



BSc. (HONS) COMPUTING

BREAD

PROJECT REPORT

SUBMITTED BY
ANURAG BHARATI
CUID 11494830
BATCH 29 A

SUBMITTED TO
HARI S. SHRESTHA
MODULE LEADER
OF ST5007CEM



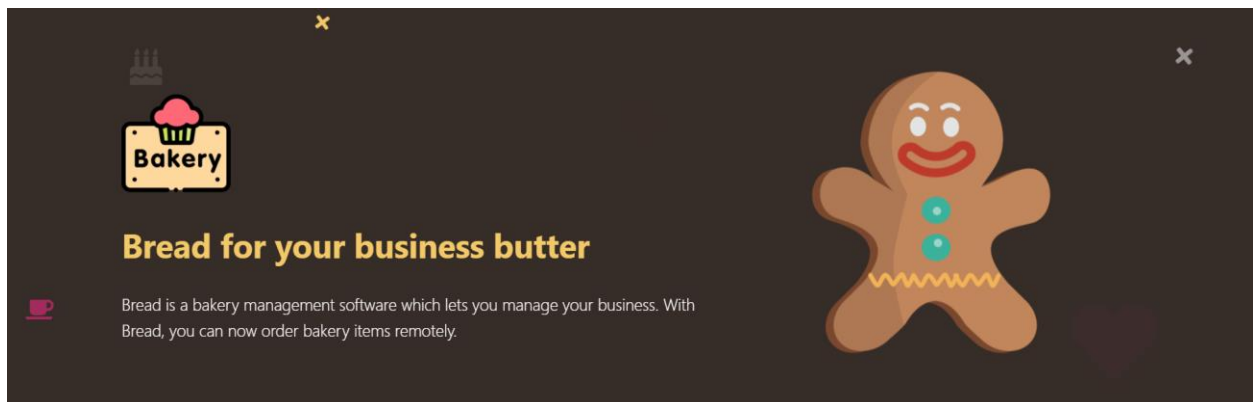
INTRODUCTION	4
BUT WHY BREAD?	4
FEATURES OF THE SYSTEM	5
REAL TIME DATA PROCESSING	5
EMAIL WITH VERIFICATION LINK	5
EMAIL NOTIFICATION ON ORDER COMPLETION	6
EMAIL NOTIFICATION ON ORDER DELETION	6
ADMIN DASHBOARD	7
THE USER INTERFACE AND THE DARK MODE	7
THE SERVER-SIDE PAGINATION AND SEARCHING	7
AUTHENTICATION AND AUTHORIZATION	8
WHATS THE AIM OF BREAD?	8
WHAT ARE MY OBJECTIVES?	8
PERSONAL OBJECTIVES	8
PROJECT OBJECTIVES	8
WHAT HAVE I LEARNED?	9
ANALYSIS	9
USE CASE DIAGRAM	10
RICH PICTURE	10
THE FRONT END	11
INTRODUCTION OF TECHNOLOGY USED	11
HYPER TEXT MARKUP LANGUAGE (HTML)	11
CASCADING STYLE SHEETS (CSS)	12
JAVASCRIPT (JS)	12
OTHER FRAMEWORK AND LIBRARY USED	12
A SCREENSHOT	12
CHALLENGES OF FRONT-END WEB DEVELOPMENT	13
ACCESSIBILITY	13
COMPATIBILITY	13
USABILITY	13
RESPONSIVE	13
THE BACKEND	14
INTRODUCTION OF BACK-END TECHNOLOGY	14
DJANGO	14
FLASK	14
WHY I USED DJANGO	15
FEATURES OF DJANGO	15

RAPID DEVELOPMENT	15
SECURE AND SCALABLE	15
VERSATILE	15
CLASS DIAGRAM	16
THE DATABASE	17
WHATS POSTGRESQL?	17
WHAT'S AND WHY MYSQL ?	17
ADVANTAGES AND DISADVANTAGES OF MYSQL	18
ADVANTAGE OF MY SQL	18
DISADVANTAGE OF MYSQL	18
HOW TO INTEGRATE MYSQL IN DJANGO	18
ERD OF BREAD	19
TESTING	20
UNIT TESTING	20
PROJECT ISSUE	24
ISSUES DURING THE PROJECT DEVELOPMENT	24
LIMITATIONS OF BREAD	24
FUTURE WORKS	24
CONCLUSION	24
LINKS	25
REFERENCES	25
APPENDIX	26
BACKUP LINKS	26
GITHUB : HTTPS://GITHUB.COM/ANURAG-BHARATI/BREAD.GIT	26
YOUTUBE: HTTPS://YOUTU.BE/FYGQSHBLQGY	26
SCREENSHOTS OF BREAD	26

INTRODUCTION

Project management has become more significant than any other time. As business turns out to be more perplexing and more noteworthy works are being doled out to the group, the management becomes a lot harder ([Liberatore & Pollack-Johnson, 2003](#)).

A kind of software used to streamline and robotize repetitive management processes to minimize the management intricacy of big errands is known as management software. Businesses have been using this kind of software commonly to manage their business, and individuals have used them to manage their expenses, time, or bills. Some of the types of management software are Record management software, Task management software, Trade management software, Asset management software, and much more.



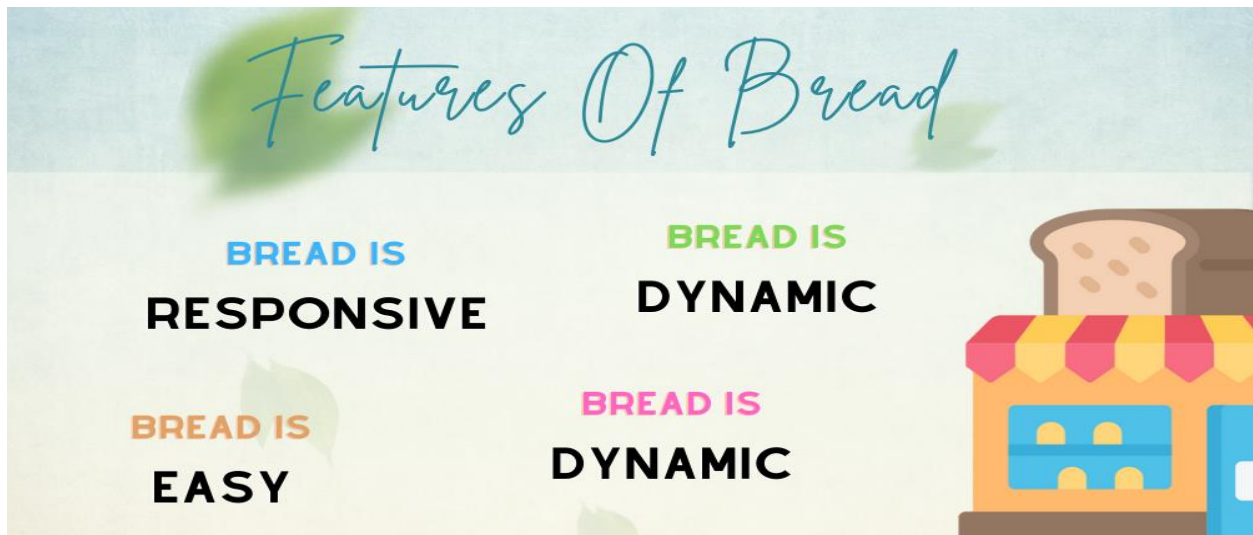
Bread is a business management web application that deals with the management aspects of businesses like bakery and café stores. Bread has been built with care to make the menu and the ordering process of food items specific to a business much easier, collaborative, customizable and most importantly, safe.

BUT WHY BREAD?

Safety and time are the two main reasons behind Bread. Business staff and their customer have been working in a risky condition during this global pandemic. Putting a mask every day for the whole working hour and making the ear's hurt is no good. People don't want that, but instead, want to work in a relaxed and pain-free environment. Even if they got a mask on, all customers that use the service would not have had one or are wearing the mask ineffectively, which is no good either. With the help of Bread, their interaction would immensely get minimized and ultimately solve this issue.

FEATURES OF THE SYSTEM

Bread is a feature-rich management application baked into a compact and responsive web application. In Bread, you can add your product with ease, edit them with fewer clicks and delete the one that is irrelevant to your current menu. The core feature of Bread is its real-time data handling. Some of the features of Bread are mentioned below.

