

Table of Contents

[Task1: Problem Statement Formulation and definition 3](#_Toc153989913)

[Task2: Creating the No-SQL MongoDB Database and Data Modeling 4](#_Toc153989914)

[Creation of a No-SQL MongoDB 4](#_Toc153989915)

[CRUD operations 4](#_Toc153989916)

[Data modelling 10](#_Toc153989917)

[Usage of MongoDB Index 10](#_Toc153989918)

[Query Diagnosis and Analysis 11](#_Toc153989919)

[Task3: Using Django to build the Web Application using Bootstrap 13](#_Toc153989920)

[Creation of VirtualEnv for Django 13](#_Toc153989921)

[Project settings used 15](#_Toc153989922)

[Connectivity of Django with MongoDB 16](#_Toc153989923)

[Django Template Language 18](#_Toc153989924)

[Model –View-Template 19](#_Toc153989925)

[Django Admin site 21](#_Toc153989926)

[CSRF tokens used in forms 22](#_Toc153989927)

[Django forms 23](#_Toc153989928)

[Incorporation of Bootstrap 24](#_Toc153989929)

[Task 4 – Overall GUI and working, Report, GitHub, Video and Reflection 24](#_Toc153989930)

[Overall Navigational GUI 24](#_Toc153989931)

[Functionality 26](#_Toc153989932)

[Reflection 28](#_Toc153989933)

# Task1: Problem Statement Formulation and definition

Inspired by my love for expressing affection through flowers, I conceived the idea of developing flower boutique website. Flowers, with their vibrant colors and diverse shapes, have always held special meaning to me. Hence, the name BloomsByD was born, with "D" representing my initial, Dana. During my exploration of various websites, I came across "Arabian Flora," which, unfortunately, proved to be text-heavy and lacked user-friendliness due to its complex layout. Additionally, the ordering process was time-consuming, and I couldn't find any flower reviews to aid my purchasing decisions.

Motivated by these shortcomings, my goal became clear: to create a visually soothing website, employing a calm color scheme that caters to eye sensitivity. It would also prioritize user-friendliness, ensuring easy navigation and browsing. One crucial aspect I aimed to incorporate was a review section, allowing users to share their experiences and read feedback from others about specific flowers.

In addition to these improvements, the website would offer various functionalities to enhance the user experience, such as:

* Register
* Login/Logout
* View selected category
* View all categories
* Add review
* View review for flowers

# Task2: Creating the No-SQL MongoDB Database and Data Modeling

## Creation of a No-SQL MongoDB



Figure 1.Creation of the database

## CRUD operations

* Create

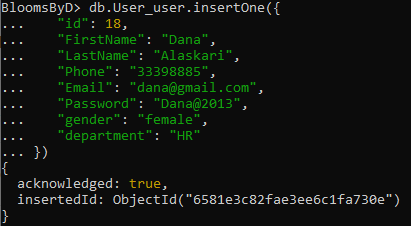


Figure 2.inserting one document

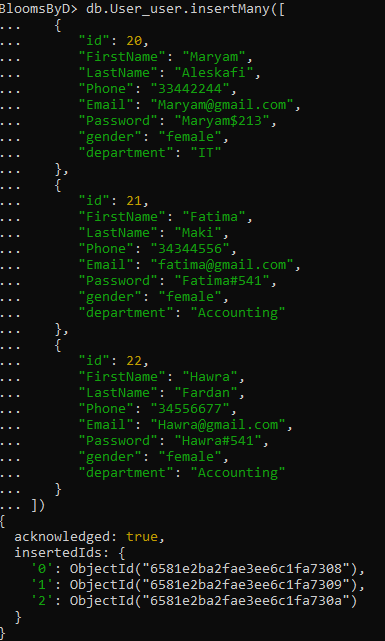


Figure 3.Inserting many documents at once

* Read



Figure 4.Listing all the documents in the database using the find command



Figure 5.Query to find all males which has the first name of Ali



Figure 6. Retrieve all female employees working in departments other than Accounting

* Update

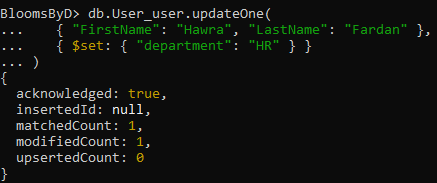


Figure 7.updating hawra department to HR

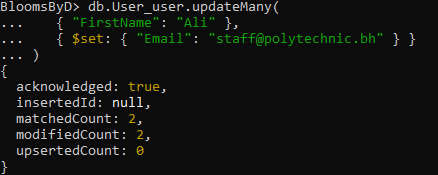


Figure 8.Query to update Many to all that have Ali as their firstname to set their email to staff@polytechnic.bh

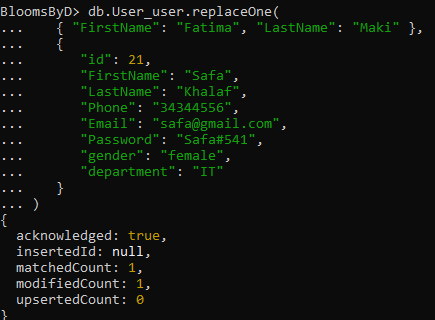


Figure 9.Replacing a document with whole new one but the object remain the same



Figure 10.Delete one query



Figure 11.Delete many query

## Data modelling

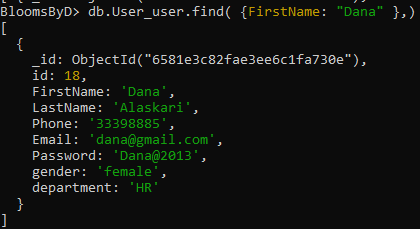


Figure 12. The data modeling concepts used is reference document and the relationship between the departments and employee is one to many

## Usage of MongoDB Index



Figure 13.creating an index on the departments

## Query Diagnosis and Analysis

The "explain" method with the "ExecutionStatus" parameter is used for diagnosing queries in MongoDB. This method triggers the query optimizer to evaluate different plans, selects the most optimal plan, executes it, and provides detailed statistics about its execution.



