Czech University of Life Sciences Prague

Faculty of Economics and Management

Department of Informatics and System Engineering

Master's Thesis

University Parking System

Author: Aryana Haji

Supervisor: Ing. Martin Pelikán, Ph.D

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

DIPLOMA THESIS ASSIGNMENT

Aryana Haji, BSc

Informatics

Thesis title

University Parking System

Objectives of thesis

The main objective of this thesis is to face the problem of Parking in the University Campus by making a Website with mobile compatible for Parking System.
The mail goal of the website is to help students and employees to search and find free space at campus, And book it easily for specific hours. The benefit of achieving this website is to reduce the time taken and the hassle factor of locating an available parking space. Being able to accurately direct a driver to an available space has many environmental benefits like reduce CO2, noise and reduce traffic.

Methodology

The work consists of theoretical and practical part. The theoretical part of the work is generally based on the study of professional information sources and based on the knowledge gained will describe all the starting points for the following practical part, namely programming languages, technologies and libraries used for Web development.

The practical part will consist in the creation of a website focused on the creation and display of events. An analysis of the website and definition of basic functionalities that will be implemented in this diploma thesis will be performed. Based on this analysis, the layout of the website in terms of navigation will be performed, as well as the selection of the necessary technologies, libraries and a suitable database for the commissioning of this website.

The proposed extent of the thesis

60-80

Keywords

Web development, HTML, Django, Parking system

Recommended information sources

Gongjun Yan; Weiming Yang; Danda B. Rawat and Stephan Olariu, "SmartParking: A Secure and Intelligent Parking System", IEEE Intelligent Transportation Systems Magazine (Volume: 3, Issue: 1, Spring 2011).

H. Wang and W. He, "A Reservation-based Smart Parking System", Computer Communications Workshops (INFOCOM WKSHPS) 2011 IEEE Conference on Shanghai, pp. 690-695, 2011.

Mohandes, M., Deriche, M., Abuelma'atti, M. T., & Tasadduq, N. (2019). Preference-based smart parking system in a university campus. IET Intelligent Transport Systems, 13(2), 417–423. https://doi.org/10.1049/iet-its.2018.5207.

Expected date of thesis defence

2022/23 SS – FEM

The Diploma Thesis Supervisor

Ing. Martin Pelikán, Ph.D.

Supervising department

Department of Information Engineering

Electronic approval: 26. 11. 2022

Ing. Martin Pelikán, Ph.D.

Head of department

Electronic approval: 28. 11. 2022

doc. Ing. Tomáš Šubrt, Ph.D.

Dean

Prague on 30. 03. 2023

Declaration

I declare that I have worked on my master's thesis titled "University Parking System" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the master's thesis, I declare that the thesis does not break any copyrights.

In Prague on 31.03.2023 ______________________________

Acknowledgement

Foremost, I would like to send my sincere gratitude to my supervisor **Ing. Martin Pelikán, Ph.D**. Dean of the Faculty of Economics and Management, at Czech University of Life Sciences Prague for the continuous support of my thesis study and research. I express my deep sense of gratitude for his paintient, motivation, immense knowledge and guidance in completing my master's thesis.

Besides my advisor, I must express my very profound gratitude to all of my professors, my fellow students for their unwavering support and ongoing encouragement during my years of study as well as during the process of conducting research for and writing this thesis.

Lastly, yet most importantly, I would like to thank my father, Dr. Mohammed Amin Haji. and my mother, Mrs. Jangeen Abdo, my sibling and my friends for supporting me emotionally throughout my life and for giving me strength to chase my dreams.

University Parking System

Abstract

This project aims to create and build parking system for University campus in order to solve the problem of the hassle factor of locating an available parking space and reduce the time taking while finding an available zone to book. Build a user friendly web application with efficient functionality, easy and understandable to use and interact with. Improving the proposed system based on achieving acceptable and scalable web application interface by utilizing different programming languages together. Provide the ability to have and own parking account with flexibility to check vacant slots and book it easily recording to users need.

This paper deal with proposed parking system starting with briefly understanding the need of implementing it, what get used for achieving the working system and how the system interact with users. Explain in details the development process and discuss how it could be improved in future.

Keywords: Web development, HTML, Django, Parking system

Univerzitní parkovací systém

Abstraktní

Tento projekt si klade za cíl vytvořit a vybudovat parkovací systém pro univerzitní kampus, aby se vyřešil problém s problémem hledání volného parkovacího místa a zkrátila se doba hledání volné zóny pro rezervaci. Vytvořte uživatelsky přívětivou webovou aplikaci s efektivní funkčností, snadnou a srozumitelnou pro použití a interakci. Vylepšení navrženého systému založené na dosažení přijatelného a škálovatelného rozhraní webové aplikace společným využitím různých programovacích jazyků. Poskytněte možnost mít a vlastnit parkovací účet s flexibilitou kontrolovat volná místa a snadno si je rezervovat a nahrávat podle potřeby uživatelů.

Tento článek se zabývá navrhovaným parkovacím systémem počínaje stručným pochopením potřeby jeho implementace, toho, co se používá k dosažení funkčního systému a jak systém interaguje s uživateli. Vysvětlete podrobně proces vývoje a diskutujte o tom, jak by se dal v budoucnu zlepšit.

Klíčová slova: Web development, HTML, Django, Parking system

Contents

1. Introduction

Currently, with the growth of population and economic development, the number of vehicle is also getting increased, searching for vacant parking space is being the main issue for all drivers. Time wasting while searching for an available spot, parking a vehicle illegally and a lot of environmental impacts are all reasons to come up with a great and huge solution to solve and help people to easily with safety find and book vacant spots.

Parking is limited almost in every major city in the world, same things goes on at CZU campus where it is always a challenging for employees and students to directly have a slots for their vehicles. Developing parking system can have a big impact to solve and limit traffic issues. Defining and improving a system with an interactive user interface, where users can easily understand it, interact with it, find vacant slot and be able to book it even before a month or a day is what we are providing in this project. This proposed system can also be used and implemented in different areas not just University campus. A key aspect of parking system is efficient use of free zones which allows more vehicles to be parked in such area, reduce traffic overcrowding, also parking system can have a huge impact in safety and help in protecting your health by reducing CO2 and smart parking system allows drivers to obtain real-time parking information and alleviates parking contentions, the most important thing is facilitating the process and time taking for people to find an available spot for their vehicles. Web development aims to make people aware of your service, understand why your business are important to them to use. A website offers a straightforward method of showing the credibility of a business, and the way a person represents his business online is vital for attracting more customers or visitors. Therefore, your website design should be handled in the best possible way, because a professional presentation speaks volumes as testimony to your business. Structuring your website in simple and understanable way for attracting users.

This paper view the structure of parking system, what technologies get used to improve and develop the system, how it got implemented with real scenarios in order to facilitate parking process for people. Web development aims to make people aware of your service, understand why your business are important to them to use.

1.1 History of Parking system

When we go back to 1920s, there was no systematize of parking anywhere yet. People would park their vehicles anywhere in the streets, while you are at university of you went for shopping it was illigal to leave your car anywhere and come back to it on anytime. Here comes to the main reason of facing traffiic congestion problem. The first working parking meter was designed by Holger George Thuessen and Gerald A. Hale. Hale and Thuessen started working on the parking meter in 1933 because of the assigned project by Carl Magee. The parking meter they designed was called The Black Maria. The first parking meter was installed in Oklahoma City on July 16th in the year 1935. Magee wasn't the first one to file patent for a parking meter. The first patent for a parking meter was filed by Roger W. Babson on August 30, 1928. Babson was an entrepreneur in the early 20th century.

1.2 History of parking garage

Carriage buildings served as the first parking garages as we know them now at the turn of the 20th century. These areas, which were formerly used to house horses, were repurposed to house cars, and as cars became more widely available, there was a need for more buildings specifically designed to shield the exposed interiors from the weather. In order to save space, several early designs from the 1920s and 1930s had substantial pulley systems that lifted cars up to various floors in stacked towers. Others resembled standard Art Deco office buildings with glass windows and big doors. Nobody on the street would likely realize that these structures were made exclusively to house cars. The Hotel for Autos, a 24-story skyscraper in New York City that could accommodate more than 1,000 vehicles, was one such mega-tower for automobiles that was inspired by this. In contrast, only 270 vehicles can fit in New York City's biggest automated parking garage. Because to the work of many architects and designers who kept experimenting with parking garages' general designs and structural capabilities, parking garages were a typical part of life by the 1960s. Bertrand Goldberg created one of the most well-known parking structures with Chicago's Marina Tower, combining spaces for people to live, work, play, and park. Around 20 storeys of workplaces and housing make up a spiraling parking structure.

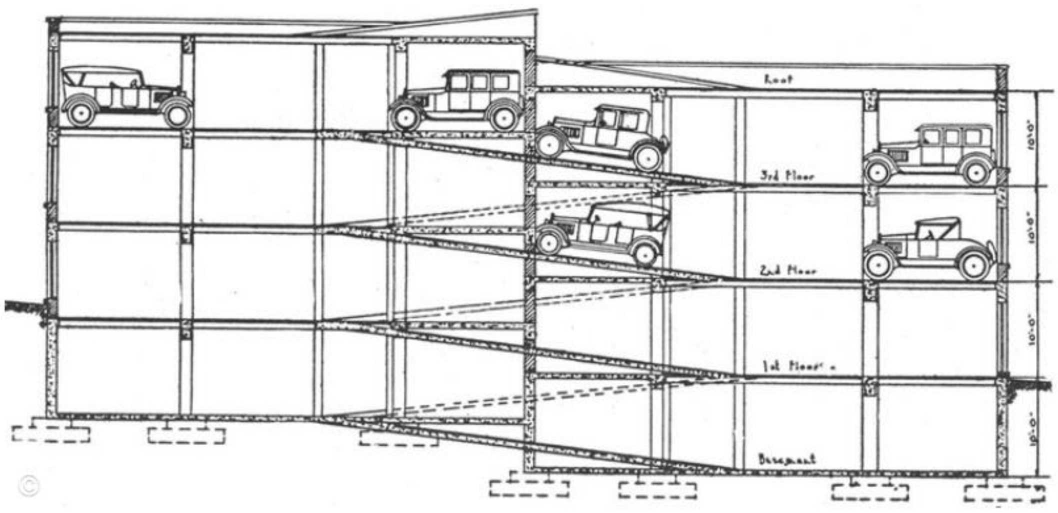


Figure 1: Parking garage (Source https://www.archdaily.com/)

2. Objectives and Methodology

2.1 Objective

The main objective of this thesis is to face the problem of Parking in the University Campus by making a Website with mobile compatible for Parking System. The mail goal of the website is to help students and employees to search and find free space at campus, and book it easily for specific hours. The benefit of achieving this website is to reduce the time taken and the hassle factor of locating an available parking space. Being able to accurately direct a driver to an available space has many environmental benefits like reduce CO2, noise and reduce traffic.

2.2 Methodology

Methodology The work consists of theoretical and practical part. The theoretical part of the work is generally based on the study of professional information sources and based on the knowledge gained will describe all the starting points for the following practical part, namely programming languages, technologies and libraries used for Web development. The practical part will consist in the creation of a website focused on the creation and display of events. An analysis of the website and definition of basic functionalities that will be implemented in this diploma thesis will be performed. Based on this analysis, the layout of the website in terms of navigation will be performed, as well as the selection of the necessary technologies, libraries and a suitable database for the commissioning of this website.

After that we started developing the project with a basic design with some functionality till we reached to the current design. Iteration waterfall development process model has been used in for this project as shown in figure 2.

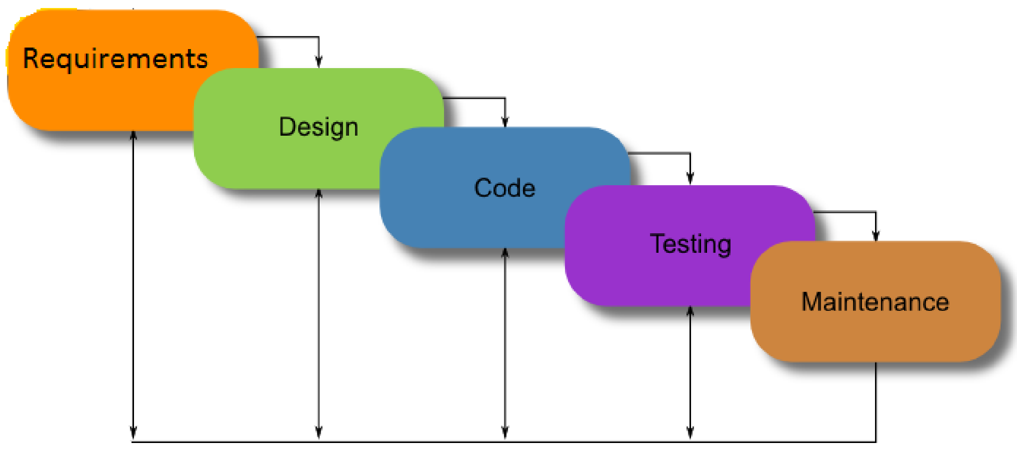


Figure 2: Project Development Process Model

Iteration waterfall development process model is very simple one, it includes many phases or steps as shown below:

Requirements: All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification doc.

Design: The requirement specifications from first phase are studied in this phase and system design is prepared. System Design helps in specifying hardware and system requirements and also helps in defining overall system architecture.

Implementation: With inputs from system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality which is referred to as Unit Testing.

Testing: All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.

Maintenance: There are some issues which come up in the development process and especially in the testing phase. To fix those issues patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment. Maintenance includes: Fixing errors, Updating, Upgrading, and Adding new functionality.

Organization: This documentation contains five chapters. Each chapter presents the required information about a part of the whole system. By reading the five chapters, all the aspects that included in the development of this project will be explained and clarified.

The chapters of the documentation are organized as follows:

Chapter 1: Introduction

This chapter gives the reader an introduction on the parking systems and the need of it in these modern days.

Chapter 2: Objective and Methodology

This chapter showes the objective and methodology of the proposed parking system.

Chapter 3: Literar review

In this chapter, all the technologies, tools, and software components that used to achieve this proposed system are explained. This chapter is giving a brief introduction to each technology used, the functionalities of each one of them is also identified.

Chapter 4: Practical part

This chapter present the proposed system for parking system, which work with different programming languages seprated with building the front end of the website by using HTML, CSS and Java Script, completing back end of the system by using Python and django framework as well as sqlite for database saved.

Chapter 5: Conclusion and Future Work

This chapter concludes the documentation by summarizing the overall work and indicating possible future works of the proposed system that can be done to improve the quality of the system in general.

3. Literature Review (Technologies Used)

In this chapter, the web application technologies and components which have been used to develop this project are described briefly.

As our parking system based on developing a web application for university which deals with students and employees need. Before improving the proposed system we need to make sure of understanding some technologies and terms used.

3.1 Different between website and web application

A website is a group of connected web pages that are kept on a server. The web pages, which primarily serve informational or promotional purposes, may include text, photographs, videos, and other types of media. Websites can be seen on a range of gadgets, including desktop computers, laptops, tablets, and smartphones, and are accessed through a web browser. There is two type of website, Dynamic and static website. Dynamic is the more complex one as it could contain a databse as backend to interact with users. Statistic is the simple one because it doesn’t consist of any database that the user can interact with, it’s simply crrated with using HTML, CSS programming langauges.

On the other hand, a web application is a piece of software that runs on a web server and can be accessed using a web browser. Online applications, which can include interactive features and functions like online shopping, social networking, email, and productivity tools, are frequently more complex than websites. They frequently demand user authentication and might save user information on a server. There are various types of web apps such as Static web applications, Dynamic web applications, E-Commerce web applications, Single-page web applications, Portal web applications, Content management system web applications, Animated web applications, and Rich Internet web applications.

