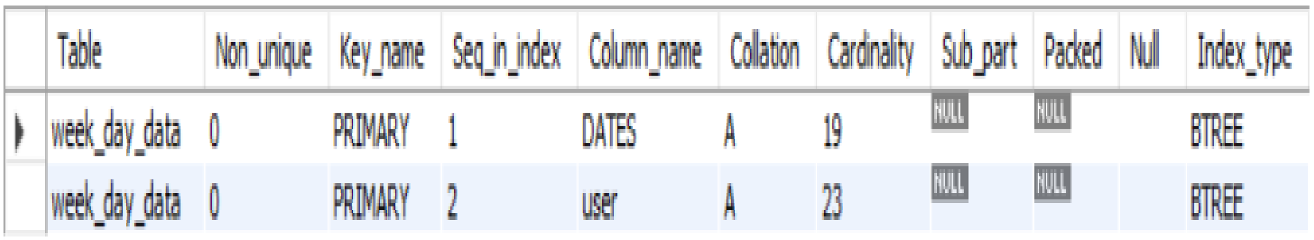


Table	Non_unique	Key_name	Seq_in_index	Column_name	Collation	Cardinality	Sub_part	Packed	Null	Index_type
week_day_data	0	PRIMARY	1	DATES	A	19	NULL	NULL		BTREE
week_day_data	0	PRIMARY	2	user	A	23	NULL	NULL		BTREE

SQL QUERIES

Basic SQL Queries for Creating and Populating the Database

```
-- MariaDB dump 10.19  Distrib 10.4.19-MariaDB, for Win64(AMD64)
--
-- Host: localhost    Database: userdb
-- ------------------------------------------------------
-- Server version         10.4.19-MariaDB

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!40101 SET NAMES utf8 */;/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0
*/;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Table structure for table `growth`
--

DROP TABLE IF EXISTS `growth`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `growth` (
`date` date NOT NULL,
`growth_percentage` float unsigned NOT NULL DEFAULT 0,
`user_id` varchar(45) NOT NULL,
PRIMARY KEY (`date`,`user_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `growth`
--
```

```
LOCK TABLES `growth` WRITE;
/*!40000 ALTER TABLE `growth` DISABLE KEYS */;
INSERT INTO `growth` VALUES ('0000-00-00',0,''), ('2021-10-17',0,'q'),('2021-10-18',0,'q'),('2021-10-
19',0,'q'),('2021-10-20',0,'q'),('2021-10-21',0,'q'),('2021-10-22',0,'q'),('2021-10-23',0,'q'),('2021-10-
24',0,'q'),('2021-11-04',72.3325,'q'),('2021-11-05',35.1351,'q'),('2021-11-06',20,'q'),('2021-11-
07',30,'q'),('2021-11-10',0,'q');
/*!40000 ALTER TABLE `growth` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `invoice`
--

DROP TABLE IF EXISTS `invoice`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `invoice` (
 `user` varchar(45) DEFAULT NULL,
`invoice_id` int(11) NOT NULL,
`client` varchar(500) DEFAULT NULL,
`date` date DEFAULT NULL,
`amount` int(11) DEFAULT NULL,
`phone` int(11) DEFAULT NULL,
`email` varchar(45) DEFAULT NULL,
`payment_mode` varchar(45) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `invoice`
--

LOCK TABLES `invoice` WRITE;
/*!40000 ALTER TABLE `invoice` DISABLE KEYS */;
INSERT INTO `invoice` VALUES ('q',0,'hima','2021-10-18',8920,NULL,NULL,NULL),('q',0,'adi','2021-10-
23',150,NULL,NULL,NULL),('q',0,'adi','2021-10-23',150,NULL,NULL,NULL),('q',0,'adi','2021-11-
05',2460,NULL,NULL,NULL);
/*!40000 ALTER TABLE `invoice` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `product`
```

```
--

DROP TABLE IF EXISTS `product`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `product` (
  `user` varchar(45) NOT NULL,
  `product_id` int(11) NOT NULL,
  `product_name` varchar(45) DEFAULT NULL,
  `product_price` int(11) DEFAULT NULL,
  `stock` int(11) DEFAULT NULL,
  `total_sale` int(11) DEFAULT NULL,
  `profit_per_item` int(11) DEFAULT NULL,
  PRIMARY KEY (`user`,`product_id`),
  CONSTRAINT `product_ibfk_1` FOREIGN KEY (`user`) REFERENCES `user_login` (`username`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `product`
--

LOCK TABLES `product` WRITE;
/*!40000 ALTER TABLE `product` DISABLE KEYS */;
INSERT INTO `product` VALUES ('q',1,'cars',2000,-20,48,800),('q',2,'bedsheet',200,-
39,84,100),('q',3,'pepsi',30,-28,78,5);
/*!40000 ALTER TABLE `product` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `profit`--
DROP TABLE IF EXISTS `profit`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `profit` (
  `date` date NOT NULL,
  `prof` int(11) DEFAULT NULL,
  `user_id` varchar(45) NOT NULL,
  PRIMARY KEY (`date`,`user_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
/*!40101 SET character_set_client = @saved_cs_client */;
```

```
--
-- Dumping data for table `profit`
--

LOCK TABLES `profit` WRITE;
/*!40000 ALTER TABLE `profit` DISABLE KEYS */;
INSERT INTO `profit` VALUES ('2021-10-17',14000,'q'),('2021-10-18',20,'q'),('2021-10-19',20,'q'),('2021-
10-20',20,'q'),('2021-10-21',20,'q'),('2021-10-22',20,'q'),('2021-10-23',20,'q'),('2021-10-
24',20,'q'),('2021-11-02',100,'q'),('2021-11-03',4460,'q'),('2021-11-04',600,'q'),('2021-11-
05',925,'q'),('2021-11-07',5430,'q'),('2021-11-10',60,'q');
/*!40000 ALTER TABLE `profit` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `sale2`
--

DROP TABLE IF EXISTS `sale2`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `sale2` (
  `date` date NOT NULL,
  `user` varchar(45) NOT NULL,
  `sale_id` varchar(45) DEFAULT NULL,
  `product_id` int(11) NOT NULL,
  `product_name` varchar(45) DEFAULT NULL,
  `sale_count` int(11) DEFAULT NULL,
  `profit` int(11) DEFAULT NULL,
  `cost` int(11) DEFAULT NULL,
  PRIMARY KEY (`date`,`user`,`product_id`),
  KEY `user` (`user`), CONSTRAINT `sale2_ibfk_1` FOREIGN KEY (`user`) REFERENCES `user_login`
(`username`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `sale2`
--

LOCK TABLES `sale2` WRITE;
/*!40000 ALTER TABLE `sale2` DISABLE KEYS */;
```

```
INSERT INTO `sale2` VALUES ('2021-11-05','q',NULL,1,'cars',1,800,2000),('2021-11-
05','q',NULL,2,'bedsheet',1,100,200),('2021-11-05','q',NULL,3,'pepsi',5,25,90),('2021-11-
07','q',NULL,1,'cars',6,4800,12000),('2021-11-07','q',NULL,2,'bedsheet',6,600,200),('2021-11-
07','q',NULL,3,'pepsi',6,30,30),('2021-11-10','q',NULL,3,'pepsi',12,60,150);
/*!40000 ALTER TABLE `sale2` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `tempinvoice`
--

DROP TABLE IF EXISTS `tempinvoice`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `tempinvoice` (
  `user` varchar(45) NOT NULL,
  `date` date NOT NULL,
  `product_id` int(11) NOT NULL,
  `product_name` varchar(45) DEFAULT NULL,
  `price` int(11) DEFAULT NULL,
  `quantity` int(11) DEFAULT NULL,
  `amount` int(11) DEFAULT NULL,
  `client` varchar(45) NOT NULL,
  PRIMARY KEY (`date`,`user`,`product_id`,`client`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `tempinvoice`
--

LOCK TABLES `tempinvoice` WRITE;
/*!40000 ALTER TABLE `tempinvoice` DISABLE KEYS */;
INSERT INTO `tempinvoice` VALUES ('q','2021-10-18',1,'cars',2000,5,10000,'adi'),('q','2021-10-
18',1,'cars',2000,4,8000,'hima'),('q','2021-10-18',2,'bedsheet',200,5,1000,'adi'),('q','2021-10-
18',2,'bedsheet',200,4,800,'hima'),('q','2021-10-18',3,'pepsi',30,5,150,'adi'),('q','2021-10-
18',3,'pepsi',30,4,120,'hima'),('q','2021-10-23',3,'pepsi',30,5,150,'adi'),('q','2021-11-
05',1,'cars',2000,1,2000,'adi'),('q','2021-11-05',2,'bedsheet',200,2,400,'adi'),('q','2021-11-
05',3,'pepsi',30,2,60,'adi');
/*!40000 ALTER TABLE `tempinvoice` ENABLE KEYS */;
UNLOCK TABLES;
```

```
--
-- Table structure for table `user_login`
--

DROP TABLE IF EXISTS `user_login`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `user_login` (
  `userid` varchar(50) DEFAULT NULL,
  `username` varchar(45) NOT NULL,
  `password` varchar(45) NOT NULL,
  PRIMARY KEY (`username`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
/*!40101 SET character_set_client = @saved_cs_client */;

--
-- Dumping data for table `user_login`
--

LOCK TABLES `user_login` WRITE;
/*!40000 ALTER TABLE `user_login` DISABLE KEYS */;
INSERT INTO `user_login` VALUES (NULL,'q','q'),(NULL,'qq','qq'),(NULL,'qqq','qqq');
/*!40000 ALTER TABLE `user_login` ENABLE KEYS */;
UNLOCK TABLES;

--
-- Table structure for table `week_day_data`
--

DROP TABLE IF EXISTS `week_day_data`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!40101 SET character_set_client = utf8 */;
CREATE TABLE `week_day_data` (
  `user` varchar(45) DEFAULT NULL,
  `DATES` date NOT NULL,
  `MONDAY` int(11) DEFAULT NULL,
  `TUESDAY` int(11) DEFAULT NULL,
  `WEDNESDAY` int(11) DEFAULT NULL,
  `THURSDAY` int(11) DEFAULT NULL,
  `FRIDAY` int(11) DEFAULT NULL,
  `SATURDAY` int(11) DEFAULT NULL,
  `SUNDAY` int(11) DEFAULT NULL,
  PRIMARY KEY (`DATES`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

```
/*!40101 SET character_set_client = @saved_cs_client */;
--
-- Dumping data for table `week_day_data`
--

LOCK TABLES `week_day_data` WRITE;
/*!40000 ALTER TABLE `week_day_data` DISABLE KEYS */;
INSERT INTO `week_day_data` VALUES ('q','2021-09-04',NULL,NULL,NULL,NULL,NULL,6,NULL),('q','2021-
09-05',NULL,NULL,NULL,NULL,NULL,NULL,4),('q','2021-09
06',10,NULL,NULL,NULL,NULL,NULL,NULL),('q','2021-09-07',NULL,9,NULL,NULL,NULL,NULL,NULL),('q','2021-
09-08',NULL,NULL,8,NULL,NULL,NULL,NULL),('q','2021-09-
09',NULL,NULL,NULL,6,NULL,NULL,NULL),('q','2021-09-10',NULL,NULL,NULL,NULL,10,NULL,NULL),('q','2021-
09-11',NULL,NULL,NULL,NULL,NULL,8,NULL),('q','2021-09-
23',NULL,NULL,NULL,34,NULL,NULL,NULL),('q','2021-09-24',NULL,NULL,NULL,NULL,2,NULL,NULL),('q','2021-
09-27',4,NULL,NULL,NULL,NULL,NULL,NULL),('q','2021-10-
17',NULL,NULL,NULL,NULL,NULL,NULL,86),('q','2021-10-
18',NULL,NULL,NULL,NULL,NULL,NULL,19),('q','2021-10-25',5,NULL,NULL,NULL,NULL,NULL,NULL),('q','2021-
11-03',NULL,NULL,6,NULL,NULL,NULL,NULL),('q','2021-11-
04',NULL,NULL,NULL,21,NULL,NULL,NULL),('q','2021-11-
05',NULL,NULL,NULL,NULL,42,NULL,NULL),('q','2021-11-
07',NULL,NULL,NULL,NULL,NULL,NULL,18),('q','2021-11-10',NULL,NULL,12,NULL,NULL,NULL,NULL);
/*!40000 ALTER TABLE `week_day_data` ENABLE KEYS */;
UNLOCK TABLES;

