Cover page

Arden University

Restaurant management system

8978

Table of contents

Table of Contents

Chapter 1
Introduction

1.1

Chapter introduction

For my dissertation project to finalize my studies at Arden University for my Computer Science undergraduate program, I decided to build a web application using Django framework with the help of other technologies like HTML, CSS and JavaScript. The application I want to build will represent a booking and management system for restaurants. I decided to build this project as I have a special interest in web development, and this project will test and potentially improve my knowledge, skills and abilities to research and solve real world problems.

A restaurant management system is something that has been around for quite a while now, it is not something new, but something very important in the modern world with self-service options almost in any shop or restaurant. It helps to have this option as some people find it more attractive than having to interact with staff, or it purely saves time to others, by not having to wait in the queue to order something, allowing them to spend time with friends and family instead. Also, the modern average person is used to doing most of things on their smartphones, so giving them this option could leave more satisfied customers for a restaurant. It offers the convenience of browsing the menu and placing orders from their phone. The other huge benefit that such system offers is the possibility to view and manage all incoming orders efficiently for the kitchen staff, other feature is the ability to see the order status of the order, from the moment it is received, is it ready or still in preparation.

The reason I decided to choose Django as the development framework for this project is because Django framework takes care of many repetitive steps and procedures of a development routine, allowing developers to focus on the important parts of the application. According to the information found on the Django official website (Djangoproject.com. 2025), Django is “Ridiculous fast”, allowing to get a project from an idea to a prototype in a very short amount of time. Another huge benefit of using Django is that it has a dozen of included functions, focused on doing specific tasks to speed the production, from authenticating a user, to content administration features, allowing to build complex projects like social media platforms or e-commerce sites. It also takes security seriously, offering built-in solutions that protects against SQL-injections, cross-site scripting and cross-site request forgery.

Also, Django projects can be scaled easily, and they enable heavy traffic handling too.

1.2

Choice justification

From my own past experiences with some specific restaurants and bars, some of them are using the old fashion style of managing customers and orders, by taking orders at the counter or at the table relying on a person, which of course makes the experience more personal and individual with human interaction, but on a weekend or evening, it can get so busy that one would have to wait a very long time to be served. Other restaurants still use paper menus, which I find overwhelming with amount of text and images on a small piece of paper, that it gets hard to find anything. There are also cases with restaurants that does not use a proper organized system that sometimes they forget about someone’s order, leading to frustration and bad customer experience. This type of issues come along with the personal human interaction between customer and waiter which is great for some people, but these restaurant workers are affected my multiple external factors, and they are being put under a lot of pressure from their managers and rude customers which is something that occurs fairly often in such environments, and when this happens, a human can get a bit bitter, and this may have a bad impact when they try to interact with other, more friendly and positive customers. The automated, online reservation and ordering system will exclude these unpleasant situations, but at the same time it will also exclude the more positive things that come with human interaction too unfortunately.

1.3

Research questions

To make the research process more organized, and structured, so I can measure my progress and check what has been achieved or not in the end, I will set some questions that align with the objectives of this project.

“Can a Django application solve some existing issues in some old-fashioned restaurants, and how?”

“What features it should absolutely have in order to make the restaurant more efficient, and more convenient for their customers?”

“Can such project be designed, and developed single handed by one person, or it requires a team?”

“What challenges may be encountered during the planning and development stages?”

1.4

Background and overall intention

As I mentioned above in paragraph 1.2, some restaurants often have queues at the counter, with people waiting to just order one product, and they have to wait very long time, so a web application that would allow them to order something on their phone could come really handy. Also often on a weekend, when for example a couple wants to go out and have a dinner at their favourite restaurant, but when they get there are not free tables, this can be very annoying. Yes they could try and call, wait long time before someone will pick up the phone, but it doesn’t always work and it’s just waste of time, while a web application could be easily accessed on their phone, it would show them if there is any table available at the specific time, and then they could work with it, if nothing is available at the time, they could move dinner for another time or go somewhere else, it would just save a lot of precious time. It will also exclude the human factor, as people can easily get distracted, or their judgment may be affected by lots of exterior factors and they can easily make mistakes.

1.5

Aims and objectives

Aim:

- To design a functional system that will help restaurant to run their operations more efficient, by making table reservations and food ordering more convenient and easier.
- To attempt to implement this system in a form of a Django-based web-application.

Objectives:

- To gather and analyse requirements needed to optimize and improve how some restaurants operate.
- To develop a functional, simplistic, user-friendly interface, to be as straight-forward as possible
- To test and evaluate the application against the requirements gathered

1.6 Deliverables

A Django application that will allow to make table reservations, place orders online, show order history, add items to favourites list, and leave reviews. Also, it will have a dashboard for the kitchen staff to see new orders and manage reservations and stock availability.

Usability and performance reports based on testing results and users testing.