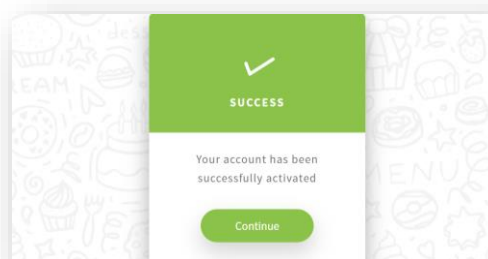
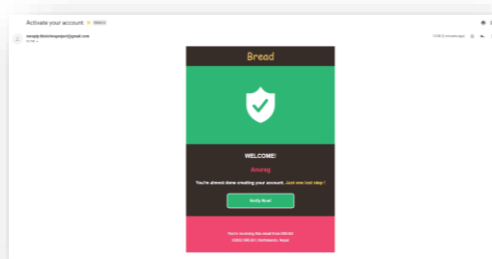


REAL TIME DATA PROCESSING

Customer requests are processed quickly and displayed on the dashboard. Products can be added, updated, and deleted within seconds, with the limiting factor being the internet connection.

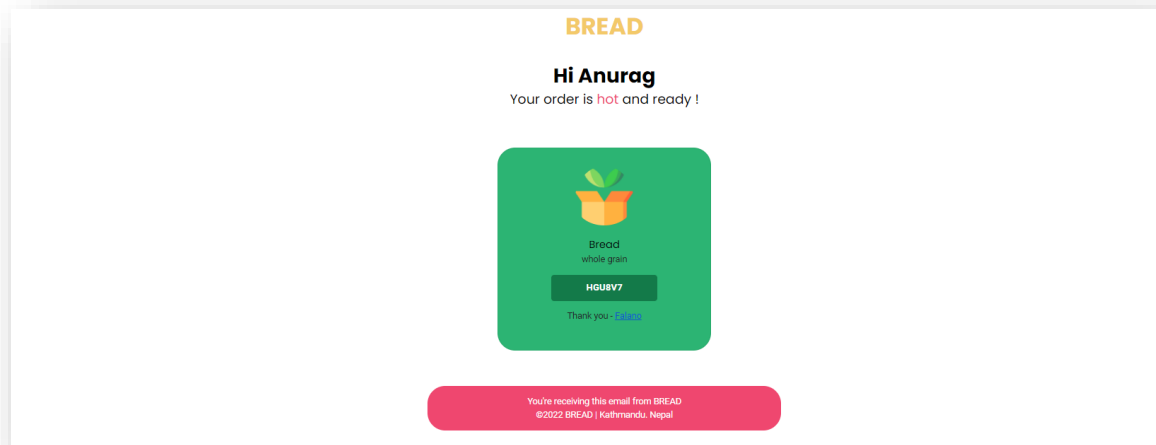
EMAIL WITH VERIFICATION LINK

Upon registration completion, an email with an appropriate attachment is sent to the owner. The email will send in the background, and the customer can navigate to the login page without getting stuck.



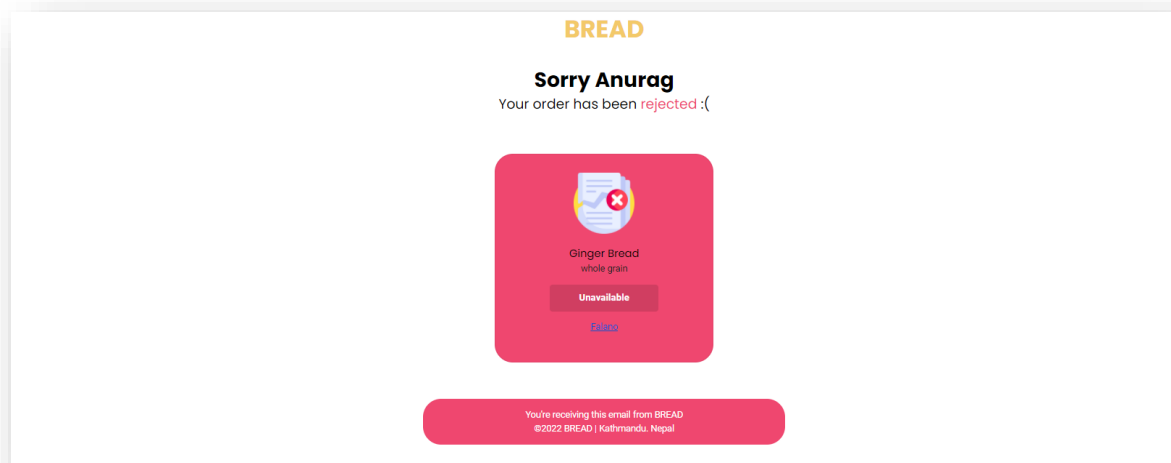
EMAIL NOTIFICATION ON ORDER COMPLETION

Upon order completion, an email with an appropriate attachment is sent to the owner. The email sending will happen in the background, so the staff would not have to wait for the email to be sent and can continue their work.



EMAIL NOTIFICATION ON ORDER DELETION

Upon order deletion, an email with an appropriate attachment is sent to the owner. The email sending will happen in the background, so the staff would not have to wait for the email to be sent and can continue their work.

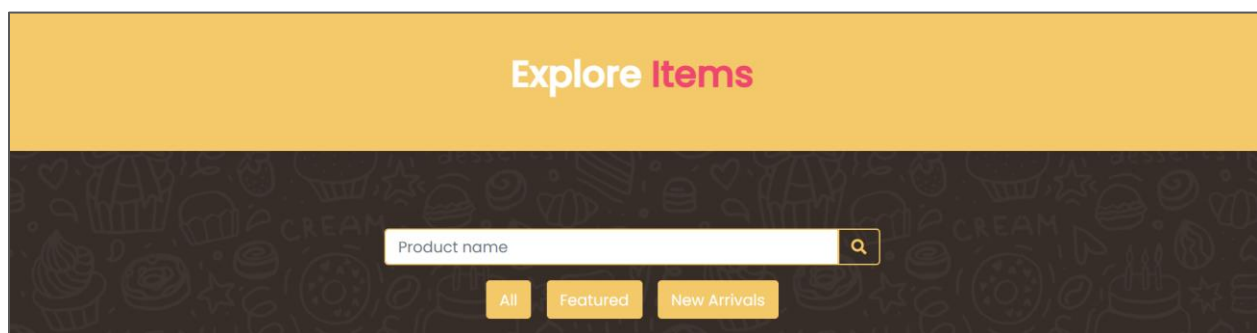
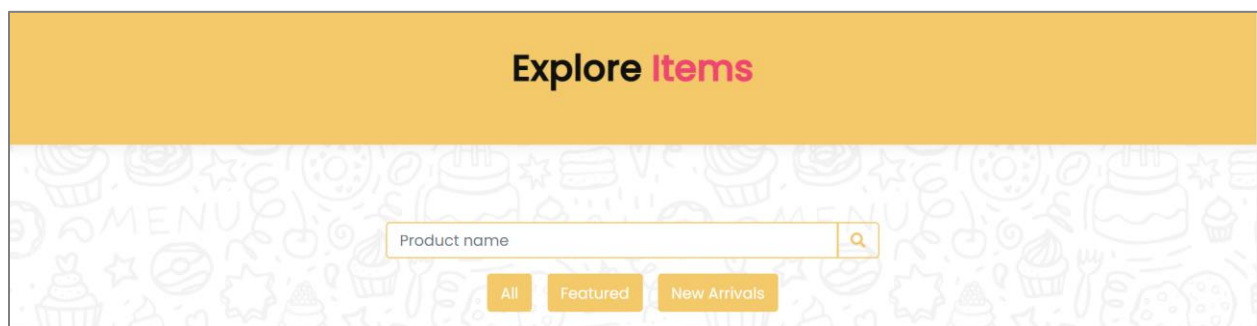


ADMIN DASHBOARD

Staff and the manager can easily see the metrics and the data in one place. The dashboard shows the general info such as staff count, customer count, product count, order count, delivery count and order list containing orders that have not been processed yet. The admin dashboard panel also shows a doughnut chart that contains the details about orders and their status.

THE USER INTERFACE AND THE DARK MODE

Bread features a simple and interactive, responsive user interface which is vivid and has a built-in dark mode that remembers its state throughout the web application life cycle. It also features some minimal animations, click feedback, and shows the message feedback of every main interaction.