/*!40103 SET TIME_ZONE=@OLD_TIME_ZONE */;
/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2021-11-11 21:55:41
```

SQL Queries : -

Simple :-

OPERATIONS	SAMPLE QUERY	SYNTAX
Create	**1. To create user login table in the database** Create table user login(Userid varchar(50) default null,' Username varchar(45) not null, Password varchar(45) not null, Primary key username);	CREATE TABLE table_name (column_1 datatype, column_2 datatype, column_3 datatype)
Insert	**1. To register user** ❖ INSERT INTO USER_LOGIN(username, password,name,shopname,shopaddress) VALUE (?,?,?,?,?)	INSERT INTO table_name (column_1, column_2, column_3) VALUES (value_1, 'value_2', value_3);
Select	**1. To fetch List of Products for unique user** ❖ SELECT * FROM product where user=?	SELECT column1, column2, ...FROM table_name;
	2. To count Number of Products for unique use ❖ SELECT count(*) as count FROM product where user=?	

Continued:

OPERATIONS	SAMPLE QUERY	SYNTAX
	3. To fetch Sales Data ❖ SELECT * FROM sale2 where user=?	
	4. For Authentication of User Login ❖ SELECT * FROM USER_LOGIN WHERE USERNAME= ? AND PASSWORD=?; **5. For fetching Data for Profile** ❖ SELECT * FROM USER_LOGIN WHERE USERNAME= ?	
Delete	**1. To delete a product from Database** ❖ DELETE FROM PRODUCT WHERE product_id=? and user=?;	DELETE FROM table_name WHERE some_column = some_value;
Update	1. **For Updating stock value and total sale of a Product** ❖ UPDATE product setstock = stock-? ,total_sale=total_sale+? Where product_id=? and user=?;	UPDATE table_name SET some_column = some_value WHERE some_column = some_value;
Order By	**1. To Rank the products based on their sales** ❖ SELECT product_id,	SELECT column_name FROM table_name

Continued:

OPERATIONS	SAMPLE QUERY	SYNTAX
	2. To implement sorting ❖ Select p.date,p.prof,g.growth_ percentage from profit p,growth g where p.date=g.date and p.user_id=? and g.user_id=? order by p.date desc	
Group by	**1. To Fetch data for Doughnut / Pie Chart** ❖ SELECT product_name, sum(profit) as sum from sale2 where user=? GROUP BY product_name ;	SELECT column_name, COUNT(*)FROM table_name GROUP BY column_name;

Complex : -

OPERATION/ FUNCTIONS	SAMPLE QUERY
INTERVALS & CURDATE()	**1. To Calculate and Insert Growth Percentage into Growth Table** ❖ INSERT INTO GROWTH SELECT p1.date, ((p1.prof- p2.prof)/p1.prof)*100 ,p1.user_id FROM profit p1,profit p2 WHERE p2.date=CURDATE()- INTERVAL 1 DAY and p1.date=CURDATE() and p1.user_id=? ON DUPLICATE KEY UPDATE growth_percentage= (select (((p1.prof- p2.prof)/p1.prof)*100) FROM profit p1,profit p2 WHERE p2.date=CURDATE()- INTERVAL 1 DAY and p1.date=CURDATE() and p1.user_id=?

Continued…

OPERATIONS	SAMPLE QUERY	SYNTAX
On Duplicate Key	**1.Handles Sales of the Product** ❖ INSERT INTO sale2(date,user,product_id, product_name,sale_count,cost) VALUES(CURDATE(),?,?,?,?,?)ON DUPLICATE KEY UPDATE sale_count=if(user=? and product_id=?,sale_count+',salecount+', sale_count),profit=if(user=?and product_id=?,profit+' + profit + ',profit);	INSERT INTO t1 SELECT c, c+d FROM t2 ON DUPLICATE KEY UPDATE b = VALUES(b);
	2. To Insert sales count way_day table ❖ INSERT INTO week_day_data(user,dates,' + day + ') VALUES(?,CURDATE(),?) ON DUPLICATE KEY UPDATE ' + day + '=if(user=? ,' + day + '+' + salecount + ',' + day + ')	
NESTED QUERY	**1. To fetch data for handling Profit** ❖ INSERT INTO profit(user_id,date,prof) (SELECT user,(date),sum(profit) from sale2 WHERE user=? and date=curdate()) ON DUPLICATE KEY UPDATE prof=(SELECT sum(profit) from sale2 WHERE user=? and date=curdate());	

Continued:

OPERATIONS	SAMPLE QUERY
Queries using Tuple Variables	**1. To Fetch data for Profit and Growth Line Charts** ❖ SELECT p.date,p.prof,g.growth_percentage from profit p,growth g where p.date=g.date and p.user_id=? and g.user_id=?;
	2. For periodically updating profit value ❖ UPDATE sale2 s,product p set profit=s.sale_count*p.profit_per_item WHERE s.product_id=p.product_id and s.user=p.user;