1.7

Approach

For this project I decided to go with Agile methodology, as it focuses on flexibility, cyclic development. It breaks big projects in smaller pieces, tasks, easier to manage and change in each iteration if required, allowing developers to deliver prototypes more often, test it, get feedback from users and them improve based on it. The main focuses of Agile are iterative development, adaptability, frequent delivery and customer involvement.

Figure 1: Agile methodology.

As per tools and technologies, I will be using PyCharm as my development IDE (Integrated Development Environment). I will install Python on my machine for this, and Django framework that will allow me to keep things organized and use some advanced features that Django offers, like users' accounts, cookies, databases, and seamless communication of front-end of application with back-end for interactive and personalised experience of each user.

Git and GitHub will help with version control of my application, allowing me to save progress before attempting to implement new features without risk of ruining the working version of the application.

Also, I will be using HTML with CSS obviously, these are the basic technologies to any website or web-application, but also, I will be using JavaScript for interactive changing of layout of the web pages.

Django will allow me to have an admin dashboard as a bonus, for the restaurant Manager, where he can see orders, the menu and update easily when needed, besides the Dashboard that will be created from scratch with only features required and more user-friendly interface.

1.8

Brief structure of Thesis

- **Chapter 1**: Introduction
- **Chapter 2**: Literature
- **Chapter 3**: Research Methodology

- **Chapter 4**: System Analysis and Design
- **Chapter 5**: Implementation and Testing
- **Chapter 6**: Critical Evaluation, Conclusion, and Future

1.9 Summary

In this chapter, I prepared the foundation for the report document, I stated the background, the reasons behind my decision to go with this project, my objectives and scope. In the next chapter I will be focusing on analysing the similar systems that already exist on the market, researching the potential impact of such application, and will discuss such projects from legal, social, and ethical points of view.

Chapter 2
Literature review

2.1 Introduction

The purpose of the literature review in a project involving the development of a Django-based system plays a critical role in guiding the project's foundation, focus, and progression. It helps to explore and understand existing landscape, technologies, frameworks, and methodologies related to the project. Also, by reviewing previous work, it becomes possible to pinpoint areas that haven't been adequately addressed and find opportunities for improvement. Another benefit of reviewing literature is identifying requirements and features for the project.

2.2 Overview of generic and specific areas

Technology has revolutionized restaurant management by streamlining operations, improving customer experiences, and enhancing overall efficiency. The areas that a restaurant can use a

well-structured system are managing new orders, billing, managing inventory. Also, it can help with keeping track of order history, everyone's favourite items and sales quantities of each product. It can make the ordering process much easier and convenient with the use of a mobile application.

Another benefit of such system is that with personal data collected in the app, it can be leveraged to see patterns, habits and allow to suggest appropriate products and services for each user, create attractive offers and promotions.

2.3

Existing research on chosen topic

To be able to achieve any results with any project, it is desired to consider competition on the market, by doing some form of research, observe, analyse, highlight the goods, the bads, and draw conclusions based on that. This will give a wide understanding of which features are really important and needed. Understanding the competitors' strengths and weaknesses can give an advantage in the marketing stage.

After doing some research on the internet, I found a few very interesting case studies about some restaurant management systems that are already on the market, and they came up with various solutions to address the challenges that some restaurants might need help with.

One brand that offers these services and has helped some restaurants to achieve great results is UpMenu. One of their recent clients they helped is Michelangelo 301, a pizza restaurant located in Florida, US, which according to a case study ran by Bartoszek (2024), they helped their client to generate over $800 thousands of revenue from online orders in under a year and saved over them over $100 thousands on third-party commission. Not only that, but it also helped the client to gain brand recognition, improved their operational efficiency and overall management control. This was all achieved by building a branded mobile app that helped with online ordering process, a feedback feature that was used to collect reviews that was then used to improve the service quality.

Another example of how the use of technologies boosted sales and revenue is Starbucks. How exactly they did this, and what were the outcomes is discussed in a case study ran by Chaturvedi (2024), where he found that Starbucks had to take advantage of new technologies to stay ahead of competitors. To achieve this, they started to invest heavily in research and partnerships, and came up with a mobile app that offered online ordering and payment features, a reward program, and exclusive offers. As a result, millions of new customers started to use the app actively, customers loyalty increased and more people would come back again. Also due to mobile ordering system and targeted promotions, revenue increased significantly. All these changes lead to the company strengthening their position on the market and gave them an edge against competitors.

2.4

Identify the research gap

While modern technologies and systems transformed restaurant management at the core, there will always be things to be improved that could be addressed. These gaps can provide opportunities for improvement and innovation. For example, many existing systems treat table reservations and ordering food to the table as two separate actions, not related between, which lead to inconsistent user experience, redundant actions from customers that could potentially be addressed.

Another common issue in such environment is poor communication between front-of-house and back-of-house which could lead to delays with order preparation or orders to be lost, and that will leave customers unhappy. A well-structured all-around system could reduce the frequency of such errors by a big margin.