Website	Web application
Represent a content of the page where the user can view and read	Users interact with the page not only view it.
Information here is publicly accessible	Information here is restricted to registered users
It's easy to develop since it requires only the basic web technologies	It's more challenging, because it requires a higher level of security and functionalities based on its purpose
Website help to inform	Web application serve to assist

Table 1: Website vs Web application

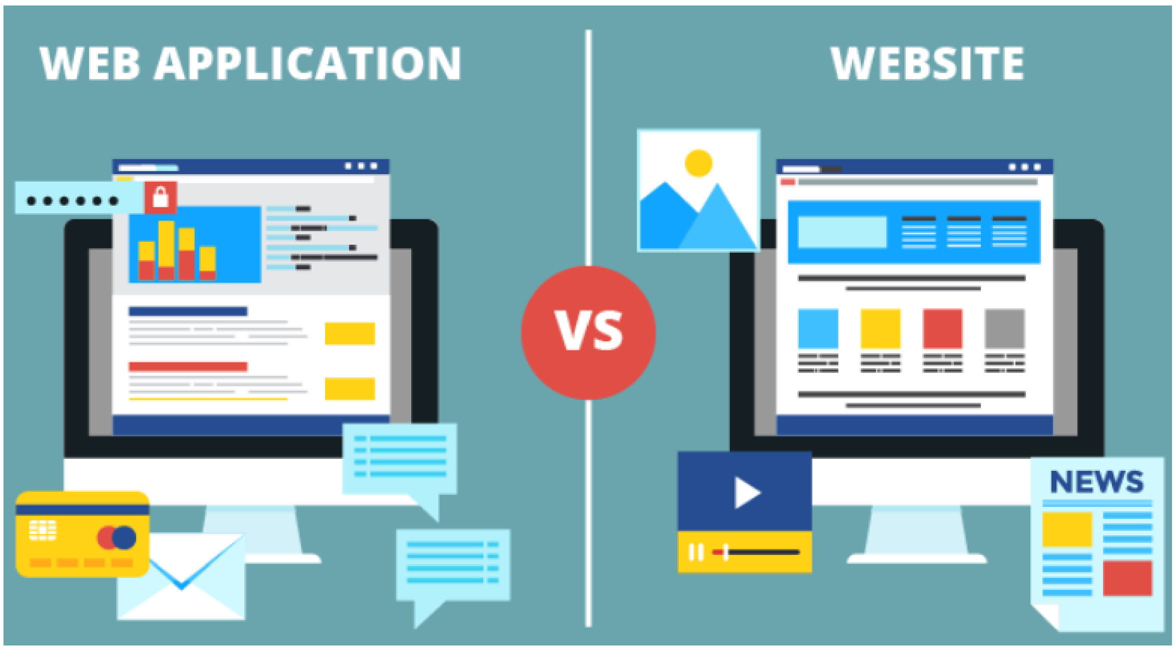


Figure 3: Difference between website and web application (Source https://gmedia.net.id/)

After understanding the difference between website and web application, will explain in details what we used for building parking system web application. The main structure of creating a web application is starting with creating the interface which called front end of the web, after that connecting it with database and server side which refers to back end of the web.

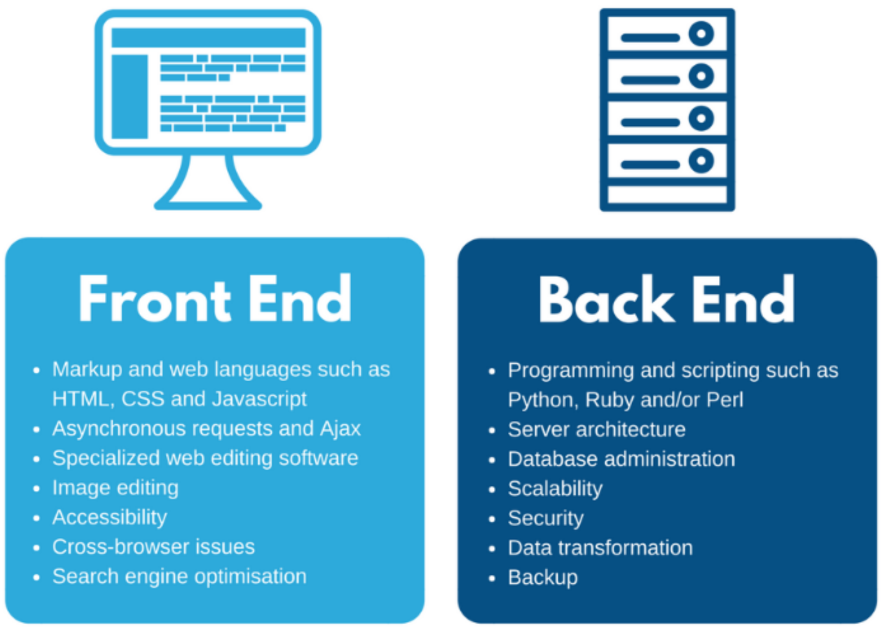

Front End
Markup and web languages such as HTML, CSS and Javascript
Asynchronous requests and Ajax
Specialized web editing software
Image editing
Accessibility
Cross-browser issues
Search engine optimisation
Back End
Programming and scripting such as Python, Ruby and/or Perl
Server architecture
Database administration
Scalability
Security
Data transformation
Backup

Figure 4:Difference between Front end and back end (Source https://www.metasource.co/)

3.2 Front End

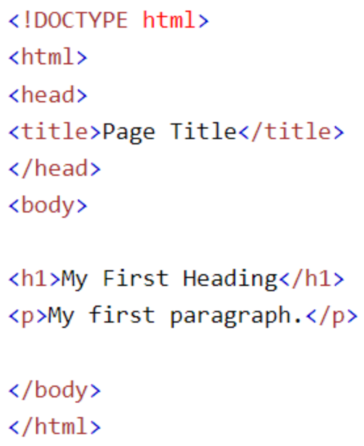
Front end is a term which is famous in web development, also known as front face. while creating your website or web application you need an interface to interact with users. So users or visitor of your site can easily find the information that they need. Therefor front end user friendly interface. Front end can be any design, layout, picture, video, color and text which can be vision and interact to users. Front end developers need to make sure that website design is good, make it understandable to users, easy to use and interact with.
I used three different languages for creating the web application interface:

3.2.1 HTML

HTML stand for Hyper Text Markup Language, A markup language is a set of markup tags and the tags describe document content. When it comes of structuring and organzing your web page, HTML are the most commonly used ones, it’s a basic programming language used to define a web stucture content by deviding it into differnet blocks for exmple you can have a part of your web page as pharagraph, other part as heading. HTML was first developed in 1993 by Sir Tim Berners-Lee. Since then this markup language has had many versions, the latest variation is labeled HTML5.

HTML is the most common used language to write web pages. It getting popular due to its advantage:

1. It is easy to understand and modify.
2. Flexible while creating and building web pages with text.
3. Can be displayed on any platforms like Windows, linux and Macintosh.
4. Provide extra features to your web page by allowing to add videos, sounds and graphics, which leads to attract more visitors.
5. Ability to add links to the web pages.



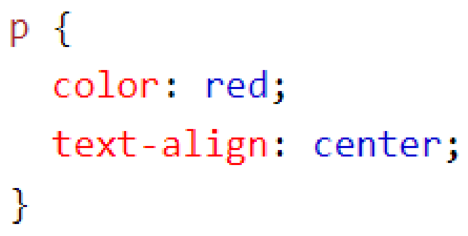
```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

Figure 5: Simple example of HTML document (Source https://www.w3schools.com/html/default.asp)

- The <!DOCTYPE html> declaration defines that this document is an HTML5 document.
- The <html> element is the root element of an HTML page.
- The <head> element contains meta information about the HTML page.
- The <title> element specifies a title for the HTML page (which is shown in the browser's title bar or in the page's tab).
- The <body> element defines the document's body, and is a container for all the visible contents, such as headings, paragraphs, images, hyperlinks, tables, lists, etc.
- The <h1> element defines a large heading
- The <p> element defines a paragraph

3.2.2 CSS

CSS stands for Cascading Style Sheet, after structing the web page you need to style it by adding some colors and organizing your paragraph aligns and more features which make your web page. Basically controlling the way of presenting the HTML document.



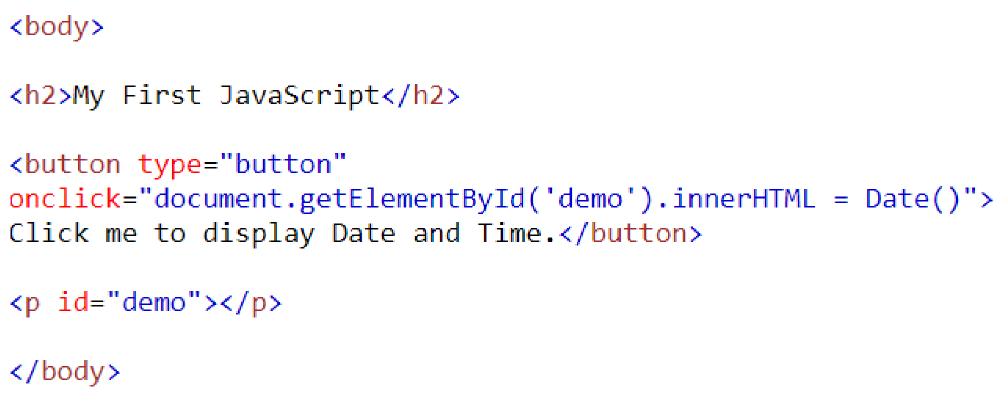
```
p {
  color: red;
  text-align: center;
}
```

Figure 6: Simple example of CSS (https://www.w3schools.com/css/default.asp)

- p is a selector in CSS (it points to the HTML element you want to style.
- color is a property, and red is the property value.
- text-align is a property, and center is the property value.

3.2.3 Java Script

Java Script is high level programming language which used to build and create and interactive and dynamic web pages and web applications to response to user actions. JavaScript is used in conjunction with CSS and HTML which help to provide functionality to our web page or web application.
Java Script can be used also for creating web based games, developing mobile application, also building server side application with platforms like Node.js.



```
<body>

<h2>My First JavaScript</h2>

<button type="button"
onclick="document.getElementById('demo').innerHTML = Date()">
Click me to display Date and Time.</button>

<p id="demo"></p>

</body>
```

Figure 7: Java script example (Source https://www.w3schools.com/js/default.asp)

3.2.4 Web design aspects

User experience: The process of creating physical and digital goods that are useful, pleasurable, and practical for people is known as user experience (UX) design. By incorporating elements of branding, design, usability, and function, UX design aims to provide products that offer people meaningful and pertinent experiences. Focusing on the user's interaction with a product, UX designers strive to make it useful, approachable, and enjoyable to use. Graphic design, architecture, interior design, software development, and industrial design are just a few of the talents and knowledge that go into UX design. To create a product that deal with users wants and expectations, the UX design process often entails research, prototyping, testing, and iteration.

User interface: Designing the appearance, feel, and interactivity of a digital product, such as a website or mobile app, is known as user interface (UI) design. The user interface (UI) includes a product's visual elements and interactive elements, such as the color, text, buttons, and animations that users interact with. Despite the fact that the two frequently overlap, UI and UX design are distinct disciplines. While UI design is in charge of a product's appearance, interactivity, usability, behaviour. The graphical components of digital products, such as mobile applications, websites, and devices, are created by UI designers who aim to make them both aesthetically pleasing and simple to use.

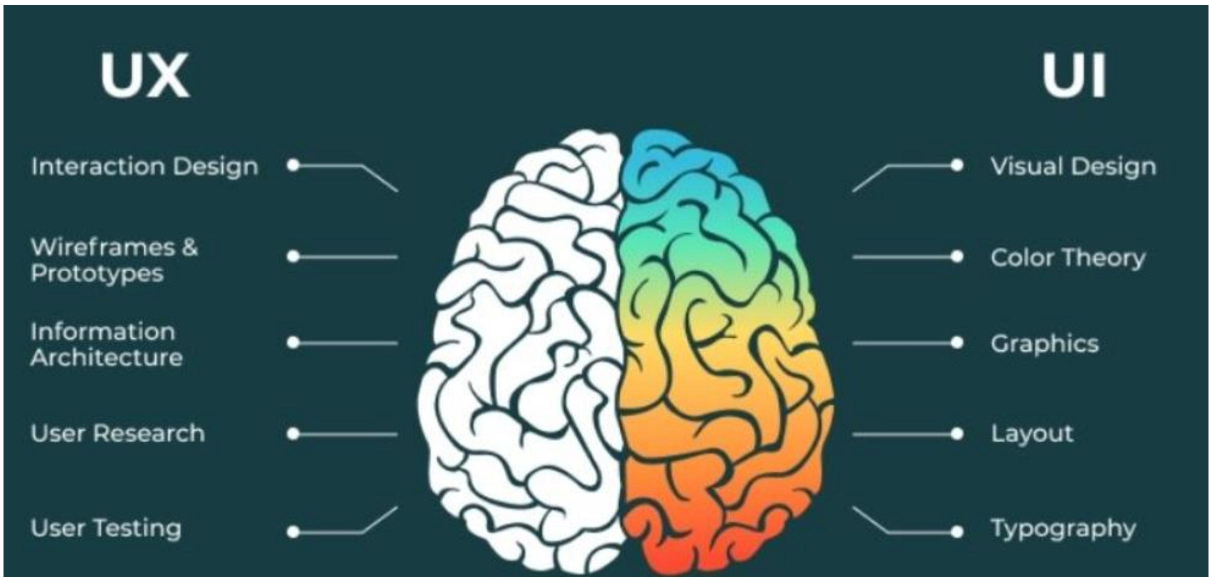


Figure 8: UX vs UI (Source https://www.mobitechspy.com/)

Usability: Is described by ISO 9241-210 as the degree to which a product, system, or service can be used by particular users to accomplish particular goals with effectiveness, efficiency, and satisfaction in a particular context of usage. Usability focuses on how the user completes their intended tasks, the tools they use to complete them, and the extent to which their demands are satisfied.

1. Learnability: The ease of learning the operation and behavior of the system for inexperienced users.

2. Efficiency: The level of productivity attainable once the expert user has already learned the system. The greater the usability of a system, the faster the user is using it, and the work is done faster.

3. Memorability: The ease of remembering the functionality of the system, so that the occasional user when returning to the system after an inactive period, does not have the need to learn how to use it again.

4. Errors: The system must have a low error rate, that is, users make few mistakes while using the system, and in case they make them help them recover easily.

5. Satisfaction: This is the most subjective attribute. It is the extent to which the user finds the system pleasant to use.

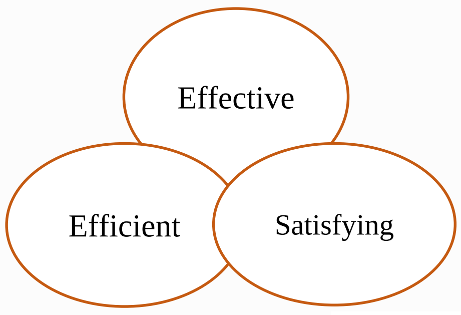


Figure 9: Usability

3.2.5 Website layouts

There is four different types of website layouts, Static, liquid, adaptive and responsive.

Static Layout: A static layout has a fixed width and does not change based on the width of the browser. This means that it is only responsive when scaling content in response to changes in the viewport's size. In this instance, the entire layout is static and only the elements' sizes are altered to display the entire template on a single screen. In other words, static layouts only allow for the desktop version of a website to be viewed on a mobile or tablet browser.

Liquid Layout: A liquid layout, also known as a fluid layout, it helps in changing the page size based on the browser width or screen size.

Adaptive layout: An adaptive layout refers to the process of adapting to the screen resolution for example mobile, tablet or disktop. It’s directly understand the size of the browser of screen and make the changes according to that. Because adaptive layouts don't provide completely responsive websites that properly fit every device size, they are most effective when users merely wish to accommodate a variety of devices and screen resolutions for their website. Only fixed website versions for specific screen resolutions will be produced by an adaptive page layout; not all screen resolutions will display the website flawlessly.
The biggest drawback of this type of adaptive layout is that it may leave too much or not enough space between the main container and the edges of the browser depending on the screen resolution.

Responsive layout: The most difficult and efficient method for designing websites that will display flawlessly on screens of any size is a responsive layout. This method combines adaptive break points with fluid relative units to create a page layout that is a hybrid of the two.

3.3 Back End

Backend is a part of a web application or software, after completing the Front end of your web application you need to take the next step forward the Backend which is a server side of a web development that helps in managing the data storage, underline logic and processing user requests. Back-end developers are responsible for creating the website's structure, writing code, and verifying that the code works. Technologies that developers can used for building complex web applications are some of programming languages such as Java, Ruby, PHP and Python the one that I used to develop my University Parking System. Creating and administering databases, managing server requests and answers, and assuring the security and scalability of a website or application are all part of back-end development. For dynamic, interactive websites and applications that can handle a lot of data and traffic, back-end development is crucial.

Reasons of the need of Backend:

1. Security: Backend is responsible to manage the security of the web application like authentication, authorization.
2. Integration: Backend helps application to communicate with other and external services by integrating with other systems and APIs.
3. Data Storage: Storing and managing data is one of the responsibilities of the Backend.
4. Scaling: To handle huge volume of data, Backend is easy to scale as much as your application need.

3.3.1 Python

Python is a high level programming language which is used in web development, machine learning, data analysis and more. First released in 1991 and was created by Guido van Rossum. Python's use of indentation to separate code chunks rather than curly brackets or other symbols is one of its distinguishing characteristics. This facilitates the reading and comprehension of Python code, but it also means that proper indentation is essential to the efficient operation of the code. Python key features are simplicity and readability, which helps to make it easy to use and learn. Also it can be used on different platforms like windows and linux. Python is an interpreted language, therefore it doesn't require compilation in order to run. The Python interpreter, on the other hand, reads the source code and runs it directly. This eliminates the requirement for a separate compilation phase and enables rapid code writing and testing. Python also can be used as backend language in many different areas for example as web frameworks, API development, data analysis and task automation.

- Web frameworks: To develop a web application Python have several frameworks which can be used to provide structure of building your web application and handle many tasks as managing databases and routes. Most popular frameworks are Django and Flask.

- API development: Developing APIs will help to communicates with other services and applications.

- Data analysis: NumPy and Scikit-lear libraries are most often used for analysing data in Python. While building your Backend you might need to process and work with data so Python might be a good option for you.

- Task automation: You might need to have and write a script that response and automate for a task, such as system manipulation or data cleaning.

3.3.2 Django Framework

When it comes to security, fast development, scalability and large community Django framework is the best choice for building a web application. It was developed in 2003 by a team of developers at Lawrence Journal World, get progressive in 2005 as an open source project. Django framework pursue the model view controller “MVC” architectural pattern which refer to model view template “MVT” also include build in features which make web development easier and faster, such as Object Relational Mapping “ORM” layer. The key principles of this framework is don’t repeat yourself “DRY” which supports developers to avoid twofold of code and makes the maintenance easier for them and reduce development time.

Steps to start using Django framework:

1- Install Django in your computer by running this command

