# CS 353 - Database Systems

## Project Final Report

**FLover**

## Group 30

Muzaffer Yasin Köktürk - 21703552

Muhammed Berk KÖSE - 21704277

Alptekin Önder - 21602728

İsmail Şahal - 21703596

**Description of the System** **4**

**Final E/R Diagram** **5**

**Relational Schemas** **5**

3.1 User 5
3.2 User Phone Number 6
3.3 Customer 6
3.4 Seller 6
3.5 Courier 6
3.6 Flower 7
3.7 Order 7
3.8 Complaint Report 7
3.9 Customer Service Employee 7
3.10 Forum Entry 8
3.11 Chocolate 8
3.12 Forum Topic 8
3.13 Attached 8
3.14 Fav_shop 9
3.15 Fav_flow 9
3.16 Includes 9
3.17 Stocks 9
3.18 Order Delivery 10

**Implementation Details** **10**

**Advanced Database Features** **11**

5.1 Views 11
5.2 Reports 12
5.3 Trigger 13
5.4 Constraint 13
5.5 Stored Procedure 13

**User Manual** **14**

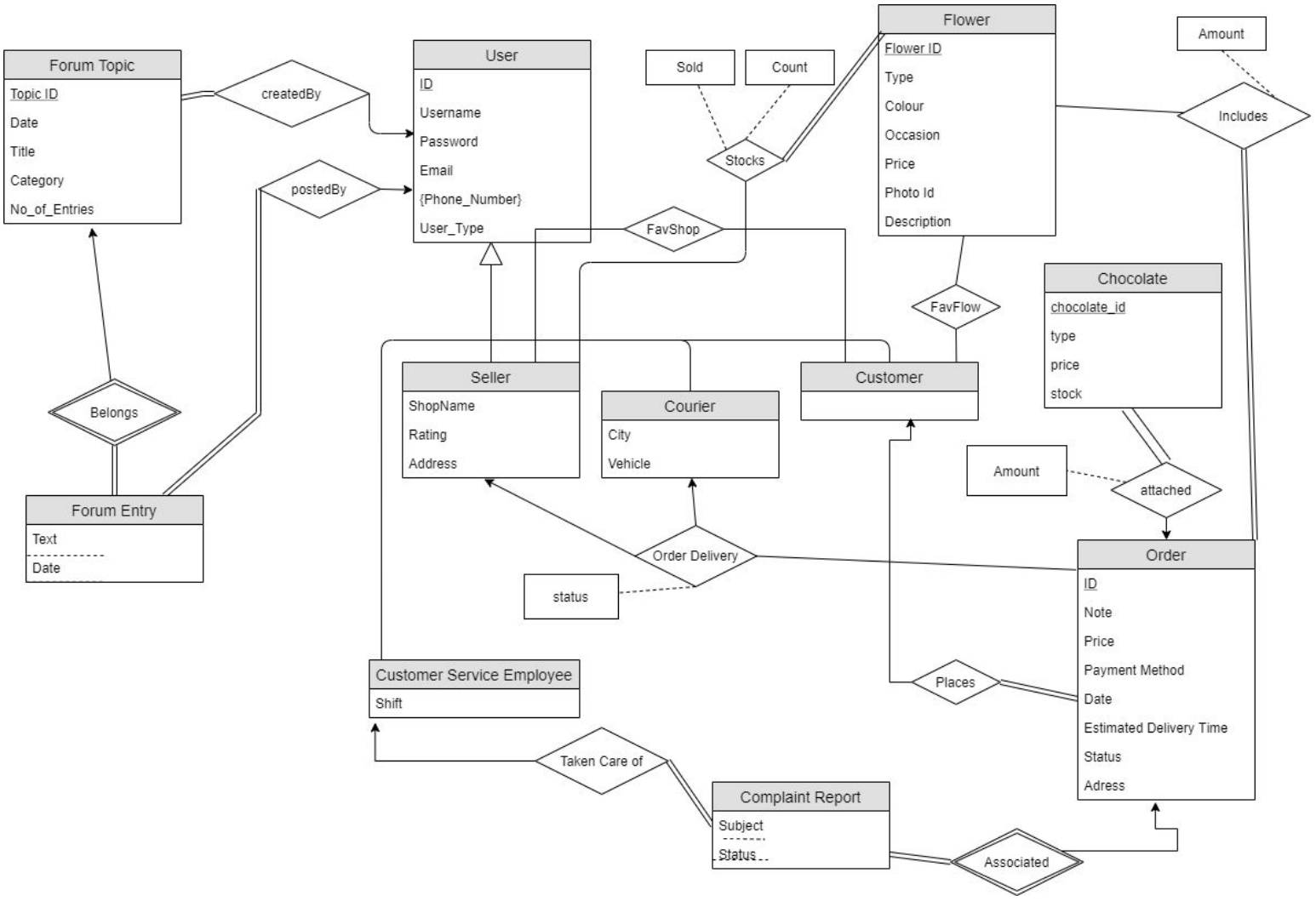
6.1 Login Page 14
6.2 Register Page 15
6.3 Index Page 16
6.4 Products page 17

6.5 Specific product page 17
6.6 About us monstrosity 18
6.7 Add a flower page 18
6.8 Delete flower page 19
6.9 Change product data page 19
6.10 Profile page 20
6.11 Forum Page 21
6.12 Forum Entry 21
6.13 Forum Topic 22
6.14 Create Forum Topic 22
6.15 Received Orders 23
6.16 Customer Service Page 24
6.17 Customer Service Report Page 25

# 1. Description of the System

Our project is developing a website where customers can make flower arrangement orders from a flower seller. Flowers have occasions and categories. Customer can choose a chocolate type to be attached to an order. A seller can add different type of flowers to its shop page. A seller will accept an order and assign it to couriers in the system. Then, a courier can view the assigned orders and choose to accept or decline the assignment. A customer can create a complaint report for an order. A customer service employee can make the necessary communications between the customer and the seller to solve this complaint. In the system, we have a forum page where each user can create a forum topic discussion and create entries.

# 2. Final E/R Diagram




# 3. Relational Schemas

## 3.1 User

- **Relational Model**
  - User(id,username,password,email)
- **Primary Key**
  - id

## 3.2 User Phone Number

- **Relational Model**
  - User_phone_number(id, phone_number)
- **Primary Key**
  - {id, phone_number}
- **Foreign Key**
  - id references to User

## 3.3 Customer

- **Relational Model**
  - Customer(id)
- **Primary Key**
  - id
- **Foreign Keys**
  - id references to User

## 3.4 Seller

- **Relational Model**
  - Seller(id, shop_name, rating, address)
- **Primary Key**
  - id
- **Foreign Keys**
  - id references to User

## 3.5 Courier

- **Relational Model**
  - Courier(id,city, vehicle)
- **Primary Key**
  - id
- **Foreign Keys**
  - id references to User

## 3.6 Flower

- **Relational Model**
  - Flower(flower_id, type, color, occasion, price, photo_id, description)
- **Primary Key**
  - flower_id

## 3.7 Order

- **Relational Model**
  - Order(id, note, price, payment_method, date, est_delivery_time, address, status)
- **Primary Key**
  - id

## 3.8 Complaint Report

- **Relational Model**
  - Complaint_report(order_id, subject, status)
- **Primary Key**
  - {order_id, subject, status}
- **Foreign Keys**
  - order_id references Order