2.5

Impact of potential solution

The proposed solution, to design and develop a management system could improve on all these points, making the customer experience more pleasant. Allowing customers to reserve tables and giving this peace of mind that they will get a table is something that the restaurant managers should keep in mind and consider.

Also, the orders management part will also improve the back-a-house experience for staff, not only making stock management easier, also reducing the probability of any missed orders and the potential customers frustration, as in a business like this, customer satisfaction is at the front.

2.6

Social, Ethical, Security concerns

The World Wide Web (WWW) has been expanding very quick in last few decades, “With this expansion comes a varied spectrum of ethical concerns and conflicts within the web development sector, though the most important and prudent ethical considerations include

user privacy, business security, and web accessibility." (Williams. 2023). Due to this, while developing this project, ethical concerns needed to be addressed too.

When storing personal information of any sorts, its necessary to ensure that that data is securely locked away from potential fraudulent actions and data leaks. To address these potential issues in time, I decided to collect and store the minimal amount of personal information required for my application to work as designed. The username is one thing that is needed as it will be used to log in to personal account, and it is not required to represent user's real name and can be a nickname. The e-mail is necessary to be able to contact the user directly in case of an emergency or anything else that could be needed in future. And the password is just to be able to identify and authenticate the users individually. The password is also encrypted when saved in the database, using a specialized encryption algorithm, to prevent anyone from accessing the user's account.

The security is also one of the important parts to be considered. That's why I decided to implement a registration and log in system, with e-mail address and a password. The password also is required to meet certain criteria, and be complex enough before creating the account, and it will be checked by 2 step validation system to make sure it is not easy to break. The potential threat of an SQL injection is also addressed using Django's inbuilt functionality that prevents such things from happening.

Transparency will also be addressed, starting with the registration page, user will have to tick a box that he agrees with the personal data use policy, with a link to the page where all the information will be stated. The link to the page will also be present in the footer of the page so the user can access it any time. Another way of improving transparency is the order status. When the user places a new order, the order status will be 'Preparing', after, when the order will be ready it will be updated to 'Ready', and the user can see the state of the order at any time in the order history tab.

The feedback system is not filtered or manipulated in any way, so all the feedback will be displayed on the page, good or bad. This encourages the freedom of speech for anyone. And the Overall rating section is affected by the feedback all together. Also, the feedback of each individual will not reveal any personal information of the specific user, except for the username, which does not provide any sensitive information about the individual.

Each user will be informed about the data retention and deletion policy on the Terms and Conditions page. It will state that any personal data stored, can be deleted on the user request, when contacted via e-mail with the appropriate message.

A feature that will be added with time is the recommendation system using AI and machine learning to suggest products based on each individual order history and other algorithms. When this will be implemented, user will be informed about the use of his personal data, how it is used, and will be asked to consent or not with this use of personal data by this system.

As the restaurant may sell alcohol, and this action is illegal to people under 18 as per Gov.uk (2025). Another feature that will have to be implemented is adding age restriction to alcohol products. When new users will be registering, they will be asked to state the date of

birth, and when they will try to add to cart age restricted products on the app, it should now allow it.

2.7

Chapter summary

In this chapter I had a closer look in to the existing literature materials regarding similar systems, discussed the use of technology in the restaurants industry, how this can improve certain aspects and add advantages against competitors. Also considered some case studies, how the implementation of technologies and online ordering system affected real businesses, what features can add value and improve customer experience. Another point that had to be addressed is the Ethical and Social concerns of such systems. After doing some research I pointed out some important details that have to be kept in mind when developing a system like this.

In the next chapter we will look at the research methodology, development methodology, gather the requirements for this project, come up with a development plan, set milestones and set a timeline for the entire development process.

Chapter 3
Research methodology.

3.1

Chapter introduction

In this chapter I will describe the methodology I have chosen to use as guidance during the development of this project. Also, will speak about the requirements gathering process, will justify the choice of this project, will outline the major tasks of development process, will provide a clear timeline as a GANNT chart to use as guidance and measure the progress along the line.

3.2

Project type justification

I decided to go with a development focused dissertations as I am more of a practical persona than theoretical. I like building, creating new things, solving real world problems, and see results, so this was an obvious and very easy decision to make. There is no specific reason why I decided to build a restaurant management system, it just seemed a reasonably challenging type of project, something that could put my knowledge to test, but at the same time something that could be achieved with time and persistence. Also, I have a passion and special interest in web-development, so this also works well for me, as I plan to focus on this direction in future career.

3.3

Methodology

Requirement Gathering

When it comes to requirements gathering stage, the best way to understand what features would be useful for a restaurant is by contacting them directly, arranging some small interview, have a chat with the owner, or manager as they are the most appropriate people to ask, with years of experience behind their backs. Another way is running a survey campaign, asking restaurant staff, and random people that attend restaurants to ask their opinions, and find what is the most mentioned and desired thing that could improve the experience for both sides and make it a more pleasant overall experience.

