E-Commerce Flower Shop

Mia Angelucci - 8546063849
Siyu (Tracy) Chen - 8794016338
Sizhe(Steven) Liu - 7943772012
Yizhou (Johnson) Lu- 5286068870
Sujit Ponguluri - 6334348636
Yixuan (Oliver) Wang - 7960757354
Haofeng (Ace) Xu - 9025766132

Table of Contents

Project Proposal

Meeting time:
1. Thur 6 - 7 p.m. (in person)
2. Sun 11 - 11:30 a.m. (with cp)

Motivation and Concept:

Our flower-centric web application is aimed at simplifying the process of finding and buying the ideal flowers for any occasion. Comparable to popular online shopping platforms such as Amazon, our application provides users with an extensive range of floral options but with a distinctive search function. Our website combines the concepts of an e-commerce platform and a flower-oriented application. The main purpose of our project is to establish an online flower-shopping platform that allows users to purchase flowers from the comfort of their own homes. The website will enable users to peruse different categories of flowers based on their meanings or specific keywords. For instance, a user can seek out flowers that symbolize "friendship," and our website will generate results featuring flowers that represent this concept.

The website will have the following features:
- User Registration and Login: Users will be able to register their accounts like a portfolio to log in to the website. They are able to view their purchase history and save their personal information for future purchases.
- User-Friendly Interface: Our application will be designed with a user-friendly interface that is easy to navigate, making it simple for users to find what they are looking for. We will ensure that the layout is visually appealing, with a range of high-quality images to showcase the available flowers.
- Search and Filter: Implemented a search engine that can match user queries with relevant flower types. Let users search for flowers based on their meanings, occasions, and other filters.
- Implement e-commerce functionality: Allow users to purchase flowers securely and implement features such as a shopping cart and checkout process.
- Reviews and Ratings: Users can leave ratings on flowers, helping other users make informed decisions.

1

- Admin Dashboard: The website will have an admin dashboard for managing products, orders, and customer information.

High-Level Requirements

The website will have the following features:
- User Registration and Login: Users will be able to register their account like a portfolio to log in to the website(as well as using "forgot password" to reset their account login information). The registration process will require users to provide basic information such as emails, usernames, and passwords. They are able to experience personalized features such as viewing purchase history and current shopping carts and saving their personal information for future purchases.
- User-Friendly Interface: Our application will be designed with a user-friendly interface that is easy to navigate, making it simple for users to find what they are looking for with clear and concise menus, categories, and labels. We will ensure that the layout is visually appealing, with a range of high-quality images to showcase the available flowers.
- Search and Filter: Implemented a search engine that can match user queries with relevant flower types. Let users search for flowers based on their meanings, occasions, and other filters.
- Implement e-commerce functionality: Provide a simulated process of payment to the users, create a payment gateway into the website, and implement features such as a shopping cart and checkout process. Users are required to input street address, city, ZIP code to continue with the simulated process. Then, a "checkout is successful" message will be displayed, and the user's cart will be cleared.
- Ratings: Users can leave ratings on flowers, helping future customers make informed decisions.
- Admin Account: will have a single user account that has the ability to add/remove products from the website, manage user lists, and access users' info (non-sensitive data) and users' past orders.

Technical Specifications:

Database (1-2 people / 12 hours) :
The database will be implemented using a relational database management system such as MySQL.
There will be four tables in the database:
- Users table: This table will store information about registered users, including their user ID, username, email address, role, and password.

- Products table: This table will store information about the flowers and gifts available for purchase, including their product ID, name, description, image, price, keywords, rating, rating number, and available quantity left in stock.
- Cart table: Will store information about the cart, such as user ID, cart ID, product ID, and quantity.
- Order history table: Will store information about the user's past orders and the information necessary to save those, such as their orderID, userId, productID, quantity, rating, and rated.

Frontend UI(1-2 people / 20 hours):
- Homepage: It will showcase a variety of flowers and gifts, along with the different categories and filters that users can use to find what they are looking for.
- User Registration and Login: Users will be able to register for an account and log in to the website to view their purchase history and shopping cart.
- Product pages: Each product page will display a detailed description of the flower or gift, along with high-quality images and customer reviews. Users will be able to add the product to their shopping cart directly from the product page.
- Search and Filter: The search bar will be prominently displayed on the homepage and all product pages, allowing users to search for flowers and gifts based on their meanings, occasions, and other filters.
- Shopping Cart: The shopping cart will allow users to view and modify their cart contents at any time. Users will be able to add or remove items from their cart and update the number of items they wish to purchase.
- Checkout process: The checkout process will be simple, with clear instructions for users to follow. Users will be required to enter their billing and shipping information (random combinations of numbers would be fine) and confirm their order before it is processed.
- Admin Dashboard: The admin dashboard will be a separate interface that is accessible only to authorized personnel. It will allow administrators to manage products, orders, and customer information and view sales reports and analytics.

