Farmer Representator

Database Systems Lab Semester Project

Session: 2021 – 2025

Submitted by:

Group-39

Muhammad Danish	2021-CS-167
Muhammad Ibrahim	2021-CS-169
Abdul Mateen	2021-CS-190
Muhammad Ahmad	2021-CS-211

Supervised by:

Mr. Nazeef Ul Haq

Mr. Numan Babar

Mr. Samyan Qayyum Wahla

Department of Computer Science and Engineering

University of Engineering and Technology

Lahore Pakistan

Contents

List of Figures

List of Tables

Project Description

Farmers in Pakistan cultivate a variety of crops, including wheat, rice, sugarcane, cotton, maize, and fruits and vegetables. They are predominantly smallholders, with most farming operations conducted on small plots of land. Farmers are the backbone of our country's economy. But this sector is in much trouble as compared to others. We have to facilitate them as much as possible. Overall, farmers and agriculture play a critical role in Pakistan's economy and society, and efforts to improve the productivity and resilience of the sector are essential for poverty reduction and sustainable economic growth in the country. So, what needs to be addressed?

Farmers often face numerous challenges when it comes to selling their crops. Some of the common issues they face include:

1. **Lack of market access:** Many farmers, especially those in remote areas, lack access to markets, which makes it difficult for them to sell their produce.
2. **Limited bargaining power:** Farmers often have limited bargaining power due to the fragmented nature of the agricultural market and the dominance of middlemen.
3. **Quality concerns:** Buyers are often concerned about the quality of the produce, which can lead to rejection or lower prices.
4. **Post-harvest losses:** Farmers may incur losses due to inadequate storage facilities, lack of transportation facilities, and spoilage.
5. **Lack of market information:** Many farmers lack access to timely and accurate market information, which can result in poor pricing decisions.

To address these issues, we are facilitating our farmers with the possible comfort and ease we can provide. So, we will perform our role as a mediator between farmers and the market. To address **issue (1) & (3)**, simply farmer, when cultivating their crops, has to reach us. We will store his product with proper care and add farmers and the product record in our system. As soon as, products are entered in the system, they are available to the market. The system is connected with multiple manufacturing companies which buy the products. **(Addressing issue(2)**) As the manufacturer buys the product, the corresponding farmer will get the profit. The company keeps in mind to benefit farmers as much as possible.

(Addressing issue(4)) As the farmer's product start getting sold, he starts getting market knowledge after analyzing his product reports.

We are also providing transport facilities to farmers to move their products from one place to another at a much more reasonable cost.
Overall this system is for facilitating farmers. Because they are the backbone of our country. If a farmer grows, it is beneficial for both the people and the country as well. So, the addressed issues are completely fulfilled by this project.

Project Actors

The followings are the actors and stakeholders of the system:

Actors

1. CEO
2. Regional Head
3. Managers (farmer, transport, Organization)
4. Farmers
5. Drivers
6. Organizations

Stakeholders

1. Farmers
2. Transport Drivers
3. Organizations(which will place orders)

Project Features

Expected Features for the project are: All the actors of the systems have different features which are as follows:

CEO

CEO will have the following features:

1. Add/update regional heads(Generate new regions)
2. Generate all types of business reports.

Regional Head

Regional Head will have the following features:

1. Add/update different managers(farms, transport, marketplace) with respect to region.
2. Generate all types of region-level business reports.

Farmer Manager

The farmer manager will have the following features:

1. Add/update different farmers.
2. Approve the requests of new farmers.
3. Approve fields added by the farmer.
4. Receive crop requests from farmers and respond to them.
5. Approve/disapprove crops requested to be sold by the farmer.
6. Generate farm-related business-level reports.

Transport Manager

The transport manager will have the following features:

1. Add/update different transport drivers.
2. Send available drivers to fields approved by the farm manager for crops to transport them to the warehouse.

3. Receive requests from drivers and respond to them.
4. Generate transport-related business-level reports.

Organization Manager

The Marketplace manager will have the following features:

1. Add/update different organizations.
2. Receive and fulfill orders placed by organizations.
3. Request a transport manager for the transport of orders from the warehouse to the organization.
4. Access to all the data related to the warehouse.
5. Generate sales-related business-level reports.

Farmer

A farmer will have the following features:

1. Create/update account.
2. Add his fields in the system.
3. Request farm manager for the sale of his crops.
4. Can view previous history of requests for sale, their corresponding status (approved or disapproved), and payments.
5. Generate related reports.