THE SERVER-SIDE PAGINATION AND SEARCHING

By considering the application's responsiveness, BREAD has featured server-side pagination, which optimizes the product menu and the table-view for mobile devices. This feature is one of the most important because it enhances the user experience and reduces the time taken to load the website. The customer page also has a search and filter options.

AUTHENTICATION AND AUTHORIZATION

Bread authentication authenticates the user before processing their requests. If an anonymous user tries to visit any of the staff sites, the user will be redirected to the login page. Similarly, If the unauthorized person tries to order a product without logging in first, the website redirects him to the login page. Bread authorizes users based on their assigned roles. If a customer tries to visit any staff pages, the website redirects the customer to the product page. On registration completion, the user must verify their identity to get the active serving. The user must verify him by going to their email and clicking the “verify” button.

WHATS THE AIM OF BREAD?

Simplifying the business management and providing an interactive food menu is the aim of Bread.



WHAT ARE MY OBJECTIVES?

PERSONAL OBJECTIVES

1. Identify and evaluate the web design of existing websites.
2. Evaluate the effectiveness of responsive and dynamic website using appropriate tests.
3. Identify web development approaches suitable for modelling good web pages.
4. To learn and develop website design skills
5. Develop a website and Complete the final report.

PROJECT OBJECTIVES

1. Create a dynamic, responsive, and easy to use web application.
2. Make the user interface clean, minimal, and comfortable for the users.
3. Create email notification, server-side pagination, searching and filtering
4. Create file upload and download service for the product image.
5. Create automatic verification and validation system.

WHAT HAVE I LEARNED?

From this journey, I have had learned so many new things since day one that I can't even explain. Initially, I learned about how the web works behind the scenes, the elements of the client-server web model and learned the core technology used to make a website.



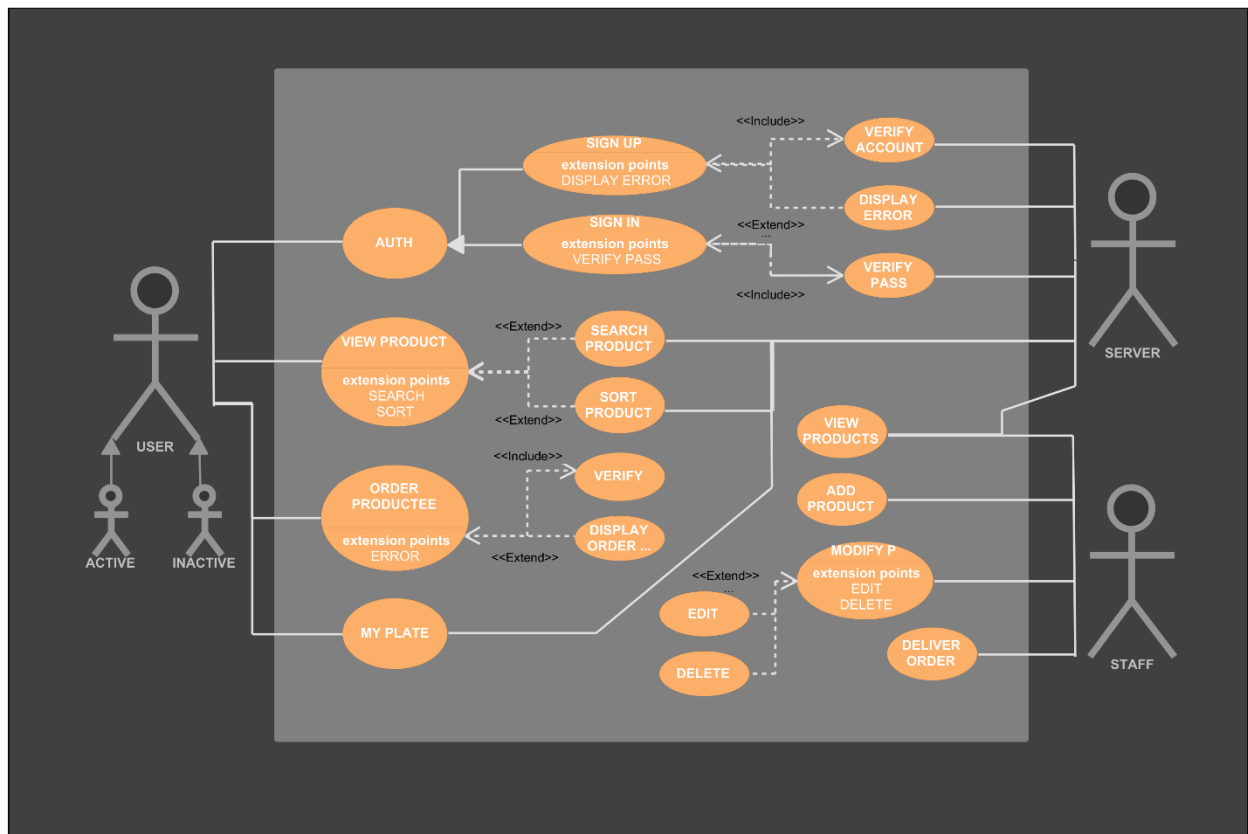
I have had learned how to layout the architecture or the frame of a web page using HTML 5, learned about how to style them using CSS 3 and, in this phase of learning, I learned about the BOOTSTRAP framework and how it helps to make a rapid website with nice looking web components. Then, I learned JavaScript, one of the core technologies that help to make a website even more dynamic. Later, I got to know about this amazing JavaScript library called jQuery that made the DOM manipulation much simpler.

After I grasped the core knowledge of front-end design, I moved to the back end, in which I got to understand Django, A python back-end framework for the web. And at last, I learned MySQL database.

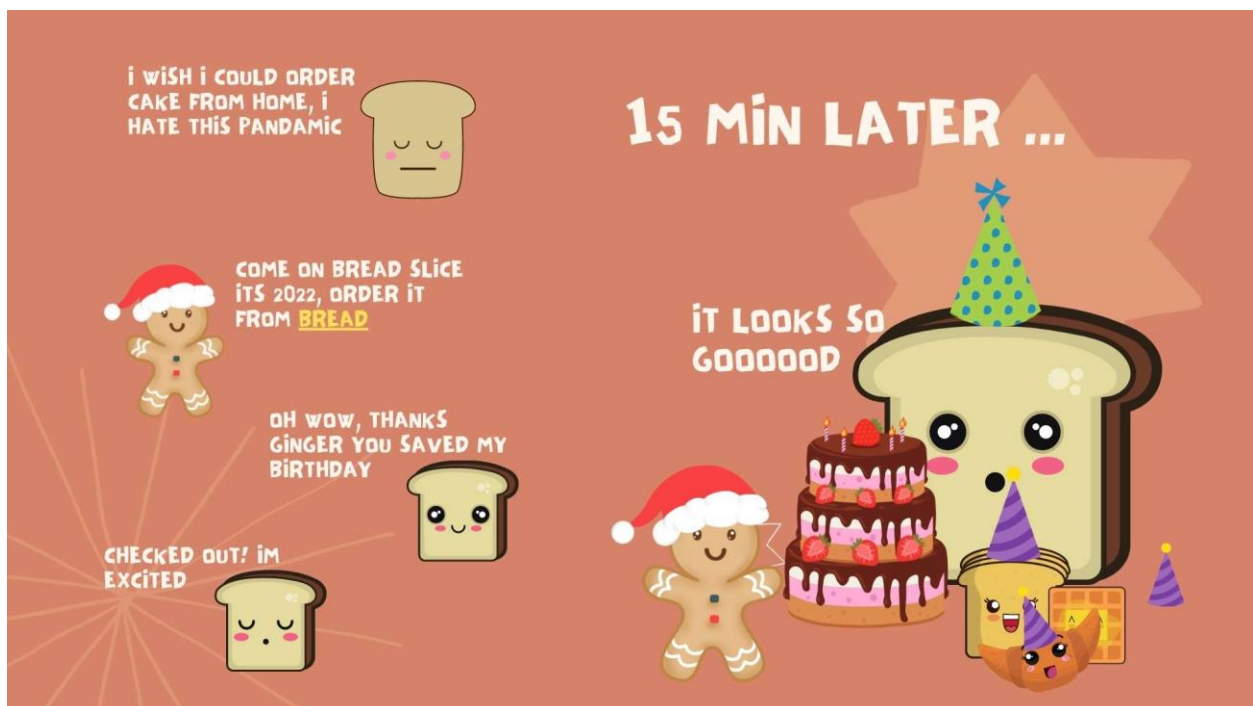
ANALYSIS

The expanding strain on associations to enhance more realistic in what they convey and how they work implies there is an unmistakable requirement for intercessions that improve the advancement abilities of representatives (Birdi, 2021). The analysis is the backbone of a project which is done to break down a complex topic into its smaller parts and with the main motive to simplify and gain a better understanding of the project.