Development Methodology

The development methodology I have chosen is Agile, as mentioned before in chapter 1.7, it enables a very flexible development process, due to which I will be able to go through multiple iterations, delivering new features one at the time, test them and do so again until I achieve the desired result.

3.4

Project planning review

Tasks

Requirements gathering

- Gather feedback from users of existing similar systems, find case studies

Requirements analysis

- Analyse collected information, decide what features are useful, what works well, what can be improved

System design

- Designing the database

Front-end design

- Creating the front page design
- Creating dashboard design
- Creating log in and register pages design

Development

- Build the Django project
- Create all necessary files
- Create Django models for database with necessary fields
- Build front page according to design
- Build log in and register page according to design
- Build dashboard according to design
- Write URL routes for all the pages and connect them between each other
- Write scripts for creating new users, and logging in
- Populate products database with product that restaurant will offer
- Retrieve these products in back end, and display them on main page as a menu
- Create Cart, and all cart related functionality, like add item to cart, and check out
- Create order history page, where will display all past orders for the specific user
- Create reviews section, with overall rating, feedback form, and all the existing reviews
- Create footer with personal restaurant information: opening days and hours, address, and contact information.

- Creating the dashboard page
- Write script to ensure that only superuser can access this page
- Populate the dashboard with all the information retrieved from database, by using scripts
- Add the stock section, to display all existing product, with buttons to update stock availability, and delete products if needed.
- Add a form on dashboard that allows to create new product

Testing

- Testing each function, making sure everything works as it should according to requirements specifications

Milestones

- Requirements gathering
- Requirements analysis
- Designing the system
- Development
- Testing
- Collect feedback, fix bugs, improve the system

GANNT chart

Resource list

- A laptop or PC
- PyCharm or other IDE
- Django framework
- Database

Risks and limitation

Potential risks and weak points for the system are mainly with the user's personal data and security. This is always a big concern even with big tech companies, it takes a lot of cybersecurity specialists to address the potential threat of a data breach, and even in big tech companies data leaks happen time to time which leads to big scandals and lawsuits against the company, but most importantly, personal and sensitive information gets leaked and could end up in the hands of people with bad intentions.

Another potential risk is related to online payments that will be implemented with time, and it's a needed function for a system like this. If transactions are not encrypted and processed safely, people may lose money, as well as the restaurant.

The main limitation I see is my lack of experience and expertise in such domain. I realize that I don't have enough knowledge to make this a robust system that will offer sophisticated features and cover all potential weak points.

3.5

Chapter summary

In this chapter I discussed more about the methodology I will be using to develop this project, also I came up with a development plan by breaking down the entire development process into smaller steps to be able to track the progress and better understand the current task. I then set some milestones, which were then used to create a GANNT chart for the entire project, that is split in weeks, so then it is easier to decide how long it should take to achieve each milestone. Also, I wrote a list of resources that I will need during the development and had a look at risks and limitations I could encounter.

In next chapter I will analyse the requirements gathered and write a list of functional and non-functional requirements for my project and draw some diagrams to visualise the system and how it interacts with users and with other components inside the application.

Chapter 4
Analysis and Design

Chapter Introduction

In this chapter I will determine the functional and non-functional requirements, describe all the feature that the web-application and the system should have, and create a design for the entire application in details. The analysing and designing parts are vital for the success of any project, as at this stage all the major decisions are made, and it is necessary to be as specific as possible so it can be measured later at the reflection stage how close I got to the original plan, where I succeeded and where I could have done better. Also, in this chapter I will design and draw UML diagrams, Use Case diagrams, and other type of diagrams to visualise the connection between each element of the system and how they interact with each other, also to illustrate the workflow and data flow.

4.1

Functional and non-functional requirements

Functional

- The user will be able to register or log in if already have been registered.
- On the registration stage, the password will be checked for validation
- Some links on the navigation bar will only appear if user is logged in, but home and log in buttons will always be there.

They will not be accessible even by writing the address directly in the URL field, those pages are closed from users that are not logged in.

- At the top of the page, will have a list of all categories, retrieved automatically from database
- When user will click on any of the category, page will scroll down to that specific category in the menu so user can see all items in that category
- The user will be able to reserve a table if there is one available.

- The user will be able to browse the menu inside the app.
- The menu will be split in categories, and each product will display under its own category
- Each product will have name, description, price, a numerical quantity field with add to cart button , and ‘add to favourites’ button, that will change to ‘remove from favourites’ if the item is in the favourites list already
- The user will be able to order food or drinks to his table online via the app.
- The main page will have most popular foods and drinks featured, one for each category
- The user will have an orders history in his account of previous orders made to order again if needed.
- Home page will have a Reviews section
- Reviews section will display a container with overall rating
- A form so user can leave feedback (only if logged in)
- And a section with all feedback messages, with comment, date, user name, and rating
- Users will be able to add product to their favourite list, and add items from there to cart.
- There will be a cart button in the bottom right corner of the screen
- When it is clicked it will reveal a container that shows all the items in the cart
- If container is displayed and button clicked again, it will hide the cart container
- Inside cart container will show all items in the cart, or if it is empty, it will say so.
- If the cart is not empty, a button will appear to allow user to check out
- It will have a footer section at the bottom of the page

That will show information about opening hours, restaurant address, and how to contact restaurant via e-mail.