Driver

A driver will have the following features:

1. Create/update account.
2. Register his vehicle in the system which will be approved by the transport manager.
3. View the previous history of orders delivered, and their corresponding payments.
4. Generate related reports.

Organization

An Organization will have the following features:

1. Create/update account.
2. Place Order.
3. View Products and history.
4. Generate related reports.

Technology Stack

Language	C#
Backend Framework	Asp.Net Core
Platform	Desktop
Frontend Technology	Guna Framework .NET
IDEs	Visual Studio Community 2022
DBMS	MS SQL Server

Use Cases

Use Case Id	U001
Name	Login Form
Actor	Employee, Driver, Farmer
PreCondition	?
Description	?
Queries	Q-1
Flow	**Main Success Scenario:** 1. Enter Email 2. Enter Password 3. Click Enter **Alternative Flows:** 1. Account is not verified 2. Password not Entered 3. Email not Entered 4. Email or Password incorrect 5. Forget Password
post Condition	?
GUI	Image

Use Case Id	U002
Name	SignUp Form
Actor	Employee, Driver, Farmer, Organization
PreCondition	?
Description	?
Queries	Q-2,3,4,5,6,7,8,9,10,30
Flow	**Main Success Scenario:** 1. Enter all required details 4. Click Enter **Alternative Flows:** 1. Select Role 2. Password not Entered 3. Email not Entered 4. Username or Password incorrect 5. Password does'nt meet the criteria 6. Email already Taken 7. Already have Account
post Condition	?
GUI	Image

Use Case Id	U003
Name	Forget Password Form
Actor	Employee, Driver, Farmer
PreCondition	?
Description	?
Queries	
Flow	**Main Success Scenario:** 1. Enter Email 2. Click Send OTP 3. Enter oTP 4. Click Enter **Alternative Flows:** 1. Email does not exist 2. OTP does not exist
post Condition	?
GUI	Image

Use Case Id	U004
Name	Add Field Form
Actor	Farmer
PreCondition	?
Description	?
Queries	Q-11, Q-12
Flow	**Main Success Scenario:** 1. Select Region 2. Enter Location **Alternative Flows:** 1. Region not selected 2. Limit exceeded
post Condition	?
GUI	Image

Use Case Id	U005
Name	Update Field Form
Actor	Farmer
PreCondition	?
Description	?
Queries	
Flow	**Main Success Scenario:** 1. Change Region 2. Change Location **Alternative Flows:** 1. Region not valid 2. Location not valid
post Condition	?
GUI	Image

Use Case Id	U006
Name	Withdraw Cash Form
Actor	Employee, Driver, Farmer
PreCondition	?
Description	?
Queries	Q13, Q-14
Flow	**Main Success Scenario:** 1. Enter Amount 2. Enter Credit Card Number 3. Click Withdraw **Alternative Flows:** 1. Amount is not greater than zero. 2. Incorrect Credit card nuber.
post Condition	?
GUI	Image

Use Case Id	U007
Name	TopUp Form
Actor	Employee, Driver, Farmer
PreCondition	?
Description	?
Queries	Q-13, Q-14
Flow	**Main Success Scenario:** 1. Enter Amount 2. Enter Credit Card Number 3. Click TopUp **Alternative Flows:** 1. Amount is not greater than zero. 2. Incorrect Credit card nuber.
post Condition	?
GUI	Image

Use Case Id	U008
Name	Add Crop Form
Actor	Farmer
PreCondition	?
Description	?
Queries	Q-15
Flow	**Main Success Scenario:** 1. Enter Name 2. Enter Price 3. Select Category 4. Click Add **Alternative Flows:** 1. Price is not greater than zero. 2. Name not selected. 3. Quantity is not greater than zero. 4. Category not selected.
post Condition	?
GUI	Image

Use Case Id	U009
Name	Change Settings Form
Actor	Employee, Driver, Farmer
PreCondition	?
Description	?
Queries	Q-16,17,18
Flow	**Main Success Scenario:** 1. Change Name 2. Change Contact 3. Change Password 4. Click Done **Alternative Flows:** 1. Name Limit. 2. Contact Limit. 3. Password does not match 4. Category not selected.
post Condition	?
GUI	Image

Use Case Id	U010
Name	Deliveries Form
Actor	Driver
PreCondition	?
Description	?
Queries	Q-18
Flow	**Main Success Scenario:** 1. Click Accept Delivery 2. Delivery changes to delivering **Alternative Flows:** 1. Reject Delivery. 2. Status changes back to pending.
post Condition	?
GUI	Image