USE CASE DIAGRAM



RICH PICTURE



THE FRONT END

The front end is the chunk of the site that a user or client connects with. Front end web development, otherwise called client-side development is the practice of creating an interactive interface for the users which is not hard to navigate. There are three core building blocks of front-end web development, HTML for the layout, CSS for its styling and JavaScript for developing the interactivity of the web application. The goal of front-end development is to guarantee that when the clients open up the site, they see the data in a format that is not difficult to peruse.

INTRODUCTION OF TECHNOLOGY USED

Front end uses HTML, CSS, JavaScript, and The Document Object Model or DOM for short at its core. The DOM is an API for HTML and XML that represents web pages in a hierarchal structure. It is used to make the web page dynamic and eases the front-end development process. I have used these technologies and its frameworks to develop Bread.



HYPER TEXT MARKUP LANGUAGE (HTML)

The Hypertext markup language is the key markup language that characterizes the structure of content on a web page. It was invented in 1989 by the creator of the world's first browser, Tim Berners-Lee. It has evolved since to HTML5. Modern HTML contains tags to handle a variety of different media types like audio, video, and canvas.

CASCADING STYLE SHEETS (CSS)

By default, the web browser renders HTML as a boring black-and-white document. CSS provides the presentation of this markup. Cascading Style Sheets, also known as CSS in short, is a styling language used to modify web page appearance. It allows web designers to add style to Web pages and have more dynamics over the visuals of a website page like font styles, shadings, spacing, coloring, animation and much more.

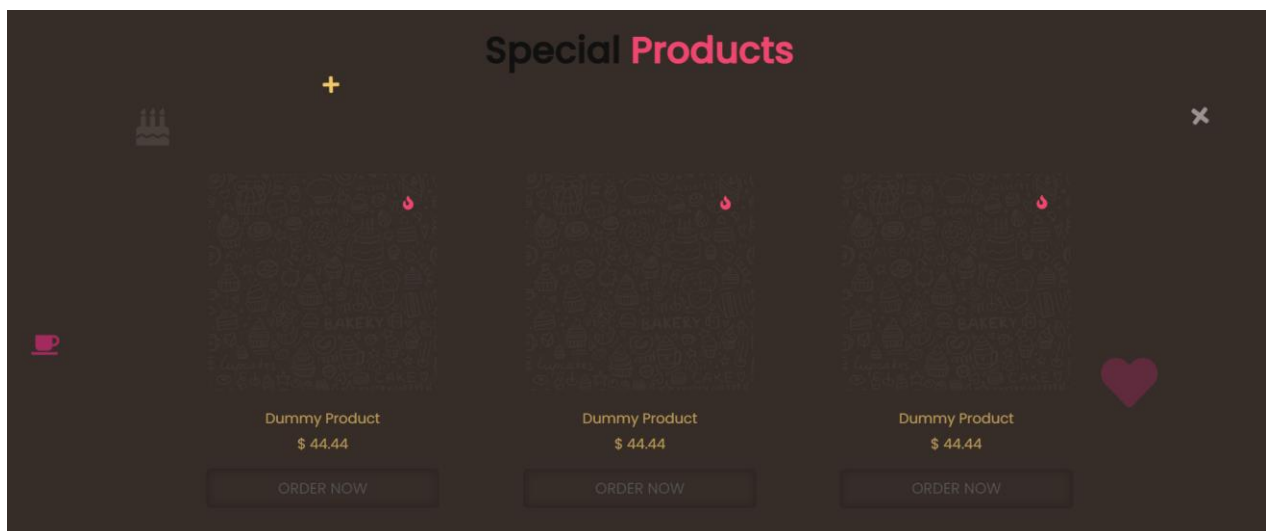
JAVASCRIPT (JS)

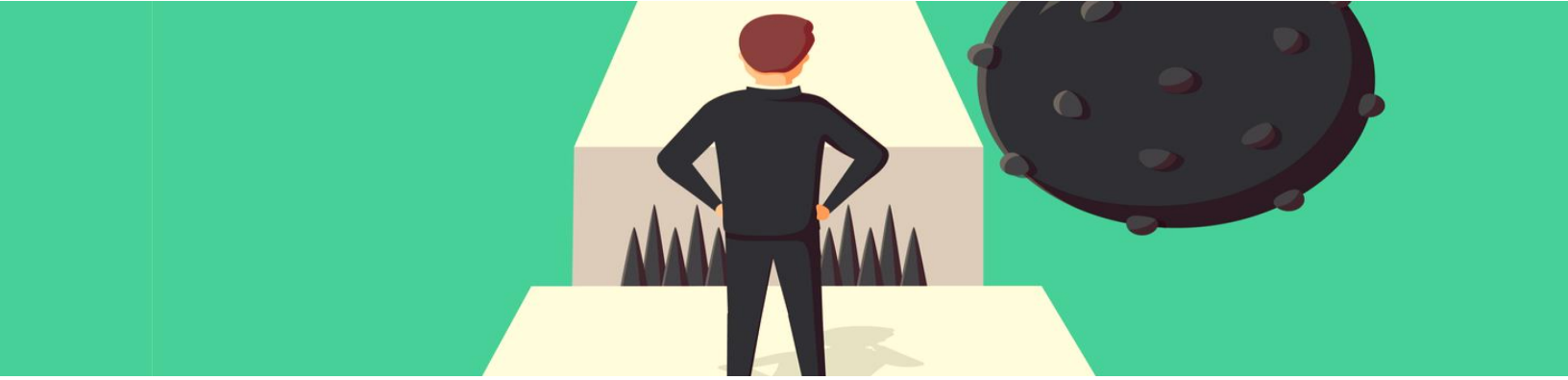
JavaScript is a high-level, multi-paradigm, single-threaded, garbage-collected, and dynamic programming language commonly used to program front end web applications but can also be used as backend programming languages using Node.js. It is the only language that is natively supported in modern web browsers. It is commonly used to add interactivity to the HTML elements via The DOM API.

OTHER FRAMEWORK AND LIBRARY USED

Bootstrap is a CSS framework used to create beautiful responsive web applications in no time. It is free and open source. This frame allows the web designer to write beautiful front-end UI without ever leaving the HTML. jQuery is a JavaScript library used to reduce the DOM selection queries. It simplifies the DOM tree traversal, operation and can even handle events. It is also free and open source. I have chosen these framework or libraries to ease the development process of BREAD.

A SCREENSHOT





CHALLENGES OF FRONT-END WEB DEVELOPMENT

The most common challenges of web development are given below

ACCESSIBILITY

When compiling ideas and planning a web application for its development phase, web designers must plan the web page in such a way that every end-user should be able to use it. Especially, for people with disabilities, Any content or feature should be fully accessible for them with ease. Implementing alt text in the logo, adding text to speech option for the article inside web page, choosing simple, clear, and easy to read fonts are some of the ways to make a website accessible for everyone.

COMPATIBILITY

There are several web browsers out there and people use different web browser to get the content available on the web. To make sure one code base covers all the available browser, front-end web designers must make the code compatible with any other browsers by following their protocol. If there is compatibility problem in a website, then the user experiences will not be the same for everyone.

USABILITY

Webpages that are user friendly and less complex entice more users and have a higher user base. Users don't want a complex layout. Complex webpage UI hinders the user's ability to accomplish a task that they intended to do on the website.

RESPONSIVE

Responsiveness is the most important aspect in web design. The user experience of a website must be equal in both mobile and desktop devices. If a web page is not responsive, it loses half of its user base because many people tend to use mobile phone all the time rather than pc or laptop. A webpage become responsive if it's web content or arrangement is flexible and fit within diverse screen sizes.

THE BACKEND

Any part of a website that is not visible to the client or its users is its back end. In engineering words, the backend is the "information access layer," and the front end is the "data presentation layer." While making web applications render on the client-side is tied in with the front-end development, back-end development is tied in with making these applications render server-side. Yet, it's more than that.

Almost 95% of today's sites are dynamic, meaning its client can change the state of the website. Because of the Back end, A user can request new data and even create one. The created data gets sent to the back end to get processed and stored while the requested one is fetched from the database and presented to the front-end.