```
Pip install django
```

2- Create your django project within your project name

```
Django-admin startproject parking-system
```

3- Create new django app with the app name

```
Python manage.py startapp parking
```

4- Data model definition by creating new file called models.py
5- Create database tables

```
Python manage.py makemigrations python manage.py migrate
```

6- Define your views for handling user requests and return HTTP responses.
7- Define URL for mapping URLS to views.
8- Define Templates which refer to the presentation layer of your application.
9- Run Django server

```
Python manage.py runserver
```

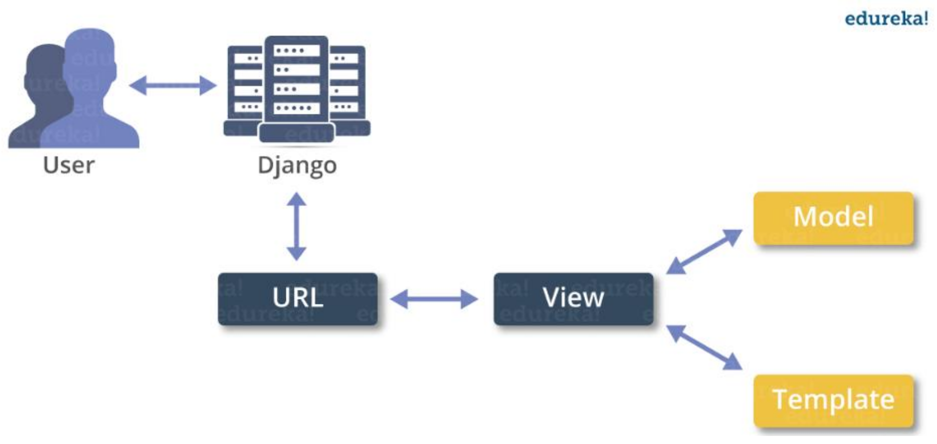


Figure 10: Django process Diagram (Source https://www.edureka.co/)

3.3.3 API

Application Programming Interface is a tool used by developers for web development which allow applications to interact and communicate with each other. There is a lot of purposes of using API but the main one is helping with integration which means integrating different software applications with each other for sharing data and communication. Software developers can extract and communicate information using an APIs' set of protocols, routines, and developer tools, which also allows apps to interact in a user-friendly way. APIs can be used for a variety of tasks that call for communication across various applications or systems, including retrieving data from servers, sending data to servers, and more.

The following are some characteristics of APIs that make them beneficial for software development:

1. Interoperability: Regardless of the programming language or platform on which they are developed, APIs enable communication between various applications.
2. Modularity: APIs let programmers divide complicated programs into smaller, easier-to-manage parts that can be independently created and tested.
3. Reusability: APIs can be applied to a variety of applications, cutting down on expenses and development time.
4. Scalability: APIs are great for creating scalable applications since they can manage massive amounts of data and traffic.
5. Security: To guarantee that only authorized users can access sensitive data or functionality, APIs can be guarded using authentication and permission techniques.

3.3.4 Database

A database is a structured collection of data that can be electronically stored and accessed via a computer system. Databases are created to make it easier to store, retrieve, edit, and delete data while carrying out various data-processing tasks. The data is managed and information is extracted from the database using a database management system (DBMS). Databases can range in size from tiny file system-based databases to big ones housed on computer clusters or on the cloud. Database design encompasses both formal methods and pragmatic factors, such as data modelling and effective data storage. Large volumes of data are stored and managed via databases in a variety of applications, such as e-commerce, healthcare, banking, and social media.

There are various kinds of databases, such as:

1. Relational databases: These databases keep information organized in tables with established connections between them. They are based on the Structured Query Language and are frequently used in corporate applications (SQL).

2. Document databases: These databases keep data in documents that are either XML, JSON, or BSON. They can hold hierarchical data structures and are adaptable.

3. Key-value stores: These databases are utilized for session management and caching, and they store data as key-value pairs.

4. Column-oriented databases: These databases handle enormous volumes of data quickly because they store data in columns rather than rows.

5. Graph databases: These databases can handle complicated interactions between data points because they store data in nodes and edges.

Time-series databases, object-oriented databases, and NoSQL databases are some other varieties of databases. The specific needs of the application, such as the amount of data to be stored, the complexity of the data, and the demand for scalability and performance, will determine which database type should be used.

SQL

A programming language called Structured Query Language (SQL) is used to control and work with data that is kept in relational databases. It is the common language for communicating with relational databases like Oracle, MySQL, Microsoft SQL Server, PostgreSQL, and SQLite. It is used to create, change, and query databases. Data definition language (DDL) instructions are used to create and edit database objects like tables, views, and indexes, whereas data manipulation language (DML) commands are used to insert, update, and delete data. To manage and analyse huge amounts of data, SQL is frequently used in data-driven applications, such as e-commerce, banking, healthcare, and social media. For anyone dealing with relational databases, SQL is a valuable tool for managing data.

SQLite

SQLite is a database engine, software library that provides a relational database management system RDMS, that I have used for building this system. It is designed to be linked into an application and provide a rich set of APIs for accessing and manipulating data. So it's not stand alone application and can't run separately unlike other databases. Features of using SQLite as database management system is portability, Simplicity, compatibility by supporting most SQL standard syntaxes, fast and efficient and providing security features by including encryption and authentication. Open-source SQLite is free to use and distributed under the public domain license. It has a variety of capabilities, including transactions, triggers, and views, and it supports conventional SQL syntax. For developers looking for an easy-to-use, standalone database solution, SQLite is a popular choice due to its minimal size, excellent performance, and stability. It is extensively utilized in the creation of mobile applications, including well-known ones like WhatsApp, Skype, and Firefox.

3.3.5 PyCharm

An integrated development environment (IDE) for Python called PyCharm is made to assist professional developers in writing more effective, self-assured, and clean code. PyCharm Pro and PyCharm Community are the two versions that it comes in, both of which are created by the Czech company JetBrains. Out of the box, PyCharm supports the entire Python workflow, including frontend technologies, databases, and web frameworks. An intelligent code editor, code analysis, a graphical debugger, an integrated unit tester, and integration with version control systems are just a few of the features it offers. Along with cross-technology development, PyCharm also supports JavaScript, CoffeeScript, TypeScript, Cython, SQL, HTML/CSS, template languages, AngularJS, Node.js, and other languages. Professional Python developers frequently use PyCharm, which is renowned for its intelligent code editor, syntax, and error highlighting, and enhanced code comprehension and readability.

Some of the key features of Pycharm:

1. Intelligent code editor: PyCharm has an intelligent code editor that makes it easier to write Python code of the highest caliber. The use of various color schemes for keywords, classes, and functions, often known as syntax and error highlighting, improves code comprehension and readability.

2. PyCharm offers code analysis to assist in finding faults and other problems in the code.

3. Graphical debugger: PyCharm comes with a graphical debugger that lets programmers navigate through code and find problems.

4. Built-in unit tester: PyCharm provides an inbuilt unit tester that enables programmers to test their code and find bugs before distributing it.

5. Version control system integration: PyCharm interacts with well-known version control systems like Git, making it easier for developers to maintain their code and work with others.

6. Support for multiple technologies: In addition to Python, PyCharm also supports JavaScript, CoffeeScript, TypeScript, Cython, SQL, HTML/CSS, template languages, AngularJS, and Node.js.

4. Practical Part

This chapter present the proposed parking system, to understand the work of the proposed system and its functionalities and events. This chapter includes the system development life cycle of parking system web application with brief explanation of each phase. The implementation phase of system development, where the system is actually built and put into use, is referred to as the practical element of a system. This step involves writing code, configuring hardware and software, testing and debugging the system, and deploying it for end users.

4.1 System development life cycle

A project management model is essentially what a system development life cycle, or SDLC, is. It outlines the various steps required to take a project from its inception or inception to deployment and later maintenance. Phases of the SDLC allow you to swiftly develop high-quality software that is well-tested and prepared for usage in production. Development Process of creating a website of web application refers to sequence of steps involves in building, designing and deploying a website. Development process steps are Define the problem, find a solution for the problem, Planning, designing, Create the content, development, testing and maintain the website.

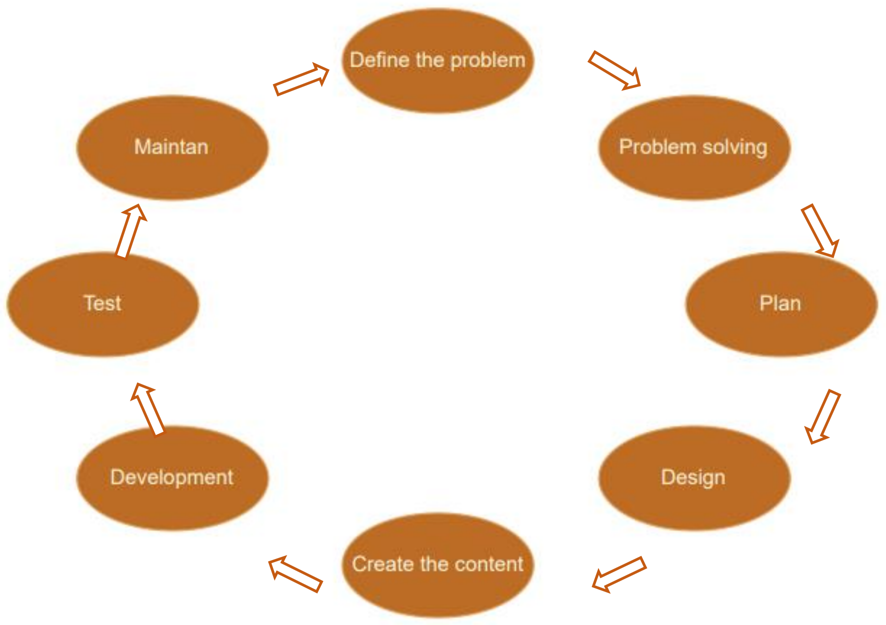


Figure 11: System development life cycle

4.2 Define the problem

Traffic congestion, time taking while finding a vacant zone, safety and health impact and environmental hazard are all reasons occurs nowadays due to the huge increase of vehicles number. This issue is not limited only in big cities, but we are facing it at CZU campus as well. Students are always being late for lectures and university and a lot of time they are not able even to park at university campus.

4.3 Problem solving

Problem solving is the process of achieving specific goal that deal with finding a solution for an issue in order to help and make things easier for people. Related to our topic, we decided to build a system for university campus where students and employees can have their own account and interact with the system by finding vacant zones and book it for specific days. Users will be able to login and access their account on anytime and anywhere to meet what they need.

Use case diagram