Figure 14.the analysis conducted prior to using an index reveals that the query is performing a collection scan (COLLSCAN).you can notice the works is 6



Figure 15.This analysis have been conducted after creating an index , you can notice the difference in the number of works and needed time

# Task3: Using Django to build the Web Application using Bootstrap

## Creation of VirtualEnv for Django



Figure .Creation of the ProjEnv



Figure . Activating the environment



Figure .starting new project



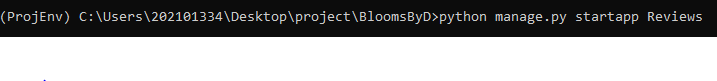




Figure .craeting the three apps that I have which are the Catogries, Reviews, User that the project will be built on.

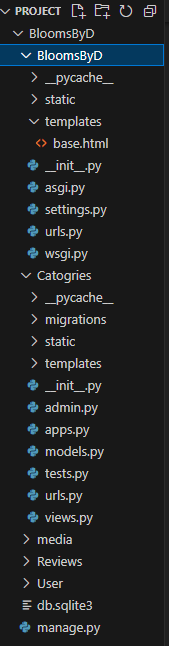
**-Project files organization**

Figure 20.structure of project files

In my project file organization, I have a project called "BloomsByD." This project folder contains important files such as "settings.py" and "base.html," along with other initial files. These files are crucial for setting up and configuring the project. Furthermore, I have created separate folders for each app within the project. These app folders include "Categories," "Reviews," and "Users." Each app folder contains its own set of initial files, as well as directories for templates and static files. This structure allows for better organization and separation of concerns within the project.

## Project settings used

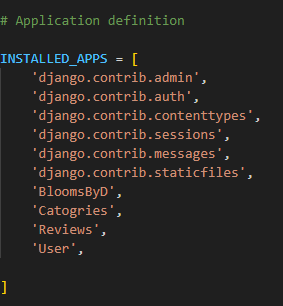


Figure 21.All the aps are added to the installed\_apps for it to function



Figure 22.This is the templates of the project

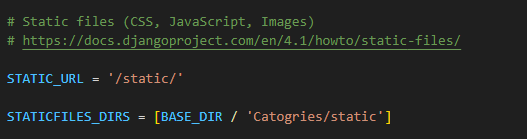


Figure 23.this is the static folder where all the images are stored

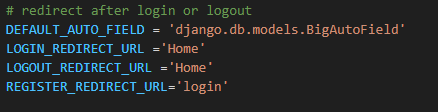


Figure 24.The urls for directing the user after the login and logout

## Connectivity of Django with MongoDB

By enabling Object Document Mapping (ODM), or the mapping of Python objects to MongoDB documents, Djongo acts as a bridge between Django and MongoDB. The clean and well-structured storage of data in the MongoDB database is guaranteed by this integration. Djongo efficiently makes it possible to integrate Django's ORM (Object-Relational Mapping) features with MongoDB, giving developers the opportunity to take advantage of both technologies' advantages in their applications.

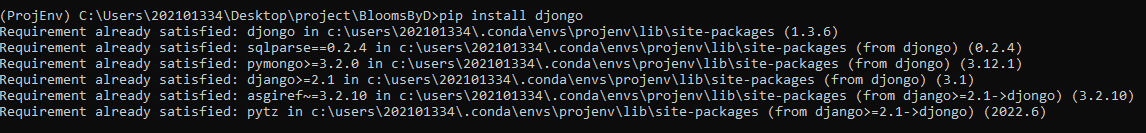


Figure 25.installing djongo in the project environment

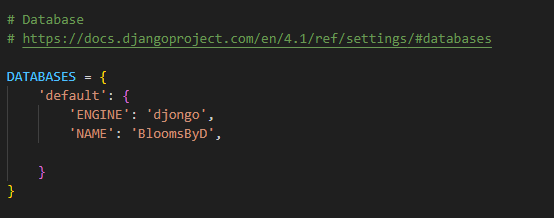


Figure 26.Connecting the Django and MongoDB defining 'djongo' as the engine for it



Figure 27.after each modification makemigrations must be runned.

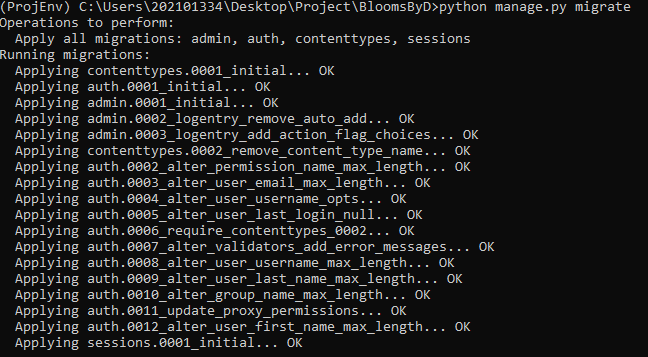


Figure 28.after running makemigrations the migrate command should be runned for migrations to be done

## Django Template Language

The Django template language allows Python code to be embedded in HTML templates using distinct syntax and tags. These tags and filters provide capabilities for executing logic, controlling flow, and manipulating data within the template. Templates typically consist of variables represented as {{example}}, which are replaced with their evaluated results by the template engine. Tags, denoted by {% example %}, offer more extensive functionalities. They can generate text in the output, control flow through loops or logic, and load external information into the template for later use.