INTRODUCTION OF BACK-END TECHNOLOGY

DJANGO

Django is a well-known Python open-source web development framework, commonly used to develop production-grade web applications significantly quick. It is a powerful and friendly system that allows backend developers to zero in on their applications by having pre-made features commonly used in the market. Django makes it more straightforward to build applications as opposed to wasting time. It is used in a wide range of tech stacks, including Instagram, Pinterest, Mozilla, and Eventbrite. Organizations wherever are effectively utilizing Django and put resources into its turn of events. In 2020, it is the fourth most needed web system as per Stack Overflow.

FLASK

Flask is a web development framework, written in Python that is made to simplify the backend development process. A framework is a bunch of bundles and libraries, that help engineers in making applications with fewer lines of code and accordingly less exertion since they are not expected to compose dull code.

It has been labelled as a microframework because:

1. It does not depend on other third parties' frameworks or libraries,
2. It has no extra component included.
3. Third party libraries can be added

WHY I USED DJANGO

Though, there are many well-known frameworks in the web development industry. Besides all, I have chosen the Django framework as it is user-friendly and easy to learn. Django is created for the quick development of perplexing web applications. Completely included right out of the case, devs have every one of the tools they need to carry out and grow effectively adaptable, solid, and viable web applications in record time. It has allowed me to create a beautiful, dynamic web application in no time and the project structuring way is less complex. Since the feature of the application is divided into several apps, which reduces coupling and increases cohesion between apps. Due to this, scaling and adding changes to the apps is easy and straightforward, and I don't have to worry about breaking other stuff while changing one.

FEATURES OF DJANGO

RAPID DEVELOPMENT

With the help of pre-loaded packages, Django makes it easier and promotes rapid application development. It has provided the wheel so that the developers don't have to make their own from scratch.

SECURE AND SCALABLE

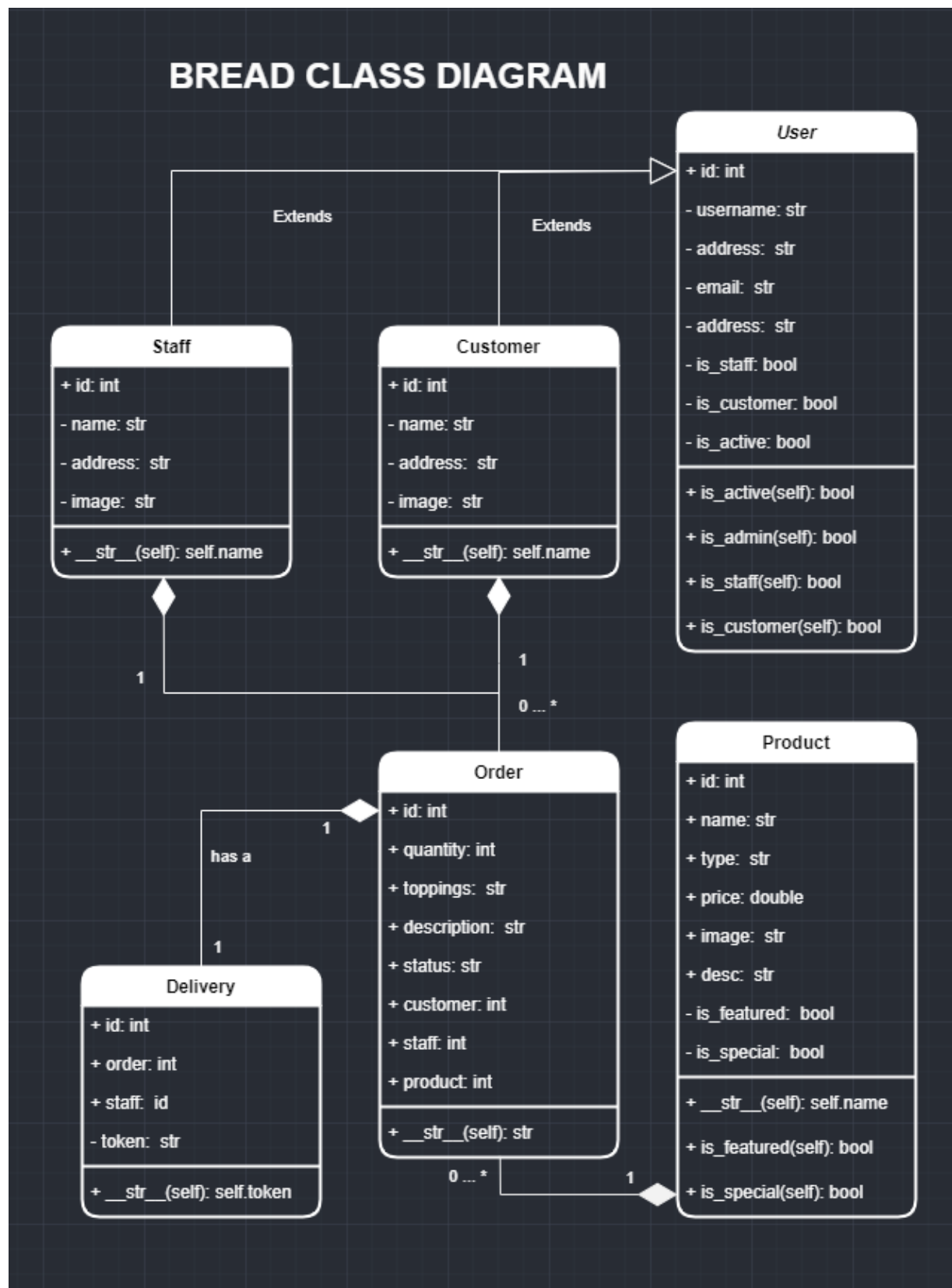
Django treats security in a serious way. It assists and keeps developers away from numerous typical security botches, like XSS, CSRF, SQL injection, and many others. The client validation framework provided by Django offers a secure method for overseeing sensitive client data like passwords and tokens.

A Django application can be quickly scalable without doing a huge amount of work. The Django object-relational mapping provides the easiest way to manage and model the database that abstracts away the database layer in the application in a remarkable way.

VERSATILE

Django is adaptable in nature which permits it to construct applications for various areas. Presently a day, Companies are utilizing Django to fabricate different sorts of uses like content administration frameworks, informal communities destinations or logical processing stages and so forth.

CLASS DIAGRAM



THE DATABASE

A database is a collection of well-structured and organized data that is set up for undertaking CRUD operation on it. Databases normally store accumulations of documents that comprise data, like client information, product information, financials, etc.

The database is ought to be managed in a regular interval of time, to keep the data normalized, and to store it efficiently. Database management software is specifically made for that purpose only. It is a program, which allows users to create a new database and make new tables inside the database that may have a relation with other tables.

WHATS POSTGRESQL?

PostgreSQL, otherwise called Postgres, is an undertaking open-source database administration framework. It upholds both SQL and JSON for common and non-common inquiries for extensibility and SQL consistence. PostgreSQL upholds progressed information types and execution enhancement highlights, which are just accessible in costly business database, similar to Oracle and SQL Server.

WHAT'S AND WHY MYSQL ?

MySQL commonly pronounced as 'my-sequel' is a free and open-source RDBMS that has been used in the industry since its formation. It deals with the relational table containing organized set of data which may be related to other table.

I have used my SQL for the following reason

1. Devs don't need to pay to use it as it has been delivered under an open-source permit.
2. It handles an enormous subset of the database packages.
3. It utilizes a standard SQL language which is common in the database community.
4. It is supported in many frameworks and with several languages including JAVA, C++, C#, PERL, PHP, PYTHON, and many others.
5. It functions splendidly even with enormous informational collections and works rapidly.
6. It is adaptable. The open-source GPL license allows developers to change their programming to accommodate their own behavior.

ADVANTAGES AND DISADVANTAGES OF MYSQL

ADVANTAGE OF MY SQL

MySQL is a cross-stage database server. It can run on various stages like Linux, Solaris and Windows and so forth. It is a decent decision for those projects that focus on various stages, especially web applications. It is a part of the famous Linux Apache MySQL PHP server stack which is utilized worldwide. MySQL support numerous stages with various dialects like C, C++, PHP, PERL, JAVA, Python and so on.