A use case diagram graphically depicts the system's behaviour. It is used to show how various system actors and the software interact with one another. A use case diagram can be created using programs like Microsoft Visio, UML Class Diagram, or Open UML. Use case diagrams are used to illustrate needs as seen from the perspective of the user. They are created top-down and offer a summary of all possible outcomes for a given system.

Related to our parking system, the use case diagram is showing how actor can interact with system.

- Register or login to the system
- Check vacant zones
- Booking
- Print ticket
- Check out from parking zone
- Log out from the system

we have two different users use cases, first one is new user and second one is already registered user.

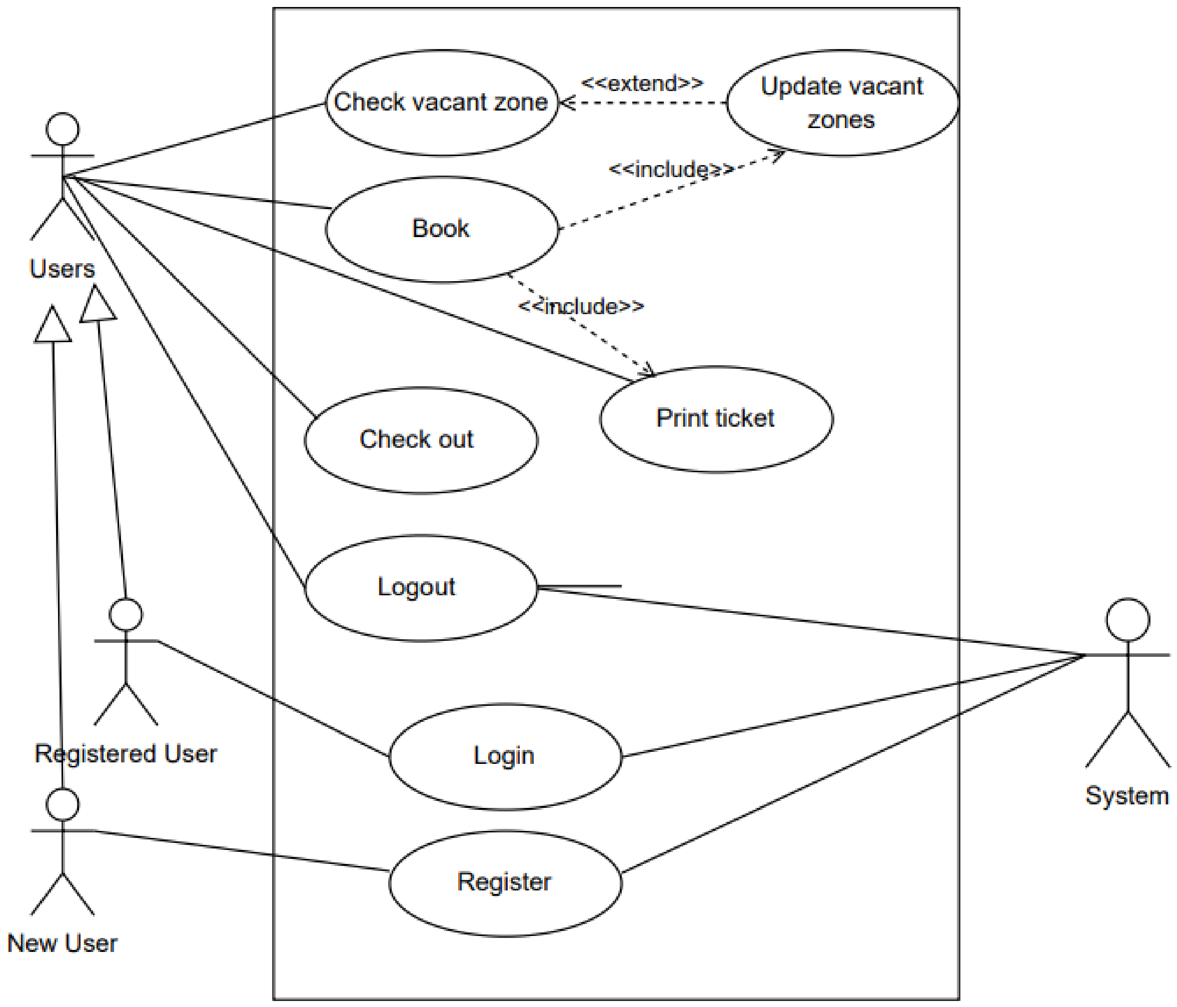


Figure 12: System Use case

4.4 Plan

Planning means completing a number of tasks in order to design and develop the website that will be able to draw visitors. Determining the target audience, the website's aims and objectives, and the features and functionality needed to achieve those goals all fall under this category. A content strategy and information architecture may also be developed during the planning phase, as well as wireframes or prototypes of the website's layout and design. In order to ensure that the website or application is created in a way that meet the demands of the target audience and accomplishes the specified goals and objectives, the planning phase is a crucial step in the web development process. The main goal of creating this system is to build an interactive and useable parking system which helps users to easily book an available zone in three different locations. Planned to have different interfaces in order to meet user scopes. The scope of my project is to let the user create an account in parking system application in order to log in to their account and be able to view application contents, free zones, book a zone, print out the ticket details, check out from reserved zone and log out from the account.

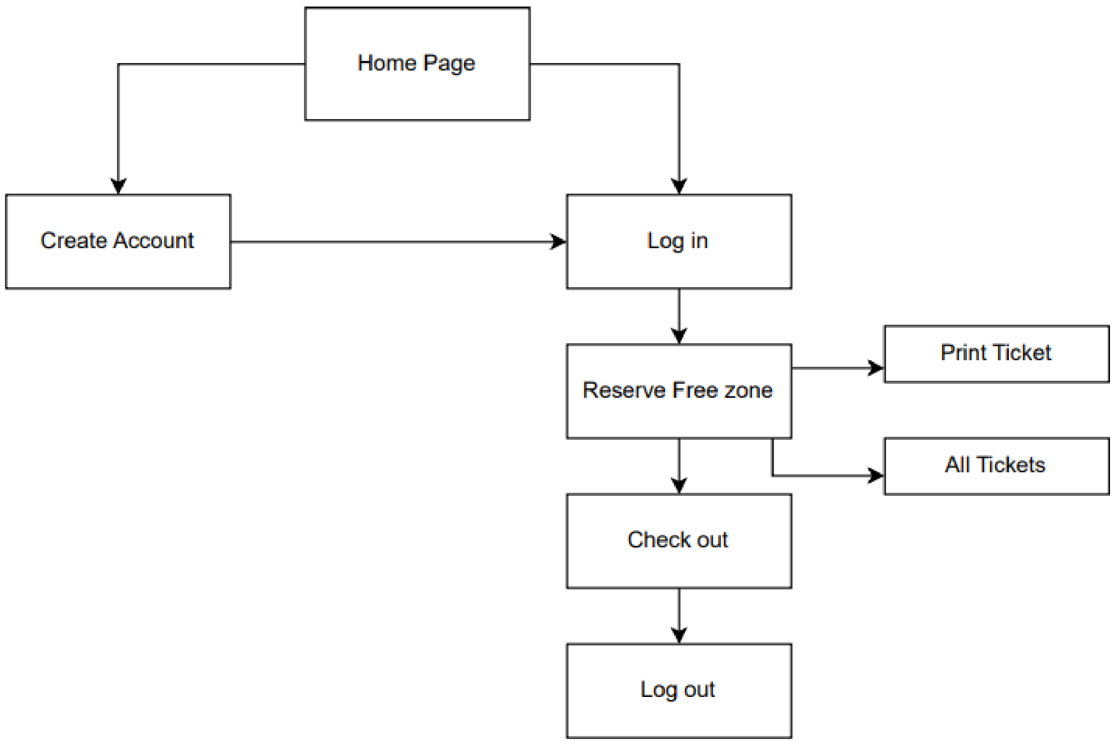


Figure 13: Basic plan diagram of parking system

Activity diagram

Activity diagram is type of UML (Unified Modelling Language) diagram, refers to sequence of actions and activities within a system or process. The activity diagram consists of a series of nodes and edges that represent the steps and transitions between them. The nodes represent the activities or actions, while the edges represent the transitions or flows between them. It can be used as business workflows, software systems and manufacturing process. Activity diagrams can be created using various tools, including SmartDraw, Venngage, and Lucidchart.

Elements of activity diagram:

1. Actions: In an activity diagram, each step or task that makes up a process is represented by an action. They can be straightforward, like sending an email, or intricate, like running a calculation.

2. Control flow: Arrows are used in activity diagrams to show how control moves from one action to another. These arrows, which can also include decision points and loops, represent the order in which actions are carried out.

3. Activity diagrams can be divided into swimlanes, which stand in for the many participants or departments in the process. This makes it clearer who is in charge of each action.

4. Forks and joins: Activity diagrams can depict parallel processing using forks and joins. A join brings these paths back together when a fork divides the control flow into several different paths.

Activity diagrams feature distinct start and finish points that signify the beginning and conclusion of the process being depicted.

4.4.1 System operation Diagram

The boundary between a system and its environment is defined by a system operation diagram, which also depicts the entities that interact with the system. It is a straightforward and cooperative diagram that can assist a team in gaining a thorough grasp of a system.As first activity diagram for our parking system, it shows how the process start and all activities that can be done by users.

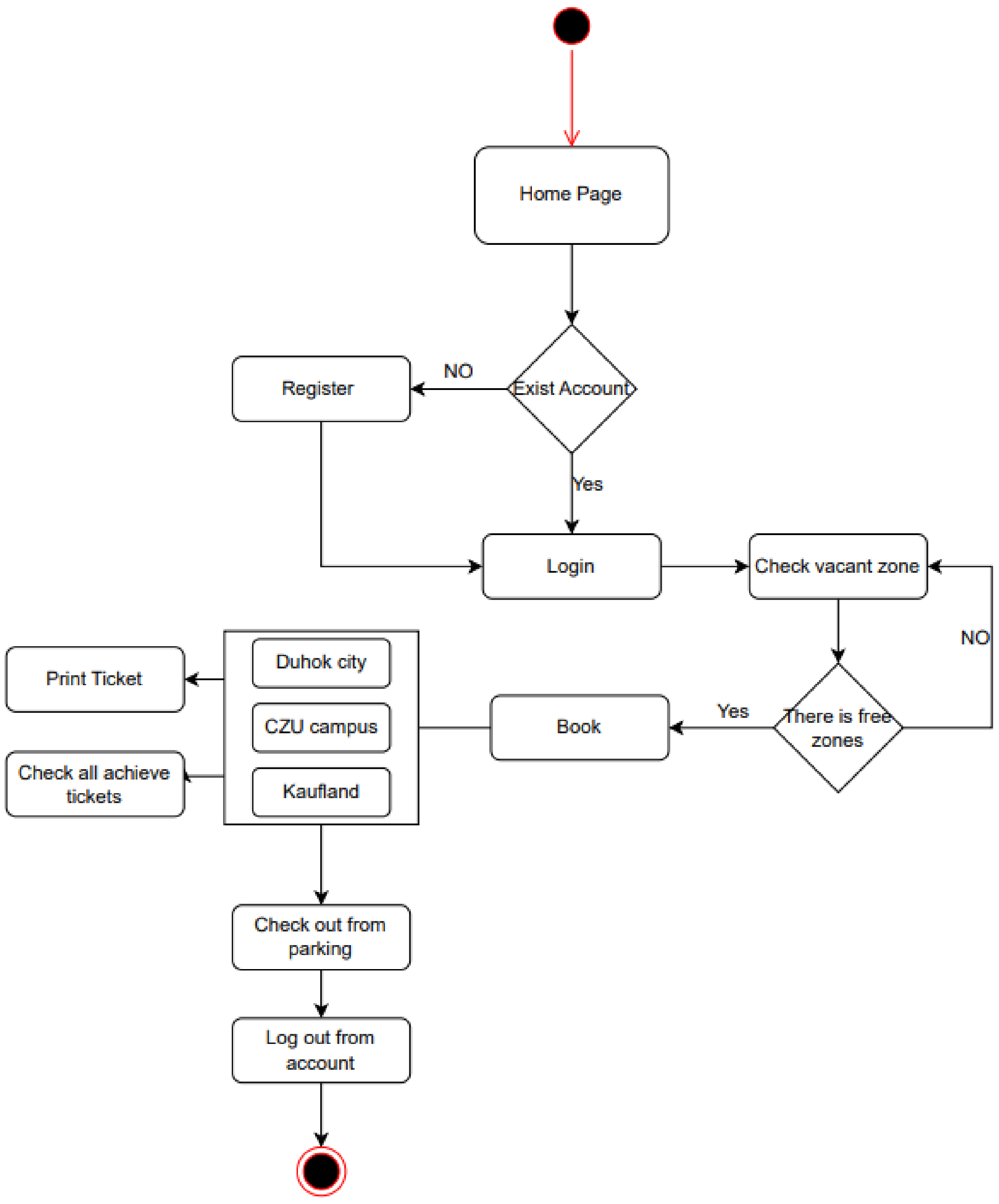


Figure 14: Activity diagram of System operation

4.4.2 Booking second location diagram

This activity diagram shows the process of booking another vacant place with already valid one. As our system doesn’t support having multiple booking, you have to check out from the previous one first to be able to have a new booking.

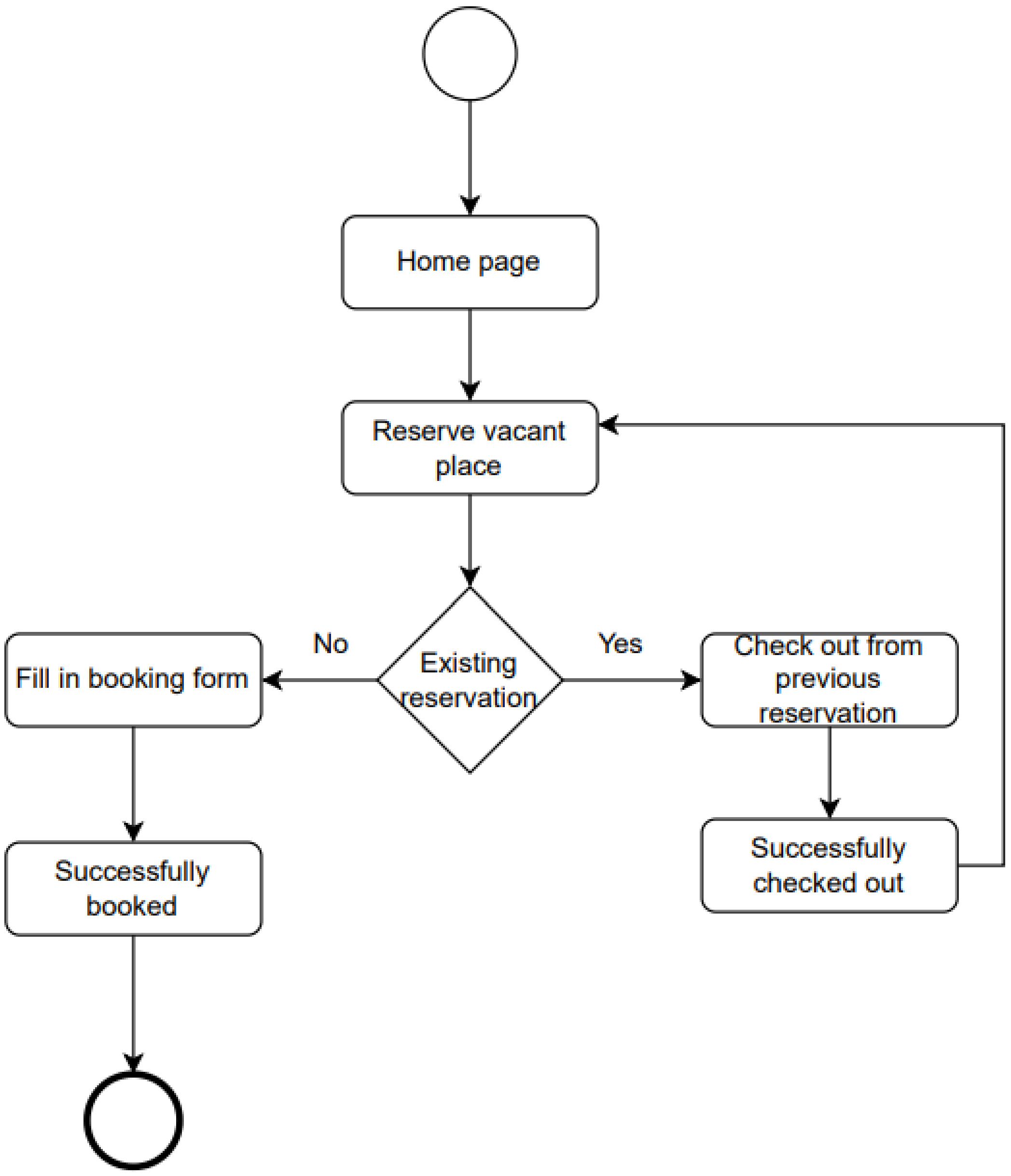


Figure 15: Booking second location use case

4.5 Design

Design phase is when your website takes the shape including the creation of all visual content, such as images, videos and all information that was gathered through the planning phase. This includes first step of the website structure and website visual design, will consist of Header, Footer, Side bar, Table and image.
Header shows on the top of your page usually involves important section about your website like contact details, website logo, navigation menu and different sections that user can access it to provide more information about the website.
Footer take place in the end of your web page, I used for copyright of the website.

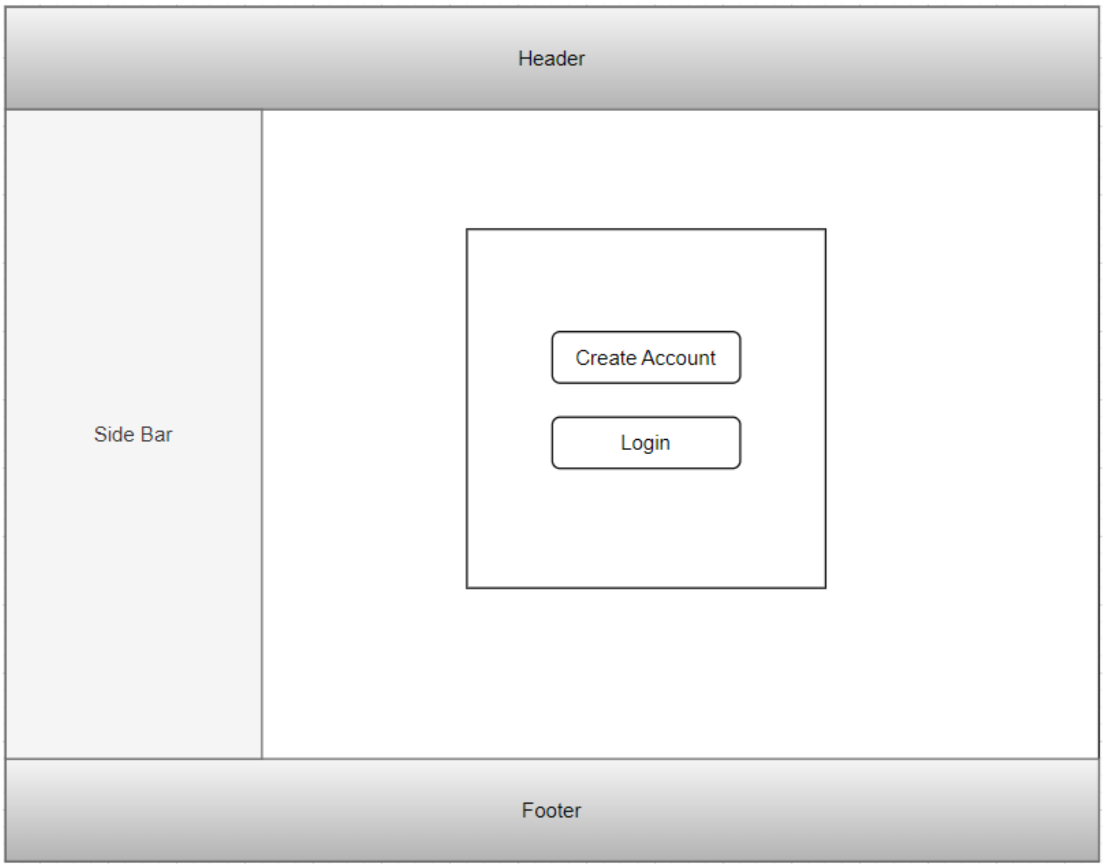


Figure 16: Basic design of login page

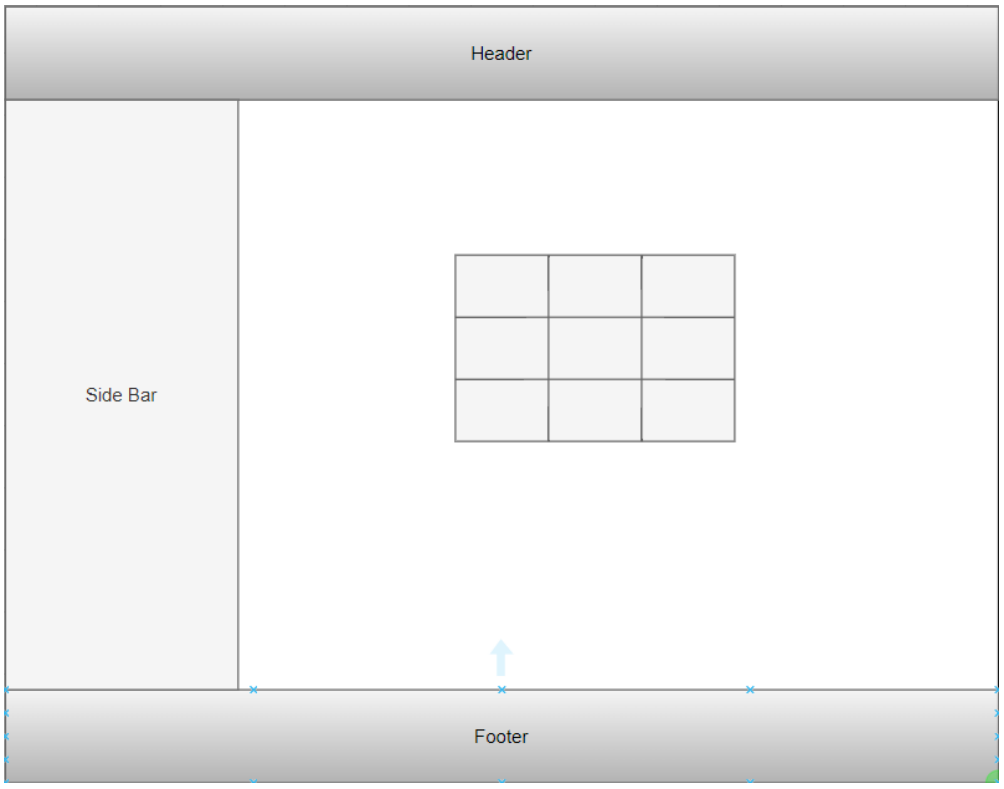


Figure 17: Parking System Interface

4.6 Create the content

This steps contain developing content for the website, including text, image and other multimedia.

4.6.1 Header

A web page's header is a crucial component that often contains a company name, logo, search bar, navigational tags, and more. It can be seen on all pages of the website and is often at the top of the screen. Both HTML and CSS can be used to construct a header.
The header can be modified using CSS to alter the font size, editing the text, add padding. The user experience can be enhanced and the website's navigation made simpler with a well-designed header.

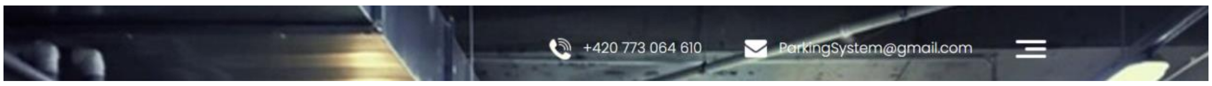


Figure 18: Header of parking system web application

4.6.2 Footer

A web page's footer, which is normally found at the bottom of the page, is a crucial component that includes details like copyright notices, contact details, and links to other pages on the website. The global site footer, which typically contains copyright notices, contact information, and links to other pages on the website, can also be defined using the HTML "footer" element.

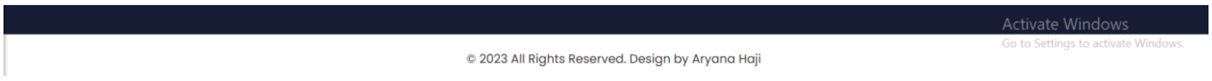
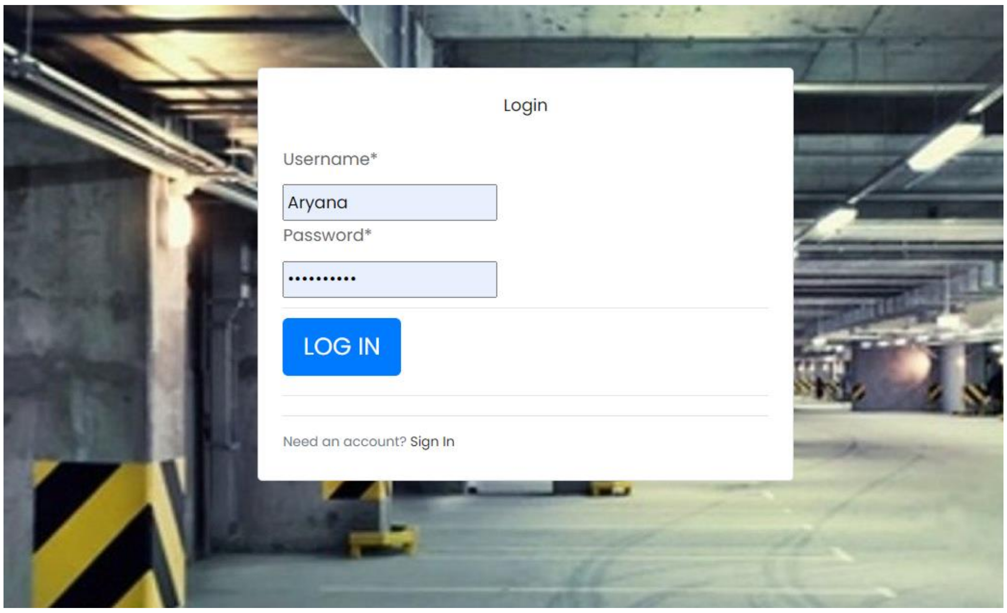

© 2023 All Rights Reserved. Design by Aryana Haji

Figure 19: Footer of parking system web application

4.6.3 Login

A website's login page, which enables users to access their accounts by entering their login information, is a crucial component. HTML may be used to add inputs for each field and wrap them in a "form" element to process the input to produce a login page.


Login

Username*

Aryana

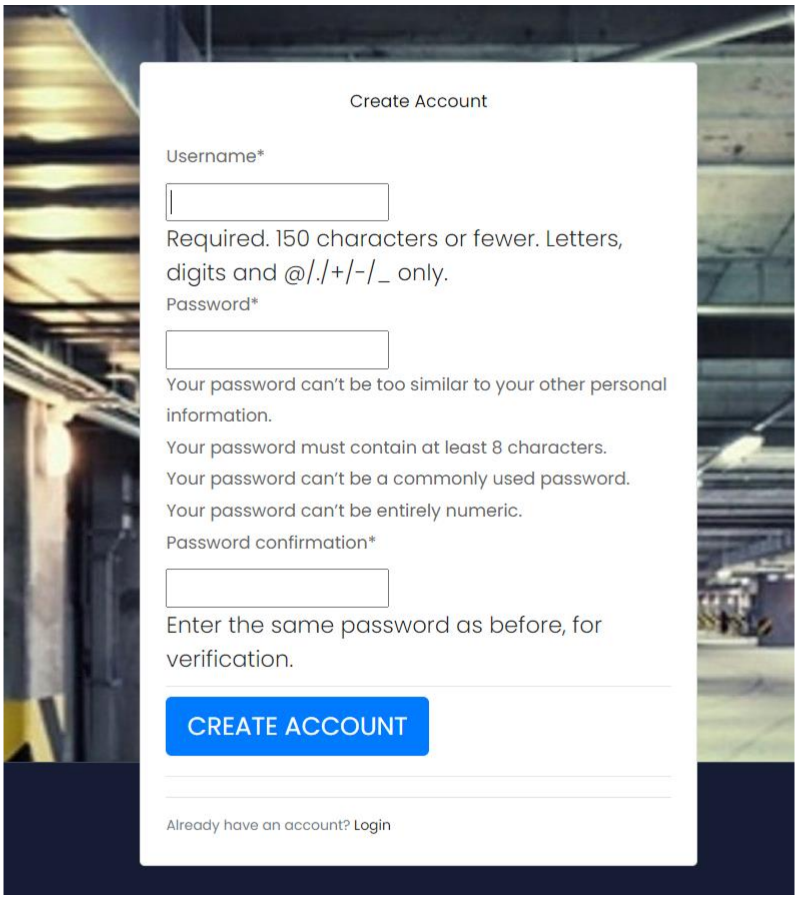
Password*

••••••••••

Figure 20: Login Page

4.6.4 Create an account

HTML can be used to create a sign-up form in web development to establish an account page. To process the input, the sign-up form can be enclosed in a "form" element, and inputs can be inserted for each field, including "name," "email," "password," and "confirm password." CSS can be used to style the sign-up form to add a border, padding, and a certain background color.


Create Account

Username*

Required. 150 characters or fewer. Letters, digits and @/./+/-/_ only.

Password*

Your password can't be too similar to your other personal information.
Your password must contain at least 8 characters.
Your password can't be a commonly used password.
Your password can't be entirely numeric.

Password confirmation*

Enter the same password as before, for verification.

CREATE ACCOUNT

Already have an account? Login

Figure 21: Create account page

4.6.5 Sidebar menu

A sidebar menu is a navigation menu with links to various website parts that is often found on the left or right side of a web page. The sidebar may be made using the HTML div> element, and it can be styled using CSS with a certain width, height, position, and background color. By keeping crucial navigation links visible and reachable on the screen even as the user scrolls down the page, a sidebar menu in web design can enhance user experience. By keeping the call-to-action button or contact form on the screen and in the user's thoughts, a sidebar menu can also aid in increasing conversions.

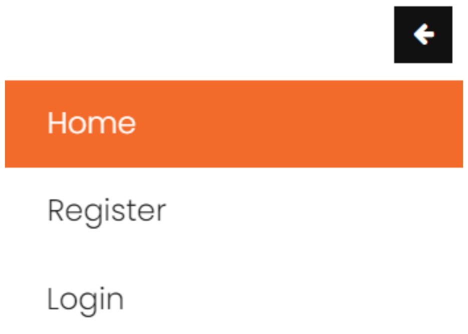


Figure 22: Menu bar before logging in

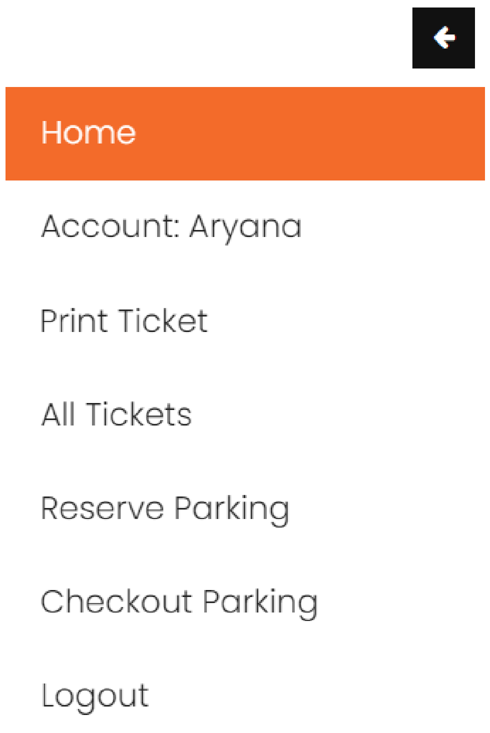
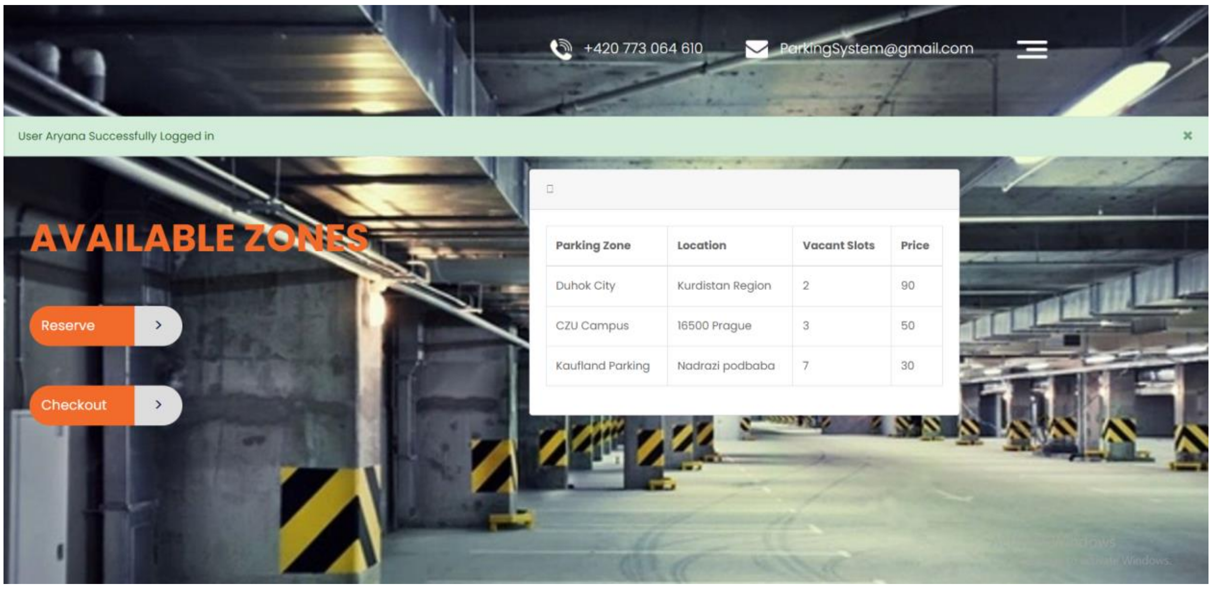


Figure 23: Menu bar after login to the account

4.6.6 Main page

The homepage, the initial page a visitor sees when arriving at a website, serves as its main page. A website's main page should give visitors a quick summary of the site's content and direct them to the relevant pages. Links to other website pages, such as the About Us, Contact Us, or Services pages, may also be found there. For a good user experience, a website's home page should be well-designed, aesthetically pleasing, and simple to navigate.
str



User Aryana Successfully Logged in

Parking Zone	Location	Vacant Slots	Price
Duhok City	Kurdistan Region	2	90
CZU Campus	16500 Prague	3	50
Kaufland Parking	Nadrazi podbaba	7	30

Figure 24: Main page of parking system

4.7 Development

Development process means how to turn out all programming languages works together in order to build the website or web application. The actual construction of the website or web application takes place during the development stage of web development. This stage, which comes after the planning and design phases, entails creating the website's or application's functional component. While front-end developers create the user interface and make sure the website is responsive and user-friendly, back-end developers construct the website's framework, write code, and confirm that the code functions. The website or application can be deployed and launched for end users after it has been created and tested. This part will include some section of code used in building parking system.

4.7.1 URL

Started with page URL which refer to Uniform Resource Locator, related to typed address browsers to get page access in internet. Each page of the system will have different URL named. URLs can be used to link to external resources like photos, videos, or other websites as well as to connect multiple sections of a website together.