Server (2-3 people / 20 hours):
- Use Java to create a server that can be connected with clients.
- Accept search requests and return matched products with the keywords with information.
- Create connections between server and databases to allow modifications on the database on product/user information.
- Threading: We will use multithreading to handle multiple client requests simultaneously.
- Login: compare the user input with the stored username/password in the database and return the boolean result.

Detailed Design Document:

Hardware and Software Requirements:
- a. Hardware:
  - A server with adequate processing power, memory, and storage to handle the web app's requirements.
  - Develop computers with necessary hardware configurations for efficient development and testing.
- b. Software:
  - Server: Java, Apache Tomcat
  - Frontend: HTML, CSS, JavaScript, React
  - Database: MySQL
  - Version Control: Git, GitHub
  - IDE: Eclipse or Visual Studio Code

Class Diagram and Inheritance Hierarchy:
- a. Main Classes:
  - User
    - Variables: user_id, username, email, password, order_history
    - Methods: register(), login(), viewOrderHistory(), updateProfile(), addproduct(),
  - Product
    - Variables: product_id, name, description, keyword, image_url, price, stock, ratings, rating_num
    - Methods: viewProductDetails(), addReview()
  - Order
    - Variables: order_id, user_id, product_id, quantity
    - Methods: createOrder(), updateOrder(), cancelOrder()
  - Admin (inherits from User)
    - Methods: manageProducts(), manageOrders(), manageUsers()
- b. Relationships:
  - User (1) --- (0..*) Order
  - Product (1) --- (0..*) Order

AdminController

- ♂ᶠ databaseURL : String
- ♂ᶠ databaseUser : String
- ♂ᶠ databasePassword : String
- ● getproduct(HttpSession) : String
- ● getadminhome(HttpSession) : String
- ● updateproduct(int, Model, HttpSession) : String
- ⚠ updateproducttodb(int, String, int, int, String) : String
- ⚠ removeProductDb(int, HttpSession) : String
- ● getCustomerDetail(HttpSession) : String

CartController

- ♂ᶠ databaseURL : String
- ♂ᶠ databaseUser : String
- ♂ᶠ databasePassword : String
- ● usercart(Model, HttpSession) : String
- ● userorder(Model, HttpSession) : String
- ● addToCart(int, HttpSession) : ResponseEntity<String>
- ● updateCart(int, int, int) : String
- ● updateRating(int, int, int, int, RedirectAttributes) : String
- ● payment(Model, HttpSession) : String
- ● checkout(int, RedirectAttributes) : String

Util

- ● ˢ FillProducts(List<Product>) : void

Product

- ■ id : int
- ■ name : String
- ■ description : String
- ■ price : String
- ■ image : String
- ● ᶜ Product()
- ● ᶜ Product(int, String, String, String, String)
- ● getId() : int
- ● setId(int) : void
- ● getName() : String
- ● setName(String) : void
- ● getDescription() : String
- ● setDescription(String) : void
- ● getImage() : String
- ● setImage(String) : void
- ● getPrice() : String
- ● setPrice(String) : void

UserController

- ♂ᶠ databaseURL : String
- ♂ᶠ databaseUser : String
- ♂ᶠ databasePassword : String
- ● returnIndex(HttpSession) : String
- ● index(Model, HttpSession) : String
- ● userlog(Model) : String
- ● userlogin(String, String, HttpSession, Model) : String
- ● registerUser() : String
- ● buy() : String
- ● getproduct(Model) : String
- ● newUseRegister(String, String, String, String) : String
- ● search(String, Model) : String
- ● profileDisplay(Model, HttpSession) : String
- ● updateUserProfile(int, String, String, String, String, HttpSession) : String

Database Schema:

Here's the database schema for the flower/gift e-commerce web app. The schema includes tables for Users, Products, cart, and order history.

   a. Users table:

     CREATE TABLE `users` (
      `user_id` int(11) NOT NULL AUTO_INCREMENT,
      `username` varchar(45) NOT NULL,
      `password` varchar(64) NOT NULL,
      `role` varchar(45) NOT NULL,
      `enabled` tinyint(4) DEFAULT NULL,
      `email` varchar(110) NOT NULL,
      `address` text NOT NULL DEFAULT '',
      PRIMARY KEY (`user_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
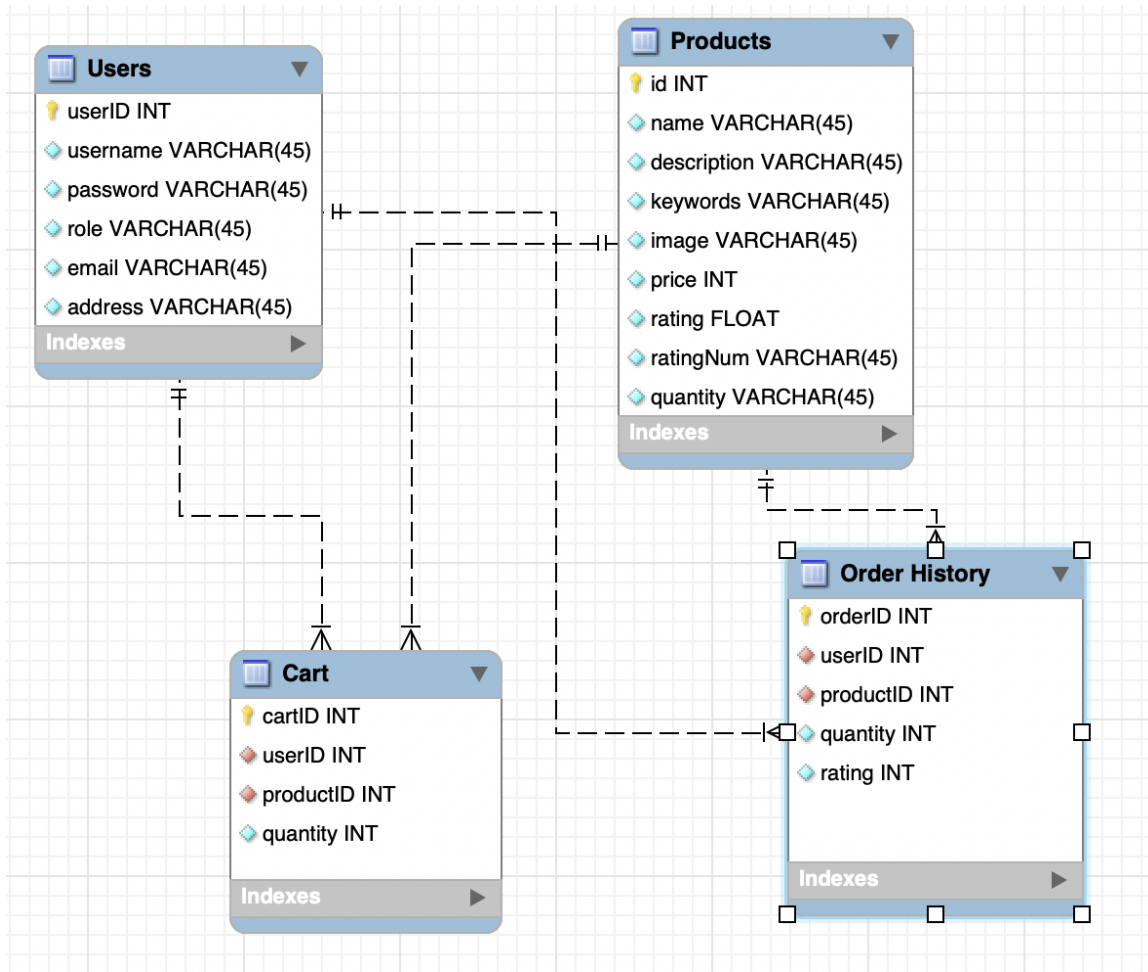
b. Products table:
```
CREATE TABLE `products` (
  `id` int(11) NOT NULL AUTO_INCREMENT,
  `name` varchar(255) NOT NULL,
  `description` text NOT NULL,
  `keywords` varchar(255) NOT NULL,
  `image` text NOT NULL,
  `price` int(11) NOT NULL,
  `rating` float NOT NULL,
  `ratingNum` int(11) NOT NULL,
  `quantity` int(11) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

c. Cart Table:
```
CREATE TABLE `cart` (
  `cart_id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `product_id` int(11) NOT NULL,
  `quantity` int(11) NOT NULL,
  PRIMARY KEY (`cart_id`),
  KEY `fk_user_id` (`user_id`),
  KEY `fk_product_id` (`product_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```