Use Case Id	U011
Name	Add Vehicle Form
Actor	Driver
PreCondition	?
Description	?
Queries	Q-19
Flow	**Main Success Scenario:** 1. Enter Registration Number 2. Click Enter **Alternative Flows:** 1. Registration Number already exists. 2. Vehicle Limit exceeded.
post Condition	?
GUI	Image

Use Case Id	U012
Name	Update Vehicle Form
Actor	Driver
PreCondition	?
Description	?
Queries	Q-20
Flow	**Main Success Scenario:** 1. Change Registration Number 2. Click Enter **Alternative Flows:** 1. Registration Number already exists. 2. Vehicle Limit exceeded.
post Condition	?
GUI	Image

Use Case Id	U013
Name	Approve/Reject Farmer Request Form
Actor	Farmer Manager, Region Head
PreCondition	?
Description	?
Queries	Q-21
Flow	**Main Success Scenario:** 1. Accept the request 2. Farmer's status change to approved **Alternative Flows:** 1. Reject the request.
post Condition	?
GUI	Image

Use Case Id	U014
Name	Approve/Reject Field Request Form
Actor	Farmer Manager, Region Head
PreCondition	?
Description	?
Queries	Q-22
Flow	**Main Success Scenario:** 1. Accept the request 2. Field's status change to approved **Alternative Flows:** 1. Reject the request.
post Condition	?
GUI	Image

Use Case Id	U015
Name	Assign Tasks to driver Form
Actor	Transport Manager, Region Head
PreCondition	?
Description	?
Queries	Q-23,24,25
Flow	**Main Success Scenario:** 1. Select Order 2. Select Driver 3. Click Assign **Alternative Flows:** 1. Order not selected. 2. driver not selected.
post Condition	?
GUI	Image

Use Case Id	U016
Name	Approve/Reject Driver Request Form
Actor	Driver Manager, Region Head
PreCondition	?
Description	?
Queries	Q-26
Flow	**Main Success Scenario:** 1. Accept the request 2. Driver's status change to approved **Alternative Flows:** 1. Reject the request.
post Condition	?
GUI	Image

Use Case Id	U017
Name	Approve/Reject Vehicle Request Form
Actor	Transport Manager, Region Head
PreCondition	?
Description	?
Queries	Q-22
Flow	**Main Success Scenario:** 1. Accept the request 2. Field's status change to approved **Alternative Flows:** 1. Reject the request.
post Condition	?
GUI	Image

Use Case Id	U018
Name	Change Active Vehicle Form
Actor	Driver
PreCondition	?
Description	?
Queries	Q-27
Flow	**Main Success Scenario:** 1. Click on activate 2. Accept pop up **Alternative Flows:** 1. Click on already active vehicle
post Condition	?
GUI	Image

Use Case Id	U019
Name	Accept/Reject Order Form
Actor	Org. Manager, Region Head
PreCondition	?
Description	?
Queries	Q-28
Flow	**Main Success Scenario:** 1. Accept the Order 2. Order's status change to approved **Alternative Flows:** 1. Reject Order
post Condition	?
GUI	Image

Use Case Id	U020
Name	Accept/Reject Organization Form
Actor	Org. Manager, Region Head
PreCondition	?
Description	?
Queries	Q-29
Flow	**Main Success Scenario:** 1. Accept the Organization 2. Organization's status change to approved **Alternative Flows:** 1. Reject Request
post Condition	?
GUI	Image

Use Case Id	U021
Name	Delete Field Form
Actor	Farmer
PreCondition	?
Description	?
Queries	Q-14
Flow	**Main Success Scenario:** 1. Click on Delete 2. Field's status change to deleted
post Condition	?
GUI	Image

Use Case Id	U022
Name	Change Driver's Status Form
Actor	Driver
PreCondition	?
Description	?
Queries	Q-26
Flow	**Main Success Scenario:** 1. Click on offline 2. Status changes to offline
post Condition	?
GUI	Image

Use Case Id	U023
Name	Bids Form
Actor	Organization
PreCondition	?
Description	?
Queries	
Flow	**Main Success Scenario:** 1. Select Product 2. Enter Price 3. Enter number of units 4. Click Enter **Alternative Flows:** 1. Price is not greater than the average price. 2. Product not selected. 3. Number of units not available. 4. Units not entered.
post Condition	?
GUI	Image

User Interface Details

ucfbur

Classes

Credentials

```
public class Credentials
{
    private int Id { get; set; }
    private string Email { get; set; }
    private int Pasword { get; set; }
    private int Type { get; set; }

}
```