```
 urlpatterns = [
    path('', include('source.urls')),
    path('admin/', admin.site.urls),
    path('user/signup/', users_views.signup_view, name='signup'),
    path('user/login', users_views.user_login, name='login'),
    path('user/logout/', users_views.user_logout, name='logout'),
    path('book/',
login_required(parkingzoneviews.ReservationView.as_view()),
name='book'),
    path('ticket/',
login_required(parkingzoneviews.Ticket_Pdf.as_view()),
name='ticket'),
    path('all_tickets/',
login_required(parkingzoneviews.Display_Tickets.as_view()),
name='all-tickets'),
    path('checkout/', parkingzoneviews.check_out, name='checkout')
    ]
```

4.7.2 Booking process code

```
class ReservationView(View):
    def get(self, request):
        try:
            user_reservation =
Reservation.objects.get(customer=request.user, checked_out=False)
            if user_reservation:
                messages.warning(self.request, 'Please Check Out
Your Previous Reservation')
                return redirect('index')
        except ObjectDoesNotExist:
            pass

        reservation = ReservationForm()

        return render(request, 'parking_zones/booking.html',
{'form': reservation})

    def post(self, request):
        try:
            user_reservation =
Reservation.objects.get(customer=request.user, checked_out=False)
            if user_reservation:
                messages.warning(self.request, 'Please Check Out
Your Previous Reservation')
```

```
                    return redirect('index')

            except ObjectDoesNotExist:
                pass

        reservation_form = ReservationForm(data=request.POST)

        if reservation_form.is_valid():
            start_date =
reservation_form.cleaned_data['start_date']
            finish_date =
reservation_form.cleaned_data['finish_date']
            parking_zone =
reservation_form.cleaned_data['parking_zone']
            plate_number =
reservation_form.cleaned_data['plate_number']

            parkingzone =
Parking_Zone.objects.get(name=parking_zone)
            if parkingzone.vacant_slots == 0:
                messages.warning(self.request, 'Parking Zone
Full!')
                return redirect('index')

            reservation = reservation_form.save(commit=False)
            reservation.customer = request.user
            reservation.parking_zone = parking_zone
            reservation.ticket_code = create_ticket_code()
            reservation.save()
            #parkingzone =
Parking_Zone.objects.get(name=parking_zone)
            parkingzone.occupied_slots += 1
            parkingzone.save()
            vacantslots = int(parkingzone.num_of_slots) -
int(parkingzone.occupied_slots)
            parkingzone.vacant_slots = vacantslots
            parkingzone.save()
            messages.info(request, 'Successfully Booked')
            return redirect('index')

        return render(request, 'parking_zones/booking.html',
{'form': reservation_form})
```

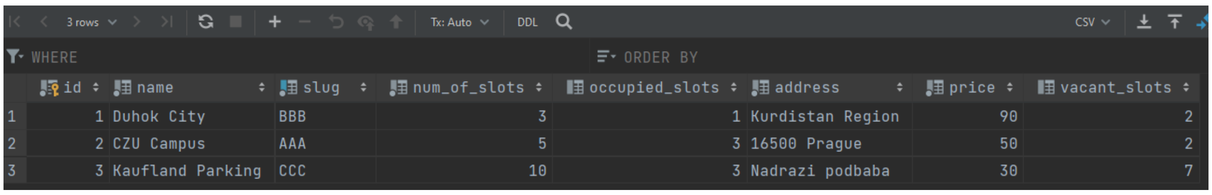
4.7.3 Tickets code

```
class Ticket_Pdf(View):

    def get(self, request):

        today = timezone.now()
        reservation =
Reservation.objects.filter(Q(customer=request.user,
checked_out=False) | Q(customer=request.user,
checked_out=True)).first()
        if reservation:
            params = {
                'today': today,
                'reservation': reservation,
                'request': request
            }
            return Render.render('parking_zones/ticket.html',
params)
        else:
            messages.warning(self.request, f'No Parking
reservation exists for {self.request.user}')
            return redirect('index')
```

4.7.4 Check out code

```
def check_out(request):
    try:
        reservation =
Reservation.objects.get(customer=request.user, checked_out=False)
        if reservation:
            reservation.checked_out = True
            reservation.save()
            parking_zone_name = reservation.parking_zone.name
            parking_zone =
Parking_Zone.objects.get(name=parking_zone_name)
            parking_zone.occupied_slots -= 1
            parking_zone.vacant_slots += 1
            parking_zone.save()
            messages.info(request, 'Successfully Checked Out')

    except ObjectDoesNotExist:
            messages.warning(request, f'No Parking reservation
exists for {request.user}')