d. Order History Table
```
CREATE TABLE `order_history` (
  `order_id` int(11) NOT NULL AUTO_INCREMENT,
  `user_id` int(11) NOT NULL,
  `product_id` int(11) NOT NULL,
  `quantity` int(11) NOT NULL,
  `rating` int(11) NOT NULL DEFAULT -1,
  PRIMARY KEY (`order_id`),
  KEY `fk_order_user_id` (`user_id`),
  KEY `fk_order_product_id` (`product_id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
```
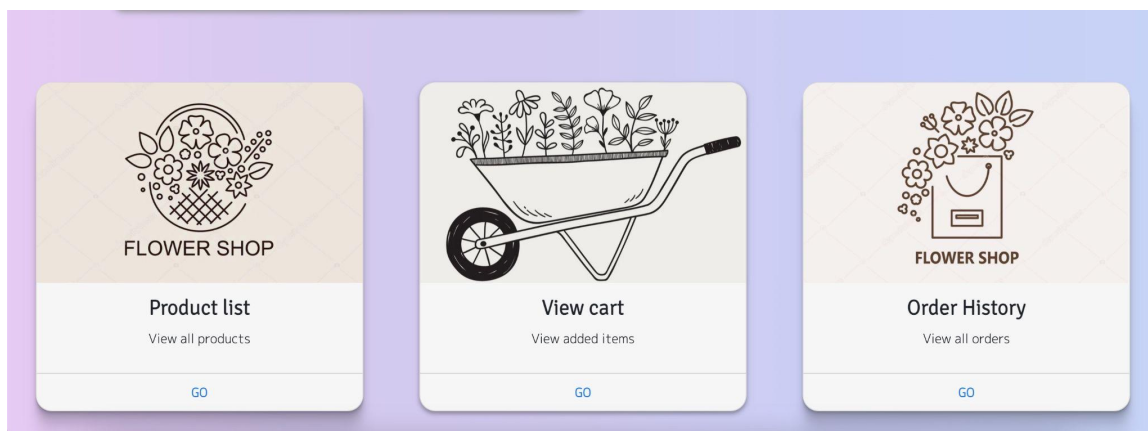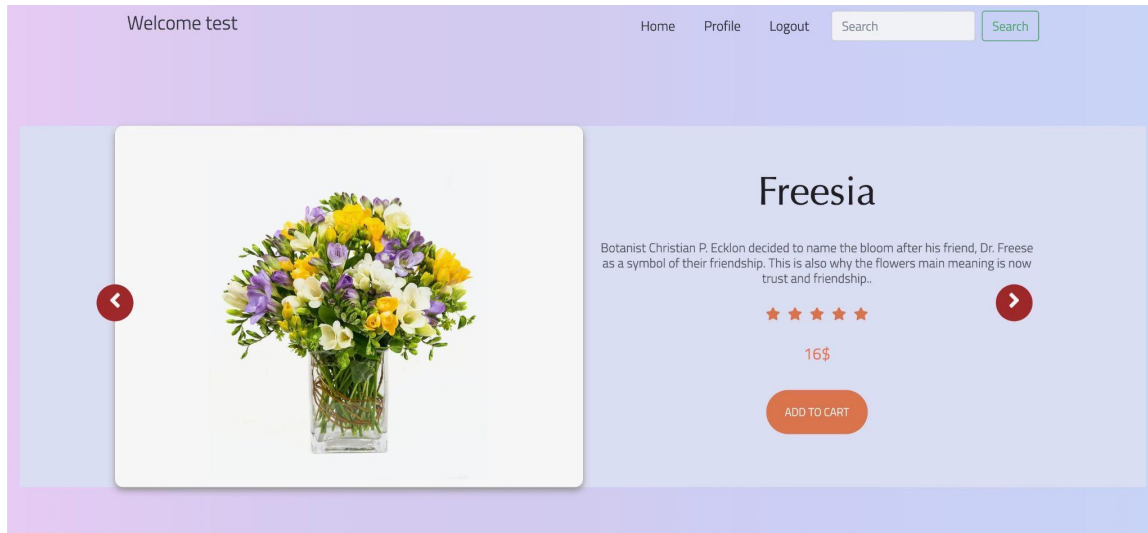
Frontend UI:

a.   Login page:

b. Main home page:

c. View the cart:

Serial No.	Product Name	rating	Preview	Quantity	Price	Description	Update
13	Daisy	5.0		1	12	The meaning of a daisy flower can be purity, innocence, new beginnings, joy and cheerfulness. In the Victorian Era, daisies symbolised innocence, loyalty and an ability to keep things secret.	Update
							Checkout

d. View all products:

Serial No.	Product Name	rating	Preview	Quantity	Price	Description	Add To Cart
1	Red Rose	2.7		8	15	Most of us are familiar with what the red rose means, having been used across cultures to represent love and romance for centuries. The meaning of red roses is universally understood to be love and passion.	Add To Cart
2	White Rose	5.0		10	16	White roses symbolize purity, youthfulness, and innocence. Young love, eternal loyalty, and new beginnings are also commonly tied to the meaning of white roses, making them a popular choice for weddings and romantic occasions.	Add To Cart
3	Allium	5.0		10	14	Allium symbolizes unity, good fortune, prosperity, humility, and patience—these blooms make the perfect gift to wish someone good luck on a new endeavor.	Add To Cart
4	Sunflower	5.0		10	17	Sunflowers symbolize loyalty and adoration thanks to the myth of Clytie and Apollo. And, because of their association with the sun, sunflowers are well-known for being a happy flower and the perfect bloom for a summer flower delivery to brighten someones mood!	Add To Cart

e. Checkout page:

Write your imaginary card number. The Address, City and Zipcode areas are required. Click Submit Payment for purchasing.

Payment Informations

Card Number

MM / YY

CCV

Street Address

City

ZIP code

Submit Payment

VISA

XXXX XXXX XXXX XXXX

name   EXP   CCV
Trojan   XX/XX   XXX

f. Order history page:

Serial No.	Product Name	rating	Preview	Quantity	Price	Description	Update
5	Tulipp	5		1	15	The most known meaning of tulips is perfect and deep love. As tulips are a classic flower that has been loved by many for centuries they have been attached with the meaning of love. Theyre ideal to give to someone who you have a deep, unconditional love for, whether its your partner, children, parents or siblings.	Update
6	Peony	3		1	14	Generally symbolic of love, honor, happiness wealth, romance, and beauty, the peony is traditionally given on special occasions as an expression of goodwill, best wishes, and joy.	Update

Rating has been updated successfully.

g. Admin page:

Welcome Back, Admin

Manage your data from this Admin Panel

Products

Manage all the products here.

Manage

Customers

Manage all the customer here.

Manage

Serial No.	Product Name	Preview	Quantity	Price	Description	Delete	Update
1	Red Rose		10	15	Most of us are familiar with what the red rose means, having been used across cultures to represent love and romance for centuries. The meaning of red roses is universally understood to be love and passion.	Delete	Update
2	White Rose		10	16	White roses symbolize purity, youthfulness, and innocence. Young love, eternal loyalty, and new beginnings are also commonly tied to the meaning of white roses, making them a popular choice for weddings and romantic occasions.	Delete	Update
3	Allium		10	14	Allium symbolizes unity, good fortune, prosperity, humility, and patience—these blooms make the perfect gift to wish someone good luck on a new endeavor.	Delete	Update
4	Sunflower		10	17	Sunflowers symbolize loyalty and adoration thanks to the myth of Clytie and Apollo. And, because of their association with the sun, sunflowers are well-known for being a happy flower and the perfect bloom for a summer flower delivery to brighten someones mood!	Delete	Update
5	Tulip		10	15	The most known meaning of tulips is perfect and deep love. As tulips are a classic flower that has been loved by many for centuries they have been attached with the meaning of love. Theyre ideal to give to someone who you have a deep, unconditional love for, whether its your partner, children, parents or siblings.	Delete	Update

10

Test Document for Flower E-commerce Web App

Black Box Test:
- Test the user registration functionality by providing valid input data (email, username, and password). The user should be successfully registered and redirected to the login page with a success message.