There are different secure and consistent association systems accessible to interface with the MySQL servers. These associations incorporate named pipes, TCP/IP attachments and UNIX Sockets. Info is safeguarded through secret phrase and the beneficial thing about these passwords is that they are put away in scrambled structure and cannot break these perplexing encryption calculations. MySQL is worldwide perceived as the greatest solid and dependable database. It is being utilized in famous web applications including Facebook, WordPress, and Twitter.

DISADVANTAGE OF MYSQL

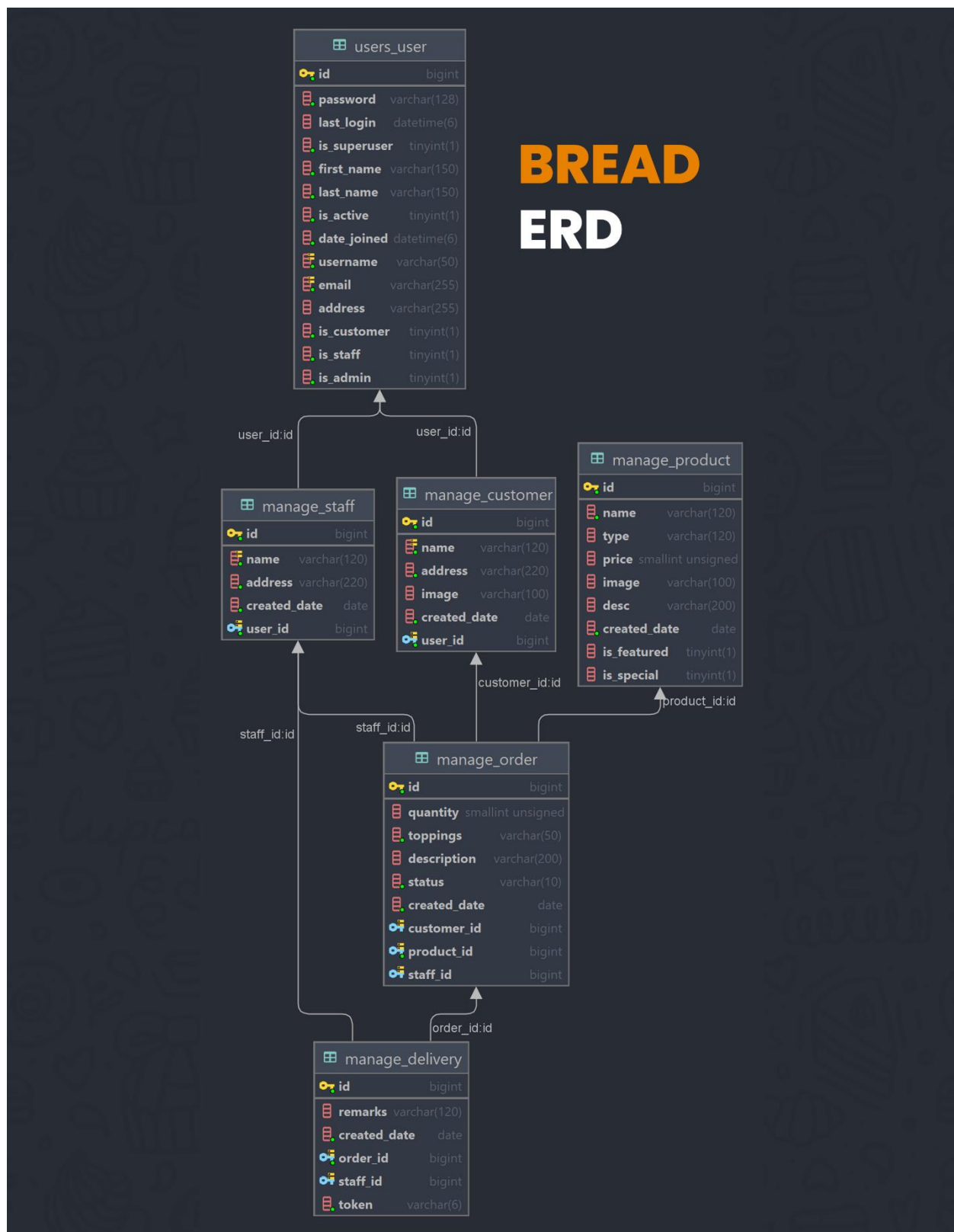
MySQL doesn't uphold an extremely enormous database size proficiently, doesn't deal with exchanges effectively and is inclined to information defilement. It does not have a decent troubleshooting mechanism so troubleshooting would be time-consuming and costly.

HOW TO INTEGRATE MYSQL IN DJANGO

To include MySQL as the back-end database for a Django project, we want to follow a straightforward rule:

1. Install the MySQL Server and MySQL Python driver which is utilized by Django in order to talk with the database.
2. Make the MySQL database and the client
3. Update settings Django
4. Execute the Django movement and make the venture tables

ERD OF BREAD



TESTING

Testing is a way of executing a program with a bent on finding flaws. Testing is done to add value to a program and adding value by testing it is improving the quality or reliability of the program ([Myers et al., 2011](#)).



Testing is one of the core practices in the web development that is done to minimize the fault that may occur in the release of the system. It is proven that testing software before its release reduces the cost and improves user experiences.

UNIT TESTING

Unit test is one of many testing in white-box-test that is done to test the littlest piece of code that can be logically disconnected in a framework. Unit test practices a "unit" of code in confinement and contrasts real and intended outcomes ([Olan, 2003](#)).



I have done unit testing of URL, Views, Model and Forms in Django. In URL testing, I have tested if the URL is resolved or not by first retrieving URL details from the "url's.py" file. I have used the reverse function, provided by Django to retrieve the URL. Then, I have used the resolved function with the constraints being the reversed URL and passed a function that is, expected to run when the URL is triggered.

```

1  from django.test import SimpleTestCase
2  from django.urls import reverse, resolve
3
4  from manage.views import *
5
6  class TestUrls(SimpleTestCase):
7
8      # Reverse : retrieve url details from url's.py
9      # Resolve : resolving URL paths to the corresponding view functions
10     # Assert : func/class == func/class
11
12     # MOD
13     def test_mod_order_url_is_resolved(self):
14         url = reverse('modify-order')
15         self.assertEqual(resolve(url).func.view_class, ModifyOrder)
16
17     def test_mod_product_url_is_resolved(self):
18         url = reverse('modify-product')
19         self.assertEqual(resolve(url).func.view_class, ModifyProduct)
20
21     def test_edit_order_url_is_resolved(self):
22         url = reverse('edit-order', kwargs={'id': 0})
23         self.assertEqual(resolve(url).func, edit_order)
24
25     def test_edit_product_url_is_resolved(self):
26         url = reverse('edit-product', kwargs={'id': 0})
27         self.assertEqual(resolve(url).func, edit_product)
28
29     def test_delete_order_url_is_resolved(self):
30         url = reverse('delete-order', kwargs={'id': 0})
31         self.assertEqual(resolve(url).func, delete_order)
32
33     def test_delete_product_url_is_resolved(self):
34         url = reverse('delete-product', kwargs={'id': 0})
35         self.assertEqual(resolve(url).func, delete_product)
36
37     # MISC
38     def test_order_summary_url_is_resolved(self):
39         url = reverse('order-summary')
40         self.assertEqual(resolve(url).func, order_summary)
41

```

In the Views test, I have checked if the function inside the Views.py file is working as expected or not and, in the Models test, I have checked if the method inside the Model class is returning value as expected or not and at last in the Forms test, I have checked the form validity.

```

1 class TestViews(TestCase):
2
3     order = None customer = None product = None staff = None
4
5     @classmethod
6     def setUpTestData(cls):
7         cls.client = Client() cls.staff = create_staff() cls.customer = create_customer()
8         cls.product = create_product() cls.order = Order.objects.create(staff=cls.staff,product=cls.product,quantity=1,toppings='test',
9         description='test',customer=cls.customer,)
10
11         cls.staff_list_url = reverse('staff-list') cls.customer_list_url = reverse('customer-list') cls.product_list_url = reverse(
12         'product-list') cls.order_list_url = reverse('order-list') cls.delivery_list_url = reverse('delivery-list')
13
14         cls.create_staff_url = reverse('create-staff') cls.create_customer_url = reverse('create-customer') cls.create_product_url =
15         reverse('create-product') cls.create_order_url = reverse('create-order') cls.create_delivery_url = reverse('create-delivery')
16
17         cls.edit_order_url = reverse('edit-order', kwargs={'id': cls.order.id})
18         cls.delete_order_url = reverse('delete-order', kwargs={'id': cls.order.id})
19
20     def setUp(self):
21         self.client.login(username="test_staff", password="123456")
22
23     def test_create_staff(self):
24         response = self.client.post(self.create_staff_url, {'name': 'test','address': 'test','email': 'test','username': 'test',
25         'password': 'test','retype_password': 'test'})
26         self.assertEqual(response.status_code, 302)
27         self.assertRedirects(response, '/manage/staff-list/')
28         self.assertTrue(Staff.objects.filter(name__exact="test") is not None)
29
30     def test_create_customer(self):
31         response = self.client.post(self.create_customer_url, {'name': 'test','address': 'test','email': 'test','username': 'test',
32         'password': 'test','retype_password': 'test'})
33         self.assertEqual(response.status_code, 302)
34         self.assertRedirects(response, '/manage/customer-list/')
35         self.assertTrue(Customer.objects.filter(name__exact="test") is not None)