Delivery

```
public class Delivery
{
    private int Id { get; set; }
    private int VehicleId { get; set; }
    private int DeliveryType { get; set; }
    private int SourceId { get; set; }
    private int DestinationId { get; set; }
    private int ProductId { get; set; }
    private int CurrentStatus { get; set; }
    private int ManagerId { get; set; }
}
```

Delivery Details

```
public class DeliveryDetails
{
    private int DeliveryId { get; set; }
    private int Status { get; set; }
    private DateTime TimeStamp { get; set; }
}
```

Driver

```
public class Driver
{
    private int Id { get; set; }
    private int DrivingLicence { get; set; }
    private int Status { get; set; }
    private int ManagerId { get; set; }
    private int AccountType { get; set; }

}
```

Employee

```
public class Employee
{
    private int Id { get; set; }
    private int Salary { get; set; }
    private int Designation { get; set; }
    private int Status { get; set; }
}
```

Farmer

```
public class Farmer
{
    private int Id { get; set; }
    private int Status { get; set; }
    private string Address { get; set; }
    private int ManagerId { get; set; }
    private int AccountType { get; set; }
}
```

Field

```
public class Field
{
    private int Id { get; set; }
    private int FarmerId { get; set; }
    private int RegionId { get; set; }
    private string Address { get; set; }
    private int Status { get; set; }
    private string ManagerId { get; set; }
}
```

FieldProduct

```
public class FieldProduct
{
    public int FieldId { get; set; }
    public int ProductId { get; set; }

}
```

Lookup

```
public class Lookup
{
    private int Id { get; set; }
    private string Value { get; set; }
    private int Category { get; set; }
}
```

Office

```
public class Office
{
    private int Id { get; set; }
    private string Address { get; set; }
    private int WalletId { get; set; }
}
```

Order

```
public class Order
{
    private int Id { get; set; }
    private int ProductId { get; set; }
    private int Qunatity { get; set; }
    private int RequestedPrice { get; set; }
    private int Status { get; set; }
    private int OrganizationId { get; set; }
    private DateTime OrderDate { get; set; }
    private int ManagerId { get; set; }
}
```

Organization

```
public class Organization
{
    private int Id { get; set; }
    private string OrganizationName { get; set; }
    private string Email { get; set; }
    private string Address { get; set; }
    private int RegionId { get; set; }
    private int WalletId { get; set; }
    private int CredentailId { get; set; }
}
```

Person

```
public class Person
{
    private int Id { get; set; }
    private string FirstName { get; set; }
    private string LastName { get; set; }
    private string PhoneNo { get; set; }
    private int CNIC { get; set; }
    private string Email { get; set; }
    private int RegionId { get; set; }
    private DateTime RegistrationDate { get; set; }
    private int Gender { get; set; }
    private int WalletId { get; set; }
    private int CredentialId { get; set; }

}
```

Product

```
public class Product
{
    private int Id { get; set; }
    private string Name { get; set; }
    private int Quality { get; set; }
    private int UnitPrice { get; set; }
    private int CategoryId { get; set; }
    private int Status { get; set; }
    private int RemainingUnits { get; set; }
    private int ManagerId { get; set; }

}
```

Region

```
public class Region
{
    private int RegionId { get; set; }
```

```
        private string RegionName { get; set; }
    }
```

Subscribtions Details

```
public class SubscribtionsDetails
{
    private int PersonId { get; set; }
    private DateTime SubscribtionDate { get; set; }

}
```

Transaction Details

```
public class TransactionDetails
{
    private int Id { get; set; }
    private int RecevierId { get; set; }
    private int SenderId { get; set; }
    private int Amount { get; set; }
    private int TransactionType { get; set; }
    private DateTime TimeStamp { get; set; }
}
```

Vehicle

```
public class Vehicle
  {
  private int Id { get; set; }
  private int DriverId { get; set; }
  private int RegistrationNo { get; set; }
  private DateTime RegistraionDate { get; set; }
  private int VehicleStatus { get; set; }
  private int ManagerId { get; set; }
  }
```

Wallet

```
public class Wallet
{
    private int Id { get; set; }
    private int Type { get; set; }
    private int TotalAmount { get; set; }
}
```

Warehouse

```
    public class Warehouse
    {
        private int Id { get; set; }
        private int MaxCapacity { get; set; }
        private string Address { get; set; }
    }
```

Warehouse Product

```
    public class WarehouseProduct
    {
        private int WarehouseId { get; set; }
        private int ProductId { get; set; }
    }
```