## 3.9 Customer Service Employee

- **Relational Model**
  - Customer_service_employee(id, shift)
- **Primary Key**
  - id
- **Foreign Keys**
  - id references User

## 3.10 Forum Entry

- **Relational Model**
  - forum_entry(topic_id, date, text, id)
- **Primary Key**
  - {topic_id, date, text}
- **Foreign Key**
  - topic_id references Forum Topic
  - id references to User

## 3.11 Chocolate

- **Relational Model**
  - Chocolate(chocolate_id, type, price)
- **Primary Key**
  - chocolate_id

## 3.12 Forum Topic

- **Relational Model**
  - Forum_topic(topic id, date, title, category, id, no_of_entries)
- **Primary Key**
  - topic_id
- **Foreign Key**
  - id references to User

## 3.13 Attached

- **Relational Model**
  - Attached(type, id, amount)
- **Primary Key**
  - type
- **Foreign Keys**
  - type references to Chocolate
  - id references to Order

## 3.14 Fav_shop

- **Relational Model**
  - Fav_shop(customer_id, seller_id)
- **Primary Key**
  - {customer_id,seller_id}
- **Foreign Keys**
  - customer_id references Customer
  - seller_id references Seller

## 3.15 Fav_flow

- **Relational Model**
  - Fav_flow(customer_id, flower_id)
- **Primary Key**
  - {customer_id, flower_id}
- **Foreign Keys**
  - customer_id references Customer
  - flower_id references Flower

## 3.16 Includes

- **Relational Model**
  - Includes(flower_id, order_id, amount)
- **Primary Key**
  - {flower_id, order_id}
- **Foreign Keys**
  - flower_id references Flower
  - order_id references Order

## 3.17 Stocks

- **Relational Model**
  - Stocks(flower_id, seller_id, sold, count)
- **Primary Key**
  - {flower_id, id}
- **Foreign Keys**
  - flower_id references Flower
  - seller_id references Seller

### 3.18 Order Delivery

- **Relational Model**
  - Order_delivery(order_id, courier_id, seller_id, status)
- **Primary Key**
  - {(order_id, courier_id, seller_id)}
- **Foreign Keys**
  - order_id references Order
  - courier_id references Courier
  - seller_id references Seller

# 4. Implementation Details

For our project, we have used Django framework of Python. We have written our code in Visual Studio Code and used Gitkraken for version control. Since this was our first time using Django, we had to learn the project structure of Django. We did not have any web design experience prior to this project, so we had to learn how to make connections between HTML scripts and Python code. Thankfully, Django made this connection simple and easy to control. However, we had problems with executing raw SQL queries because this was not the standard way of accessing data in Django. There were limited resources online. Also, for the models where we used inheritance, Django did not create tables that we could use. Online resources did not match our experience.

Muzaffer Yasin Köktürk: Part of SQL queries and update a product page, index page.

Muhammed Berk Köse: Register, login functionalities and HTML codes, index page.

Alptekin Önder: Part of SQL queries and delete a product page, product page.

İsmail Şahal: Part of SQL queries and register product page, product page.

# 5. Advanced Database Features

## 5.1 Views

### View 1

```
CREATE VIEW IF NOT EXISTS order_for_courier AS
SELECT date, est_delivery_time, address, status
FROM Order
WHERE id IN  (SELECT order_id
              FROM Order_delivery
              WHERE courier_id = currentUser)
```

This view is used to list the orders that were assigned to the courier who has logged in to the system. Since the courier does not need information like price, flower ids or note, we create a view for this situation.

### View 2

```
CREATE VIEW IF NOT EXISTS order_for_seller AS
SELECT price, note, date, est_delivery_time, status
FROM order
WHERE id IN (SELECT order_id
             FROM Order_delivery
             WHERE seller_id = currentUser)
```

This view is used to list the orders that the seller has received. Since the seller does not need information like the payment method or address, we create a view for this situation.

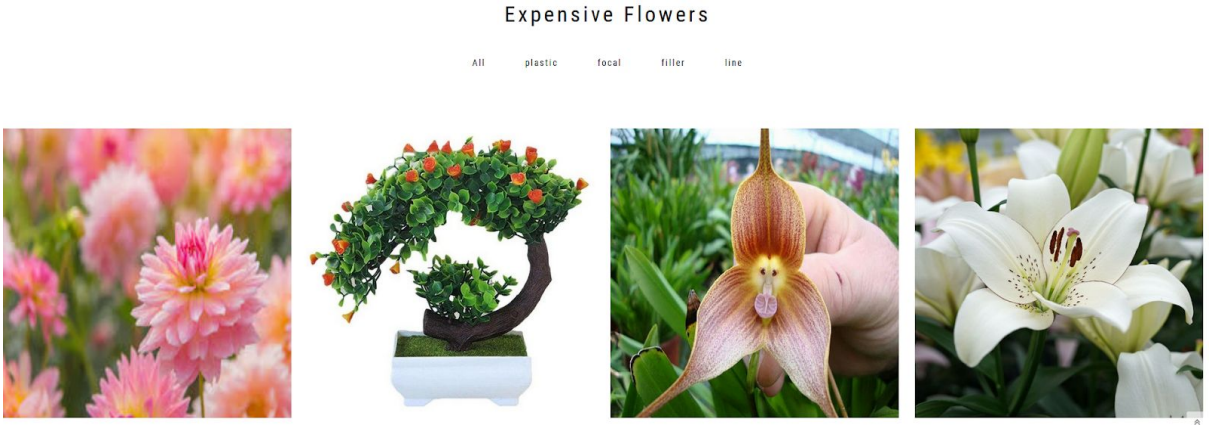
# 5.2 Reports

## Report 1

### SELECT * FROM theApp_flower

### ORDER BY price DESC LIMIT 4

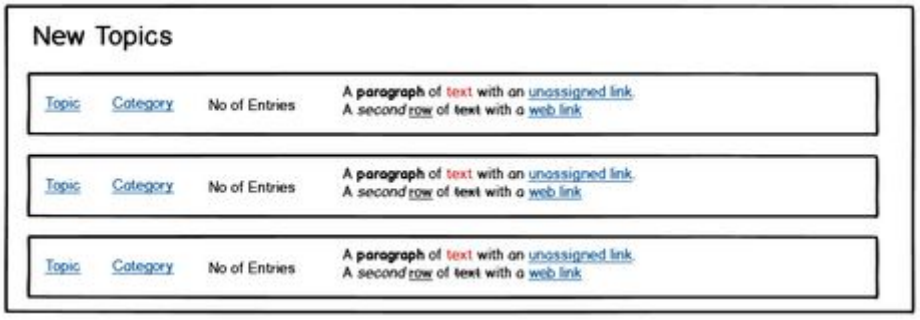
Explanation: This complex query returns the 4 flower with the highest price for our main page. We show these flowers in the page as expensive Flowers.




## Report 2

### SELECT * FROM theApp_forumTopic ORDER BY date DESC LIMIT 3

Explanation: This complex query returns last 3 forum topic that are entered. We use this feature to display last topics in the forum page.