```

```

1 class TestModels(TestCase):
2     order = None customer = None product = None staff = None
3
4     @classmethod
5     def setUpTestData(cls):
6         cls.staff = create_staff() cls.customer = create_customer() cls.product =
7         create_product() cls.order = Order.objects.create(staff=cls.staff,product=cls.
8         product,quantity=1,toppings='test',description='test',customer=cls.customer,)
9         cls.delivery = Delivery.objects.create(staff=cls.staff,order=cls.order,
10         remarks="delivered")
11
12     def test_order_name_assigned_on_creation(self):
13         self.assertEqual(
14             self.order.__str__(),
15             self.product.name + " by " + self.customer.name
16             + " in " + self.order.created_date.__str__()
17         )
18
19     def test_token_generation_upon_delivery(self):
20         self.assertEqual(
21             self.delivery.token,
22             Delivery.objects.get(id=self.delivery.id).token
23         )

```

```
1 class TestForms(SimpleTestCase):
2
3     def test_staff_form_valid_invalid_data(self):
4         valid_form = StaffForm(data={
5             'name': 'test',
6             'address': 'test',
7             'email': 'test',
8             'username': 'test',
9             'password': 'test',
10            'retype_password': 'test'
11        })
12        invalid_form = StaffForm(data={
13            'name': 'test',
14            'address': 'test',
15        })
16        self.assertTrue(valid_form.is_valid())
17        self.assertFalse(invalid_form.is_valid())
18        self.assertEqual(len(invalid_form.errors), 4)
19
20    def test_customer_form_valid_invalid_data(self):
21        valid_form = CustomerForm(data={
22            'name': 'test',
23            'address': 'test',
24            'email': 'test',
25            'username': 'test',
26            'password': 'test',
27            'retype_password': 'test'
28        })
29        invalid_form = CustomerForm(data={
30            'name': 'test',
31            'address': 12,
32        })
33        self.assertTrue(valid_form.is_valid())
34        self.assertFalse(invalid_form.is_valid())
35        self.assertEqual(len(invalid_form.errors), 4)
```

PROJECT ISSUE

ISSUES DURING THE PROJECT DEVELOPMENT

In the development phase of bread, I have encountered many different issues that I have addressed through comprehensive research and development. Most of the issue was coming from the front end and some from the backend. The problems have taught me that:

1. The Bootstrap class should be in order, and the end class element can override the initial design of the bootstrap.
2. In the Django model or ORM, On delete perimeter should have a cascade if an object is dependent.

LIMITATIONS OF BREAD

As of now the things that limits BREAD are:

1. The Database is not in 3rd normal form
2. Some content on the page is missing alt text.
3. The animation in special list of product becomes out of sync if more than 3 is added.

FUTURE WORKS

I am planning to add more features to BREAD in the future. I will be adding more error handling, remove some of its limitation. I am also planning to learn more about networking and hosting.

CONCLUSION

Bread is the butter for your business. It has been fruitfully designed to ease the management process and hopefully stop the spread of covid in some way. People, mainly the one with small to medium businesses, will definitely add some value to their businesses by using this small but useful piece of web application.

Bread has solved most of the front-end development challenges, if not all so, the audience for bread is vast. The backend is properly optimized for production and needs little to no maintenance.

LINKS

You can have a look at the apps demo and the test run on [YOUTUBE](#) and can find the source-code of Bread on [GITHUB](#).

REFERENCES

Birdi, K. (2021). Insights on impact from the development, delivery, and evaluation of the CLEAR IDEAS innovation training model. *European Journal of Work and Organizational Psychology*, 30(3), 400-414.

Liberatore, M. J., & Pollack-Johnson, B. (2003). Factors influencing the usage and selection of project management software. *IEEE Transactions on Engineering Management*, 50(2), 164-174.

Myers, G. J., Sandler, C., & Badgett, T. (2011). *The art of software testing*. John Wiley & Sons.

Olan, M. (2003). Unit testing: test early, test often. *Journal of Computing Sciences in Colleges*, 19(2), 319-328.