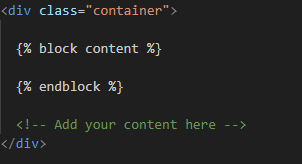
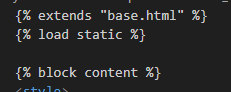


Figure 29.The ‘block’ content tag is defined in the ‘base.html’ template file and serves as a designated space that indicates to the template engine that the child template can replace this section with its own content.



*Figure 30. The extends tag at the top of each template indicates that it builds upon another template, which is base.html. The load static tag is used to access static files like images. The block content defines a section that can be overridden by a child template.*

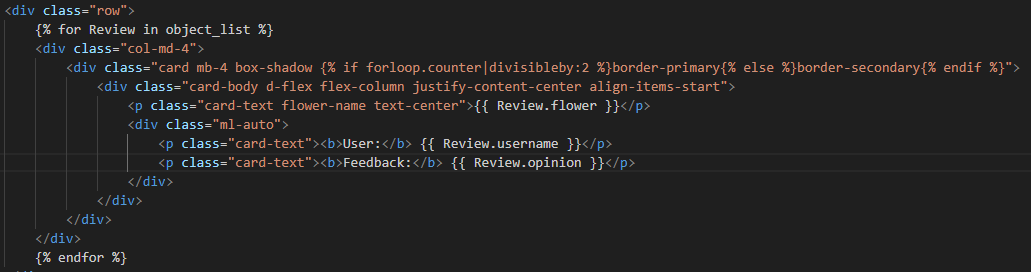
**

Figure 31.This is a for loop that iterates over the reviews. The variables flower name, username, and opinion are enclosed within double curly brackets `{{ }}` to display their values in the page.

## Model –View-Template

The Model-View-Template (MVT) design pattern, which is implemented differently in Django but has some similarities to the Model-View-Controller (MVC) model, is used in the framework. The model component functions as the data access layer and controls database objects in the MVT architecture. The user interface functionality is managed by the template, which also represents the presentation layer. As the controller, the view component helps the model and template communicate with each other. The view in the MVT pattern interacts with the model to retrieve and manipulate data, and it also executes the business logic. The answer that Django produces in response to the user's request is then generated by rendering a template. An essential element of the MVT design, the view it manages the application's general behavior and coordinates the flow of data.

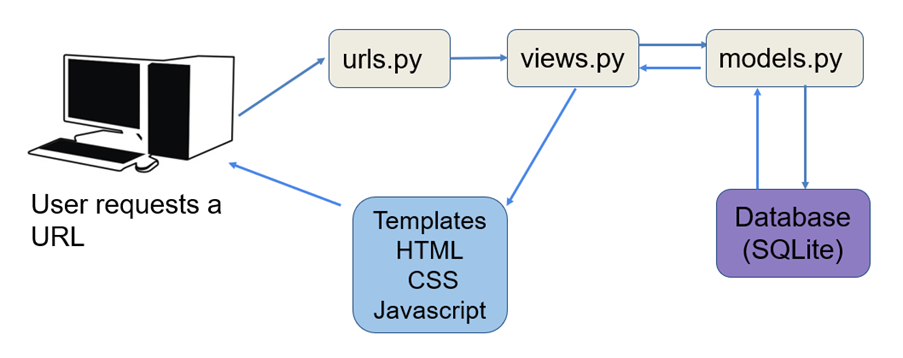


Figure 32.Model-view-template image

To apply what we have discussed:

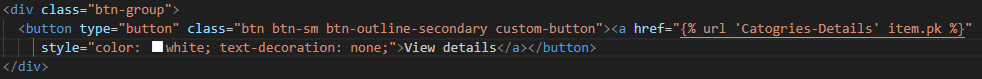


Figure 33. when clicked on this button , a request will be sent to the categories-details URL



Figure 34. this is the URL , where its mapped to a view called catogriesDetails that will do the work

## Django Admin site

Figure 35. the view will get the category model and display all details in the redirected template name which is the "views.html"