INNER JOIN JOIN	**1. To fetch Sales Data for particular product** ❖ SELECT s.date,s.sale_count,s.sale_count*p.profit_per_item as profit ,s.user,p.product_name, p.product_price from sale2 s INNER JOIN product p on s.product_id=p.product_id and s.user=? and p.user=?; **2. Displays Sales History** ❖ SELECT date,sale_count, profit,user, product_name, product.price FROM sale2 , product JOIN sale2 ON product.product_id= sale2.product_id and p.user=?;

AGGREGATE :-

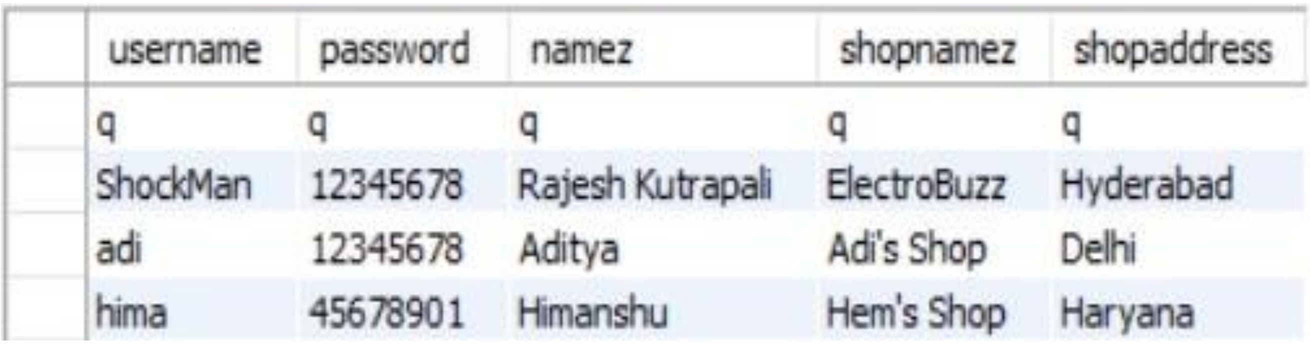
OPERATIONS	SAMPLE QUERY	SYNTAX
SUM()	**1. To Fetch total no of stock** ❖ SELECT sum(stock) as stock FROM product where user=?	SELECT SUM(column_name) FROM table_name;
	2. To fetch the total profit of a user ❖ SELECT SUM(profit) FROM sale2 where user=?;	
	3. To Fetch Total Stock Available for particular user ❖ SELECT sum(stock) as stock FROM product where user=?	
AVG()	**1. To Fetch data for Bar Graph** SELECT AVG(SUNDAY), AVG(MONDAY), AVG(TUESDAY), AVG(WEDNESDAY), AVG(THURSDAY), AVG(FRIDAY),	SELECT AVG(column_name) FROM table_name;

Continued:

OPERATIONS	SAMPLE QUERY	SYNTAX
	2. To fetch average monthly profit data for unique user ❖ SELECT month(curdate()) as month,AVG(p.prof) as avg FROM profit p WHERE month(p.date)=month(curdate()) And p.user_id=?;	

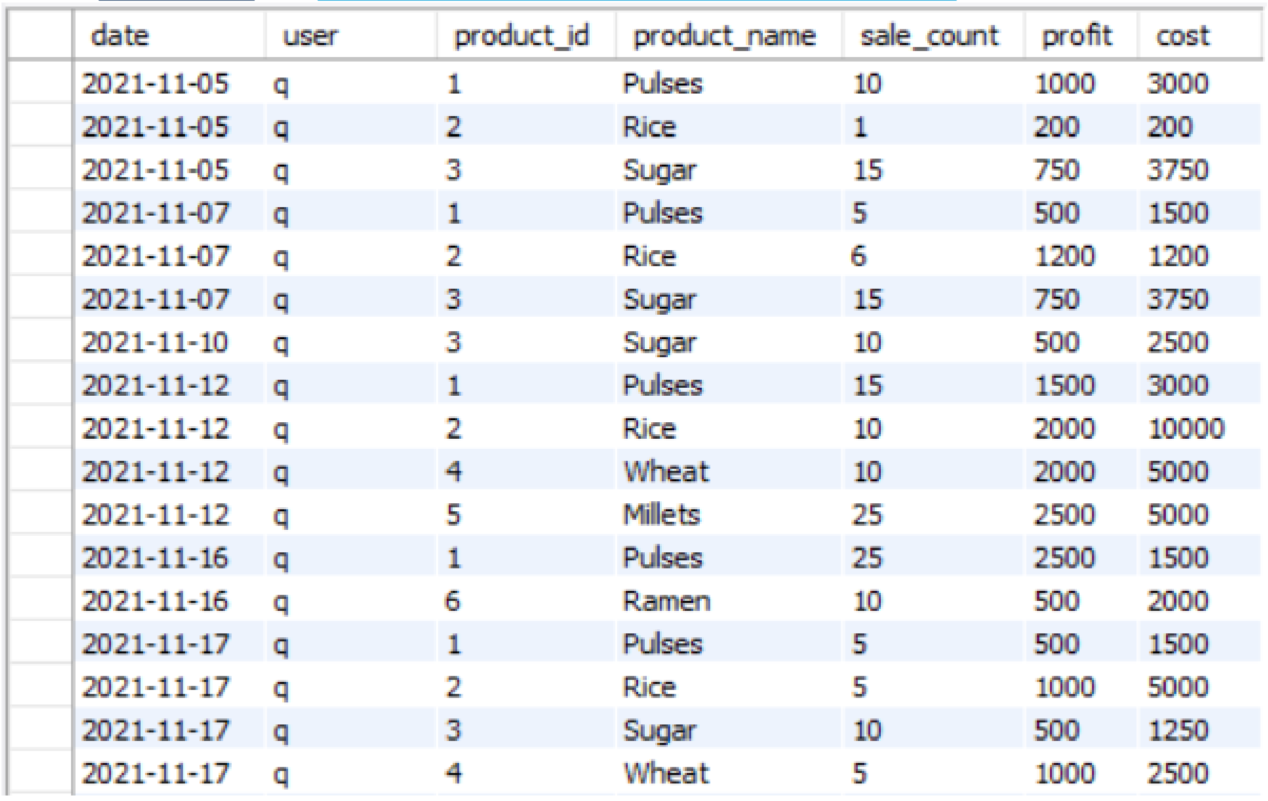
POPULATED TABLES

USER



username	password	namez	shopnamez	shopaddress
q	q	q	q	q
ShockMan	12345678	Rajesh Kutrapali	ElectroBuzz	Hyderabad
adi	12345678	Aditya	Adi's Shop	Delhi
hima	45678901	Himanshu	Hem's Shop	Haryana

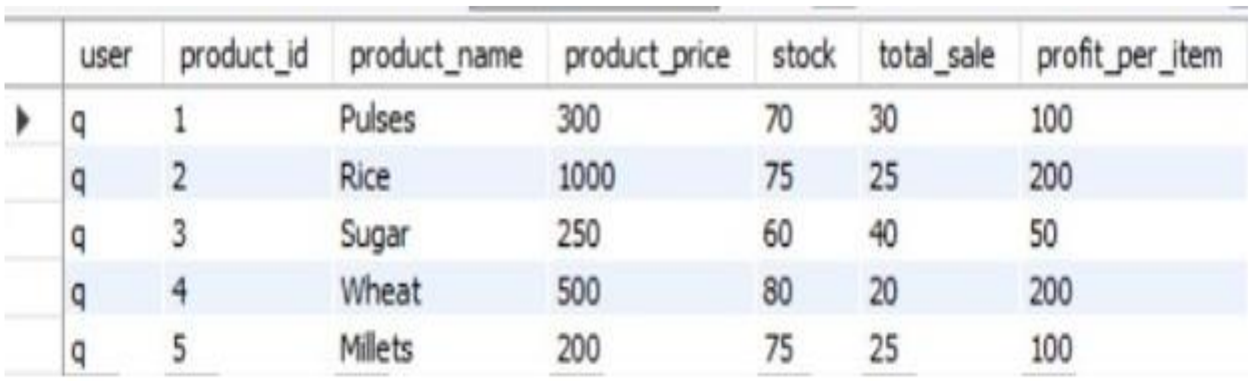
SALES_DATA



date	user	product_id	product_name	sale_count	profit	cost
2021-11-05	q	1	Pulses	10	1000	3000
2021-11-05	q	2	Rice	1	200	200
2021-11-05	q	3	Sugar	15	750	3750
2021-11-07	q	1	Pulses	5	500	1500
2021-11-07	q	2	Rice	6	1200	1200
2021-11-07	q	3	Sugar	15	750	3750
2021-11-10	q	3	Sugar	10	500	2500
2021-11-12	q	1	Pulses	15	1500	3000
2021-11-12	q	2	Rice	10	2000	10000
2021-11-12	q	4	Wheat	10	2000	5000
2021-11-12	q	5	Millets	25	2500	5000
2021-11-16	q	1	Pulses	25	2500	1500
2021-11-16	q	6	Ramen	10	500	2000
2021-11-17	q	1	Pulses	5	500	1500
2021-11-17	q	2	Rice	5	1000	5000
2021-11-17	q	3	Sugar	10	500	1250
2021-11-17	q	4	Wheat	5	1000	2500

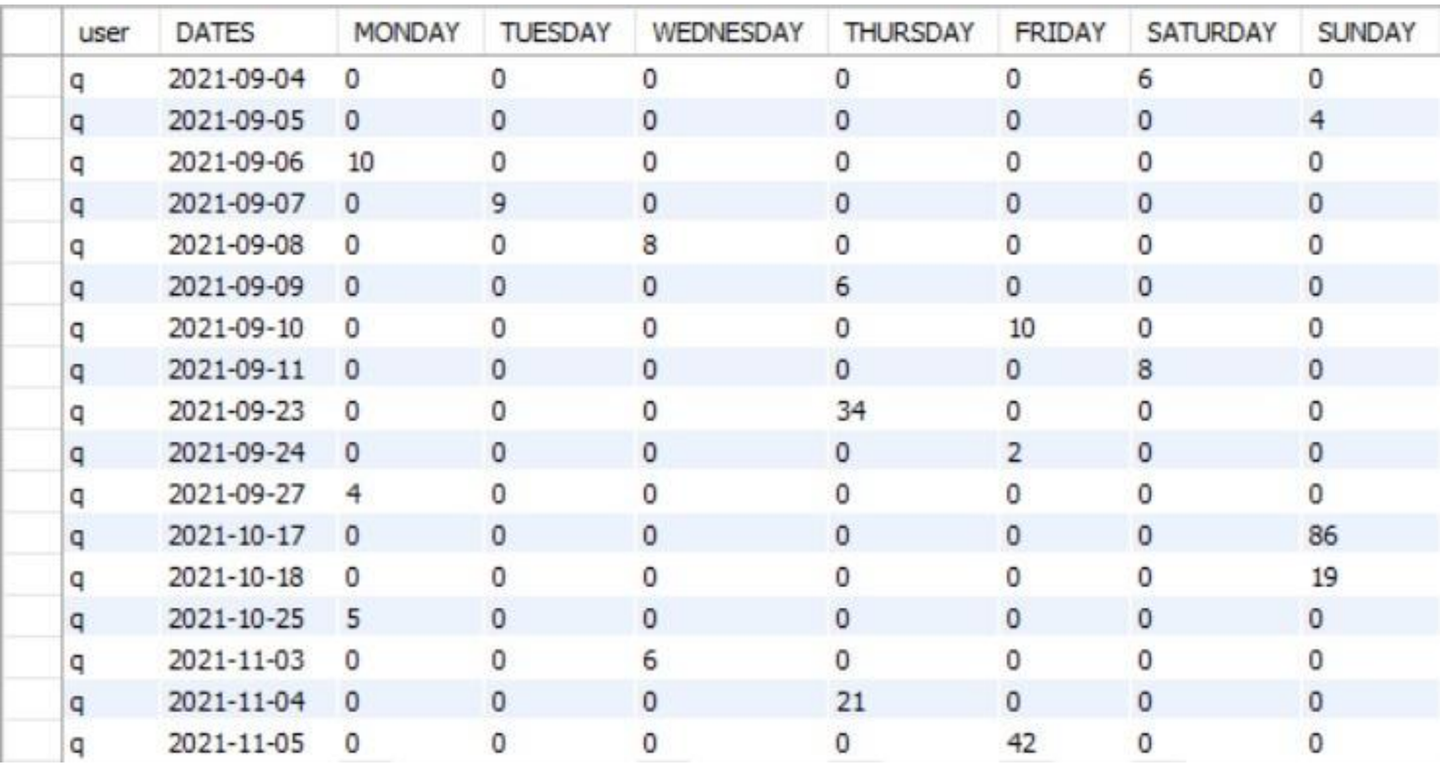
CONTINUED..

PRODUCTS



user	product_id	product_name	product_price	stock	total_sale	profit_per_item
q	1	Pulses	300	70	30	100
q	2	Rice	1000	75	25	200
q	3	Sugar	250	60	40	50
q	4	Wheat	500	80	20	200
q	5	Millets	200	75	25	100

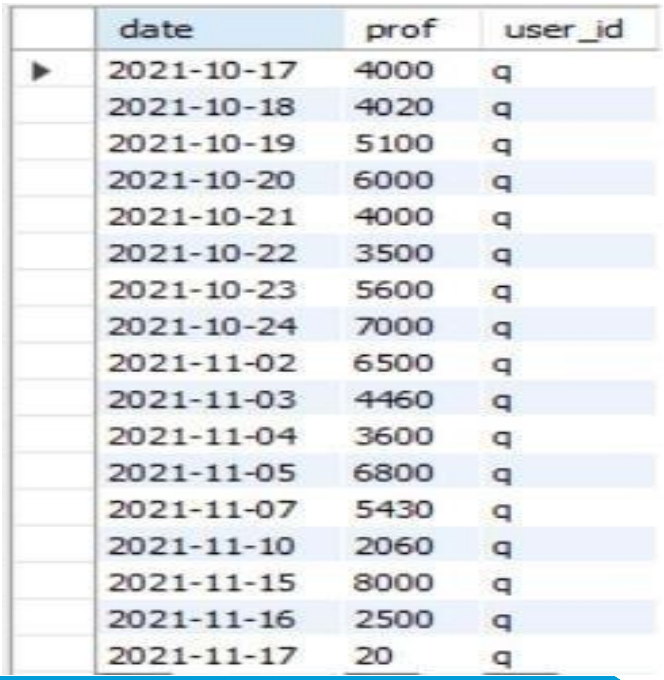
WEEK_CHART



user	DATES	MONDAY	TUESDAY	WEDNESDAY	THURSDAY	FRIDAY	SATURDAY	SUNDAY
q	2021-09-04	0	0	0	0	0	6	0
q	2021-09-05	0	0	0	0	0	0	4
q	2021-09-06	10	0	0	0	0	0	0
q	2021-09-07	0	9	0	0	0	0	0
q	2021-09-08	0	0	8	0	0	0	0
q	2021-09-09	0	0	0	6	0	0	0
q	2021-09-10	0	0	0	0	10	0	0
q	2021-09-11	0	0	0	0	0	8	0
q	2021-09-23	0	0	0	34	0	0	0
q	2021-09-24	0	0	0	0	2	0	0
q	2021-09-27	4	0	0	0	0	0	0
q	2021-10-17	0	0	0	0	0	0	86
q	2021-10-18	0	0	0	0	0	0	19
q	2021-10-25	5	0	0	0	0	0	0
q	2021-11-03	0	0	6	0	0	0	0
q	2021-11-04	0	0	0	21	0	0	0
q	2021-11-05	0	0	0	0	42	0	0

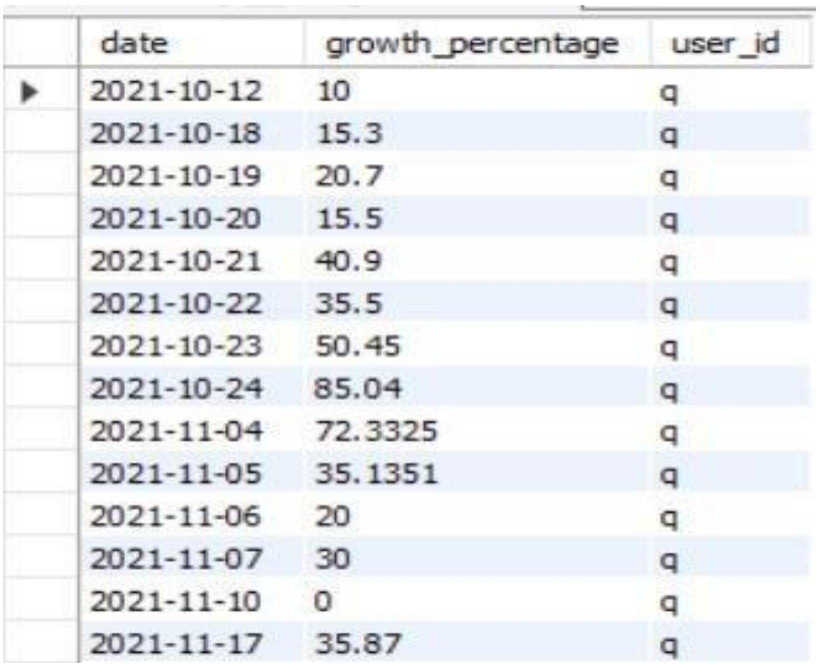
CONTINUED..

PROFIT



	date	prof	user_id
▶	2021-10-17	4000	q
	2021-10-18	4020	q
	2021-10-19	5100	q
	2021-10-20	6000	q
	2021-10-21	4000	q
	2021-10-22	3500	q
	2021-10-23	5600	q
	2021-10-24	7000	q
	2021-11-02	6500	q
	2021-11-03	4460	q
	2021-11-04	3600	q
	2021-11-05	6800	q
	2021-11-07	5430	q
	2021-11-10	2060	q
	2021-11-15	8000	q
	2021-11-16	2500	q
	2021-11-17	20	q

GROWTH_PERCENTAGE



	date	growth_percentage	user_id
▶	2021-10-12	10	q
	2021-10-18	15.3	q
	2021-10-19	20.7	q
	2021-10-20	15.5	q
	2021-10-21	40.9	q
	2021-10-22	35.5	q
	2021-10-23	50.45	q
	2021-10-24	85.04	q
	2021-11-04	72.3325	q
	2021-11-05	35.1351	q
	2021-11-06	20	q
	2021-11-07	30	q
	2021-11-10	0	q
	2021-11-17	35.87	q

DATABASE CONNECTIVITY

The database has been connected to the frontend user interface using

ReactJs as the scripting language.

The code for establishing the connection is as follows:

```
const express=require('express')

const app=express()

app.use(cors());

app.use(express.json())

const db=mysql.createConnection({

    user: process.env.USER,
    host: process.env.HOST,

    password: process.env.PASSWORD,

    database: process.env.DATABASE

})
```

The above code serves to establish the connection between the backend (database) and frontend (user interface).

The ReactJs codes corresponding to each functionality being implemented are mentioned along with the respective functionality in the following pages.

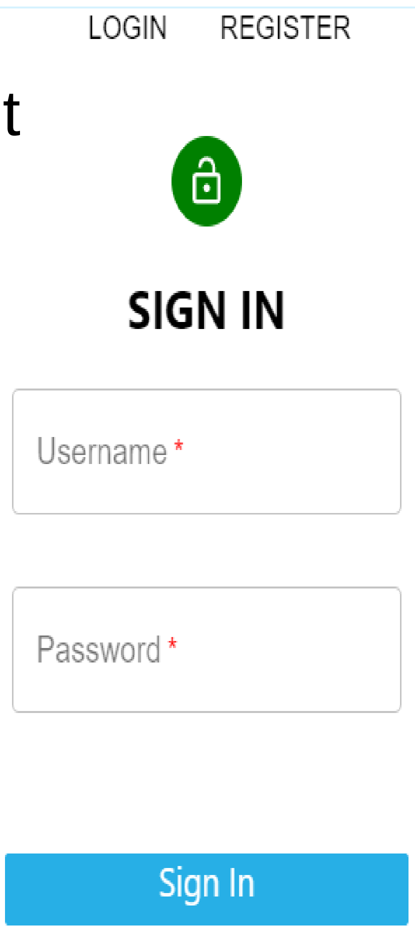
FUNCTIONALITIES

•**SIGN IN** :

To access the application. It is mandatory to pass into the application.

CODE:

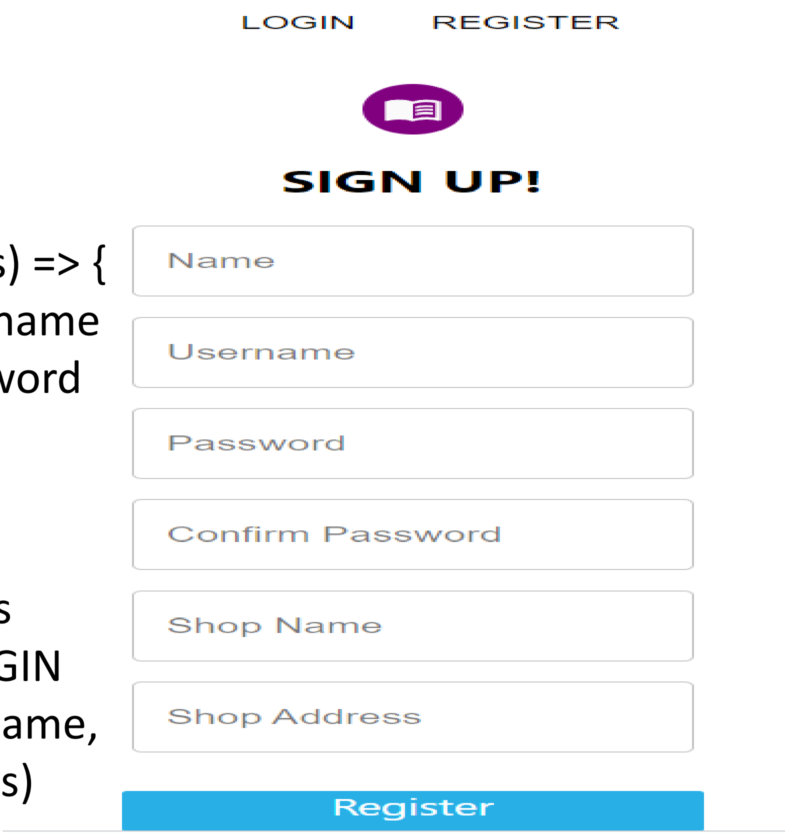
```
app.post('/login', (req, res) =>
 {
const username =
req.body.username
const password =
req.body.password
user_global = username
var qry = 'SELECT * FROM
USER_LOGIN WHERE USERNAME=
? AND PASSWORD=?;'
db.query(qry, [username, password], (err, result) => {
            if (err) {
                        return res.send({ err: err })
                  }
            if (result.length > 0) {
                        return res.send({ message: 'yes' })
                        }
            else {
                        return res.send({ message: 'no' })
                  }
   })
 })
```



CONTINUED..

Register :

For new Users.



LOGIN REGISTER

SIGN UP!

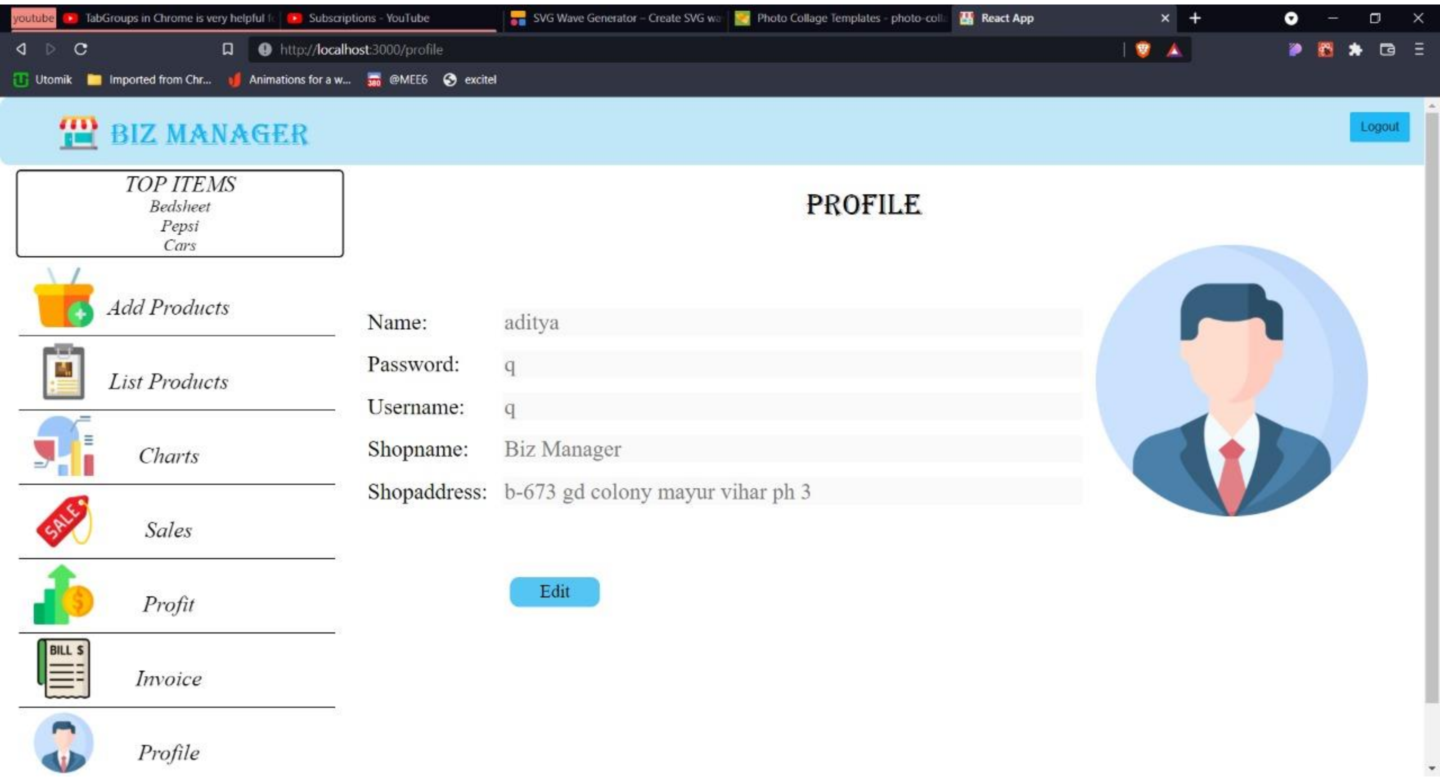
Name
Username
Password
Confirm Password
Shop Name
Shop Address

```
app.post('/registeruser', (req, res) => {
const username = req.body.username
const password = req.body.password
const name = req.body.name
const shopname =
req.body.shopname
const address = req.body.address
var qry = 'INSERT INTO USER_LOGIN
            (username,password,name,
            shopname,shopaddress)
            VALUE (?,?,?,?,?);'

db.query(qry, [username, password, name,shopname,address], (err,
result) => {
      if (err) {
                  console.log(err)
               }
      else {
                  console.log('user registered’)
                  return res.send("hello")
             }
   })
}
```

CONTINUED..

PROFILE: For displaying user details and allow them to edit


BIZ MANAGER

TOP ITEMS
Bedsheet
Pepsi
Cars

PROFILE

Name: aditya
Password: q
Username: q
Shopname: Biz Manager
Shopaddress: b-673 gd colony mayur vihar ph 3

CODE:

```
app.post('/profile', (req, res) => {
        const username = req.body.username
        var qry = 'SELECT * FROM USER_LOGIN WHERE USERNAME=
        ?;’
        db.query(qry, [username], (err, result) => {
            if (err) {
                        return res.send({ err: err })
                    }
            else
```

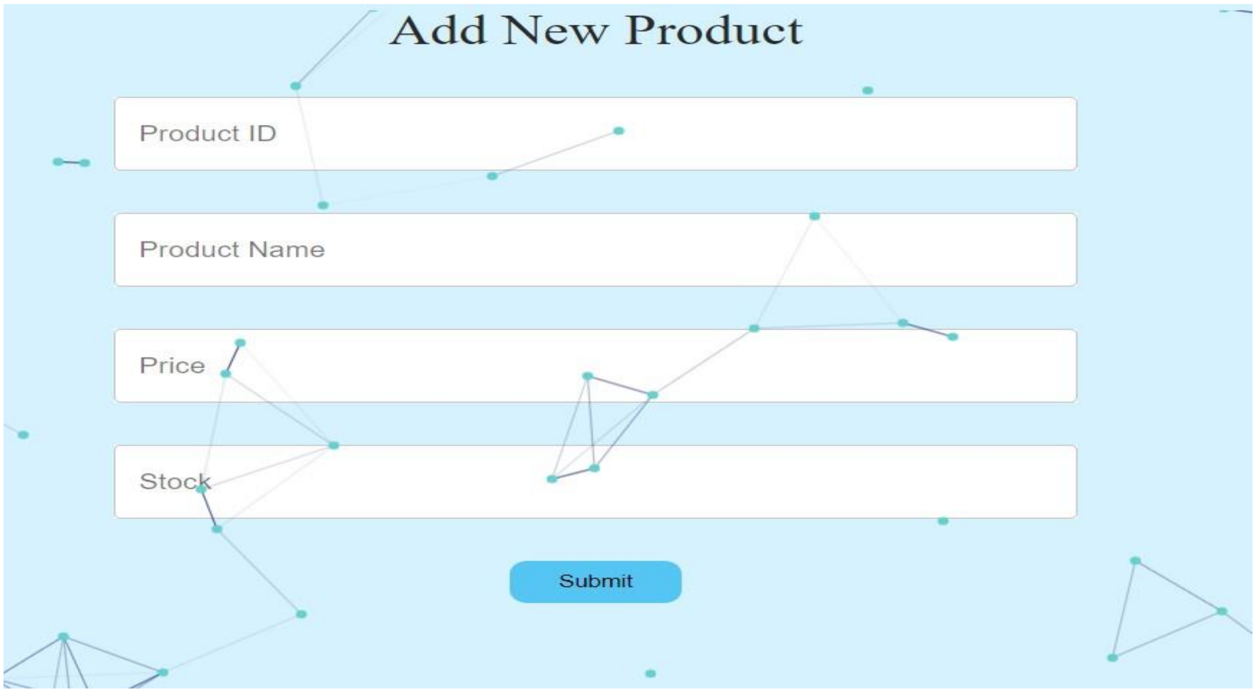
CONTINUED..