APPENDIX

BACKUP LINKS

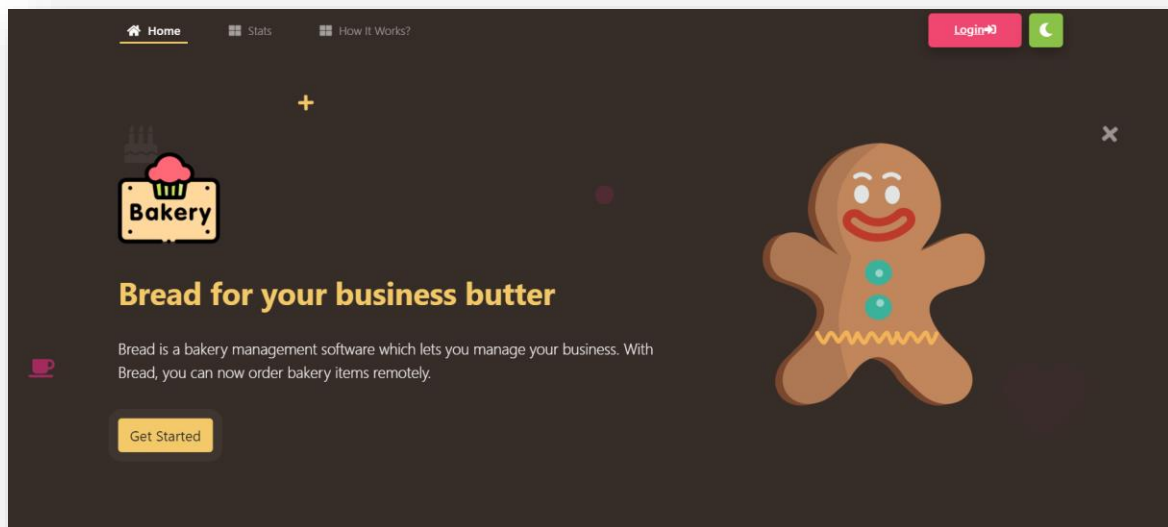
GITHUB

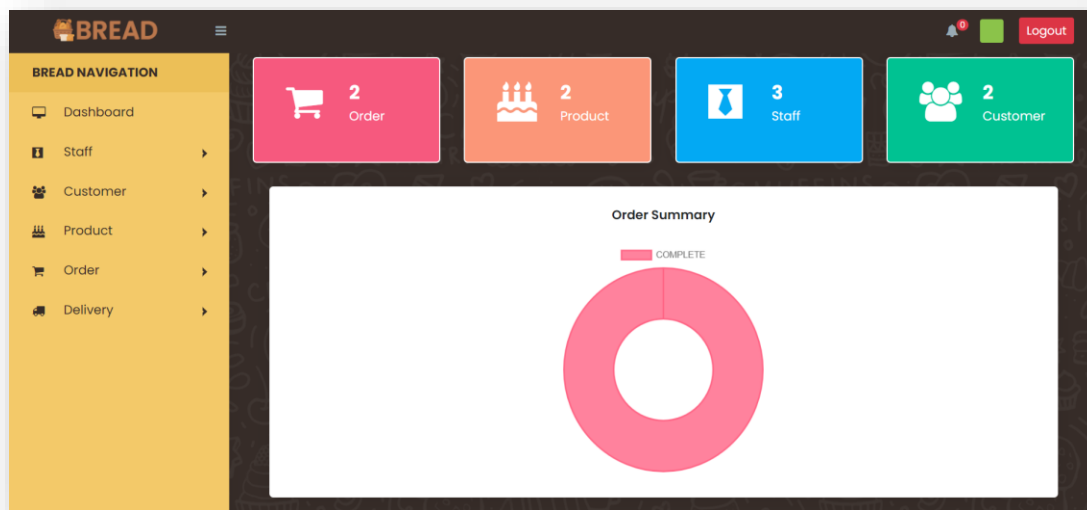
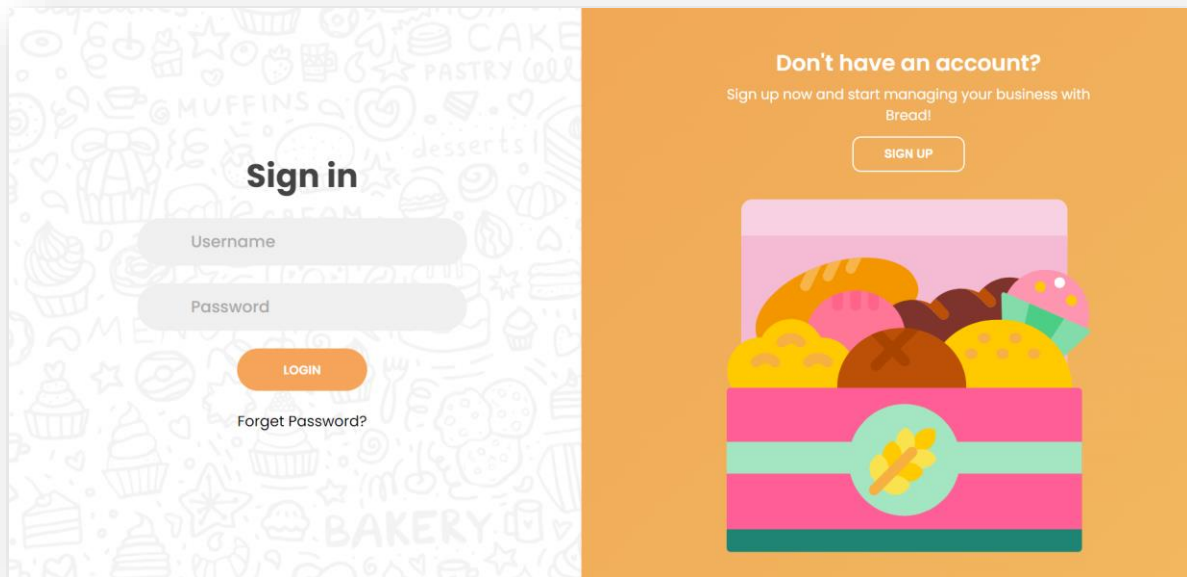
<https://github.com/Anurag-Bharati/Bread.git>

YOUTUBE

<https://youtu.be/fYGQShBLqgY>

SCREENSHOTS OF BREAD





BREAD

Logout

BREAD NAVIGATION

Dashboard

Staff

Customer

Product

Order

Delivery

Manage / Staff / List

Supplier List

#	NAME	ADDRESS	EMAIL	DATE
1	Sss	Sss	Sss	Feb. 14, 2022
2	Aaa	Aaa	Aaa@Aaa.Com	Feb. 15, 2022

1

2

Next

Last

BREAD

Logout

BREAD NAVIGATION

Dashboard

Staff

Customer

Product

Order

Delivery

Manage / Order / Modify

Edit Order

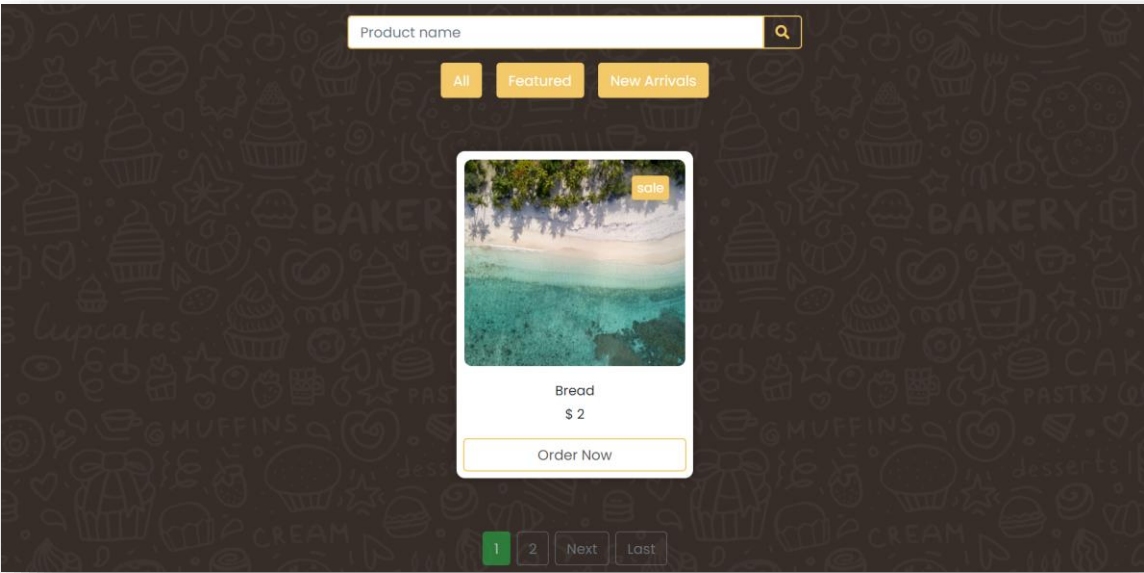
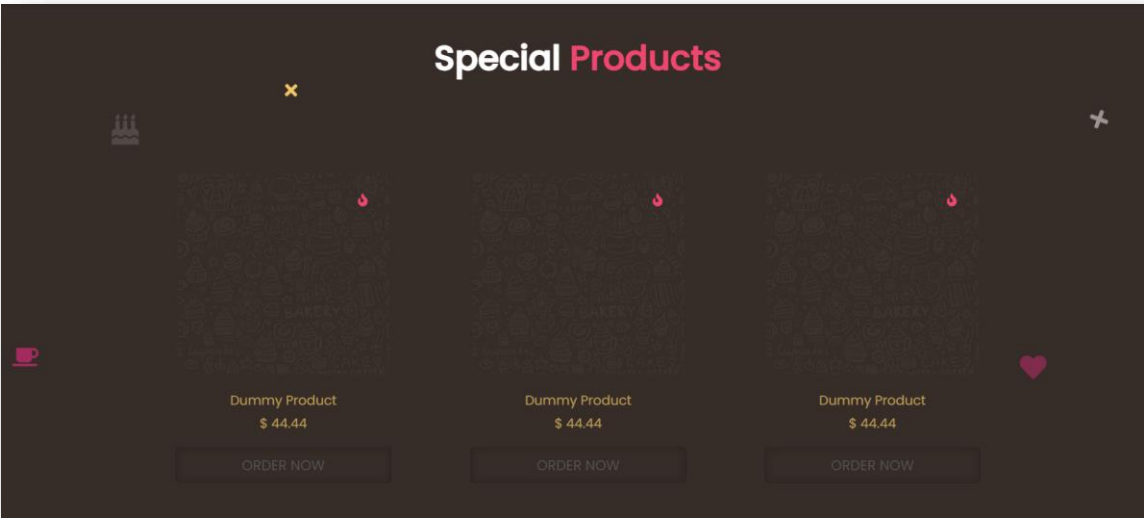
#	STAFF	PRODUCT	QUANTITY	TOPPINGS	DESCRIPTION	CUSTOMER	STATUS	EDIT	DELETE
1	Sss	Bread	1	Honey And Ginger	Glass Of Warm Water	Ccc	Complete	Edit	Delete

1

2


Next

Last



My Profile

Update Profile



Username

CCC

User ID

3

Email

CCC

Total Orders

2

← Back to shop


Active Orders

#	Staff	Product	Quantity	Date	Status
No Order Data					

Order History

#	Staff	Product	Quantity	Date	Status
1	sss	Bread	1	Feb. 15, 2022	complete
2	Waiting	Bread	3	Feb. 15, 2022	complete

Bread \$2



whole grain
multi-grain brown bread

Quantity

Toppings

Description

Create Order

[← Back to shop](#)

Update Profile

Name

ccc

Address

ccc

Image

Choose File

No file chosen

Update My Profile

About Us

Company

Quick Links

Social Pages

Lorem ipsum dolor sit amet,
consectetur adipiscing elit. sed

Privacy Policy

Home

Facebook