- App will have a dashboard page, accessible only by super users assigned by developer
- The dashboard will have next sections:
- New orders
- Will display order id
- User name

- Total order value
- And individual items in the order
- A button to confirm when order is ready
- Ready to collect orders
- Displays the order id
- A collected button to remove it from list
- Create new product
- Will have a name field
- Description field
- Price
- Category, with multiple choices pre-defined
- A button to add image
- And a submit button that will add the item to the menu list
- Manage stock
- Will list all items in the menu with name, description, picture, price.
- Also stock information, with a button to toggle the stock from ‘in stock’ to ‘out of stock’
- Items that are out of stock, on the main page in the menu will not allow users to add these item to the cart
- And a delete button that allows to delete a specific item if it is not sold anymore

Non - functional

- The top seller on main page will represent the drinks and dishes based on the greatest number of sales made, based on the information from the database.
- When user places an order he will receive a confirmation message if it was successful
- The message will be displayed at the top of the page.

- On the registration form, user will only need to provide Full name, e-mail address and a valid password that will need to meet the criteria of being longer than 8 symbols, have at least 1 capital letter, 1 number and 1 special symbol
- on the registration page:
- there will be a name field
- an e-mail field
- a password field
- and a register button
- all will be aligned and centered on the page
- a function will check database if the username is available and display message if username taken
- Information in the footer section, like opening hours, address and contact email, will be in one row, 3 columns for each section
- the featured products, top sellers, will be displayed in blocks, one for each category, In a row, with space between, spread evenly across entire width of the screen
- When page is rendered, a function will check if user is logged in, and only display certain elements on the page depending on that. Things like navigation links, or ‘add to cart’
- On log in page
- It will have the title
- A field for name
- One for password
- And a link lower to go to registration page if user does not have an account yet
- Form will be aligned and centered on the page

Security requirements

On the registration page, when I user will try to create a new account, the script will check if the username is not used already and inform the user about it. To avoid duplicates, even though they will have different ID numbers.

Also on the registration form, user will have to select a strong password. It will have to meet certain criteria, which is a good practice for protecting the account, or at least make it harder to guess. The criteria the password will have to meet are:

- Be at least 8 characters long
- Have at least one capital letter
- Have at least one number
- Have at least one special character

These criteria will be checked with a JavaScript function before the information is send to the server and inform user if all the criteria are not met.

If the password passes the JavaScript validation, the information will be sent to the server side, where this will be checked again, just another measure of precaution to ensure data integrity.

After the personal information passes all the validation steps, the account will be created, the personal information will be added to the database, and another level of protection is that the password will be hashed with a special Python function, just to add an extra layer of protection.

4.2

Use case diagrams

4.3

Activity diagrams

Registration / Log in

Figure 2: Registration/ Log in activity diagram

Create reservation

Figure 3: Reservation activity diagram

Order management

Figure 4: Placing order activity diagram

4.4

Database design

Figure 5: Database Entity-Relationship Diagram

4.5

Prototypes

Figure 6: Home page, features, menu

Figure 7: Feedback section

Figure 8: Dashboard prototype

Chapter summary

In this chapter I analysed the requirements gathered at the earlier stage, and decided what should be the functional and non-functional requirements for this project, this will help later to reflect on the overall project success. Also, in this chapter I drew the use case and activity diagrams along with database ERD, this will help to better understand the core of the project, how each component interacts with other elements of the application. At the end I provided some screenshot of the early-stage prototypes of the main elements of the application.

Chapter 5
Implementation and Testing

5.1
Chapter Introduction

After all the preparations have been finished, requirements gathered, and the system has been designed, it is now time for development and testing stage. In this chapter I will attempt to develop and implement all the features required and described earlier in Chapter 4 and run the tests to verify and ensure that all required functions are working as intended, if not make appropriate adjustments and changes to achieve the desired results.

5.2 Implementation

The development stage starts with the process of installing Django framework into our development environment, which in my case is PyCharm.

After it is successfully done, and no errors occurred in the terminal, I can proceed to creating the project by following the standard project creation procedure step-by-step as described on official Django website:

- Create Django project inside the IDE
- Create the application
- Register the application inside the settings.py file
- Create the files and templates
- Create urls.py inside the application directory
- Create a superuser
- Run all migrations
- Run the server to check if the initial setup was successful

Now with this all preparations are finished, to be able to open the project inside a browser, I need to run the local server by typing : “python manage.py runserver” in the terminal. This will launch a local server accessible on the IP address provided in the terminal as per official Django documentation (docs.djangoproject.com. 2025).