## 5.3 Trigger

```
CREATE TRIGGER chocolate_check AFTER INSERT ON Attached
REFERENCING NEW ROW AS nrow
FOR EACH ROW
WHEN (nrow.type NOT IN (
        SELECT type
        FROM Chocolate))
```

We use the type attribute in Attached table to keep the information of which chocolate type was attached to an order. However, type is not the primary key of Attached table. So, to obtain integrity, we create a trigger to check each insertion to Attached table make sure that chocolate type exists.

## 5.4 Constraint

Many attributes at tables have “NOT NULL” constraint. In example attributes like type of chocolate, flower specifies the type of flower that will be sended and it have to be filled.

## 5.5 Stored Procedure

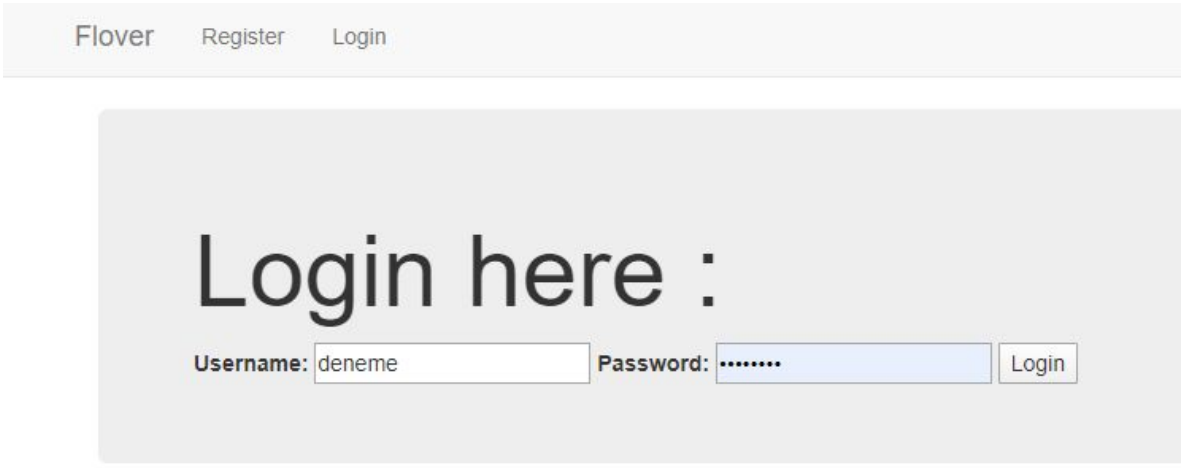
```
CREATE PROCEDURE SelectFlowers (in type varchar(64), in price integer)
AS
BEGIN
        SELECT * FROM Flower
        WHERE Flower.type = SelectFlowers.type AND  Flower.price <
        SelectFlower.price
END;
```

A stored procedure like the one above can be used to execute queries that have the same structure repeatedly without having to write the same SQL query. It can be executed with different parameters depending on the need.

# 6. User Manual

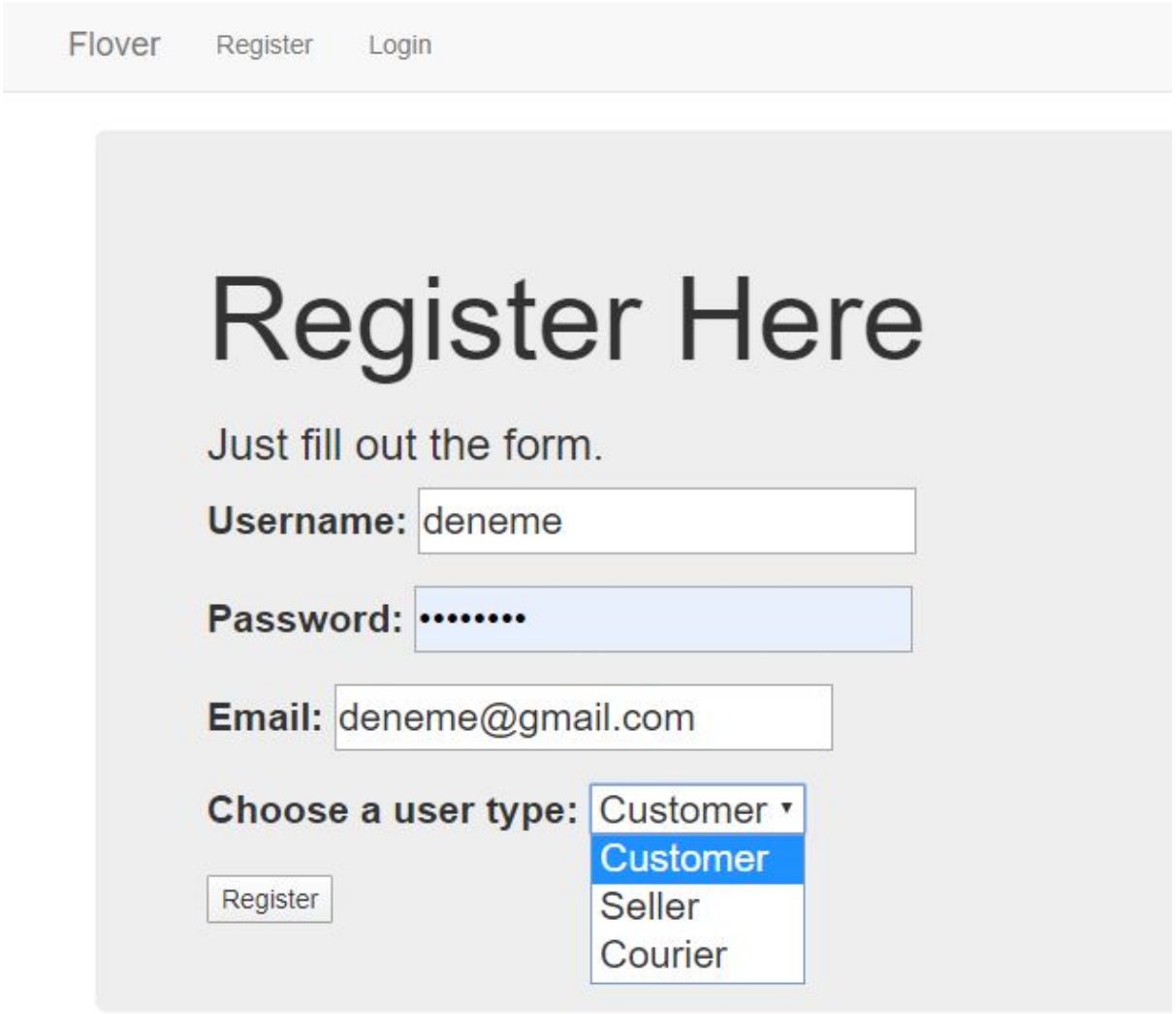
## 6.1 Login Page

Login page with 2 text places to input, which the user will fill and login if s/he already signed up.