```
                                    return res.send(result)
                }) 
})

app.post('/newprofile', (req, res) => {
                const username = req.body.username
                const name = req.body.name
                const newusername = req.body.newusername
                const password = req.body.password
                const shopname = req.body.shopname
                const shopaddress = req.body.shopaddress
                var qry = 'update user_login set name=?,username=?,
                                password=?,shopname=?,shopaddress=? where
                                username =?;’
                db.query(qry, [name,newusername,password,shopname,
                                        shopaddress,username], (err, result) => {
                       if (err) {
                                        return res.send({ err: err })
                                    }
                      else
                                        return res.send(result)
                })
})
```

CONTINUED..

ADD PRODUCTS :
To Add products in the Shop.



CODE:

```
app.post('/sentdata', (req, res) => {
            const user = req.body.user
            const id = req.body.id
            const pname = req.body.pname
            const price = req.body.price
            const profitperitem = req.body.profitperitem
            const quantity = req.body.quantity
            var qry = "INSERT INTO product(user,product_id,
            product_name,product_price,stock,profit_per_item)
            VALUES(?,?,?,?,?,?)“
```

CONTINUED..

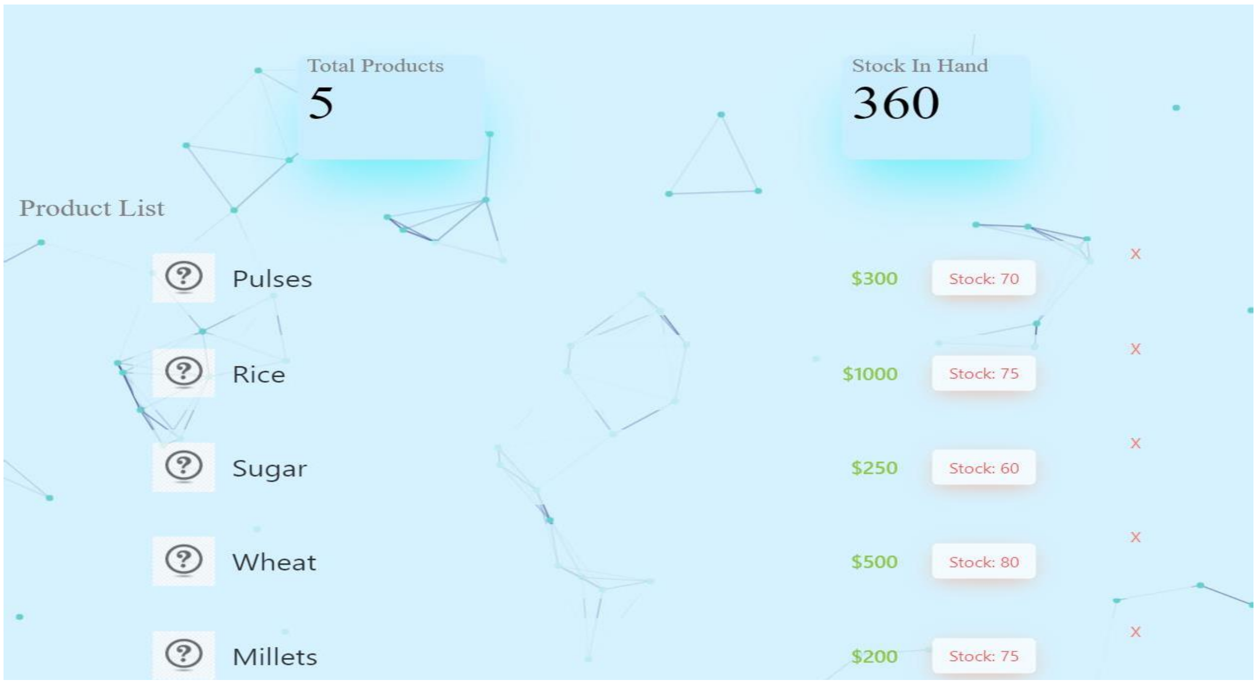
```
            db.query(qry, [user, id, pname, price, quantity,profitperitem],
            (err, result) => {
            if (err)      {
                        console.log(err)
            }

            else {
                        return res.send("hello")
                  }
    })
})
```

CONTINUED..

. LISTING PRODUCTS :

Lists all the products along with their current stock values.



Total Products

5

Stock In Hand

360

Product List

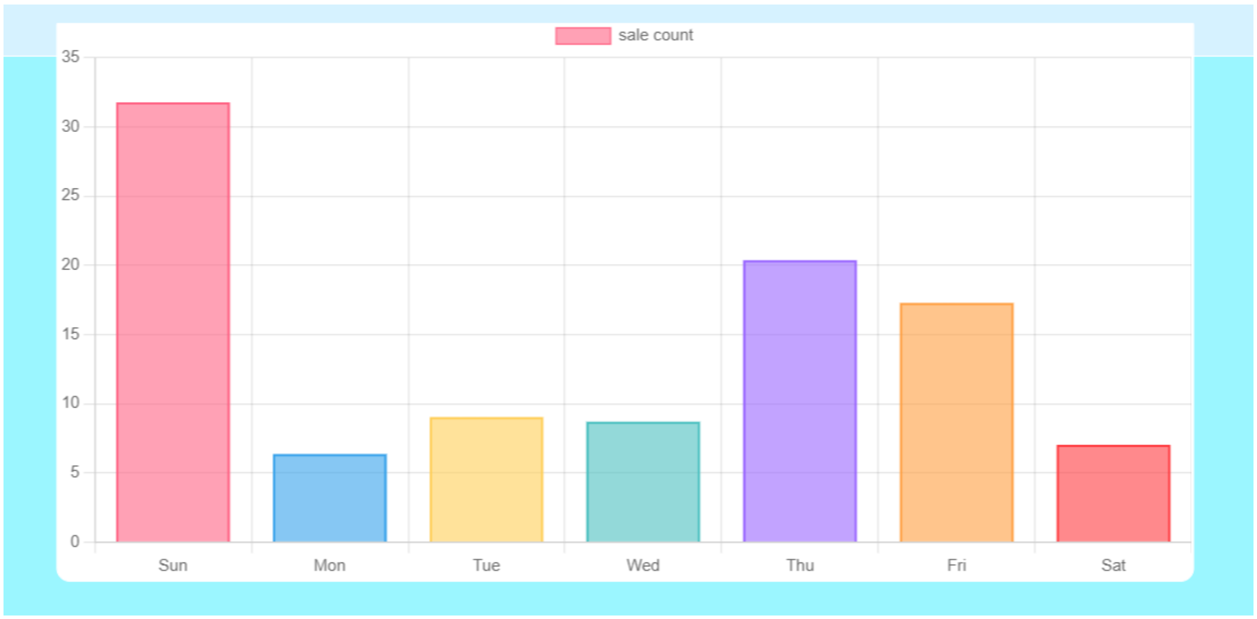
?	Pulses	$300	Stock: 70	x
?	Rice	$1000	Stock: 75	x
?	Sugar	$250	Stock: 60	x
?	Wheat	$500	Stock: 80	x
?	Millets	$200	Stock: 75	x

CODE:

```
app.get('/receiveddata', (req, res) => {
            const user = req.query.user
                    console.log(user)
            var qry = "SELECT * FROM product where user=?"
            db.query(qry, [user], (err, result) => {
                        if (err) {
                                    console.log(err)
                              }
                        else {
                                    console.log("extracted")
                                       return res.send(result)
                            }
         })
})
```

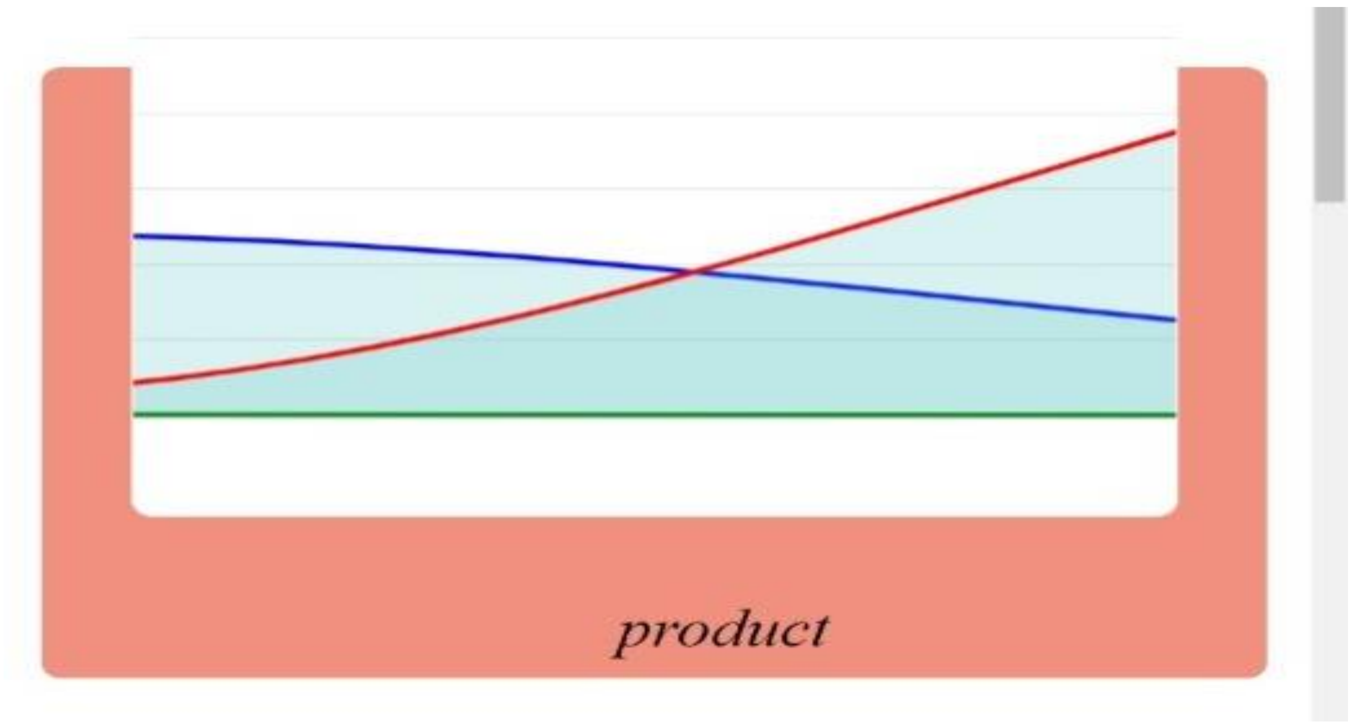
CONTINUED..

STATISTICS : Several analysis patterns based on factors affecting trade in order to monitor and enhance user's sales.



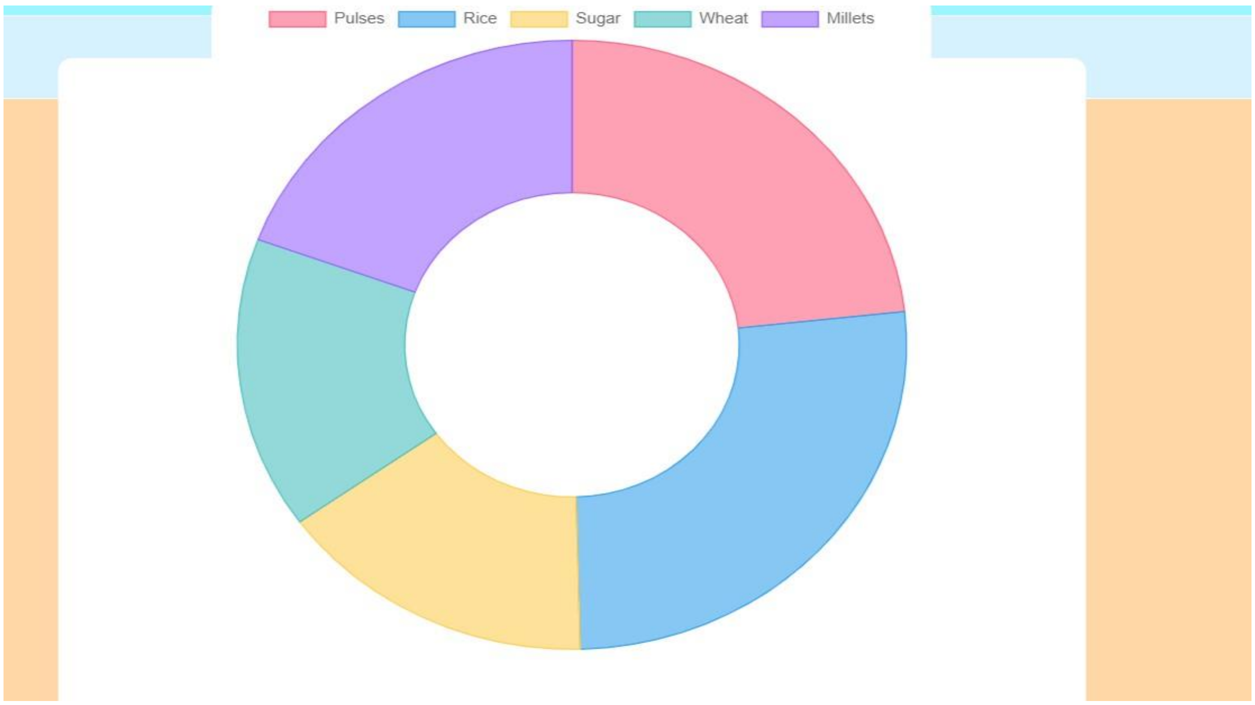
ABOUT:

THIS IS A FORM OF BAR GRAPH WHICH SHOWS WEEKLY TOTAL SALES OF SELLER.



ABOUT:

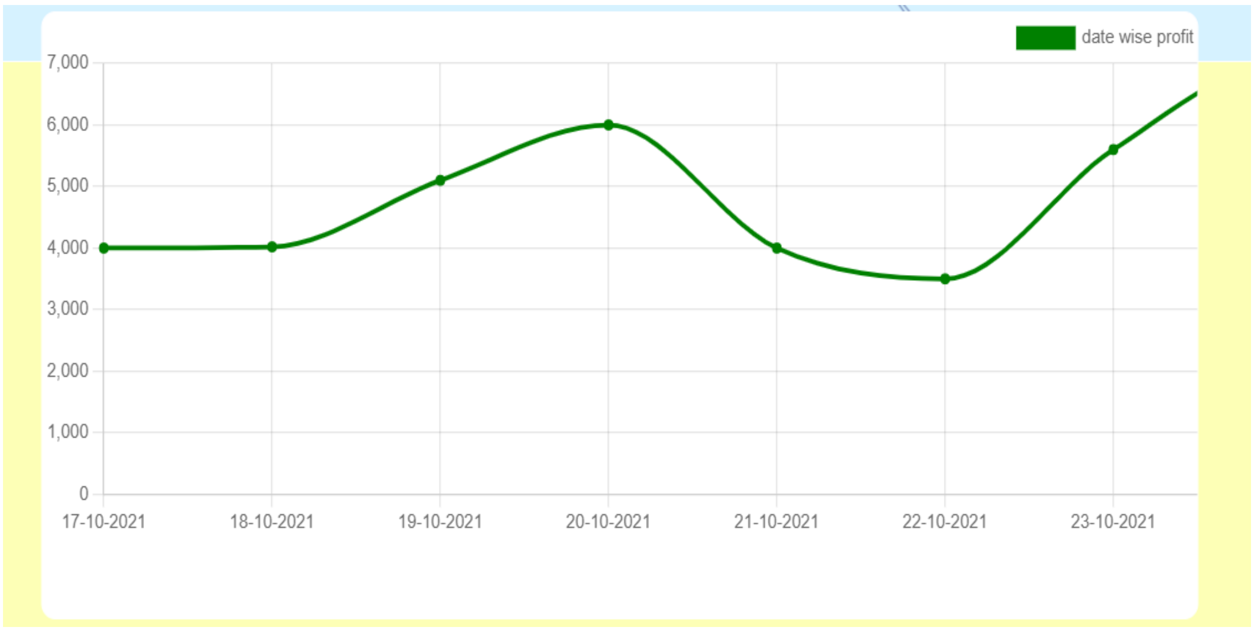
THIS IS A FORM OF LINE GRAPH WHICH SHOWS THE SALE OF THREE TOP PRODUCTS TO THE SELLER.



ABOUT:

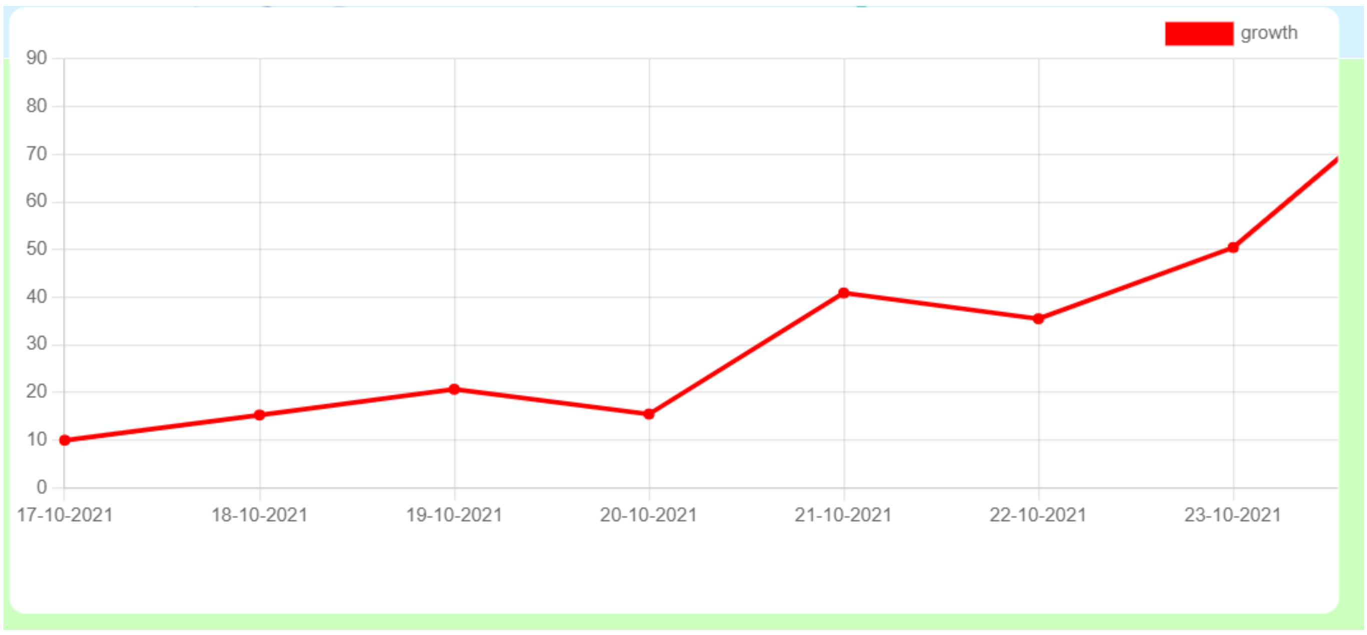
THIS IS A FORM OF PIE CHART WHICH SHOWS SALES OF DIFFERENT PRODUCTS AVAILABLE IN THE SHOP TO THE SELLER.

CONTINUED..



ABOUT:

THIS IS A FORM OF LINE GRAPH WHICH SHOWS DATE-WISE PROFIT TO THE SELLER.

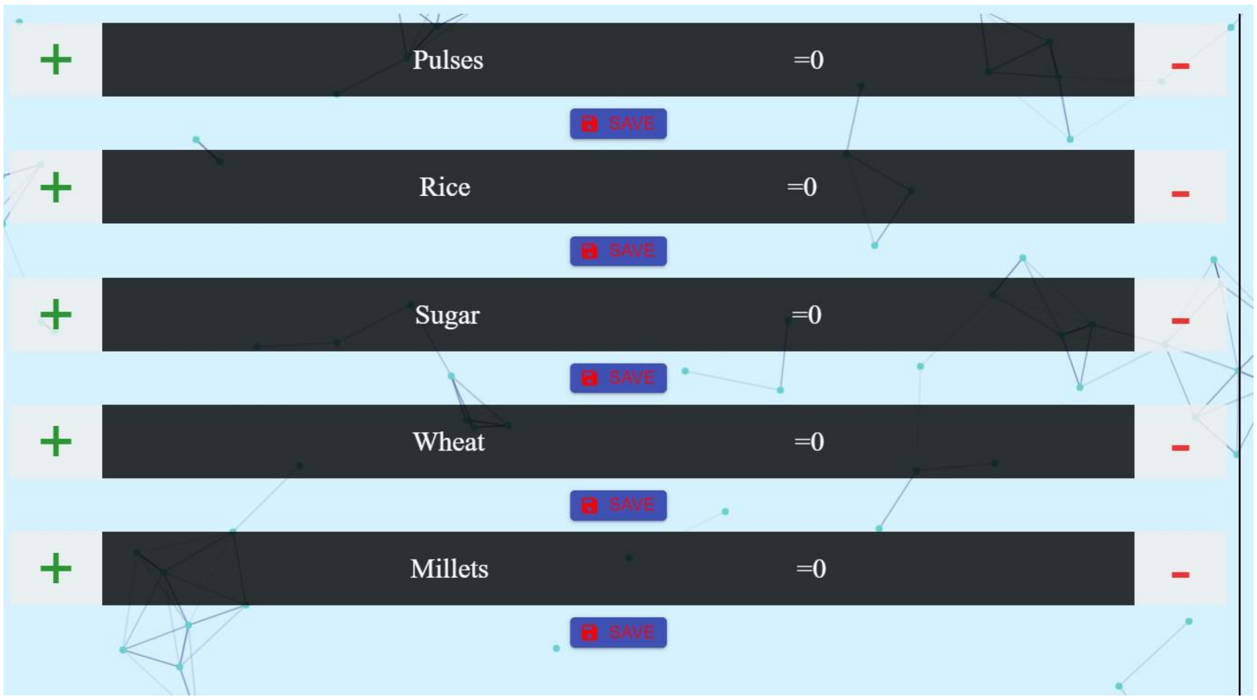


ABOUT:

THIS IS A FORM OF LINE GRAPH WHICH SHOWS GROWTH PERCENTAGE TO THE SELLER.

CONTINUED..

SALES : Sells products by updating stock values and calculates profit.



CODE:

```
app.post('/saleupdate',(req,res)=>{
            const id=req.body.product_id
            const pname=req.body.product_name
            const salecount=req.body.salecount
            const profit=req.body.profit
            const date=req.body.day
            const user=req.body.user
            var qry = 'INSERT INTO sale2(date,user,product_id,
                        product_name,sale_count,cost)
                        VALUES(CURDATE(),?,?,?,?,?)
```

CONTINUED..