Next, I need to create all my models for database according to database design described in chapter 4.4, which I have done, and they can be viewed in models.py file inside the project directory.

Registration and Log in

With the registration and log in features, I created two simplistic forms, that will request only basic information from the user, necessary to implement some basic functionality.

This is how the forms look like:

Figure 9: Log in page

Figure 10: Registration form

With the registration form, I added some validation functions, both in front-end and back-end, to check for username availability, to check if passwords match in the confirmation field, also to ensure that the password is complex enough too, so it is not that easy to break into the account.

On the log in page, the script will check the database if the user exists and if password match when the form is submitted and returns the message in case something did not work right.

Placing orders

The order placing process is fairly simple, in the menu, for each product, there is a quantity and a “add to cart” button, that will save that specific product to the user’s cart Session data:

Figure 11: Menu interface

When an item is added to the cart, the cart content will update, and when the cart button in the right-bottom corner will be clicked, the cart container will appear at the bottom of the page and page will automatically scroll down to it. If the cart button is clicked again, cart will hide. Inside the cart container, besides all the items in it, there will be a button that will send the products to the database, and the cart will be cleared after if successful.

Figure 12: Cart

Dashboard interface

The dashboard for restaurant staff can be accessed by typing <address>/dashboard in the URL section of the browser, but it will only be allowed to access by users with the right clearance level, so called “superuser” that can be assigned by system developer only, in order to grant access to the dashboard only to qualified people.

On the dashboard page, will be sections for new orders, ready orders, reservations, create new product form, and a section with all products to allow to update stock availability, and delete products that are no longer served.

Figure 13: Orders list

Figure 14: Reservations and new product form

Figure 15: Stock management interface

Reservations

When I approached the stage of developing the reservation functionality, I failed with some aspects of it, and was not able to make it work as planed from the start. Although some basic features I managed to implement, others turned out to be more complex than I anticipated and would have taken me way more time to solve than I had on hand, so I decided to move on, take the failure and learn the lesson.

The parts that I manage to create is the “Make reservation” button and the form that allows to select a date, time, and amount of tables needed, and the “Reserve” button that is supposed to create the reservation and save it inside the database. The “Make reservation” button is fixed at the bottom of the page so when the page is being scrolled down or up, it always stays in place. The button itself and the form will only show to signed in users. Also the form will be hidden until the button will be clicked to keep the page neater.

Figure 16: Reservation form

Figure 17: Reservation Django model

I created the model, as in the figure 17, but something didn't want to work as I planned, I added the route for it, and the view function to handle the information sent with the form.

Figure 18: urls.py , all my paths

Figure 19: make_reservation view

Figure 20: end-point for API

In figure 20, I tried to create a function that will act as an API end-point to allow to get available timeslots for the reservation form.

5.3 Challenges on implementing the solution

While trying to implement the reservation functionality, I encountered multiple challenges, some I managed to overcome, others I was unable due to limited time and knowledge.

Even though I attempted to create a model for the reservation system, I was not able to make the reservation to be saved in the database for some reason, it would return status 200 (status 200 means success) in the terminal when form is processed, but the reservation will not be added. I was not able to spot the problem, so I moved on, as time was running out.

Also because of this issue, there is nothing in the reservation section on the dashboard interface.

Feedback section

I decided that having a feedback section is a nice feature to add, it allows to get feedback from users, to see the satisfaction level, and allow them share thought about the restaurant, and overall experience.

It has a overall rating score, that is and average value of all the feedback rating selected by users. Also, it has a form, that allows to select a rating value between 1 and 10 to represent how happy users are and also add a little message as well. Below is the feedback section, that displays all the feedback that has been sent already, it shows the user's name, the message they left, the date that feedback was sent, and the rating they selected. It displays the newer message at the top, to see the most recent ones first.

5.4 Testing

Testing is a critical component of software development, ensuring the reliability, stability, and performance of an application. In this chapter, I describe my approach to testing implemented in the development of this Django application. From verifying the correctness of individual functions to validating the behaviour of integrated components, testing helps maintain code quality and reduces the likelihood of bugs in production.

Django, being a robust web framework, offers a comprehensive suite of tools for testing, including a built-in testing framework based on Python's unit test module. These tools allow developers to write unit tests, functional tests, and integration tests to cover a wide range of scenarios. Although I will not apply these strategies in my project, it is important to be aware of them and make use of them in the future.

This chapter details the approach taken to design and implement test cases for key features of the application.

Functional testing

The best method for functional testing in a Django project would be unit-test. Django framework offers a comprehensive set of methods to implement in the app, that can automate, and check each function individually one by one, if it is programmed accordingly. But first it requires to write a python function for each function required checking. Inside the application, Django has a file called “tests.py” where all these functions need to be included. When this is all done, all that needs to be done is to run this file, at it will quickly launch all those functions inside the file that will check and return a result depending how successful it was.