Figure 36. Th is the command to create the superuser

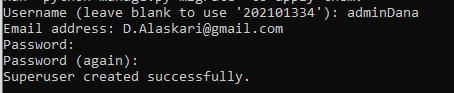


Figure 37. the superuser have been created successfully

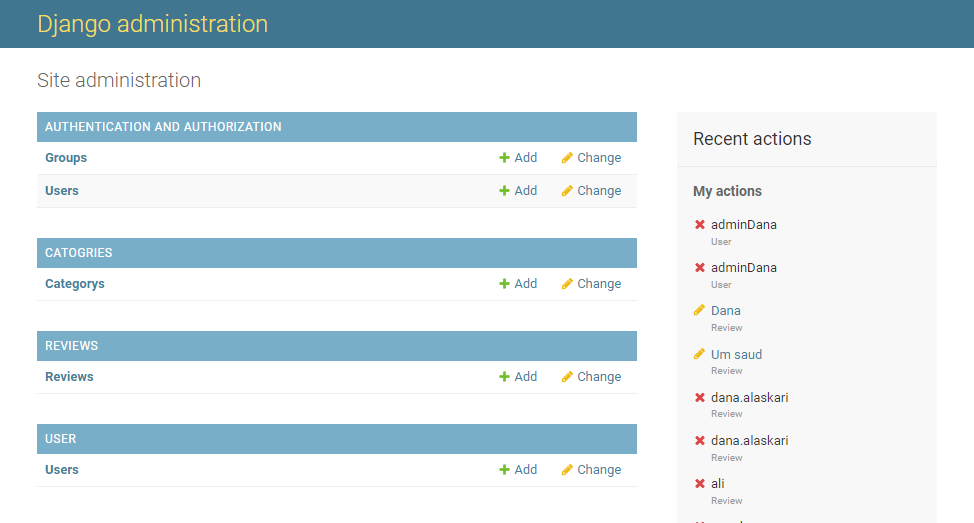


Figure 38.This is how the admin dashboard looks like where all model objects is shown, in addition to the crud operations for all objects

Figure 39.I specified the admin in the URLs file that why it could be accessed

## CSRF tokens used in forms

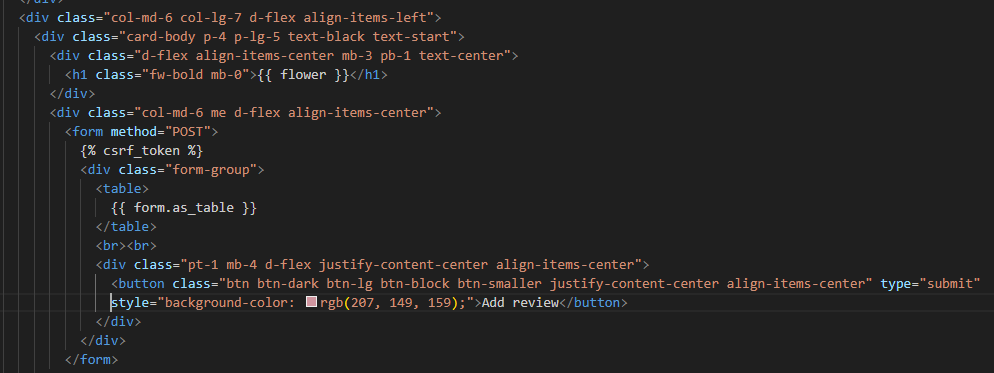


Figure 40. The "add\_review" feature utilizes the CSRF token tag as a preventive measure against malicious attacks. The server generates a token when rendering the page, which is then used to verify subsequent requests for authenticity and security purposes.

## Django forms

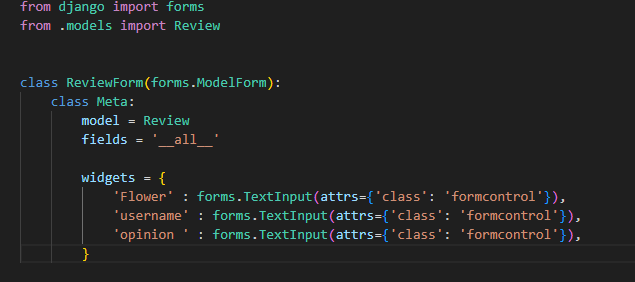


Figure 41.in the review’s app, I have a forms.py file which stores the form imported from Django to create a review Form to a chosen flower

## Incorporation of Bootstrap

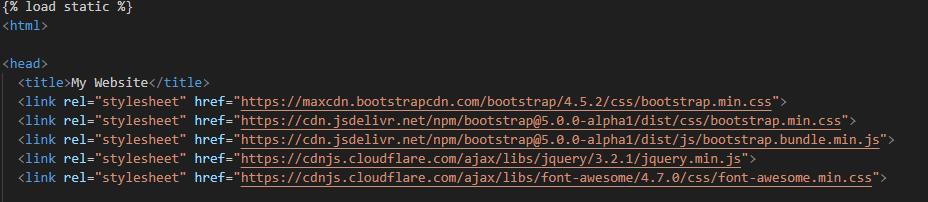


Figure 42. I have incorporated bootstrap in multiple of my pages ad one of them is the base.html, the bootstrap is used to structure and style the page

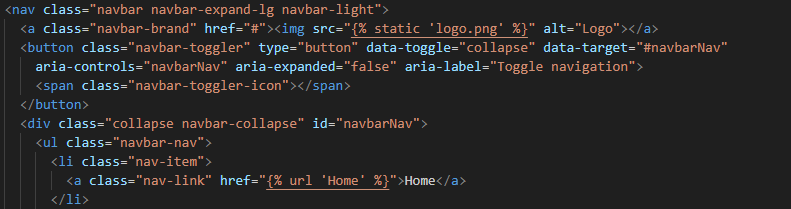


Figure 43.I have also used bootstrap in the navigation bar where I took a ready nav with its classes and restructured it to what suits me