White Box Test:
- Test the login functionality by specifying a username that exists and a password that does not match. The user should be taken back to the login page with a message "Invalid login."
- Test the login functionality by specifying a username that does not exist. The user should be taken back to the login page with a message "Invalid login."
  - Call the login function with a non-existent username and a random password.
  - Verify that the user is redirected back to the login page.
  - Check that the "Invalid login" message is displayed.
- Test the login functionality by specifying a username that exists and a password that matches. The user should be taken to the homepag.
  - Call the login function with a valid username and the correct password.
  - Verify that the user is redirected to the homepage.
- Test the admin dashboard functionality by logging in with an admin account.
  - Call the login function with a correct admin username and the password.
  - Verify that the admin has access to manage products, orders, and customer information.

```
public void testLogin() {
    // valid login credentials
    String username = "johndoe";
    String password = "mypassword";

    // valid admin login credentials
    String adminUsername = "admin";
    String adminPassword = "adminpass";

    // invalid login credentials
    String invalidPassword = "wrongpassword";
    String invalidUsername = "nonexistentuser";

    // create a user with valid login credentials
    User user = new User(username, password);
    user.save();
```

```
    // test valid login
    assertTrue(user.login(username, password));

    // test invalid password
    assertFalse(user.login(username, invalidPassword));

    // test invalid username
    assertFalse(user.login(invalidUsername, password));

    // test valid admin username
   assertTrue(user.login(adminUsername, adminPassword));

    // clean up by deleting the user
    user.delete();
}
```

Unit Test:
- Test the search functionality by providing a valid query
  - Call the search function with a valid query (e.g., a specific flower or gift type).
  - Compare the returned results with the expected output based on the given search query.

```
public void testSearchAndFilter() {
    // Test cases with various combinations of keywords and expected results
    List<Product> expectedResult1 = Arrays.asList(
    // Product class constructor: Product (String id, String name, String description)
    new Product("1", "Red Rose", " Red, rose, roses, love, girlfriend, wedding"),
    new Product("3", "Tulip", " Tulip, Wedding, love, family, girlfriend, birth, parent")
 );
assertEquals(expectedResult1, searchService.search("girlfriend"));

 List<Product> expectedResult2 = Arrays.asList(
 new Product("10", " Sunflower", " Sunflower, Happy, get, well, graduation, anniversary,
 wedding") );
 assertEquals(expectedResult2, searchService.search("happy"));

 // filtered result is empty
 List<Product> expectedResult4 = Arrays.asList();
 assertEquals(expectedResult4, searchService.search("asodhasdg"));
}
```

- Test the checkout process by providing valid billing and shipping information
    - Call the checkout function with valid billing and shipping information.
    - Verify that the order is successfully processed and the user is redirected to the "payment successful" page.
- Test the rating functionality by submitting a rating for a product
    - Call the rating function with a specific product and a rating value.
    - Verify that the rating is correctly saved in the database and displayed on the product page.

Regression Test:
- Test adding a product to the shopping cart
    - Call the add-to-cart function with a specific product and quantity.
    - Verify that the product is correctly added to the cart, with the correct quantity and price.
- Test the update and deletion of products, orders, and user accounts by an admin
    - Call the corresponding update function for products, orders, and user accounts with the modified details.
    - Query the database to verify that the changes are correctly reflected.
    - Log in as an admin and delete a specific product, order, or user account.
    - Call the corresponding delete function for products, orders, and user accounts with the target item ID.
    - Query the database to verify that the item has been removed.

Deployment Documentation

UI/UX:
- Launch website on localhost port 8080 (user lands on page homepage.html).
- Log in using the form on the login.html page and redirect the user to index.html if the username and password is correct.
- Connected the local SQL code on Google Cloud

Server:
- Create the Login, Register, and Result Controllers using SpringBoot framework and SpringMVC
    - Annotate classes with @Controller
- Use JDBC to connect the MySQL database with SpringBoot application
    - Configure the Google Cloud SQL credentials in the application properties file
- Ensure that HTML/CSS forms can communicate with SpringBoot application
    - User information should be accurately taken in
- Ensure that information gets correctly passed to the Google Cloud database

- - Register information should be saved and usable
  - Class information should come up correctly

Data Collection:
- Ensure all the data was migrated to the Google Cloud SQL database
  - Run SQL scripts to verify
- Perform a regression test of the data deployment
  - When data needs to be updated, ensure that the Google Cloud SQL database can be updated as well.
  - When certain data is missing or losing, make sure that there is a null or default value to ensure other parts of the data and calculations regarding the data will not be affected or changed.

Backend/Database:
- Create the MySQL database on Google Cloud SQL
- Use the Cloud SQL instance on Google Cloud Platform
- Import all data from the csv, for example flower name/description/price/quantity left into the corresponding table in the database
- Connect the database to server side Java code using JDBC
- Run different testing strategies (black box + white box) on database querying functionalities to ensure no problems occur
  - search for flowers with keywords as filters: for example love/friendship