This is a long and complex process, it is needed to automate the testing process on bigger scale projects, especially when multiple developers work on it. In this case it was just me from start to end, I was testing each function while I was developing it, so at this point in time, when this project is purely for research purposes and not intended for deployment, it is acceptable to keep it simple and not waste much time on writing the unit tests, but instead run a comprehensive test of each function manually which I have performed.

At the end of it, I got no error codes in the terminal or browser console.

The functions I have checked are:

- Registration form
    - Checking if the password and confirm password fields values match
    - Checking if the password is long enough and has at least one numeri symbol and one special symbol
    - Making sure the form does not submit if at least one field is empty
- Log in validation
- Elements that should only be visible to logged in users
    - Cart button and functionality
    - Make reservation button and form
    - Feedback form restriction
    - Navigation bar links are reduced and hidden
    - Access to dashboard page restricted only to superusers
- Leave feedback form
- Calculating and showing overall restaurant rating
- Retrieving and listing all feedback comments

- Retrieving and listing all menu products
- The top sellers function to list the best selling products for each category
- The function that displays “Out of stock” message instead of “add to cart” button
- The cart functionality:
    - Add to cart with quantity
    - Updating the cart container
    - Checking out the products
    - Clearing the cart after
    - Adding new entry in orders history with details
    - Each sale is associated to the right user that was logged in and placed the order
- Cart button opens the cart container when clicked, then closes when clicked again
- Clicking “Log out” button in navigation bar logs the current user out
- The responsive design layout adaptation
- Multiple browser compatibility
- On Order History page, lists all orders made in past of the user that is logged in
    - Clicking on an order, expands a container that shows all items inside that order
- Favourite functionality
    - Adding a product to the list
    - Removing the product form the list
    - On the Favourites page, displays all favourite products
- Reservations functionality
    - Clicking “Make reservation” button will reveal the form, clicking again will hide it
    - Selecting date, time and amount of tables needed and the clicking reserve sends the form and make reservation
- Dashboard

- In the “New orders” container, are displayed all new orders with details and a “Ready” button
- Inside “Ready to collect” container, shows all items that have been clicked “Ready” button in the “New orders” container
- “Create new product” form works
- Stock management section lists all products
- Clicking delete product button will first trigger and alert to confirm that the user really intends to do so and it is not just a miss click
- Clicking “Toggle stock” will change the value of that field for that specific product from “True” to “False” meaning if the product is in stock or not

After checking each of these functions, everything worked as it was meant to, no error messages in the terminal or console, meaning it was all successful, expect the reservation feature. After attempting to make a reservation, selecting the date, time and number of tables, and sending the form, it worked fine, returned no errors or any issues have appeared, but when trying to see all reservation inside the database, nothing appears to be there. This requires further investigation and debugging, but it will require an unknown amount of time, so this is a problem for another time.

Responsive design test

While developing the application, I tried to keep in mind the responsive design concepts to allow users to a comfortable use of the application on various screen sizes, so the interface feels just as usable on smaller screens as on bigger ones.

Using CSS functionality, the application will adjust the size of containers dynamically, change the margins, and hide the navigation bar so it can display by the click of a

button on the screen, in order to save space and make the overall look in line with modern standards.

Figure 21: Layout on bigger screen

Figure 22: Layout on smaller screens

As we can see the difference between Figure 21 and Figure 22, the layout adapted, elements changed in sizes, and are arranged in different order. And this effect is present across the entire page, making it more comfortable to use on a mobile device.

Different browser testing

Also, it is important to realise that there are multiple browsers on the market, some may use Google chrome, others Opera, others Safari. To ensure that the application looks and work the way it should on all platforms, the application needs to be tested in other browsers too. With this, I tested my application in Safari too, and the results were positive. All the main functions of the app have been tested multiple times, and I have spotted no issues at all. These proves that the application should run well on most Apple devices as well.

Figure 23: Testing in Safari

5.5 Chapter summary

In the end, after testing the functionality of all the main features, most of the things ended up working fine, no issues to report, the application works fine in different browsers like Google Chrome and Safari, the registration and logging in functions work as they should, the products can be added to basket and checked out successfully, the order history is saved, the items can be added to favourite list too.

Ideally, I should test the app in other browsers too, like Opera, Microsoft Edge, Mozilla. And test all the functions particularly, to spot any potential issues with data handling that could be vital.

Chapter 6
Critical evaluation, Conclusion and Future Recommendations

6.1
Chapter Introduction

In this chapter I will try to provide a summary of the project's outcomes, revisiting the goals and objectives set at the beginning of the development process. I will reflect on the effectiveness of the chosen methodologies and technologies, particularly the Django framework, in achieving the project's aims. Additionally, this chapter discusses the challenges encountered, the lessons learned, and potential areas for future improvement and expansion. By evaluating the project's journey, this section offers a holistic view of its accomplishments and the insights gained throughout the development cycle.