ER Design

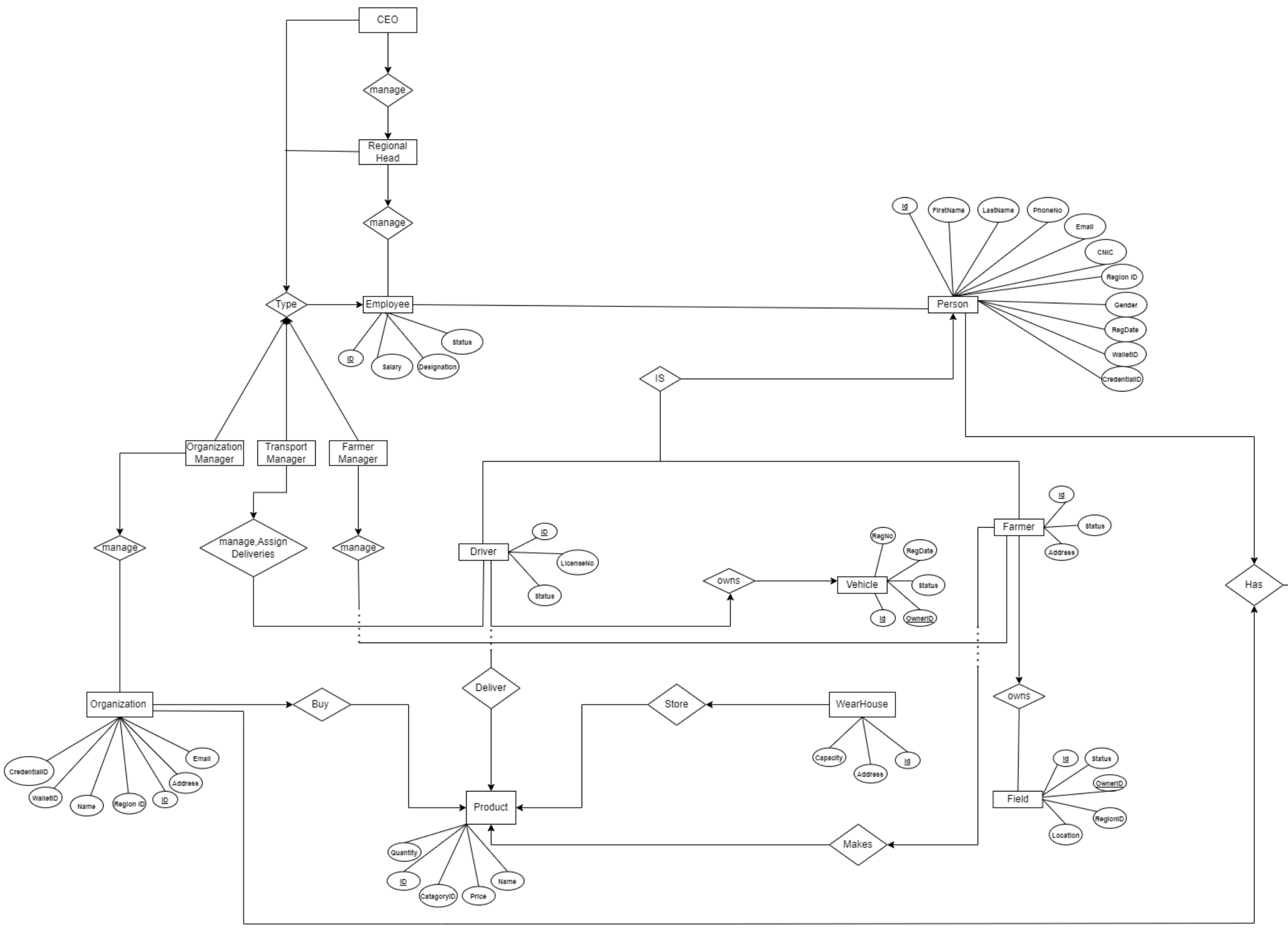

Figure 1: ER Diagram

Transactions

ucfbur

Views

V-1

```
Create View [Delivery_DriverView] as
    SELECT D.Id, D.SourceId,
CASE
WHEN L.Value='FieldToWarehouse'
        THEN (SELECT F.Address FROM Field F WHERE F.Id=D.SourceId)
WHEN L.Value='WarehouseToOrganization'
        THEN (SELECT W.Address FROM Warehouse W WHERE W.Id=D.SourceId)
WHEN L.Value='WarehouseToWarehouse'
        THEN (SELECT W.Address FROM Warehouse W
        WHERE W.Id=D.SourceId)
END AS 'SourceAddress'
, D.DestinationId,
CASE
WHEN L.Value='FieldToWarehouse'
        THEN (SELECT W.Address FROM Warehouse W
        WHERE W.Id=D.DestinationId)
WHEN L.Value='WarehouseToOrganization'
        THEN (SELECT O.Address FROM Organization O
        WHERE O.Id=D.DestinationId)
WHEN L.Value='WarehouseToWarehouse'
        THEN (SELECT W.Address FROM Warehouse W
        WHERE W.Id=D.DestinationId)
END AS 'DestinationAddress'
FROM Delivery D
JOIN Lookup L ON L.Id=D.DeliveryType
JOIN Lookup L_ ON L_.Id=D.CurrentStatus
WHERE L_.Category='DeliveryStatus' AND L_.Value='Assigned'
```

V-2

```
CREATE VIEW WarehouseProduct_FarmerManager
as
SELECT P.Id,P.Name, P.Quantity,
P.UnitPrice, P.RemaningUnits, W.Address
FROM Product P
JOIN WarehouseProduct WP ON WP.ProductId=P.Id
JOIN Warehouse W ON W.Id=WP.WarehouseId
JOIN Lookup L ON L.Id=P.Status
WHERE L.Category='ProductStatus' AND L.Value='Available'
```

V-3

```
CREATE VIEW FarmerFields_FarmerManager
as
SELECT F.FarmerId, CONCAT(P.FirstName ,
' ', P.LastName) Name, FE.Address 'FieldAddress',
R.RegionName 'Region',L.Value 'Status'
FROM Farmer F
JOIN Field FE ON FE.FarmerId=F.FarmerId
JOIN Person P ON P.Id=F.FarmerId
JOIN Region R ON R.Id=FE.RegionId
JOIN Lookup L ON L.Id=FE.Status
```

V-4

```
CREATE VIEW OrganizationDetails_OrgManager
as
SELECT O.Id, O.OrganizationName, O.Address,