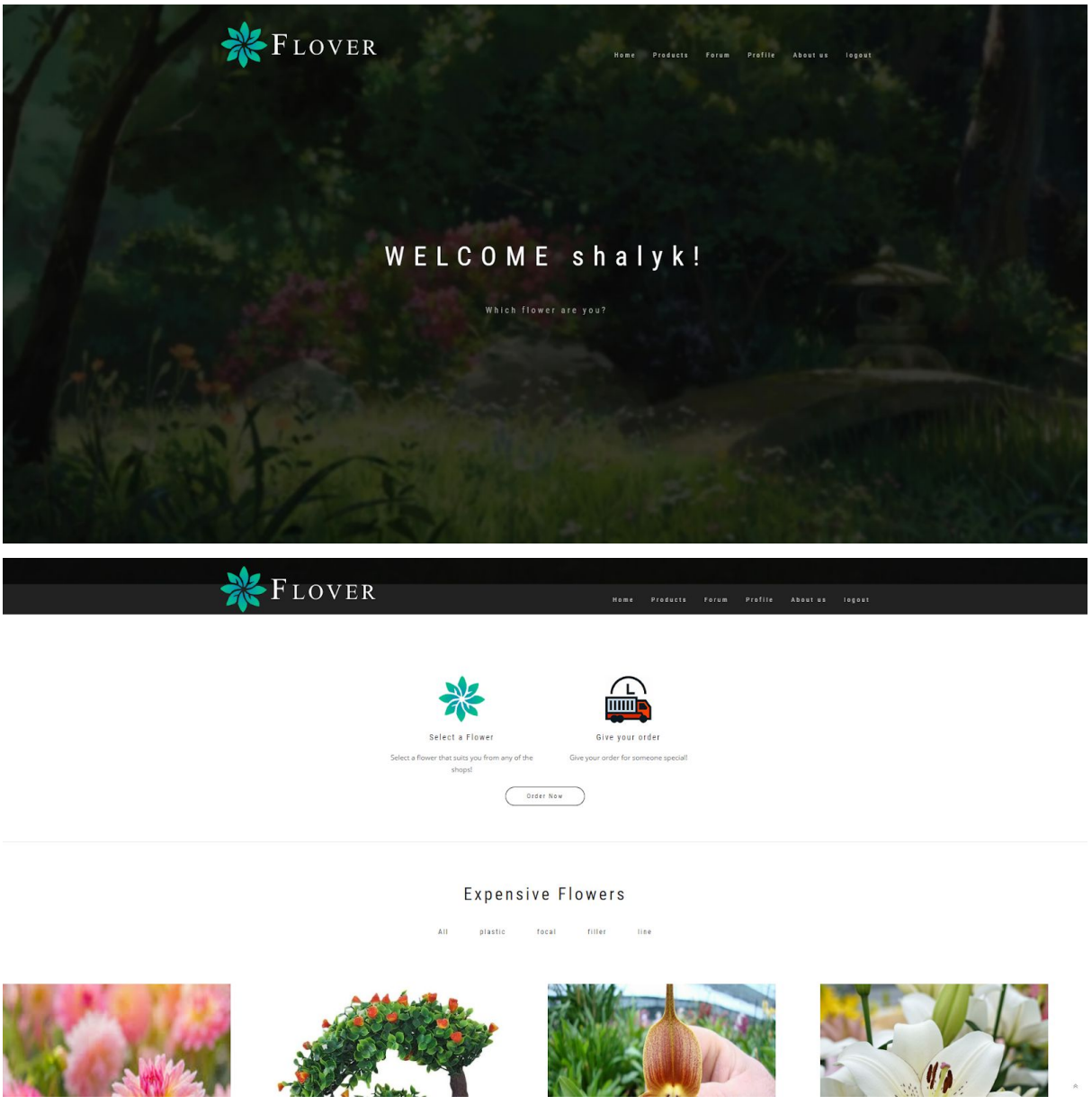
## 6.2 Register Page

Register page with 3 text and 1 select inputs. User can register his name and type, which are customer, seller, courier, in this page.



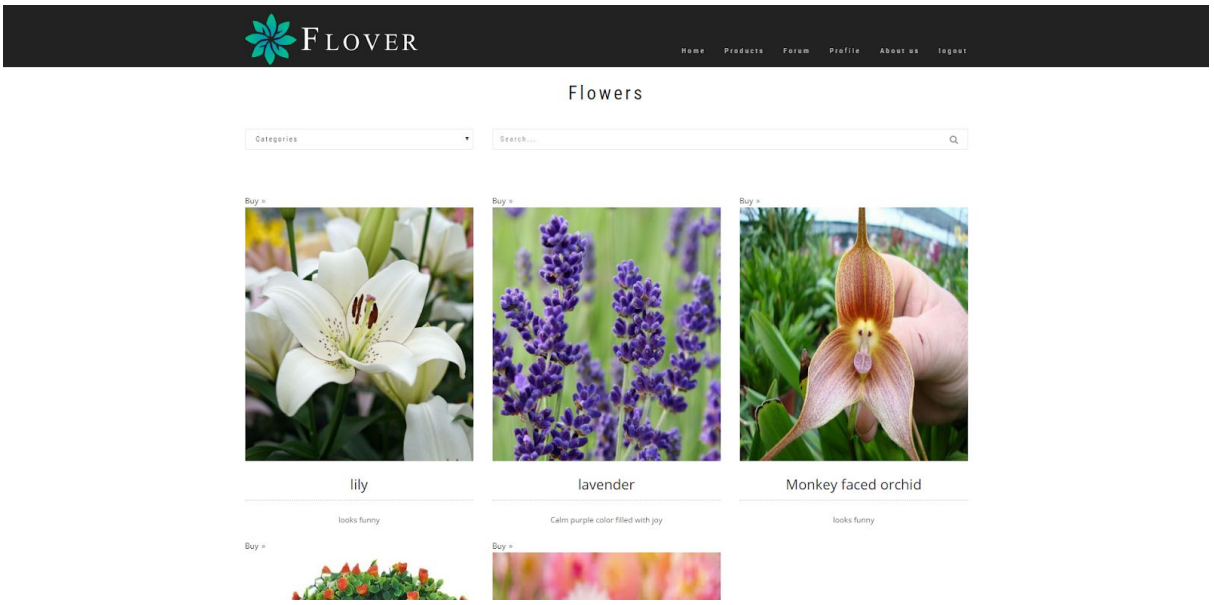
## 6.3 Index Page

Index page is the introductory to our website, it welcomes the user ex: “Welcome shalyk!”. Users can navigate to different pages of the site from the navigation bar at the right top. Also they can see the most expensive flowers at the bottom of the page. If they hit the order now button they will be sended to the products page.



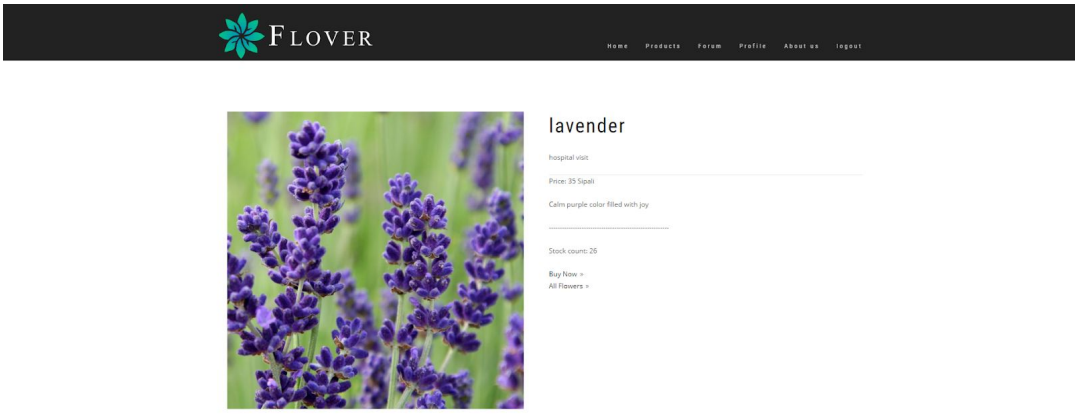
## 6.4 Products page

In products page users can see all the current flowers in database, select a category and see flowers which falls under that category. Also they can search for a flower. After they found a flower they desired they can click the buy button to get in to that flower's own page.