    return redirect('index')
```

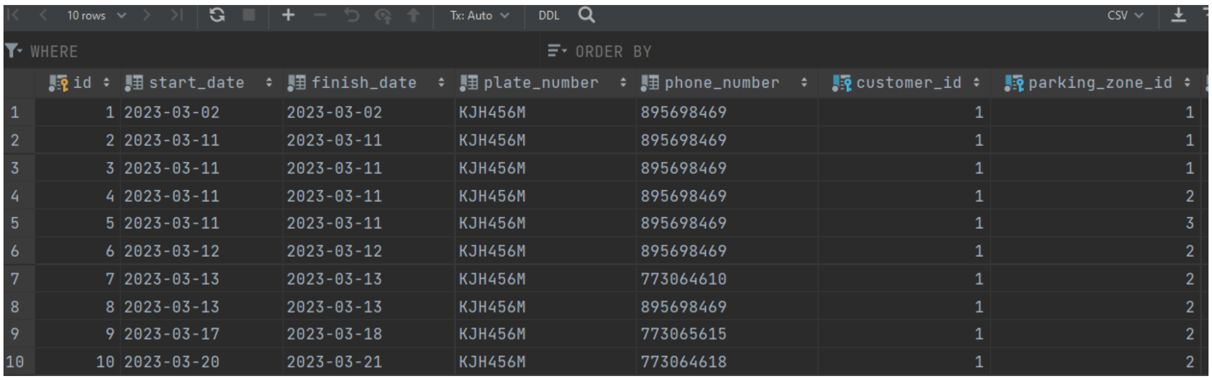
4.7.5 Database of parking zones



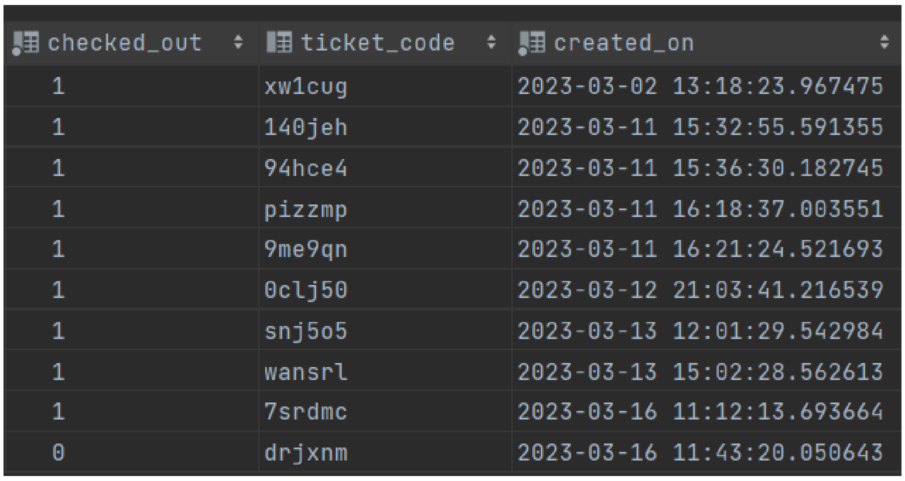
	id	name	slug	num_of_slots	occupied_slots	address	price	vacant_slots
1	1	Duhok City	BBB	3	1	Kurdistan Region	90	2
2	2	CZU Campus	AAA	5	3	16500 Prague	50	2
3	3	Kaufland Parking	CCC	10	3	Nadrazi podbaba	30	7

Table 2: Database of parking zones

4.7.6 Database of reserved parking zones


10 rows
Tx: Auto
DDL
CSV
WHERE
ORDER BY

	id	start_date	finish_date	plate_number	phone_number	customer_id	parking_zone_id
1	1	2023-03-02	2023-03-02	KJH456M	895698469	1	1
2	2	2023-03-11	2023-03-11	KJH456M	895698469	1	1
3	3	2023-03-11	2023-03-11	KJH456M	895698469	1	1
4	4	2023-03-11	2023-03-11	KJH456M	895698469	1	2
5	5	2023-03-11	2023-03-11	KJH456M	895698469	1	3
6	6	2023-03-12	2023-03-12	KJH456M	895698469	1	2
7	7	2023-03-13	2023-03-13	KJH456M	773064610	1	2
8	8	2023-03-13	2023-03-13	KJH456M	895698469	1	2
9	9	2023-03-17	2023-03-18	KJH456M	773065615	1	2
10	10	2023-03-20	2023-03-21	KJH456M	773064618	1	2



checked_out	ticket_code	created_on
1	xw1cug	2023-03-02 13:18:23.967475
1	140jeh	2023-03-11 15:32:55.591355
1	94hce4	2023-03-11 15:36:30.182745
1	pizzmp	2023-03-11 16:18:37.003551
1	9me9qn	2023-03-11 16:21:24.521693
1	0clj50	2023-03-12 21:03:41.216539
1	snj5o5	2023-03-13 12:01:29.542984
1	wansrl	2023-03-13 15:02:28.562613
1	7srdmc	2023-03-16 11:12:13.693664
0	drjxnm	2023-03-16 11:43:20.050643

Table 3: Database of reserved parking zones

4.8 Test

Implementing and testing the website to make sure everything is working, it is user friendly and meet all the requirement. The website or application is tested during this phase, which comes after the development phase, for usability, performance, security, and compatibility with various devices and browsers. To find and address any problems or faults in the website or application, developers may use a variety of testing approaches at this phase, including unit testing, integration testing, system testing, and acceptance testing. To interact with parking system and check if it’s responding to users need, I have done different scenarios.

In web development, there are various testing kinds, each with a distinct goal and scope. Some of the most typical forms of testing used in web development are listed below:

1. Unit testing: With this style of testing, individual web application modules or components are tested separately. In order to ensure that each component performs as planned and to identify any faults or defects early in the development process, unit testing is used. Using testing frameworks like Jest, Mocha, or Jasmine, unit testing is frequently automated.

2. Integration testing: Checking the interrelationships between various web application components is known as integration testing. Integrity testing checks the interoperability of the various components of the program and looks for any defects or issues that might develop due to component interaction. Integration testing can be done manually or using automated testing methods.

3. Functional testing: Testing the functionality of the web application as a whole is known as functional testing. Functional testing is done to make sure that an application complies with all requirements and specifications and performs as expected from the user's point of view. Functional testing can be carried out manually or automatically utilizing testing software.

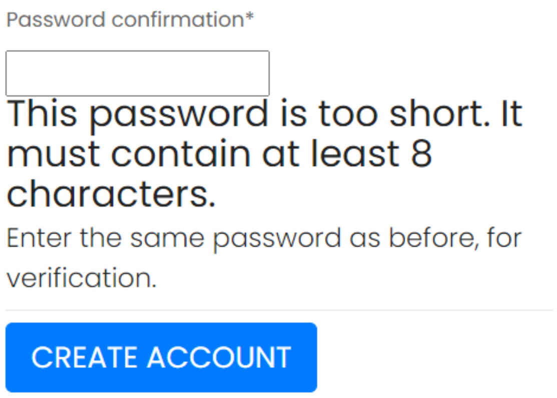
4. Performance testing: Performance testing entails evaluating how well a web application performs under various conditions, such as high traffic or a heavy load. Performance testing checks if an application can manage the anticipated load and operates effectively under pressure. Tools like Apache JMeter or LoadRunner can be used for performance testing.

5. Security testing: Testing for security involves examining the online application's security for flaws like SQL injection, cross-site scripting (XSS), and cross-site request forgery (CSRF). To make sure the application is secure and that sensitive data is secured, security testing is done. Security testing can be carried out manually or automatically using testing software.

6. User Acceptance Testing (UAT): To make sure the online application satisfies users' needs and expectations, UAT entails testing it with actual users. UAT's goal is to make sure the program is user-friendly and offers a positive user experience. Usability testing (UAT) is normally performed manually and may include focus groups, surveys, or other methods.

4.8.1 Scenario 1: Create an account to get access to the system

All you need as first step to get access and interact with the system is to create your own account with your name and generate a strong password under the requirement conditions of strong password. As you can see in figure 25, that the system interacted with the user with providing an error message about the provided password. After you meet the requirement and generate strong password your account will be created and you will get notified message as shown in figure 26.


Password confirmation*

This password is too short. It must contain at least 8 characters.

Enter the same password as before, for verification.

Figure 25: Error message for generated password

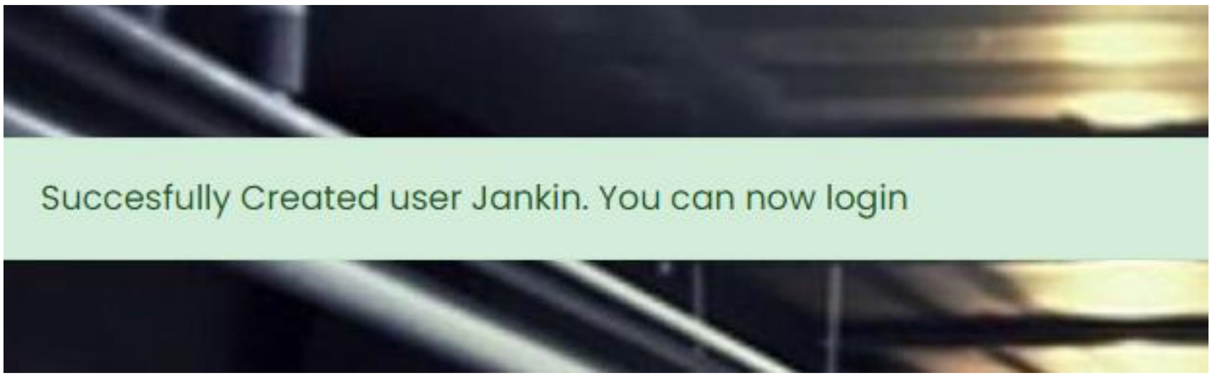
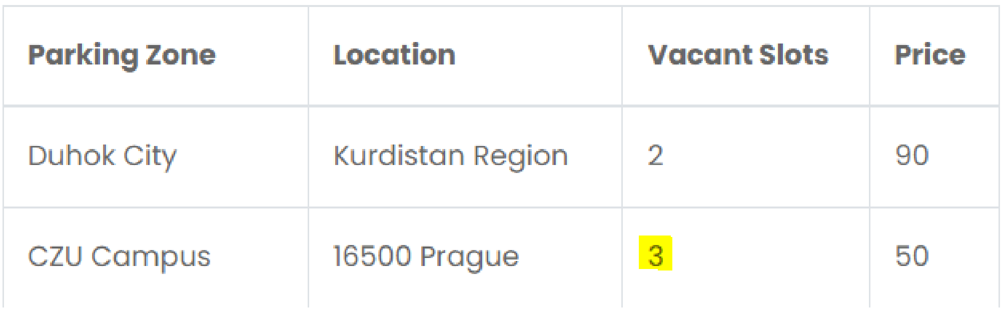

Succesfully Created user Jankin. You can now login

Figure 26: Account Created

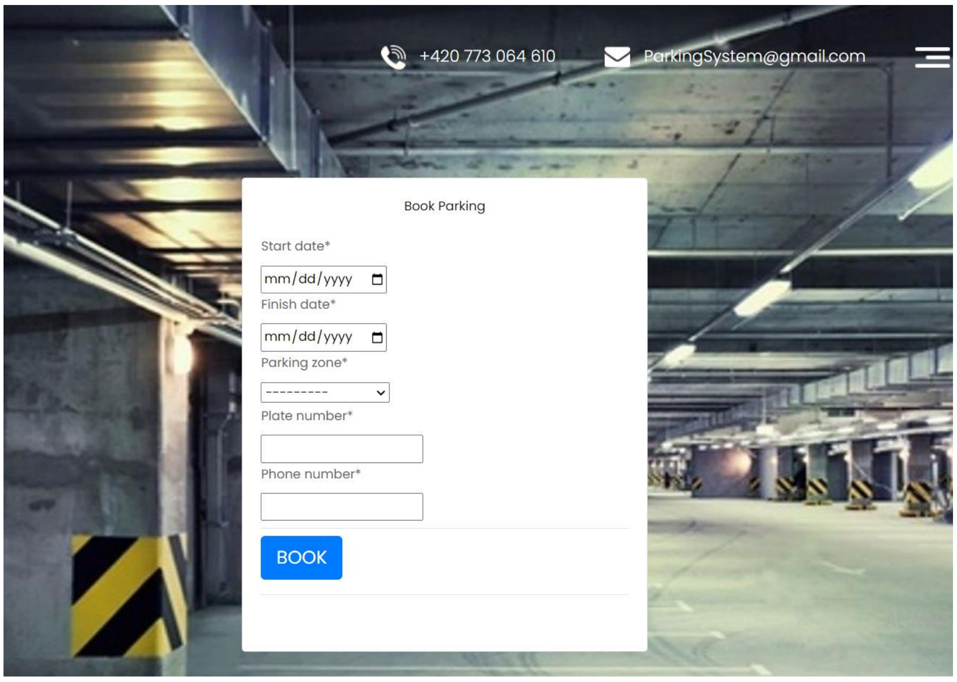
4.8.2 Scenario 2: Booking free zone at CZU campus

Main point of this system is to help users to book free zones in three different locations, after logging in to your account you will be able to view different parking location with available zones and the price of each location. Booking a zone can be done by clicking into "Reserve" bottom which will show book page for the user where they are able to book it for specific days, select specific location and insert plate and phone number. As shown in Figure 4.13. Before booking an available zone let's focus how many free spaces we have at CZU campus to make sure the system is working by seeing the changes in available zones number from before and after booking. As shown in figure 4. There are 3 available zones in CZU campus.



Parking Zone	Location	Vacant Slots	Price
Duhok City	Kurdistan Region	2	90
CZU Campus	16500 Prague	3	50

Table 4: Number of Vacant slots in CZU campus before booking



+420 773 064 610 ParkingSystem@gmail.com

Book Parking

Start date*

mm/dd/yyyy

Finish date*

mm/dd/yyyy

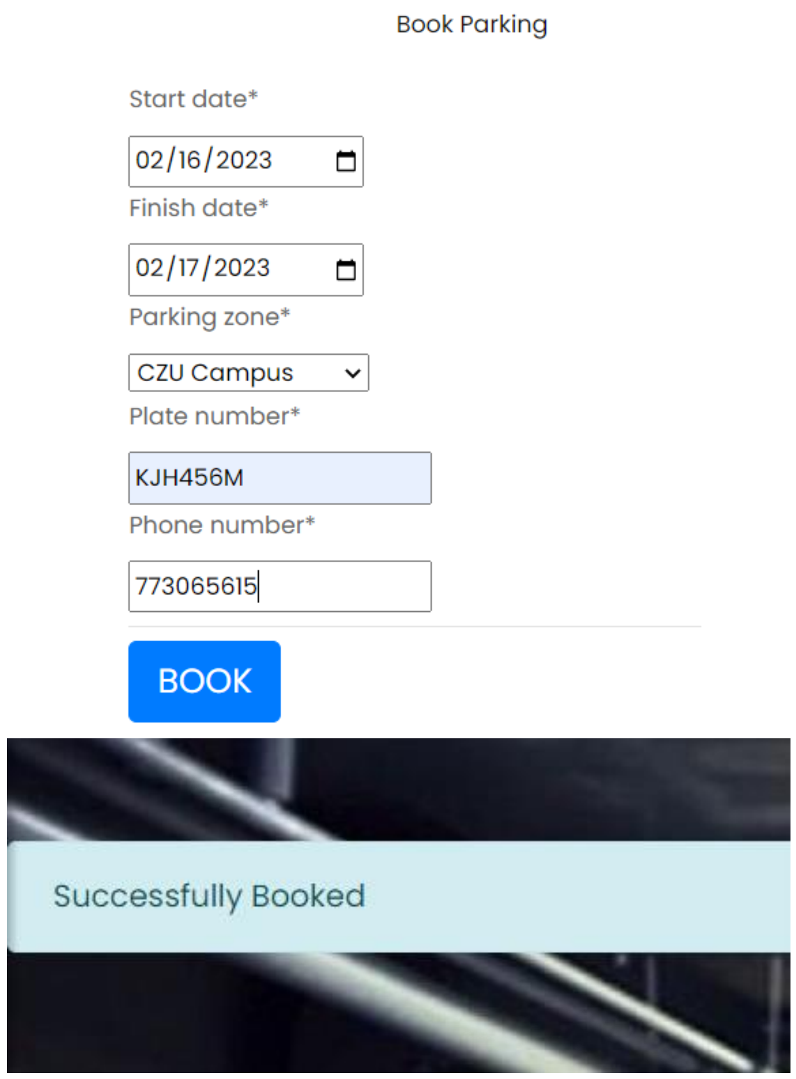
Parking zone*

Plate number*

Phone number*

BOOK

Figure 27: Book a parking process


Book Parking

Start date*

02/16/2023

Finish date*

02/17/2023

Parking zone*

CZU Campus

Plate number*

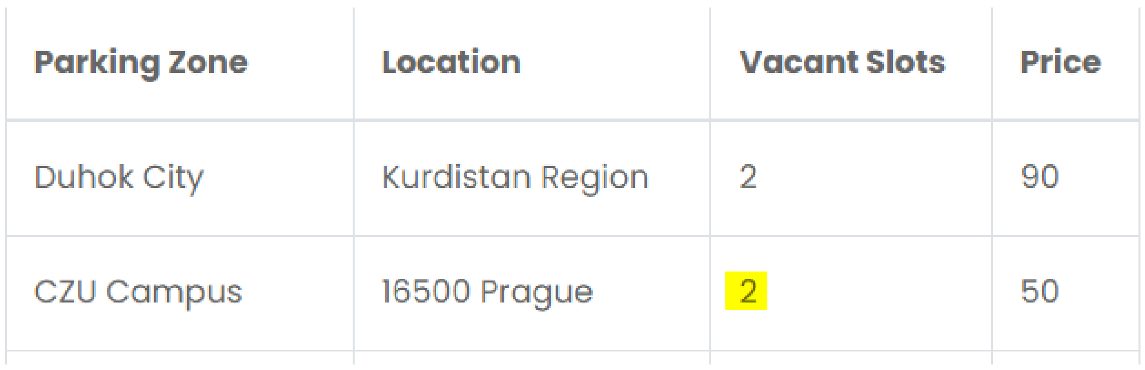
KJH456M

Phone number*

773065615

Successfully Booked

Figure 28: Parking booked



Parking Zone	Location	Vacant Slots	Price
Duhok City	Kurdistan Region	2	90
CZU Campus	16500 Prague	2	50

Table 5: Vacant slots in CZU campus after booking

4.8.3 Scenario 3: Booking another zone within the same account

This system was built on the base of each account should only have one reservation.
You as a user you are not allowed to have multiple reservation under one account, this system will avoid empty reservations.
As third scenario of implementing this system, if you try to have new booking with already existing one, system will notify you to check out first from the previous reservation as shown in figure 4.

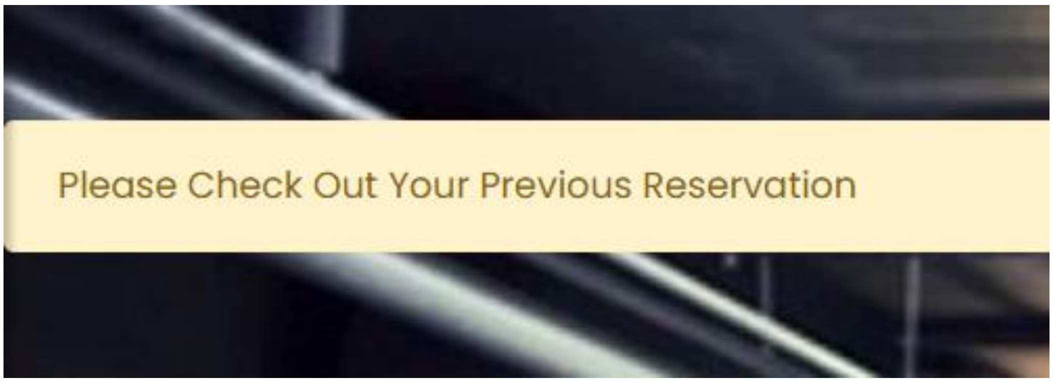
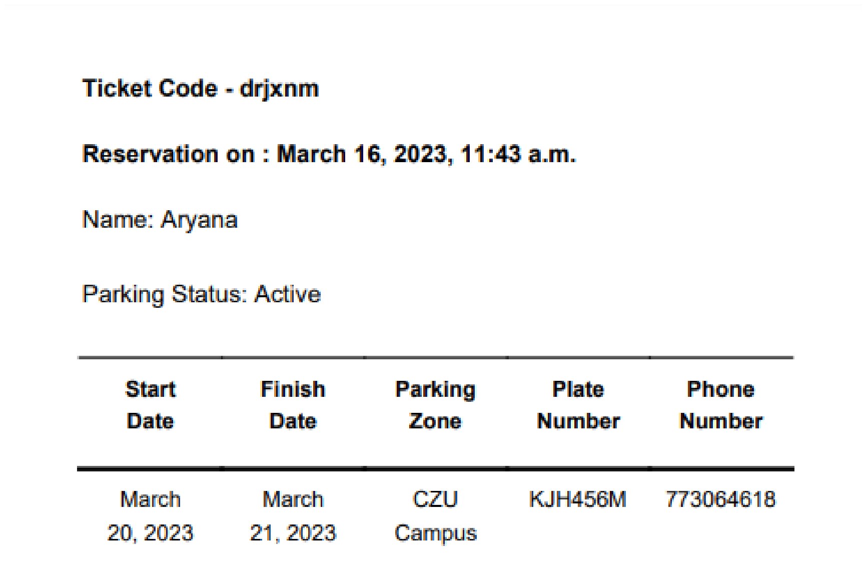

Please Check Out Your Previous Reservation

Figure 29: Notification for checking out

4.8.4 Scenario 4: Printing reserved ticked and show all tickets

One of important feature of this system is user friendly which means it’s easy to use and meet what user need. System provide you with the ability of printing your tickets with all details included, also it’s archiving all your previous tickets that you can go back to it when needed.



Ticket Code - drjxnm

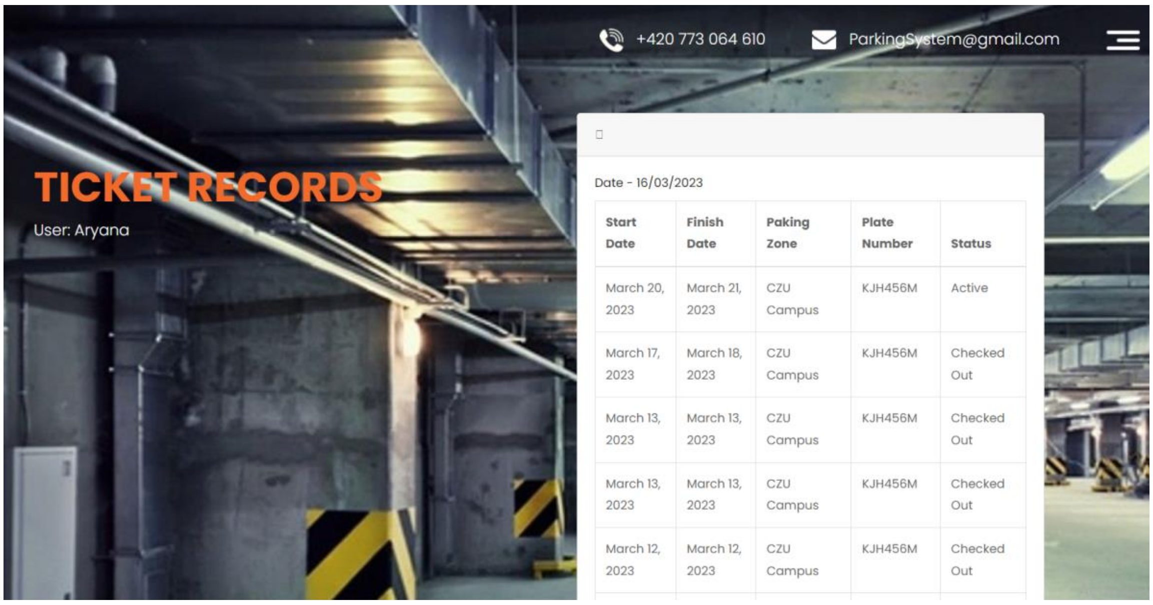
Reservation on : March 16, 2023, 11:43 a.m.

Name: Aryana

Parking Status: Active

Start Date	Finish Date	Parking Zone	Plate Number	Phone Number
March 20, 2023	March 21, 2023	CZU Campus	KJH456M	773064618

Figure 30: Print ticket details



Date - 16/03/2023

Start Date	Finish Date	Paking Zone	Plate Number	Status
March 20, 2023	March 21, 2023	CZU Campus	KJH456M	Active
March 17, 2023	March 18, 2023	CZU Campus	KJH456M	Checked Out