C.Email,R.RegionName 'Region', L.Value 'Status'
FROM Organization O
JOIN Credentails C ON C.Id=O.CredentailId
JOIN Region R ON R.Id=O.RegionId
JOIN Lookup L ON L.Id=O.Status
```

V-5

```
CREATE VIEW ManagerDetails_RegionHead
as
SELECT E.EmployeeId, CONCAT(P.FirstName,' ', P.LastName)
Name, C.Email,L2.Value 'Designation',
L.Value 'Status',L1.Value 'Gender', E.Salary
FROM Employee E
JOIN Person P ON P.Id=E.EmployeeId
JOIN Lookup L ON L.Id=E.Status
JOIN Lookup L1 ON L1.Id=P.Gender
JOIN Lookup L2 ON L2.Id=E.Designation
JOIN Credentails C ON C.Id=P.CredentailId
```

V-6

```
CREATE VIEW RegionHeadDetails_CEO
as
SELECT E.EmployeeId, CONCAT(P.FirstName,' ', P.LastName)
Name, C.Email, L.Value 'Status',R.RegionName 'Region',P.RegDate 'Registered',L1
FROM Employee E
JOIN Person P ON P.Id=E.EmployeeId
JOIN Lookup L ON L.Id=E.Status
JOIN Lookup L1 ON L1.Id=P.Gender
JOIN Lookup L2 ON L2.Id=E.Designation
JOIN Credentails C ON C.Id=P.CredentailId
JOIN Region R ON R.Id=P.RegionId
WHERE L2.Value='RegionHead'
```

Stored Procedures

S-1

```
GO
CREATE PROCEDURE stpGetFieldsofFarmer
@FarmerId int
 AS
 BEGIN
 -- SET NOCOUNT ON added to prevent extra result sets from
 -- interfering with SELECT statements.
 SET NOCOUNT ON;

 -- Select statements for procedure here
 Select * from Field where FarmerId=@FarmerId
END
GO
```

S-2

```
 GO
CREATE PROCEDURE stpGetPendingRequests_Farmers
 AS
 BEGIN
 -- SET NOCOUNT ON added to prevent extra result sets from
 -- interfering with SELECT statements.
 SET NOCOUNT ON;

 -- Select statements for procedure here
 Select F.FarmerId, CONCAT(P.FirstName,' ', P.LastName)
 Name,P.RegDate, R.RegionName,F.Address
 from Farmer F
 JOIN Lookup L ON L.Id=F.Status
 JOIN Person P ON F.FarmerId=P.Id
 JOIN Region R ON R.Id=P.RegionId
 where L.Category='FarmerStatus' AND L.Value='PENDING'
END
GO
```