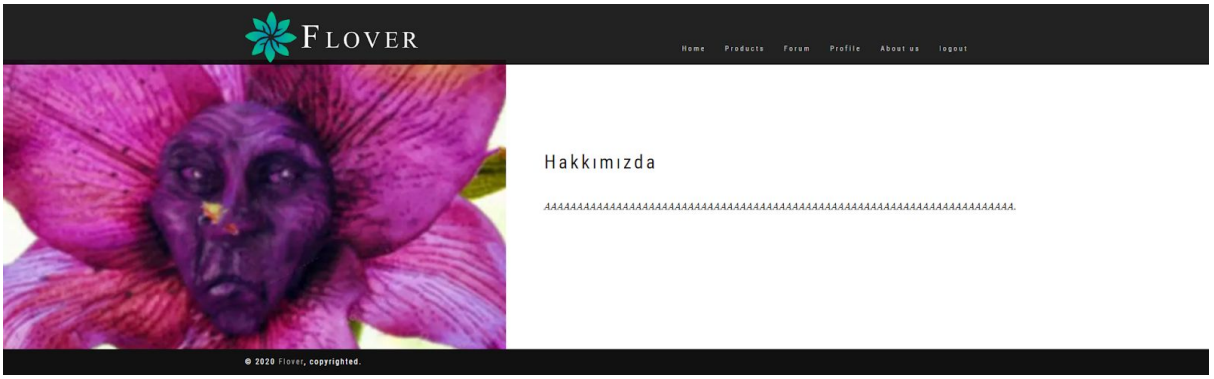



## 6.5 Specific product page

In this page user can see the products information, how much money it costs, which occasions it is used in, description of the flower etc. Then they can either select buy now to get in to the order page or they can go back by clicking all flowers button just after buy now.



## 6.6 About us

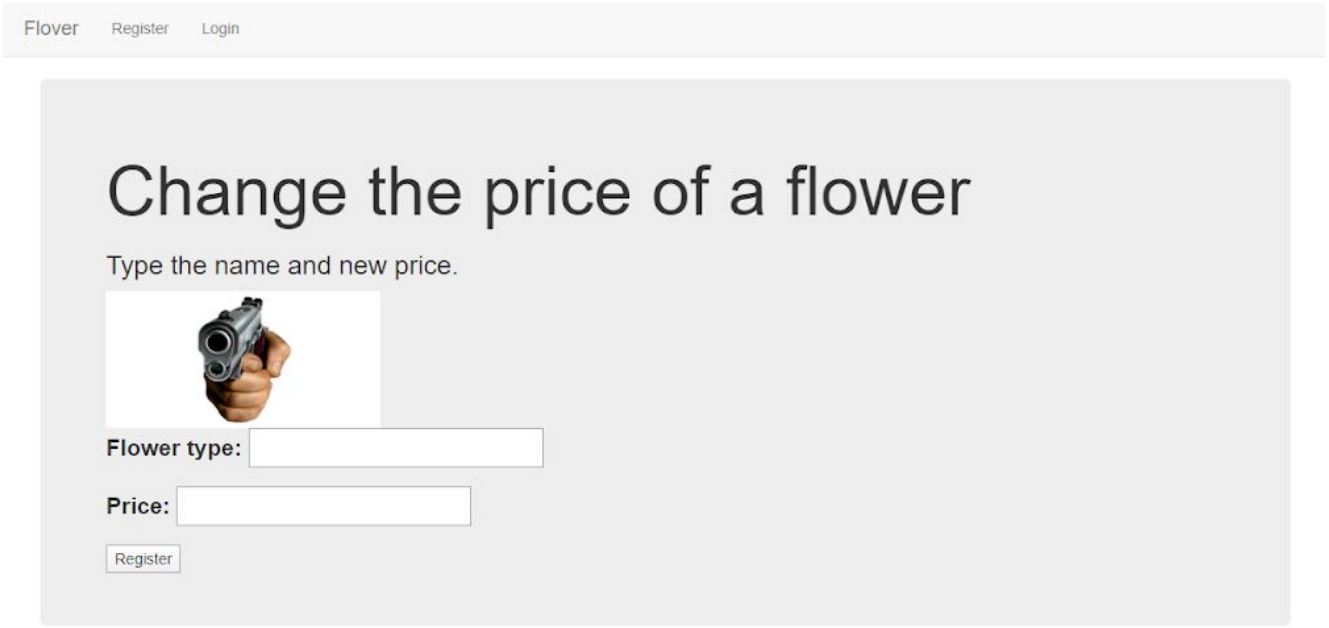

FLOVER
Home
Products
Forum
Profile
About us
logout
Hakkımızda
AAAA

## 6.8 Delete flower page

In this page sellers can delete flowers by giving its name to the input place. This page can be accessed by typing /deletion URL.

## 6.9 Change product data page

In this page sellers can change the price of a flower by giving its type. This page can be accessed by typing /changeProduct URL.