# Task 4 – Overall GUI and working, Report, GitHub, Video and Reflection

## Overall Navigational GUI

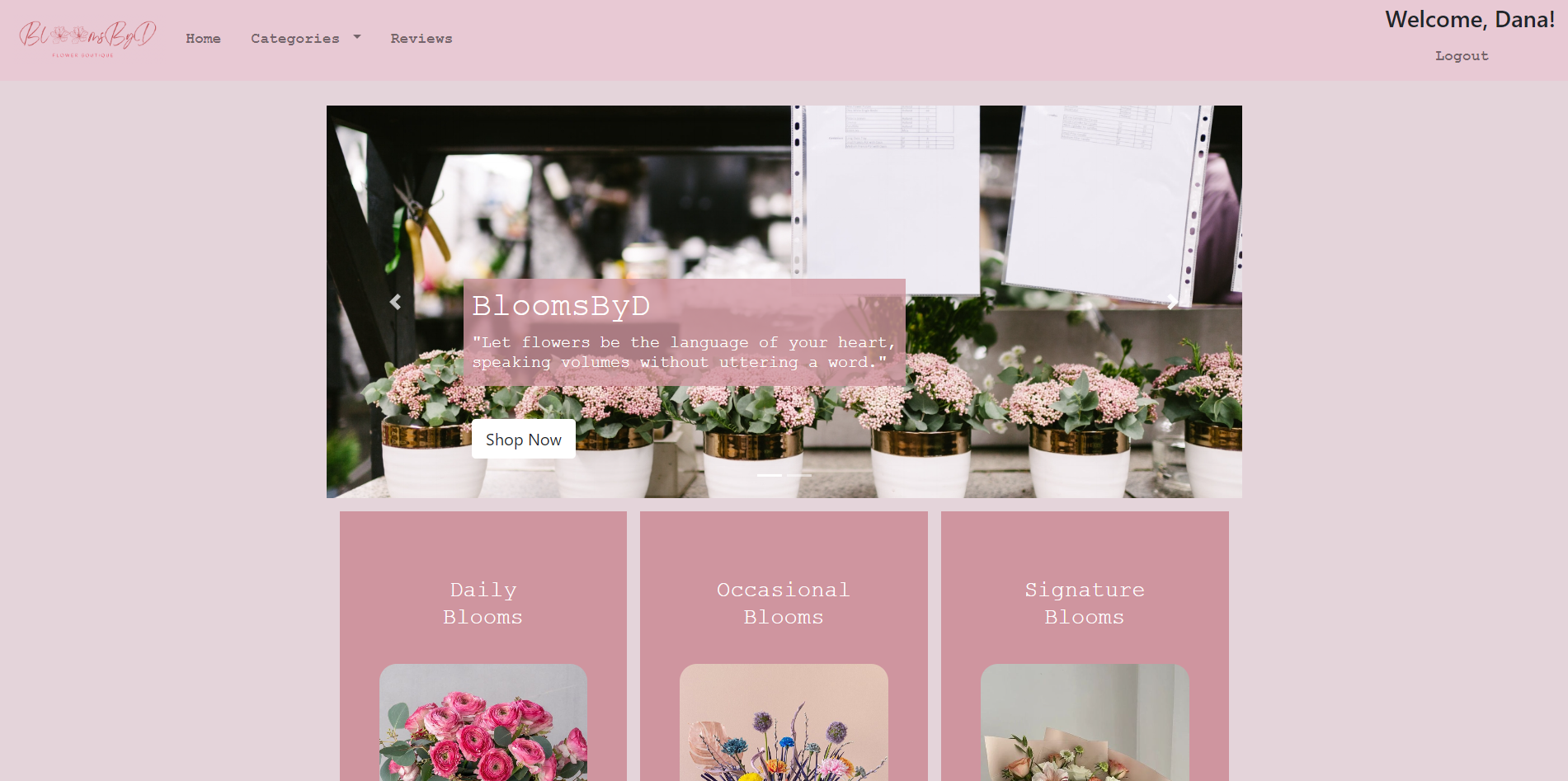


Figure 44.This is the home page where all actions can be performed from, the page prints a welcome message with the name of the logged in user



Figure 45. This is the navbar, if the user is not logged in there will be a button in the top right of the page