S-3

```
  CREATE PROCEDURE stpGetPendingRequests_Drivers
 AS
 BEGIN
 -- SET NOCOUNT ON added to prevent extra result sets from
 -- interfering with SELECT statements.
 SET NOCOUNT ON;

 -- Select statements for procedure here
 Select D.DriverId, CONCAT(P.FirstName,' ', P.LastName)
 Name,P.RegDate, R.RegionName,D.Address
 from Driver D
 JOIN Lookup L ON L.Id=D.Status
 JOIN Person P ON D.DriverId=P.Id
 JOIN Region R ON R.Id=P.RegionId
 where L.Category='DriverStatus' AND L.Value='PENDING'
END
GO
```

S-4

```
  CREATE PROCEDURE stpGetPendingRequests_Organizations
 AS
 BEGIN
 -- SET NOCOUNT ON added to prevent extra result sets from
 -- interfering with SELECT statements.
 SET NOCOUNT ON;

 -- Select statements for procedure here
 Select O.Id, O.OrganizationName,R.RegionName,O.Address
 from Organization O
 JOIN Lookup L ON L.Id=O.Status
 JOIN Region R ON R.Id=O.RegionId
 where L.Category='OrganizationStatus' AND L.Value='PENDING'
END
GO
```

S-5

```
CREATE PROCEDURE stpGetPendingRequests_Fields
 AS
 BEGIN
 -- SET NOCOUNT ON added to prevent extra result sets from
 -- interfering with SELECT statements.
 SET NOCOUNT ON;

 -- Select statements for procedure here
 Select F.Id,F.FarmerId, F.Address, R.RegionName
 from Field F
 JOIN Lookup L ON L.Id=F.Status
 JOIN Region R ON R.Id=F.RegionId
 where L.Category='FieldStatus' AND L.Value='PENDING'
END
GO
```

S-6

```
CREATE PROCEDURE stpGetPendingRequests_Products
 AS
 BEGIN
 -- SET NOCOUNT ON added to prevent extra result sets from
 -- interfering with SELECT statements.
 SET NOCOUNT ON;

 -- Select statements for procedure here
 Select P.Id, P.Name, P.Quantity, P.UnitPrice, L.Value Category
 from Product P
 JOIN Lookup L ON L.Id=P.CategoryId

 where L.Category='ProductStatus' AND L.Value='REQUEST'
END
GO
```

S-7

```
  CREATE PROCEDURE stpGetVehicles_Driver
@DriverId int
 AS
 BEGIN
 -- SET NOCOUNT ON added to prevent extra result sets from
 -- interfering with SELECT statements.
 SET NOCOUNT ON;

 -- Select statements for procedure here
 Select V.Id, V.RegNo, V.SystemRegDate, L.Value Status
 froM Vehicle V
 JOIN Lookup L ON L.Id=V.VehicleStatus

 where L.Category='VehicleStatus'
END
GO
```

S-8

```
CREATE PROCEDURE stpGetDelivery_Driver
@VehicleId int
 AS
 BEGIN
 -- SET NOCOUNT ON added to prevent extra result sets from
 -- interfering with SELECT statements.
 SET NOCOUNT ON;

 -- Select statements for procedure here
SELECT D.Id, D.SourceId,
CASE
WHEN L.Value='FieldToWarehouse'
        THEN (SELECT F.Address FROM Field F WHERE F.Id=D.SourceId)
WHEN L.Value='WarehouseToOrganization'
        THEN (SELECT W.Address FROM Warehouse W
        WHERE W.Id=D.SourceId)
WHEN L.Value='WarehouseToWarehouse'
        THEN (SELECT W.Address FROM Warehouse W
        WHERE W.Id=D.SourceId)
```

```
END AS 'SourceAddress'
, D.DestinationId,
CASE
WHEN L.Value='FieldToWarehouse'
        THEN (SELECT W.Address FROM Warehouse W
        WHERE W.Id=D.DestinationId)
WHEN L.Value='WarehouseToOrganization'
        THEN (SELECT O.Address FROM Organization O
        WHERE O.Id=D.DestinationId)
WHEN L.Value='WarehouseToWarehouse'
        THEN (SELECT W.Address FROM Warehouse W
        WHERE W.Id=D.DestinationId)
END AS 'DestinationAddress'
FROM Delivery D
JOIN Lookup L ON L.Id=D.DeliveryType
JOIN Lookup L_ ON L_.Id=D.CurrentStatus
WHERE L_.Category='DeliveryStatus' AND L_.Value='Assigned'

END
GO
```