## 6.10 Profile page

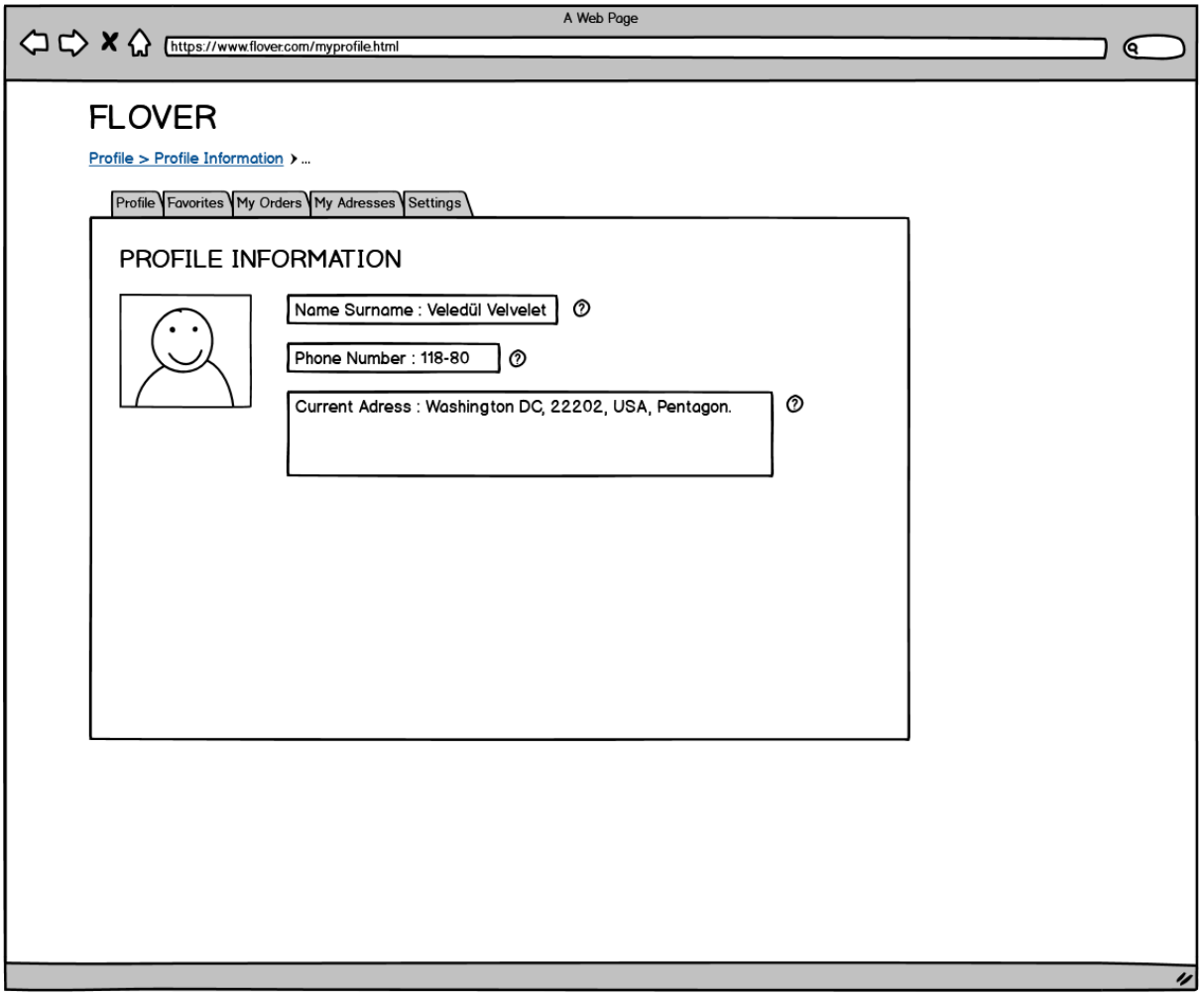
Here, the user will be able to view and edit their personal information.

It displays the favorite flowers of a user.

It displays profile information that you can change

In this page, the user will be able to view their past orders so that they can rate the shop or file a complaint.

In this page, customers will be able to view their complaints.

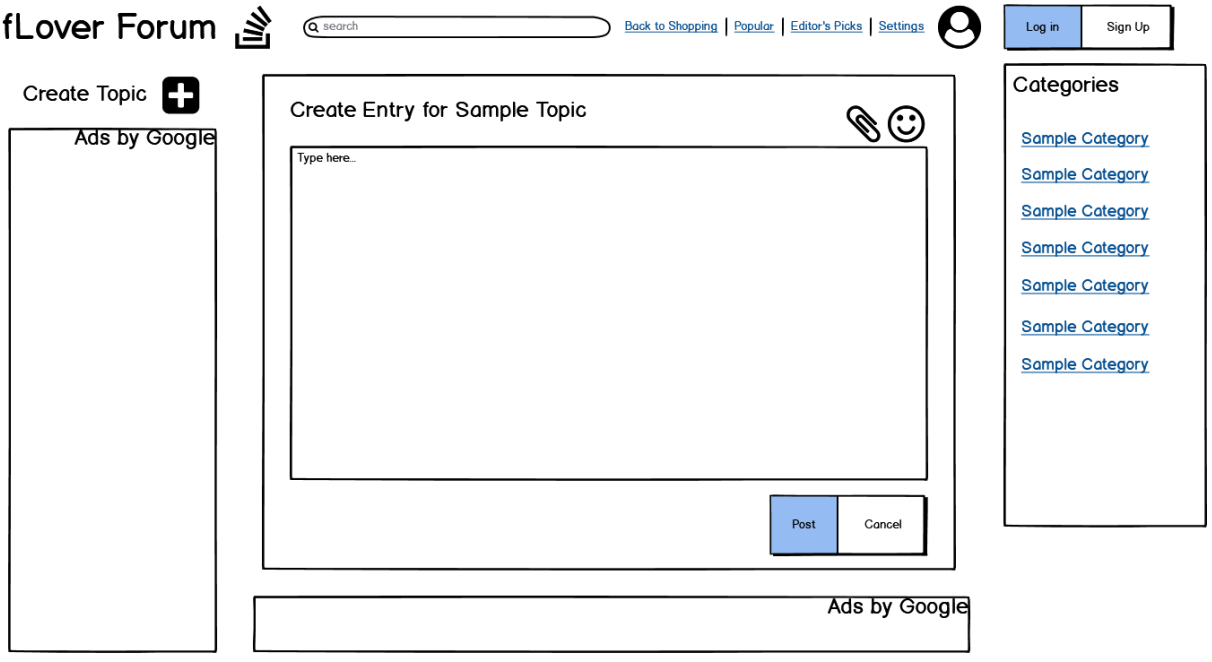


## 6.11 Forum Page

This page displays the topics in fLover Forum. Users can search different topics by using the search bar.

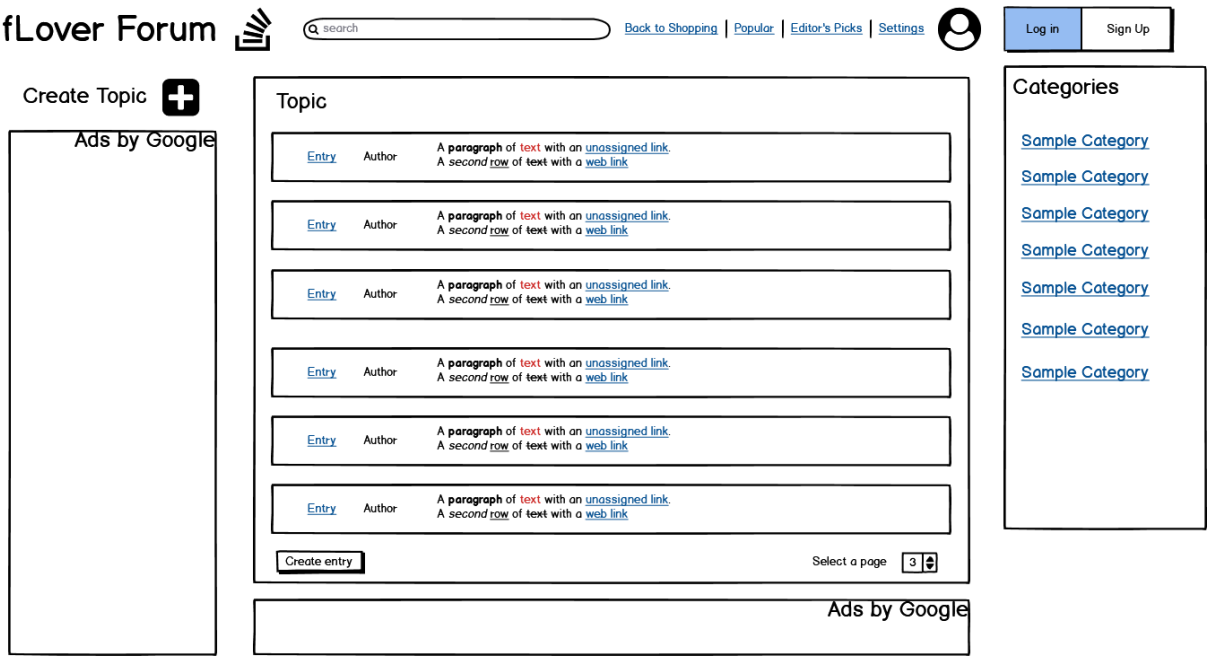
## 6.12 Forum Entry

In this page users can create a forum entry.