March 13, 2023	March 13, 2023	CZU Campus	KJH456M	Checked Out
March 13, 2023	March 13, 2023	CZU Campus	KJH456M	Checked Out
March 12, 2023	March 12, 2023	CZU Campus	KJH456M	Checked Out

Figure 31: All recorded tickets

4.9 Maintain

The life cycle of web development includes a crucial step called maintenance.
It incorporates elements of the development life cycle's testing and development phases.
In order to ensure that the web application continues to function effectively and satisfy user needs, this stage entails monitoring and updating it. A few examples of maintenance tasks are quality assurance testing, problem solving, and ongoing technical assistance. Regression tests should be run after the creation of new features to verify compatibility. Throughout the maintenance phase, you can also create new features, add new integrations, and make regular upgrades to adapt to changing business requirements.

5. Results and Discussion

Building and improving a web application with easy and understandable functionality that aims to deal with users in order to find and book vacant spots was the main solution of this thesis. As my goal was to build this system by using HTML, CSS and python with Django framework, I always get into the discussion to the reason of using this type of technologies. For example, why no Java or WordPress which is getting so familiar in designing and building great websites. My answer would always be because of Python adaptability, readability, and simplicity. It is frequently used for artificial intelligence, machine learning, data analysis, and web development. A high-level Python web framework called Django is intended for quick development and simple, practical design. It offers a collection of tools and packages, including as a potent ORM (Object-Relational Mapping) system, a templating engine, and built-in support for user authentication and security, that make it easier to construct complicated online applications. Model-View-Controller (MVC) architecture, which is used by Django, divides application functionality into three interrelated parts: the model, the view, and the controller. This frees up developers to concentrate on building clean, reusable code and makes it simple to scale and manage online applications. Also for future work and improving this system and make it working with some sensors, Python is easy and useful for that as well. As PySensors packageis only one of the many libraries and tools available in Python that make it simple to interact with sensors and gather data.

6. Conclusion and Future work

Parking system is a hot field of research due to it is wide range and diversity of applications. Today, most parking system work for people in big cities, malls, companies, and everywhere. Parking system is an effective way that provides you with safe and protected place to park your vehicles. Having parking system in your company or university or even when you are going foe shopping it redusing the congestion, time wasting while finding free zone and protects the environment from pollution. Also building a web application that interact with people for parking is an efficent and modern way to track with technology and make the process eaiser for people. Having this kind of system in our university campus will mainly solve a big issue that is facing our students, university staff and employees with finding a free space to park their vehicles. As well as it will provide us with an idea about the number of cars that entering CZU campus, maybe we will need to build new parking zones to meet the capacity of our university popultion.

Finally , as the technology improves, there will be new ways to use parking system which will help more in soliving the problems that we are facing it every day and make parking process easier and more safe to us.

The proposed system can be improved in future with using more technologies which help to use it in a wide range and be useful in many other places. I will recommend connecting web application with different type of sensors like camera, which can show a visual representation of the region behind the car, and laser sensors, which measure the separation between the vehicle and an object using laser beams. Electromagnetic and ultrasonic proximity sensors, as an object enters the electromagnetic field at the front or rear of the vehicle as it moves forward or backward, electromagnetic sensors create an electromagnetic field there and alert the driver to the object's presence. Ultrasonic sensors are more prevalent and employ sound waves to measure the distance between a vehicle and an item, giving the driver an audible warning when the vehicle approaches an object too closely. would be a great achievement for future. Also another plan for improving parking system is using Cloud based parking system. A parking management system that employs cloud computing technology to store and process parking-related data, such as occupancy, availability, and payment information, is known as a cloud-based parking system.

7. References

1- Mohandes, M., Deriche, M., Abuelma'atti, M. T., & Tasadduq, N. (2019). Preference-based smart parking system in a university campus. *IET Intelligent Transport Systems, 13*(2), 417–423. https://doi.org/10.1049/iet-its.2018.5207.

2- Sadhukhan, P. (2017). An IoT-based E-parking system for smart cities. In *International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (pp. 1062–1066), Udupi. https://doi.org/10.1109/ICACCI.2017.8125982.

3- Geng Y, Cassandras CG (2012) A new "smart parking" system infrastructure and implementation. Procedia Soc Behav Sci 54:1278–1287.

4- N. H. H. M. Hanif, M. H. Badiozaman and H. Daud, "Smart parking reservation system using short message services (SMS)", *Intelligent and Advanced Systems (ICIAS) 2010 International Conference on 2010*, pp. 1-5.

5- Quiñones, D., Rusu, C., Rusu, V.: A methodology to develop usability/user experience heuristics. Comput. Stand. Interfaces **59**, 109–129 (2018).

6- World Wide Web Consortium (W3C): Making the Web Accessible (2020). https://www.w3.org/WAI/. Accessed 29 Jan 2020.

7- H. Wang and W. He, "A Reservation-based Smart Parking System", *Computer Communications Workshops (INFOCOM WKSHPS) 2011 IEEE Conference on Shanghai*, pp. 690-695, 2011.

8- https://www.simbla.com/Why-Is-Website-Development-Important.

9- Gongjun Yan; Weiming Yang; Danda B. Rawat and Stephan Olariu, "SmartParking: A Secure and Intelligent Parking System", IEEE Intelligent Transportation Systems Magazine (Volume: 3, Issue: 1, Spring 2011).

10- https://www.parking.net/about-parking/history-of-parking

11- http://www.myreadingroom.co.in/notes-and-studymaterial/69-html/826-features-of-html.html

12- Dania Delgado, Daniela Zamora, Daniela Quiñones, Cristian Rusu, & Virginica Rusu, **User eXperience Heuristics for National Park Websites.** First Online: 10 July 2020.

13- https://www.clouddefense.ai/blog/system-development-life-cycle

14- https://www.archdaily.com/993988/exploring-the-history-and-future-of-parking-garage-designs

15- http://www.myreadingroom.co.in/notes-and-studymaterial/69-html/826-features-of-html.html

16- https://www.freecodecamp.org/news/difference-between-a-website-and-a-web-application/
17- http://www.dikonia.com/blog/how-to-extend-a-wordpress-website-into-a-business-web-app/
18- Best Parking Sensors (Review & Buying Guide) in 2023 | The Drive
19- Parking Sensors | Front, Rear, Curb, Back Up, Garage – CARiD.com
20- Reading a Sensor with Python - Problem Solving with Python
21- W3Schools Online Web Tutorials
22- MDN Web Docs (mozilla.org)
23- https://gmedia.net.id/info/news/detail/711/WEBSITE-VS-WEB-APPLICATION
24- https://www.metasource.co/fullstack_developer_vietnam_pros_cons/front-end-and-back-end/
25- https://www.edureka.co/blog/django-tutorial/
26- https://www.mobitechspy.com/design-challenges-with-ui-ux-and-solutions/

8. List of Figures

9. List of tables

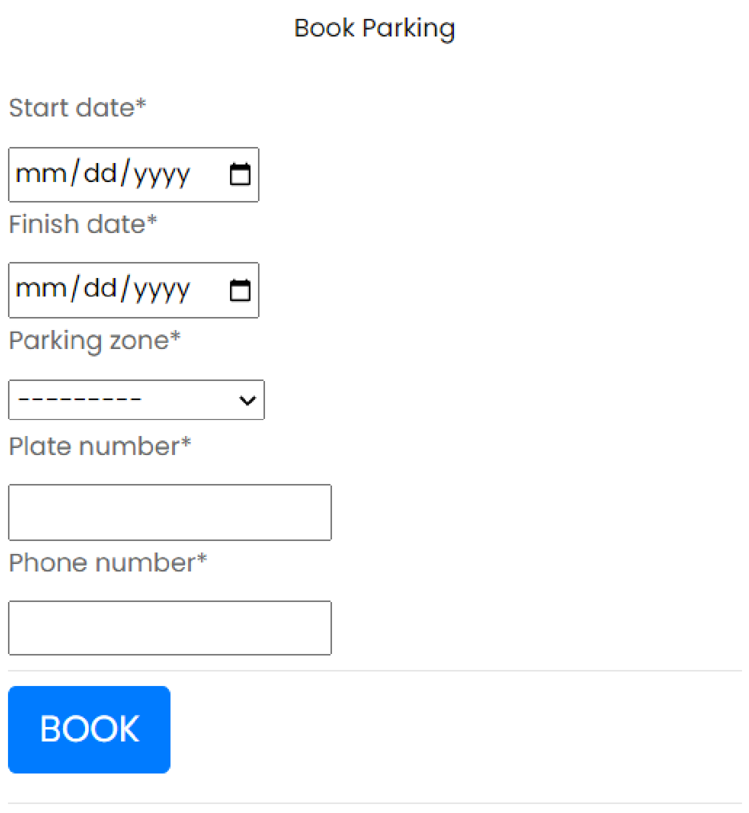
10. List of abbreviations

HTML:	HyperText Markup Language
CSS:	Cascading Style Sheets
URL:	Uniform Resource Locators
API:	Application Programming Interfaces
UX:	User Experience
UI:	User Interface
SQL:	Structured Query Language
SDLC:	System Development Life cycle

11. Appendices

This part includes some part of backend code, and view how it is reflecting in the real web application of the university parking system.

Booking View



Book Parking

Start date*

mm/dd/yyyy

Finish date*

mm/dd/yyyy

Parking zone*

Plate number*

Phone number*

BOOK

Figure 32: Book parking view

Back End code for booking functionality