```
on duplicate key
            update sale_count=
            if(user=? and product_id=?,sale_count+' + salecount +
              ',sale_count),profit=if(user=? and product_id=?,profit+’ +
            profit + ',profit);’
            var qry2 =
                        'update product set stock=stock-?
                        total_sale=total_sale+? where product_id=? and
                        user=?;’
            var qry3 = 'update sale2 s,product p set
                        profit=s.sale_count*p.profit_per_item where
                        s.product_id=p.product_id and s.user=p.user;’
  db.query(qry, [user, id, pname, salecount, profit, user, id, user, id], (err,
result) =>{
            if (err) {
                                  console.log(err)
                    }
            else {
                                 console.log('inserted in sales')
                 }
})
 db.query(qry2, [salecount, salecount, id, user], (err, result) => {
            if (err) {
                                console.log(err)
                    }
            else {
                              console.log('updated stock and total sale')
                 }
   })
```

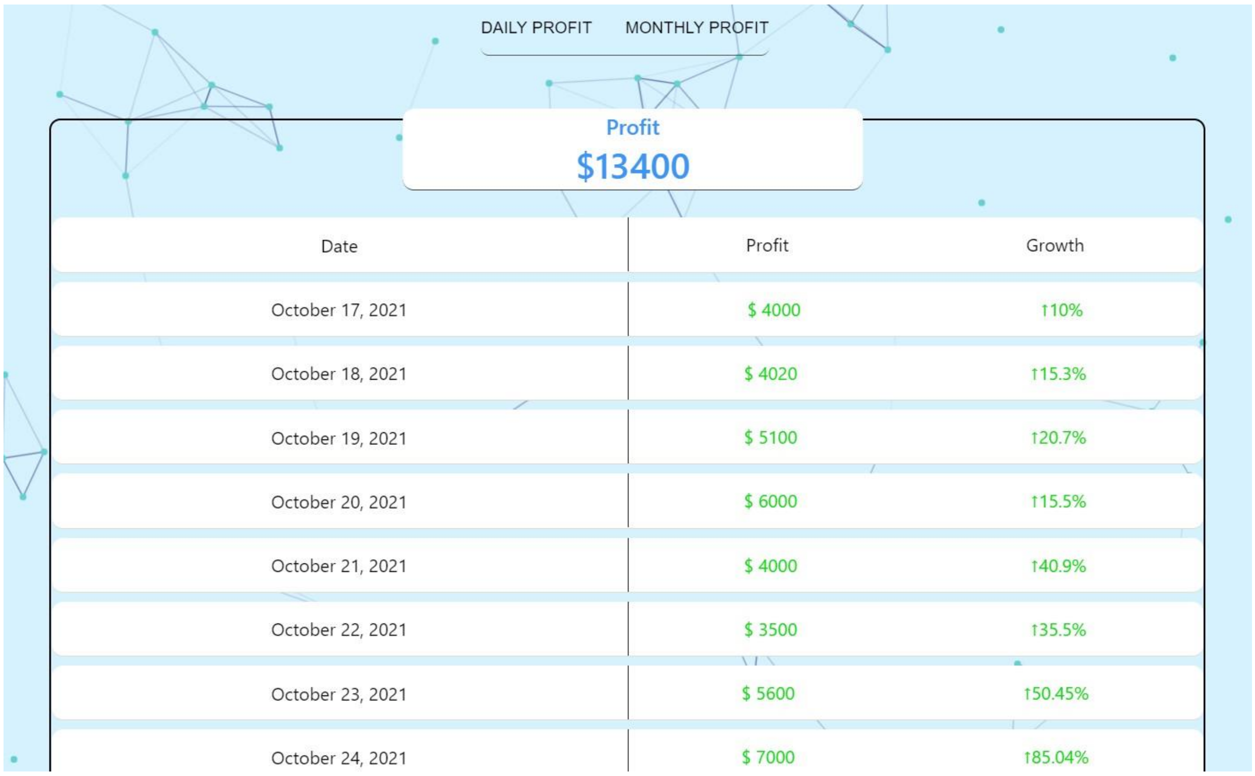
CONTINUED..

```
db.query(qry3, (err, result) => {
                        if (err) {
                                                    console.log(err)
                                    }
                      else {
                                                    console.log('updated profit')
                                                    return res.send("hello")
                              }
                  })
}
```

CONTINUED..

PROFIT :

- **DAILY PROFIT** : -



Profit
$13400

Date	Profit	Growth
October 17, 2021	$ 4000	↑10%
October 18, 2021	$ 4020	↑15.3%
October 19, 2021	$ 5100	↑20.7%
October 20, 2021	$ 6000	↑15.5%
October 21, 2021	$ 4000	↑40.9%
October 22, 2021	$ 3500	↑35.5%
October 23, 2021	$ 5600	↑50.45%
October 24, 2021	$ 7000	↑85.04%

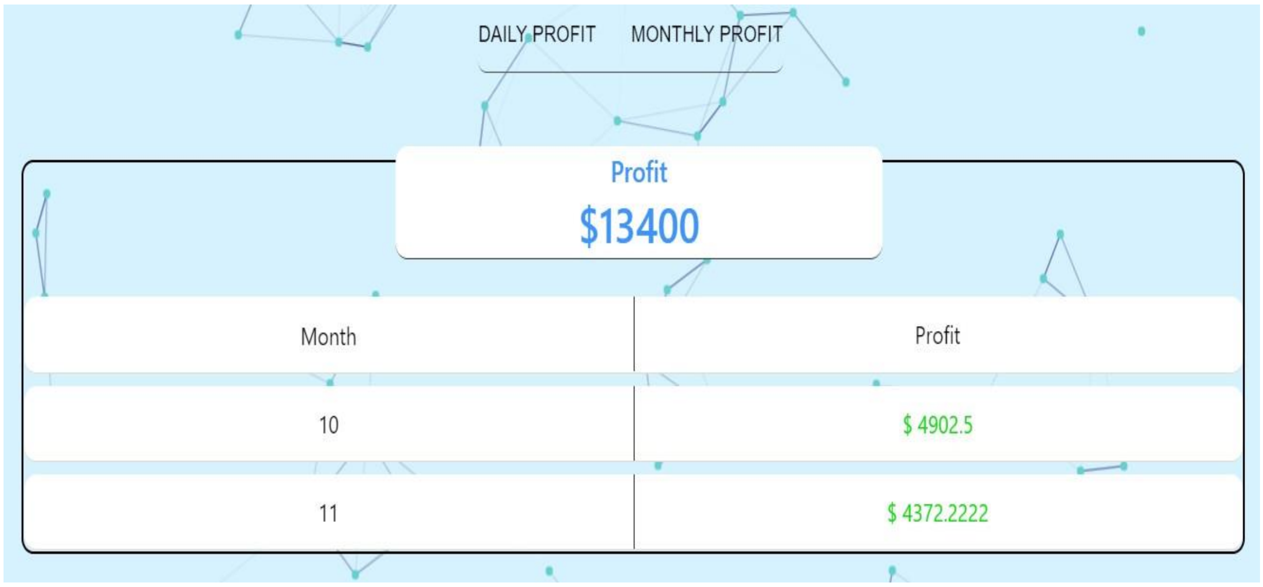
```
app.get('/dailyprofit', (req, res) => {
                const user = req.query.user
                const filter = req.query.filter
                if(filter==="true")
                var qry = "select p.date,p.prof,g.growth_percentage from
                            profit p,growth g where p.date=g.date and
                            p.user_id=? and g.user_id=? order by p.date desc
                            ;"
               else if(filter==="false")
               var qry = "select p.date,p.prof,g.growth_percentage from
                           profit p,growth g where p.date=g.date and
                           p.user_id=? and g.user_id=? order by p.date;"
```

CONTINUED..

```
db.query(qry, [user, user], (err, result) => {\
                    if (err) {
                                        console.log(err)
                              }
                   else {
                                        console.log("extracted daily profit")
                                        return res.send(result)
                         }
              })
})
```

CONTINUED..

•**MONTHLY PROFIT** :-

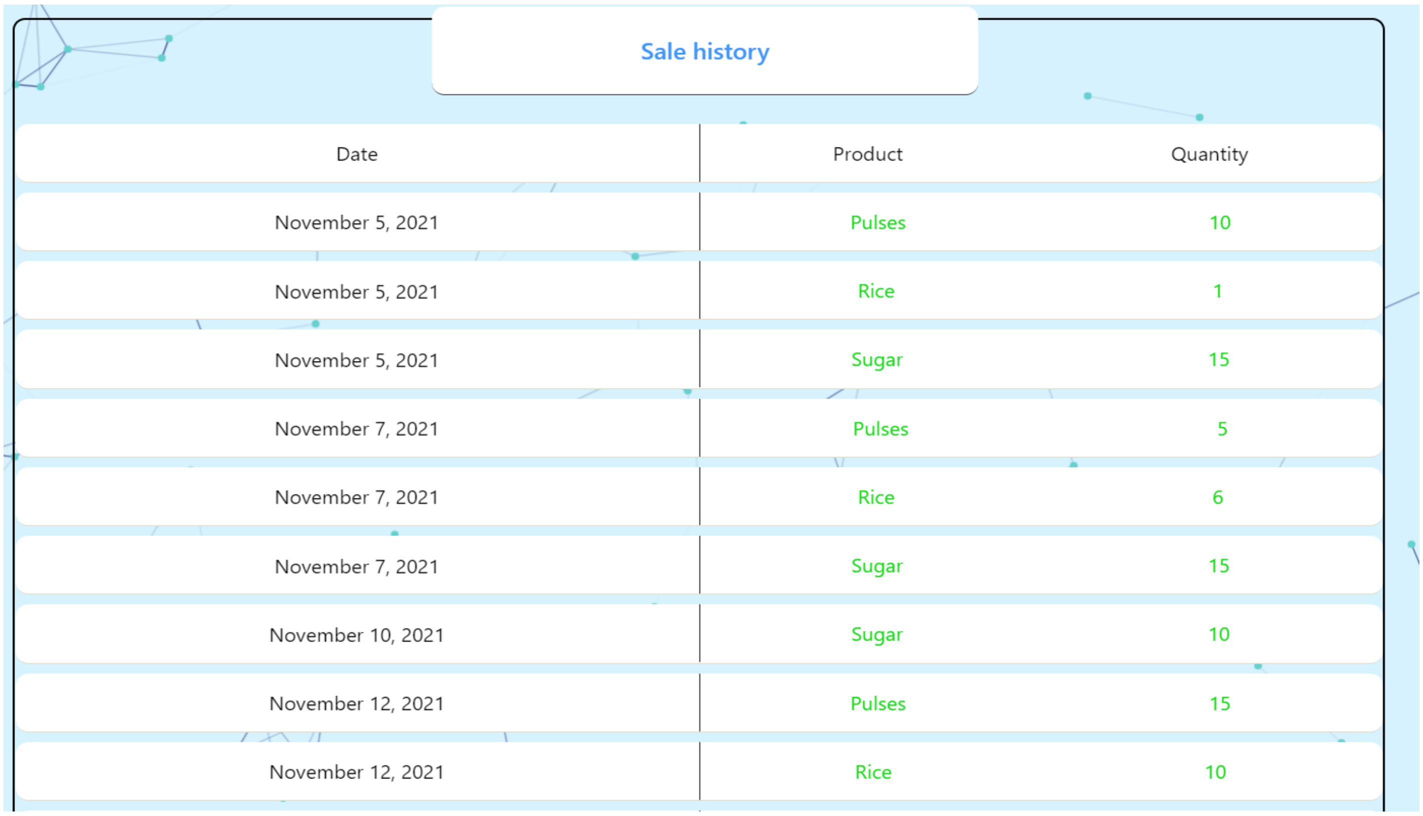


Month	Profit
10	$ 4902.5
11	$ 4372.2222

```
app.get('/monthlyprofit', (req, res) => {
                const user = req.query.user
                var qry = "select month(p.date) as month,avg(p.prof) as avg
                              from profit p where p.user_id=? group by
                              month(p.date)“
db.query(qry, [user], (err, result) => {
                 if (err) {
                                        console.log(err)
                         }
                else {
                                        console.log("extracted")
                                        return res.send(result)
                     }
            })
})
```

CONTINUED..

Sales History :



Sale history

Date	Product	Quantity
November 5, 2021	Pulses	10
November 5, 2021	Rice	1
November 5, 2021	Sugar	15
November 7, 2021	Pulses	5
November 7, 2021	Rice	6
November 7, 2021	Sugar	15
November 10, 2021	Sugar	10
November 12, 2021	Pulses	15
November 12, 2021	Rice	10

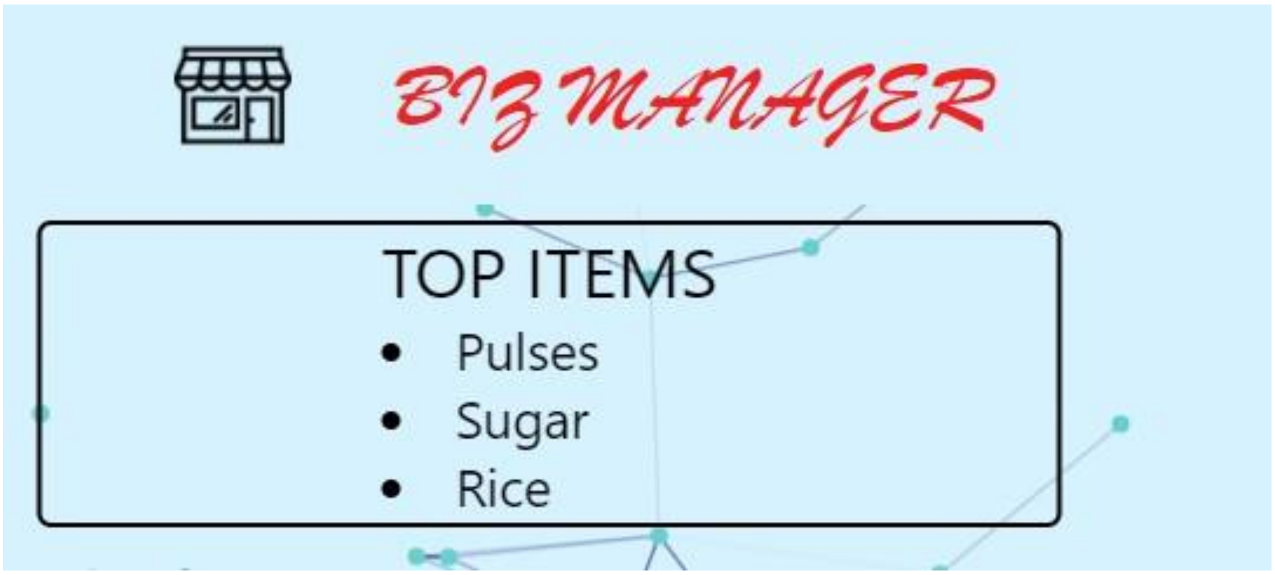
```
app.get('/salehistory', (req, res) => {
                const user = req.query.user
                const filter = req.query.filter
                if(filter==="true")
                            var qry = "select s.date,s.sale_count,s.profit,s.user,
                                          p.product_name,p.product_price from
                                          sale2 s, product p where
                                          s.product_id=p.product_id and s.user=?
                                          order by s.date desc;“
               else if(filter==="false")
                            var qry = "select s.date,s.sale_count,s.profit,
```

CONTINUED..