Triggers

ucfbur

Indexes

ucfbur

Exceptions

ucfbur

Project Plan

ucfbur

Queries

Q-1 Check account type

```
Select L.id
from Credentails C
join lookup L
on C.Type = L.Id
where c.Email = '{Credentials.Email}'
& C.Password = '{Credentials.Password}'
```

Q-2 Create credentials

```
insert into Credentails
values('{credentials.email}','{credentials.password}','
{Credentials.type}')
```

Q-3 Create wallet

```
insert into wallet
values('{wallet.type}',0)
```

Q-4 Create person

```
insert into person values('{Person.FirstName}','{Person.Lastname}','
{Person.PhoneNo}','{Person.CNIC}','
{Person.RegionId}','{Person.RegDate}','
{Person.Gender}','{Person.Walletid}','{Person.CredentialId}')
```

Q-5 Get Region Id

```
Select id
from region R
where R.RegionName = '@str'
```

Q-6 Get Walletid

```
Select MAX(id)
from wallet
```

Q-7 Get Credential id

```
Select MAX(id)
from Credentails
```

Q-8 Create Farmer

```
insert into farmer values('{Farmer.id}','{Farmer.Status}','
{Farmer.Address}',NULL,'{Farmer.AccountType}')
```

Q-9 Create Driver

```
insert into driver values('{Driver.id}','{Driver.Status}','
{Driver.Address}','{Driver.DrivingLicense}',
NULL,'{Driver.AccountType}')
```

Q-10 Create Employee

```
insert into Employee values('{Employee.id}','{Employee.Salary}','
{Employee.Status}','{Employee.Salary}','{Employee.Designation}')
```

Q-11 Create Field

```
insert into Field values('{Field.Farmerid}','{Field.Regionid}','
{Field.Address}','{Field.Status}',NULL)
```

Q-12 Update Field's Region, location

```
Update Field
set RegionId = '{Field.regionid}' , Address = '{Field.Address}'
where Id =  '{Field.id}'
```

Q-13 Alter Amount

```
Update wallet
set TotalAmount = @Amount
where id = @id
```

Q-14 Get Amount

```
Select *
from wallet
where walle
```

Q-15 Add Product

```
Insert into Product values('{Product.Name}','{Product.Quantity}','
{Product.UnitPrice}','{Product.RemainingUnits}'
,NULL,'{Product.Status}','{Product.Categoryid}')
```

Q-16 Update Person

```
Update person
set FirstName = '{person.firstname}',
LastName = '{person.Lastname}'
, PhoneNo = '{person.PHoneno}'
Where id = '{person.id}'
```

Q-17 Change Password

```
Update Credentails
set Password = @password
where Id = '{person.credentialid}'
```

Q-18 Change Delivery Status

```
update Delivery
set CurrentStatus = @status
```

Q-19 Add Vehicle

```
insert into Vehicle values('{Vehicle.RegNo}','{Vehicle.SystemRegDate}','
{Vehicle.Driverid}','{Vehicle.Vehiclestatus}','{Vehicle.Managerid}')
```

Q-20 Update Vehicle

```
update vehicle
set RegNo = '{Vehicle.RegNo}',
VehicleStatus = '{Vehicle.Vehiclestatus}'
where id = '{Vehicle.id}'
```

Q-21 Approve/ Disapprove Farmer Request

```
update farmer
set Status = @status
where id = @id
```

Q-22 Approve / Disapprove field request

```
update field
set status = @status
where id = @id
```

Q-23 Get Active Vehicle id from driver id

```
select V.Id
from Vehicle V
join Lookup L
on L.Id = V.VehicleStatus
where L.Category = 'VehicleStatus' and L.Value = 'Active' and           V.Dr
```

Q-24 Assign Product Delivery to Driver

```
insert into Delivery values
(@Vehicleid,@Deliverytype, @Sourceid,
@Destinationid, @productid, @currentstatus,@managerid)
```

Q-25 Get Delivery type(id)

```
select id
from lookup L
where L.Value = @str and L.Category = 'DeliveryType'
```

Q-26 update Driver Status

```
update driver
set status = @status
where id = @id
```

Q-27 change status of vehicle

```
update Vehicle
set VehicleStatus = @status
where id = @id
```

Q-28 change order status

```
update order
set status = @status
where id = @id
```

Q-29 change organization status

```
update Organization
set Status = @status
where id = @id
```

Q-30 Sign up Organization

```
insert into Organization values(
'{Organization.OrganizationName}','
{Organization.Address}','{Organization.Regionid}','
{Organization.walletid}','{Organization.Credentialsid}','
{Organization.Status}')
```

Q-31