6.2
Critical Evaluation

The system design is not flawless, it took a lot of energy and time to bring it to this state, but it was not enough to make it a perfect working application that could be used straight away. It still needs more work on it, fixing some bugs, improving the UI and UX aspects, and finishing the table reservation feature. In most part, I believe I delivered a good number of features that were discussed in the beginning of the project, things like registration, log in, menu, cart and check out functionality, adding products to favourites, saving order history for each user, feedback system, and the dashboard for restaurant staff that will help manage stock and new orders better.

I also believe that with this research I have addressed and even answered in detail most of the questions set in chapter 1.3.

After reading and analysing the study cases in chapter 2.3, evaluating the focus points, strengths and weaknesses of those systems, I drew some conclusions and put together a list of requirements that a system like this must have. For example, the feedback system in those other systems proved to be very useful in gathering users opinions, and what they wish the system would do better, so then the restaurant could adjust, and become a better service and increase the customers satisfaction consistently.

6.3
Recommendations for future

After the completion of any project, there must always be the discussion about the future of the project, the improvements that could be done, what other functions that could be useful and general direction where it may evolve in the future. For this project specifically, I highlighted some features that this system should have in order to be competitive with other systems on the market.

One very important feature that it should have, is the search function, to allow users to easily find the product they need, without them having to scroll through the page to find it. Although it is not something very complicated to implement, my time was limited, and I had to focus on other more important things.

Another most needed feature for such system is the ability for users to pay for their orders online. This will improve user satisfaction and make the online ordering process even better and quicker.

Something that will make the using of the system more personalised, is the recommendation system, that will analyse each user's preferred and most purchased products, and based on that it will be able to suggest other products that the user might also enjoy.

One more thing that could also come in handy, and proof to be useful, is a mobile app, for android and Apple users. This will allow a better experience overall and allow to implement even more features in the future as we discovered in the case studies analysed in chapter 2.3.

Regarding the stock management system, at the moment it only allows to see if a product is in stock or not and change its status. Ideally, this system should also allow to see the exact quantity of the product, and update it when new stock comes in, or when something is sold. This will allow a more accurate stock management.

Something else that requires some attention is the user experience (UX) and user interface (UI). I realise that the front end of the application does not look so incredible or eye-catching, but again, this was not my goal, the focus of this project was to try and create a restaurant management system with a certain set of features, that will improve a restaurant operation and make it more modern. So, with time, a specialist in UI/UX should be hired to have a look at the application and improve these aspects.

6.4 Reflection

This has been a more challenging project than I have anticipated at the start, it took more time, and knowledge to gain during development and research. But this is a good experience, as I learned a lot of new things overcoming challenges as I would have had doing something less challenging.

Regarding reservation functionality, my original idea was that orders could be attached to tables, but not necessarily. This would have allowed the restaurant staff to know which order goes to which table, without relying on the waiter memory.
Also, when making a reservation it will reduce the number of tables available, and when it's the booking time, the app would know which table you have and when you order online, it will automatically assign the order to that table for reference, so restaurant staff would know where to take the food to.

I was not able to plan this part good enough so it would all work together, there are so many pieces in this system to consider making it work well together that I simple was unable to "glue" it all together well enough.

In the end, lessons have been learned, conclusions made. With this, next projects I would have a slightly different approach, will allow more time to development and testing, do regular reviews of the timeline and objectives to make sure the development goes according to plan, maybe increase the pace. Also, because of limited knowledge from my side, it will not be a bad idea to get some help with certain aspect of the project, like researching, testing, UI/UX and accessibility features.

References

Bartoszek D. (2024). "*Michelangelo Hit Nearly $1,000,000 From Online Orders",* accessed at: https://www.upmenu.com/case-studies/michelangelo-301/

Chaturvedi S. (2024). "*Case Studies: Successful Tech Implementations in Restaurants*", accessed at: https://www.restroworks.com/blog/case-studies-successful-tech-implementations-in-restaurants/

Django.com. (2025). Available at: https://docs.djangoproject.com/en

Gov.uk. (2025). Available at: https://www.gov.uk/alcohol-young-people-law

Williams, K.S. (2023) 'The Application of Kantian Ethical Principals in Global Website Design and Development', *Journal of Multidisciplinary Research (1947-2900)*, 15(1), pp. 59–64. Available at: https://research.ebsco.com/linkprocessor/plink?id=73412d7c-b440-37db-be03-cf1f73e5da41

Appendices

The link to my GitHub Repository where the project is stored and can be accessed:

https://github.com/ITemci/dissertation

Or Clone it using this link: https://github.com/ITemci/dissertation.git

To access Admin dashboard, log in with the superuser credentials:
Username: admin

Password: admin

Then use the route <server-URL>/dashboard