## Functionality

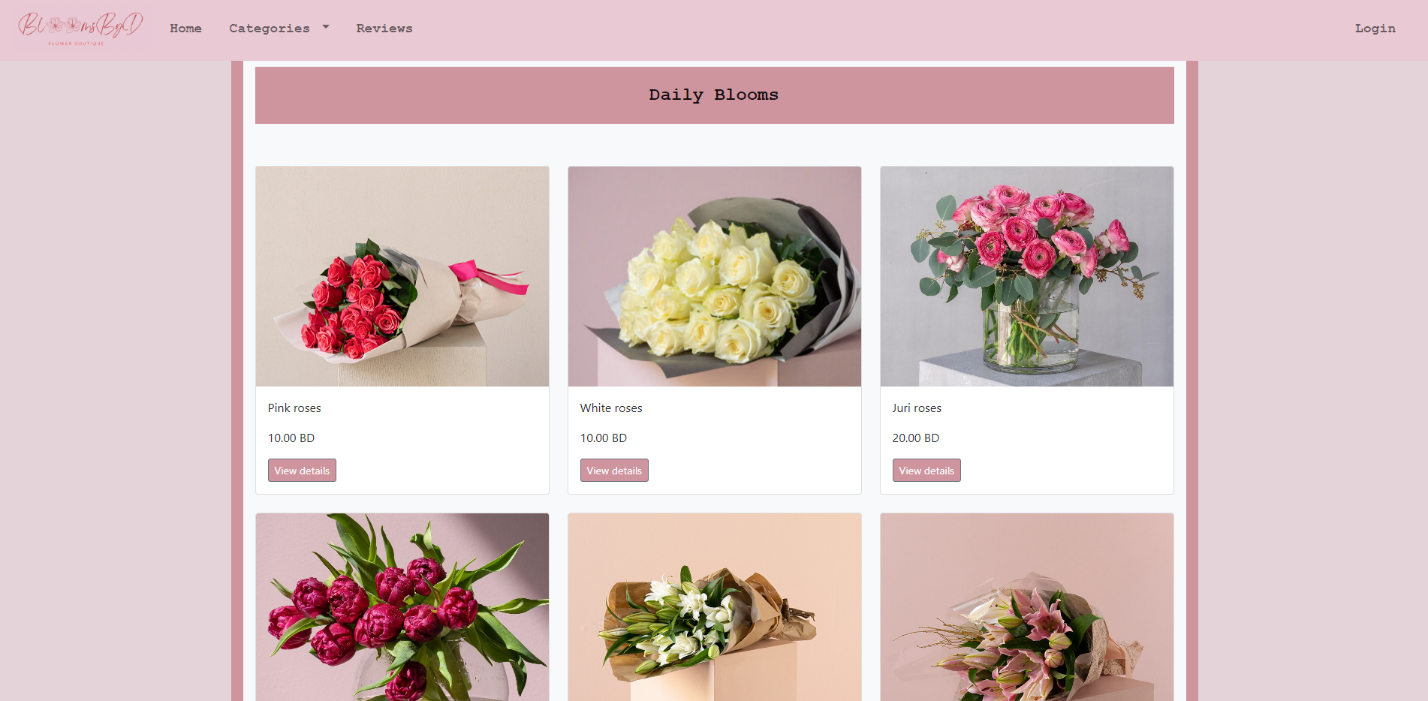


Figure 46. when the user clicks on "shop now" from the home page it will redirect him to this categories page with all flowers displayed