```
                                                    s.user,p.product_name,p.product_price
                                                    from sale2 s, product p where
                                                    s.product_id=p.product_id and s.user=?
                                                    order by s.date;“
   db.query(qry, [user], (err, result) => {
                        if (err) {
                                          console.log(err)
                                    }
                       else {
                                         console.log("sale history extracted")
                                         return res.send(result)
                               }
                })
})
```

CONTINUED..

TOP ITEMS :
Displays the best sellers of the store.



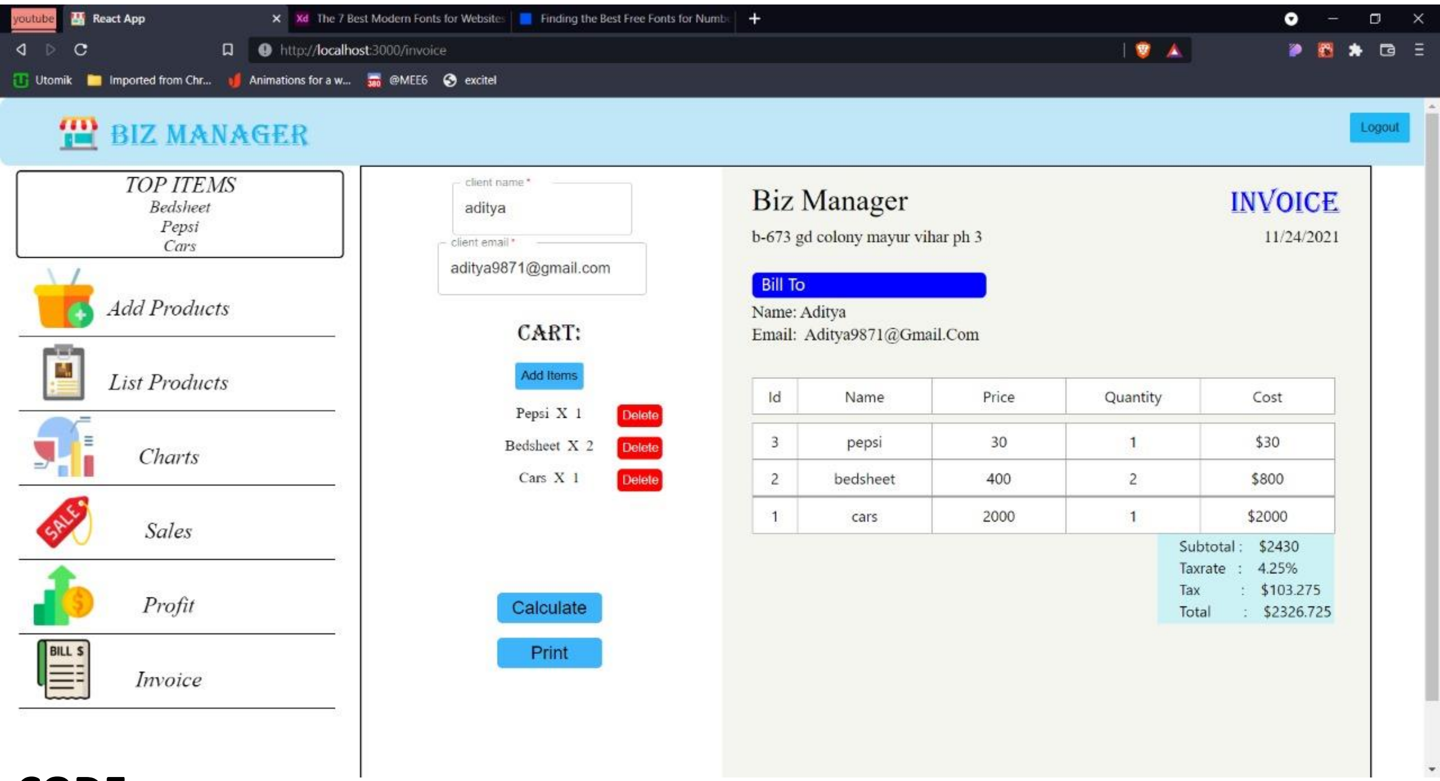
TOP ITEMS
- Pulses
- Sugar
- Rice

CODE:

```
Top Productsapp.get('/top_product', (req, res) => {
            const user = req.query.user
            var qry = 'select product_id,product_name,total_sale from
                        product where user= ? order by total_sale desc limit
                        3;'
  db.query(qry, [user], (err, result) => {
                 if (err) {
                                 console.log(err)
                        }
                 else {
                                  console.log("extracted")
                                  return res.send(result)
                         }
             })
})
```

CONTINUED..

INVOICE :.



client name: aditya
client email: aditya9871@gmail.com

CART:

Pepsi X 1
Bedsheet X 2
Cars X 1

Biz Manager
b-673 gd colony mayur vihar ph 3

INVOICE
11/24/2021

Bill To
Name: Aditya
Email: Aditya9871@Gmail.Com

Id	Name	Price	Quantity	Cost
3	pepsi	30	1	$30
2	bedsheet	400	2	$800
1	cars	2000	1	$2000

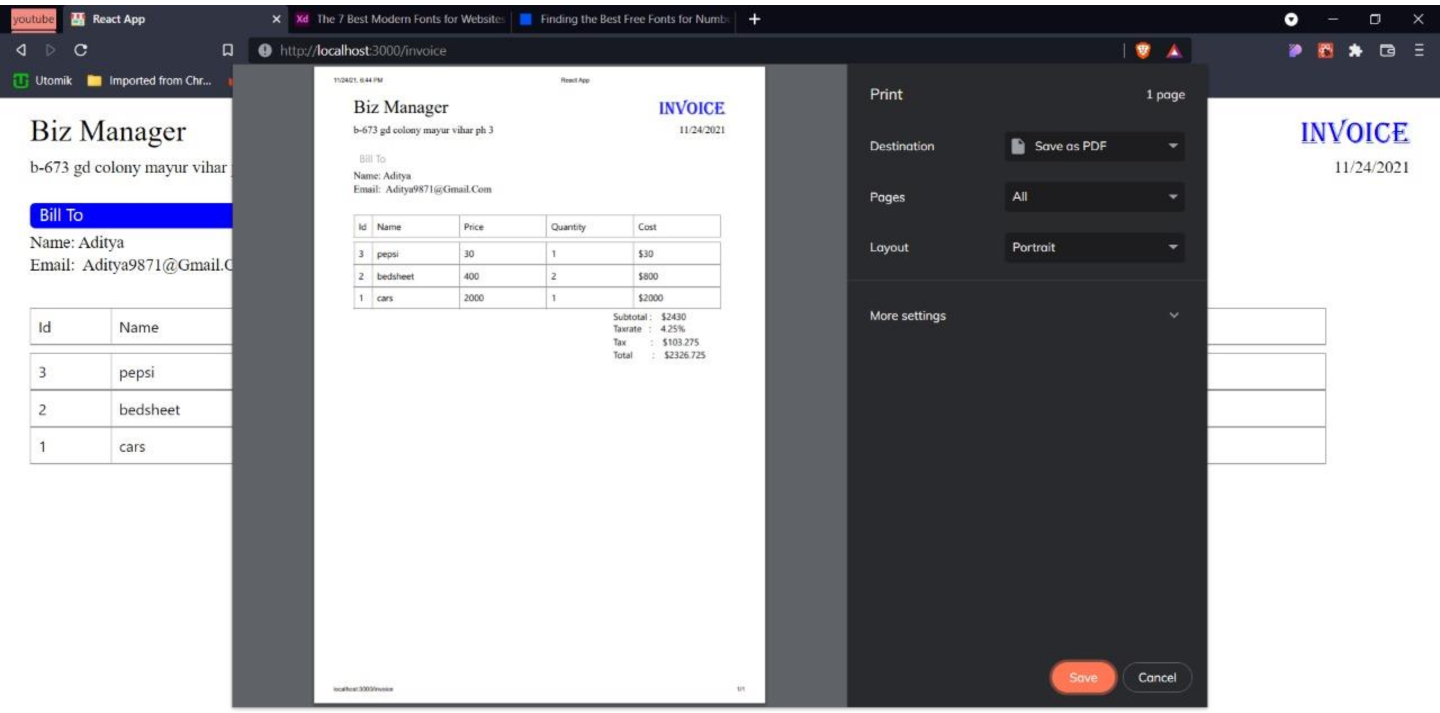
Subtotal :	$2430
Taxrate :	4.25%
Tax :	$103.275
Total :	$2326.725

CODE:

```
app.get('/newinvoicedata', (req, res) => {
                const user = req.query.user
                const client = req.query.client
                var qry = "SELECT * FROM tempinvoice where user=? and
                             client=?;“
                db.query(qry, [user, client], (err, result) => {
                   if (err) {
                                  console.log(err)
                             }
                   else {
                                  console.log("extracted temp invoice details")
                                  return res.send(result)
                           }
             })
}
```

CONTINUED..

PRINTING INVOICE :.


Biz Manager

b-673 gd colony mayur vihar ph 3

INVOICE

11/24/2021

Bill To

Name: Aditya

Email: Aditya9871@Gmail.Com

Id	Name	Price	Quantity	Cost
3	pepsi	30	1	$30
2	bedsheet	400	2	$800
1	cars	2000	1	$2000

```
Subtotal :  $2430
Taxrate  :  4.25%
Tax      :  $103.275
Total    :  $2326.725
```

Print: 1 page

Destination: Save as PDF

Pages: All

Layout: Portrait

CODE:

```
const printinvoice=(val)=>{
      var backup=document.body.innerHTML;
       var divcontent=document.getElementById(val).innerHTML;
       document.body.innerHTML=divcontent;
       window.print();
       document.body.innerHTML=backup;
       window.location.reload();
}
```

BIBLIOGRAPHY

1. Fundamentals of DATABASE SYSTEMS –Ramez-Elmasri, Shamakant-B-Navathe

https://iran-lms.com/images/images/Books/PDF/Fundamentals-of-Database-Systems-Pearson-2015-Ramez-Elmasri-Shamkant-B.-Navathe.pdf

2. Database System Concepts –H.F. Korth, Silberschatz

http://vidyadhancollege.org/wp-content/uploads/2017/05/Database-System-Concepts.pdf

3. https:/reactjs.org
4. Teacher's notes
5. Stackoverflow links

https://stackoverflow.com/questions/29716543/form-validation-using-javascript/29717516

6. Youtube links

https://www.youtube.com/watch?v=j942wKiXFu8&list=PL4cUxeGkcC9gZD-Tvwfod2gaISzfRiP9d

https://www.youtube.com/watch?v=LlvBzyy-558