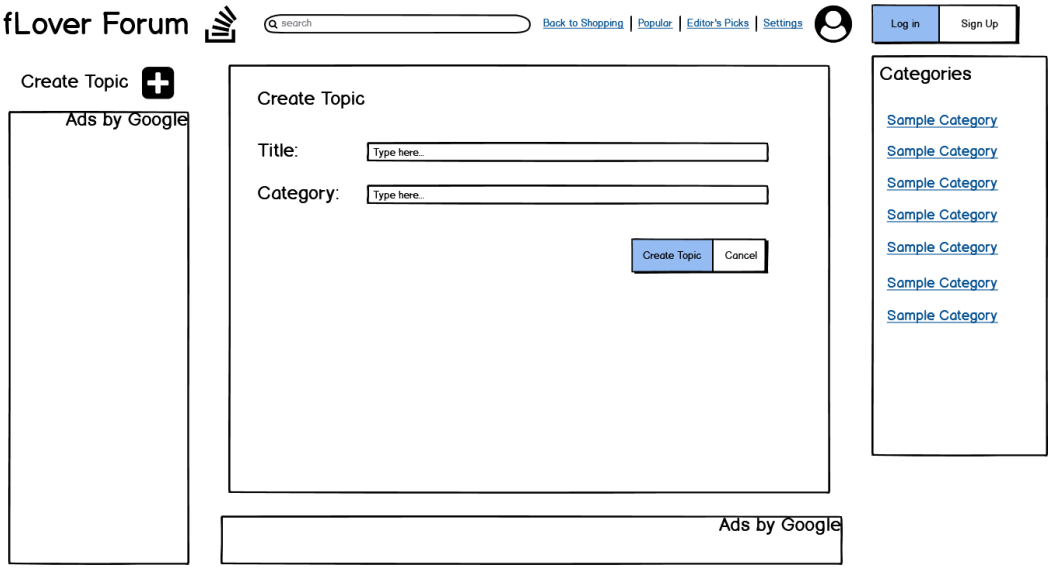
## 6.13 Forum Topic

This page displays entries for a specific topic. At the top of the page you can see the information about topic. Below that part all entries about this topic are showed with their author, text etc. At the left part




## 6.14 Create Forum Topic

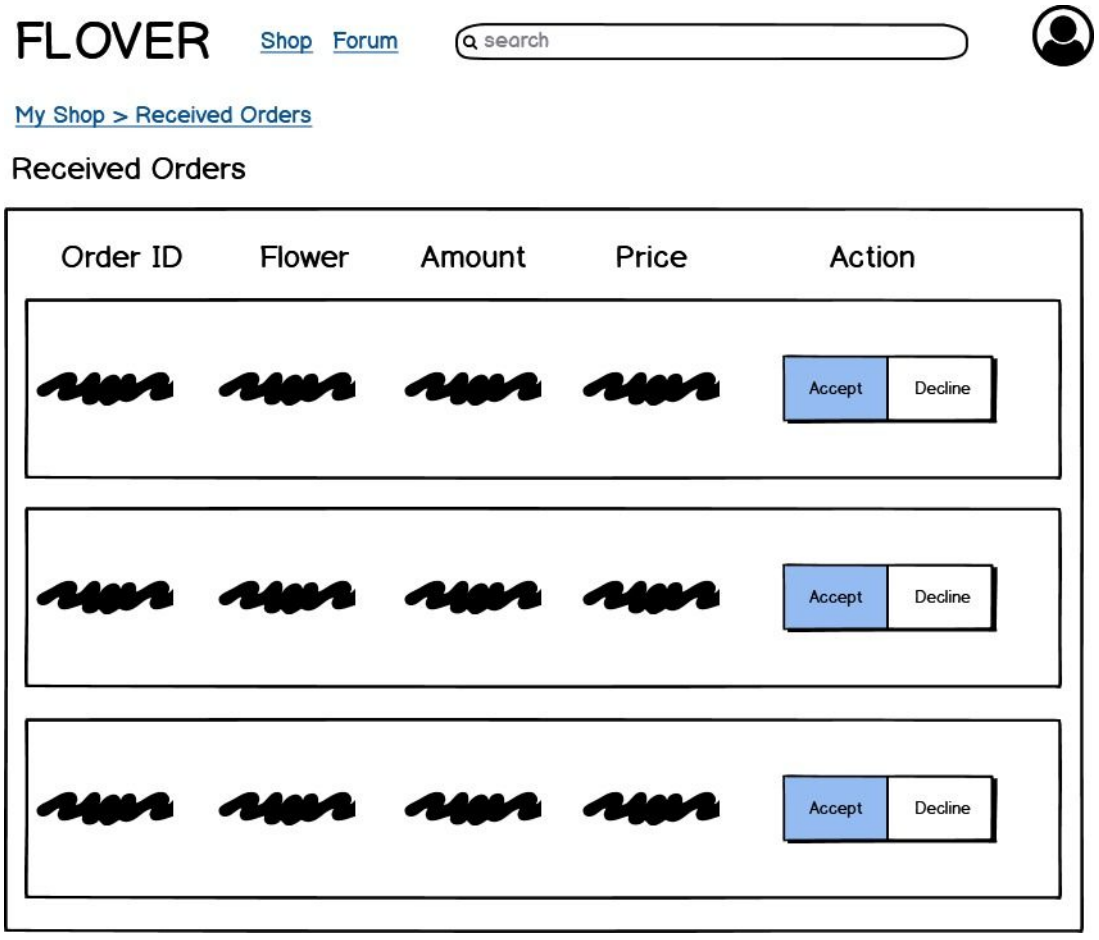
In this page, users can create a forum topic by typing a title and a category.