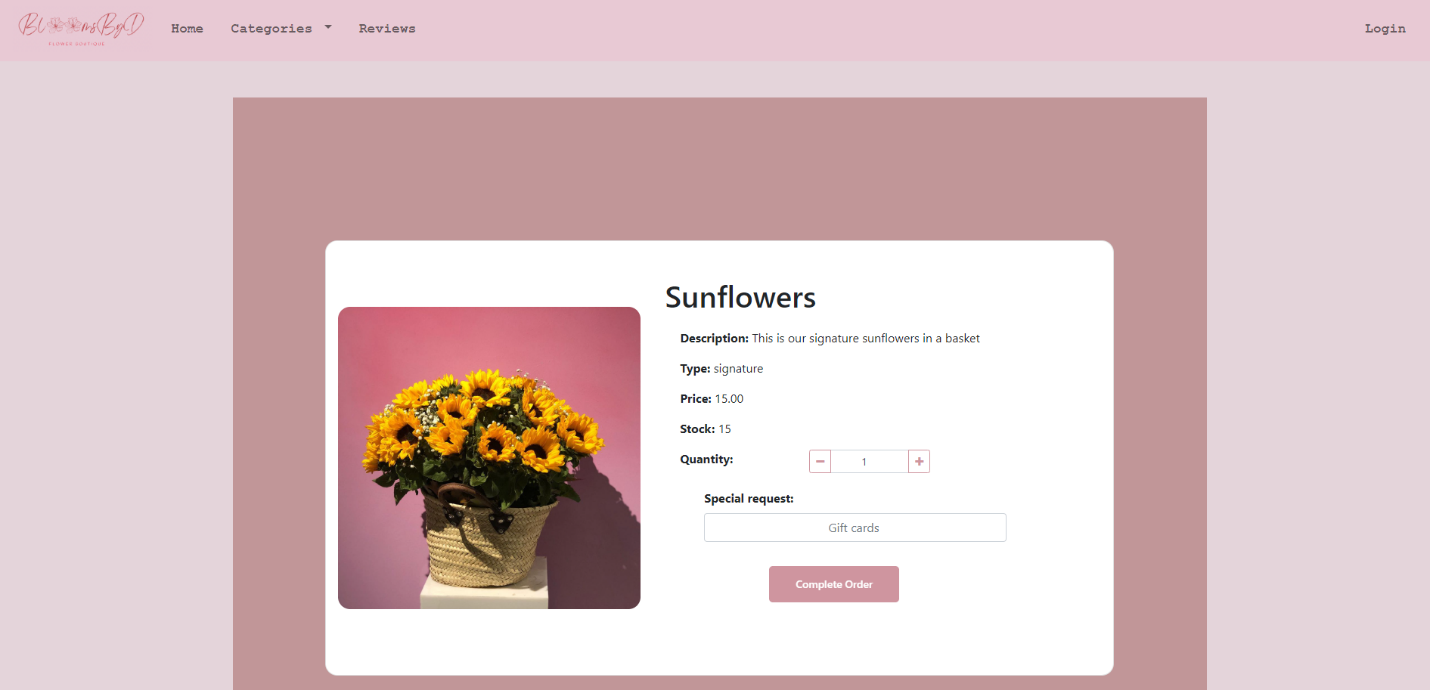


Figure 47.when clicking on "view details" flower details will be displayed

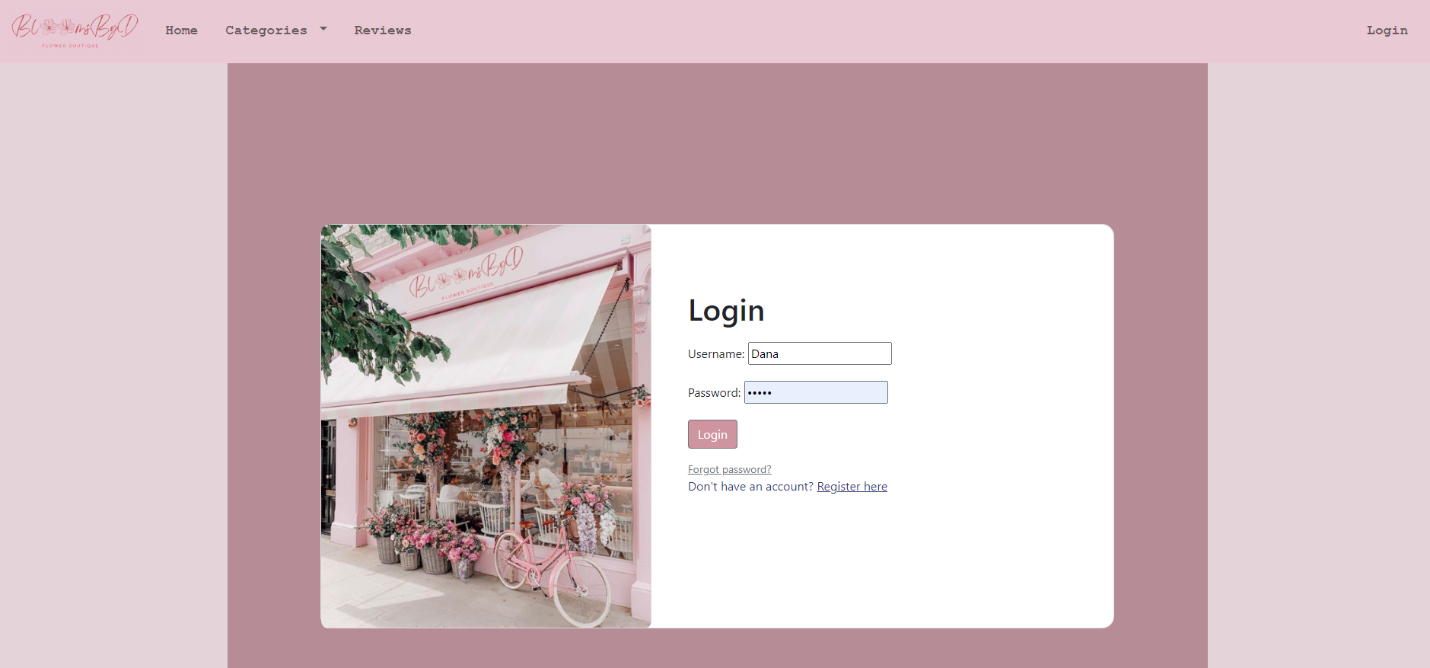


Figure 48.This is the login page

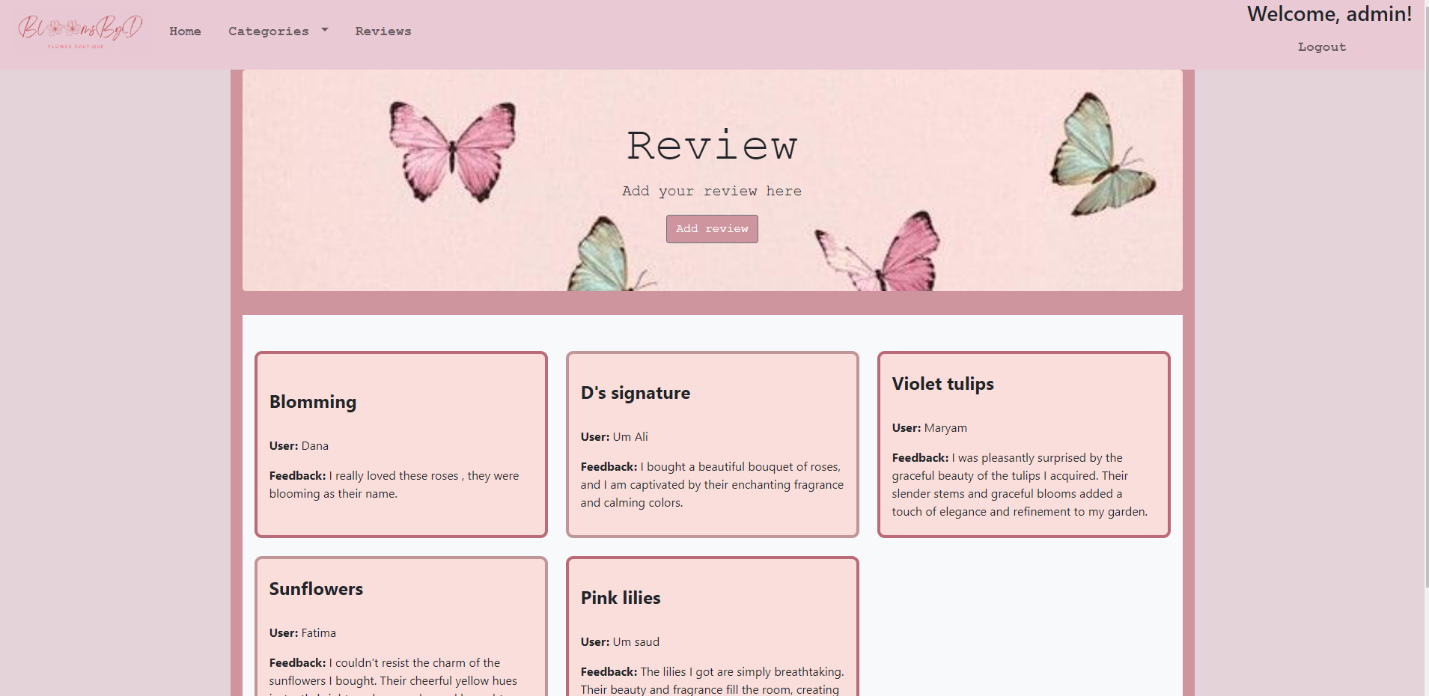


Figure 49.this is the reviews page you could either add a review and it will pop down or just look at the peoples reviews about certain flowers