```
class ReservationForm(ModelForm):
    class Meta:
        model = Reservation
        exclude = ['ticket_code', 'customer', 'checked_out']
        # Validating form fields using widgets
        widgets = {
            'start_date': DateInput(attrs={'type': 'date'}),
            'finish_date': DateInput(attrs={'type': 'date'}),
            'plate_number': TextInput(attrs={'pattern': '^K[A-
Z]{2}[0-9]{3}[A-Z]$', 'title': 'Enter a valid parking space
number'}),
            'phone_number': TextInput(attrs={'pattern': '[0-9]+',
'title': 'Enter digits only '}),
        }
```

HTML code for booking view

```
<div class="container">
    <div class="row">
      <div class="col-sm-9 col-md-7 col-lg-5 mx-auto">
        <div class="card card-signin my-5">
          <div class="card-body">
            <h5 class="card-title text-center">Book Parking</h5>
            <form class="form-signin" method="POST">
              <div class="form-label-group">
                {% csrf_token %}
                {{form|crispy}}
              </div>
              <hr class="my-2">
              <button class="btn btn-lg btn-primary text-uppercase"
type="submit">Book</button>
              <hr class="my-3">
            </form>
            {{ message }}<br>
          </div>
        </div>
      </div>
    </div>
  </div>
```

Login View

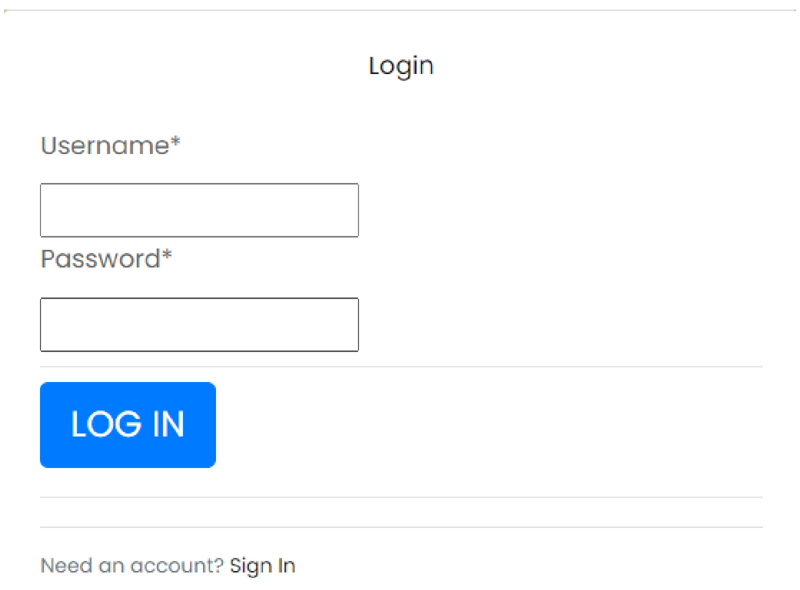


Figure 33: Log in view

Back end code for login functionality

```
def user_login(request):
    form = AuthenticationForm()
    if request.method == 'POST':
        username = request.POST['username']
        password = request.POST['password']
        user = authenticate(username=username, password=password)
        if user is not None:
            login(request, user)
            if user.is_authenticated:
                messages.success(request, f'User {username}
Successfully Logged in')
                return redirect('index')

        else:
            messages.warning(request, 'invalid Credentials')


    context = {
        'form': form
    }
    return render(request, 'users/login.html', context)
```

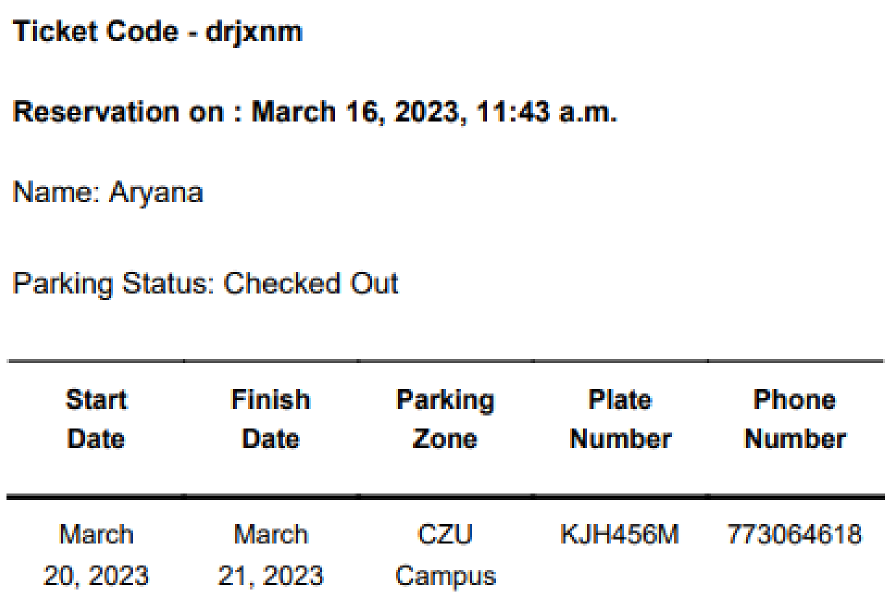
HTML for login view

```
<div class="container">
    <div class="row">
      <div class="col-sm-9 col-md-7 col-lg-5 mx-auto">
        <div class="card card-signin my-5">
          <div class="card-body">
            <h5 class="card-title text-center">Login</h5>
            <form class="form-signin" method="POST">
              <div class="form-label-group">
                {% csrf_token %}
                {{form|crispy}}
              </div>
              <hr class="my-2">
              <button class="btn btn-lg btn-primary text-
uppercase" type="submit">Log in</button>
              <hr class="my-3">
            </form>
            <div class="border-top pt-2">
              <small class="text-muted">Need an account? <a
href="{% url 'signup' %}">Sign In</a></small>
            </div>
          </div>
        </div>
      </div>
    </div>
  </div>
```

Sign in code

```
def signup_view(request):
    if request.method == 'POST':
        form = UserCreationForm(request.POST)
        if form.is_valid():
            form.save()
            username = form.cleaned_data.get('username')
            messages.success(request, f'Succesfully Created user
{username}. You can now login')
            return redirect('login')

    else:
        form = UserCreationForm()

    context = {
        'form' : form
    }
    return render(request, 'users/signup.html', context)
```

Print ticket view



Ticket Code - drjxnm

Reservation on : March 16, 2023, 11:43 a.m.

Name: Aryana

Parking Status: Checked Out

Start Date	Finish Date	Parking Zone	Plate Number	Phone Number
March 20, 2023	March 21, 2023	CZU Campus	KJH456M	773064618

Figure 34: Print ticket view

HTML code for Tickets view

```
<body>

<div class="container">
    <div class="card">
        <div class="card-header">
            <h3>Ticket Code - {{reservation.ticket_code}}</h3>
            <h3>Reservation on : {{reservation.created_on}}</h3>
        </div>

        <div class="list-group">
            <p>Name: {{ request.user }}</p>
            {% if reservation.checked_out is True %}
                <p>Parking Status: Checked Out</p>
            {% else %}
                <p>Parking Status: Active</p>
            {% endif %}
        </div>

        <table class="table">
            <thead>
            <tr>
                <th>Start Date</th>
                <th>Finish Date</th>
                <th>Parking Zone</th>
                <th>Plate Number</th>
                <th>Phone Number</th>
            </tr>
            </thead>
            <tbody>
                <tr>
                    <td>{{ reservation.start_date }}</td>
                    <td>{{ reservation.finish_date }}</td>
                    <td>{{ reservation.parking_zone }}</td>
                    <td>{{ reservation.plate_number }}</td>
                    <td>{{ reservation.phone_number }}</td>
                </tr>
            </tbody>
        </table>
    </div>
</div>

</body>
```

Main page section code

```
<section>
    <div class="container-fluid">
        <div class="row">
            <div class="col-md-5">
                <div class="full slider_cont_section">
                    <h4>{{parking_zone.name}}</h4>
                <p>{{parking_zone.address}}</p>
                <!-- <div class="button_section">
                   <a href="{% url 'book' %}">Book Spot</a>
                </div> -->
             </div>
          </div>
          <div class="col-md-7">
             <div class="col-lg-8">
                <div class="card mb-4">
                   <div class="card-header">
                       <i class="fas fa-table mr-1"></i>
                   </div>
                   <div class="card-body">
                       <div class="table-responsive">
                           <table class="table table-bordered"
id="dataTable" width="100%" cellspacing="0">
                               <thead>
                                   <tr>
                                    <th>Total Slots</th>
                                    <th>Vacant Slots</th>
                                    <th>Price</th>
                                   </tr>
                               </thead>
                               <tbody>
                                <tr>

<td>{{parking_zone.num_of_slots}}</td>
                                    <td>{{ parking_zone.vacant_slots
}}</td>
                                    <td>{{parking_zone.price}}</td>
                                    <td>
                                       <a href="{% url 'book' %}">
                                           <button class="btn btn-
primary">Book Parking</button>
                                       </a>
                                   </td>

                                </tr>


                               </tbody>
                           </table>
                       </div>
                   </div>
               </div>
             </div>
          </div>
       </div>
    </div>
 </section>
```

Header code

```
<header>
    <!-- header inner -->
    <div class="container">
       <div class="row">
          <div class="col-lg-3 logo_section">
             <div class="full">
                <div class="center-desk">
                  <!-- <div class="round">
                     <input name="q" id="search" type="text"
placeholder="Enter Name/Location" />
                   </div>
                   <div class="input-field third-wrap">
                     <button class="btn btn-primary"
type="submit">Search</button>
                   </div> -->
                <!-- <div class="logo"> <a href="/"><img src="{%
static 'source/images/logo.png' %}" alt="#"></a> </div>-->
                </div>
             </div>
          </div>
          <div class="col-lg-9">
             <div class="right_header_info">
                <ul>
                   <li><img style="margin-right: 15px;" src="{%
static 'source/images/phone_icon.png' %}" alt="#" /><a
href="#">+420 773 064 610</a></li>
                   <li><img style="margin-right: 15px;" src="{%
static 'source/images/mail_icon.png' %}" alt="#" /><a
href="#">ParkingSystem@gmail.com</a></li>
                   <!-- <li>
                      <input type="image" placeholder="Enter Name
or Location" src="{% static 'source/images/search_icon.png' %}"
alt="">
                  </li> -->
                   <li>
                      <button type="button" id="sidebarCollapse">
                         <img src="{% static
'source/images/menu_icon.png' %}" alt="#" />
                      </button>
                   </li>
                </ul>
             </div>
          </div>
       </div>
    </div>
    <!-- end header inner -->
</header>
```

Footer code

```
            <footer>
          <div class="container">

          </div>
      </footer>
      <!-- end footer -->

      <!-- copyright -->

      <div class="cpy_right">
          <div class="container">
              <div class="row">
                  <div class="col-md-12">
                     <div class="full">
                        <p>© 2023 All Rights Reserved. Design by
<a href="">Aryana Haji</a></p>
                     </div>
                  </div>
              </div>
          </div>
      </div>

      <!-- right copyright -->

   </div>
</div>
```

Github link

https://github.com/Aryanaa3/University-Parking-System.git