## 6.15 Received Orders

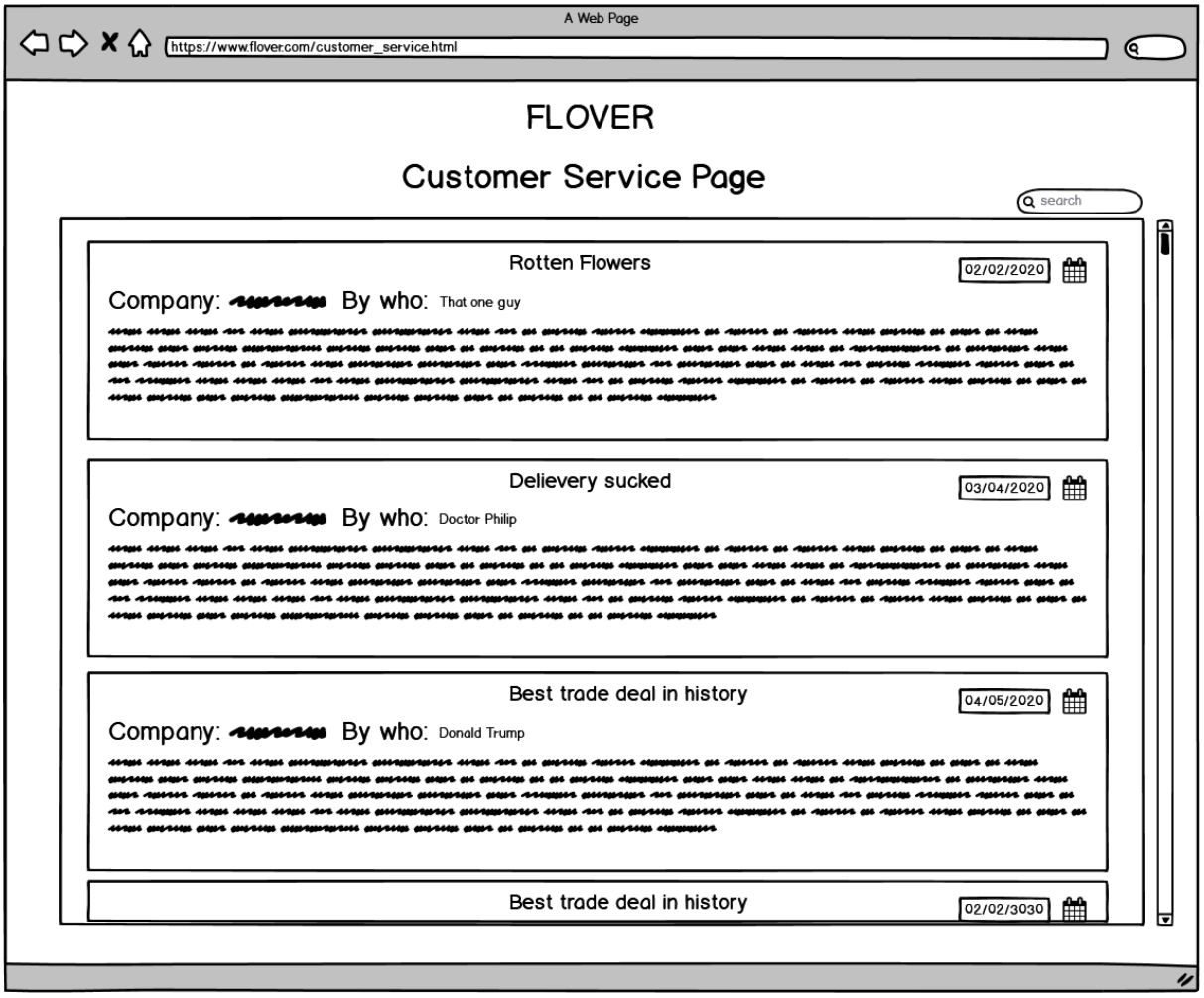
In this page seller can see the orders that customer have given. Seller can accept or decline the orders that are assigned to his/her shop.

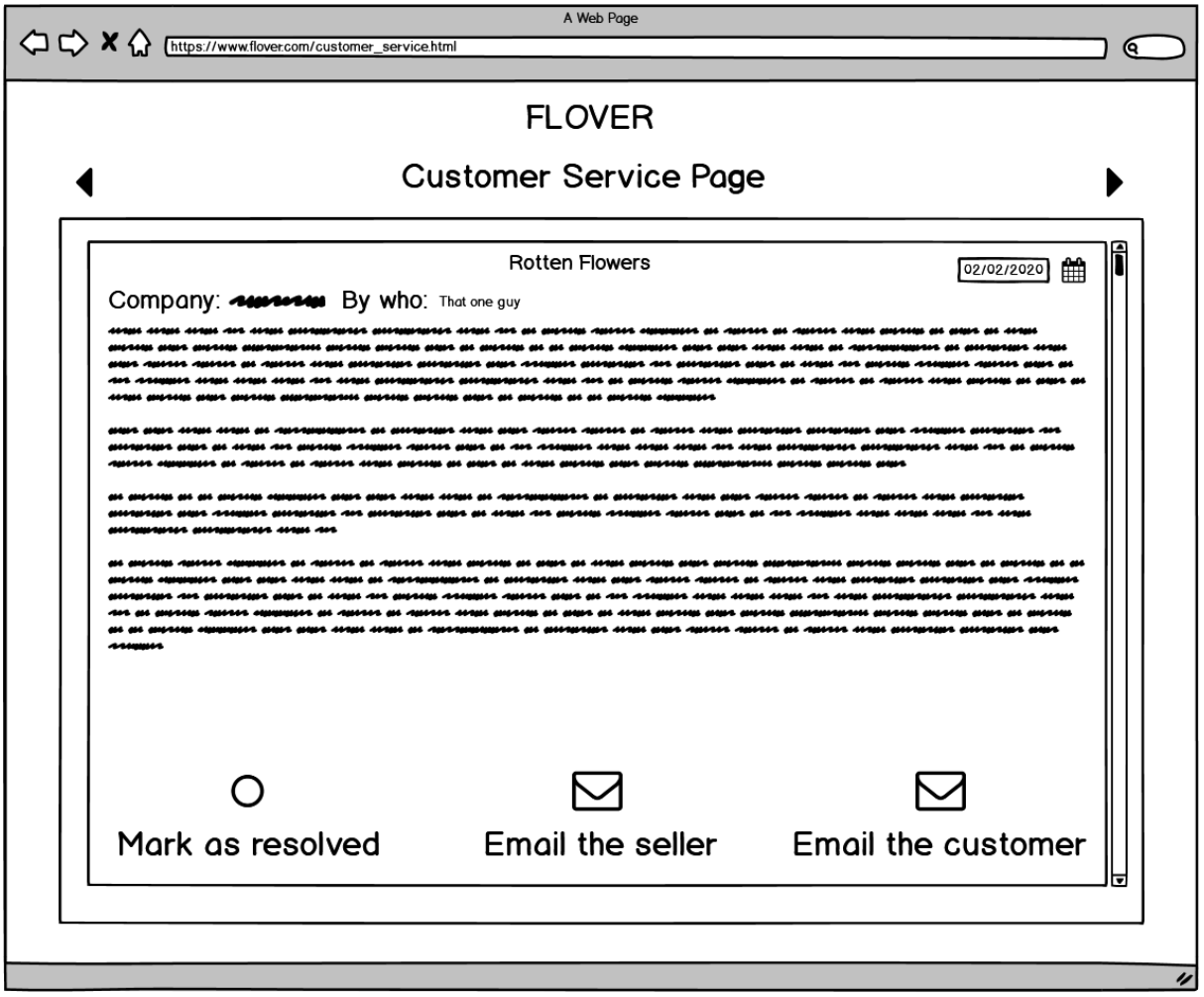
Also courier can see the orders that are assigned to him at this page. He/she can accept or decline the orders that are assigned to him.



## 6.16 Customer Service Page

At this page customer service employee can see the complaint reports that the customers entered. Employee can enter to complaint pages with clicking to one of the entries. Employee can search for a complaint report with the search bar at the right top corner.



## 6.17 Customer Service Report Page

After clicking on a complaint report customer service is directed to this page where they can see the report in detail. In this page they can email to the seller or customer or they can mark the complaint as resolved.