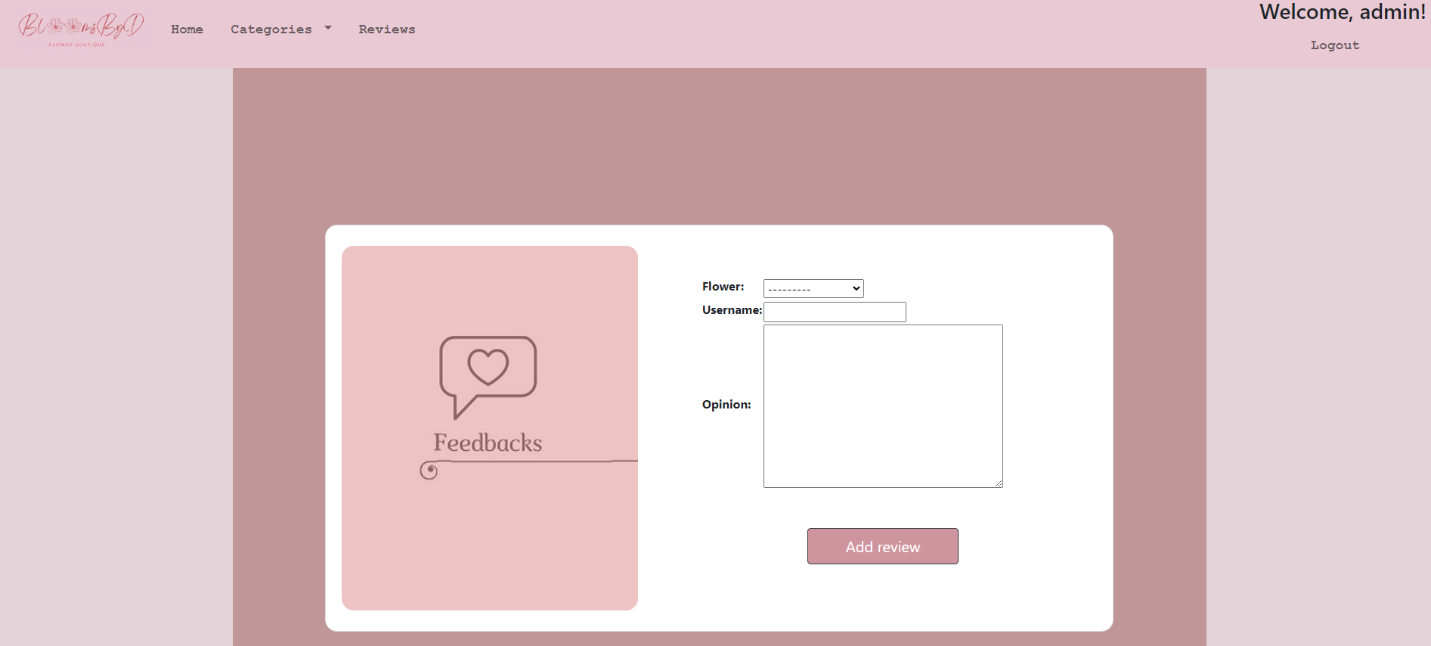


Figure 50.This is how the add review looks like, the user will be able to select the flower from the dropdown menu and add a review on it

# Reflection

It was an exceptional and demanding experience working on this project. Throughout the process, I encountered various difficulties, but I also found great satisfaction in witnessing the gradual development of my website. During the brainstorming phase, I had multiple ideas for the website's functionalities. Eventually, I selected the most functional and relevant ones, with the intention of adding more in the future. One of the initial challenges I faced was installing the required apps like MongoDB, Python, and MongoDB Compass on my MacBook. It was a bit challenging to figure out the process since there were limited resources available, this was my first obstacle. To overcome this, I decided to work on the project at the university campus. However, this presented its own challenges, as I had to be present daily, including public holidays and weekends, to make progress. I conducted thorough research, utilized the information provided in class, and used the resources available in our labs as a guideline. Furthermore, I encountered several errors and crashes when I first started. These included issues such as "no-template found," "no reserve match," and "404 page not found." Additionally, I sometimes forgot to make migrations after each change, resulting in the project not functioning properly. However, as I gained a better understanding of the development process, things became smoother. I established a schedule and created a checklist to facilitate my working strategy, which made the overall process easier. Another challenge I faced was when creating my GitHub account. I faced some issues, and there was a risk of losing my data. The problem was that the data was large, making the pushing process time-consuming. Additionally, recording the project video on campus proved to be difficult due to the lack of quiet, empty classrooms. Despite these challenges, I managed to overcome them. Due to time limitations, the frontend design of the project needed more time to work on and develop. However, I plan to address this in the future. Currently, the project only includes the basics, but I am eager to continue working on it beyond this semester and develop a business around my website. In conclusion, I thoroughly enjoyed working on this project. I am grateful to my classmates and teacher for their assistance and guidance throughout this informative and rewarding experience.